SOLPS-ITER

November 30, 2025

Preface

This manual is built from the following code repository versions:

```
SOLPS-ITER version :  a586a83
B2.5 version :  3.0.9e-311-gc2dbe8ce5
Eirene version :  3.0.9e-122-g6df4d97e
Carre version :  3.0.9e-35-g2e61eec
DivGeo version :  3.0.9e-42-gf6d9885
```

This is the combined work of many authors. Started by D.P. Coster, it includes contributions from

- D.P Coster and O. Wenisch, the original **b2plot** manual (Appendix J)
- K. and A.S. Kukushkin, the DG manual (Appendix L), with revisions by A. Chin and D. Stotler
- M. Stanojevic, Appendix D on boundary conditions
- X. Bonnin, for many contributions and the **SOLPS-ITER** updates
- S. Voskoboynikov, E. Kaveeva, and I. Senichenkov, for the **SOLPS-ITER** model equations (Appendix C) and output quantities information (Appendix H)
- S. Makarov, for the Zhdanov-Grad closure description

All errors, omissions *etc.* should be e-mailed to Xavier.Bonnin@iter.org.

Contents

Contents **3**

List of Figures **13**

List of Tables **16**

1 Introduction **18**

1.1 Obtaining and updating the code . . . 19

1.2 Requirements . . . 22

1.3 Initial set-up . . . 23

1.3.1 Installation on local machines . . . 29

1.4 Before every **SOLPS-ITER** session . . . 30

1.5 Examples . . . 30

1.6 Using a central installation . . . 32

2 The structure **33**

2.1 Setup and configuration files . . . 34

2.2 modules . . . 36

2.2.1 modules/B2.5 . . . 37

2.2.2 modules/Eirene . . . 41

2.2.3 modules/amds . . . 43

2.2.4 modules/adas . . . 43

2.3 lib . . . 44

2.4 runs . . . 44

2.5 scripts . . . 45

2.5.1 scripts/commands . . . 61

2.5.2 scripts/matlab . . . 61

2.5.3 scripts/palettes . . . 61

2.5.4 scripts/Balance . . . 61

2.5.5 scripts/nc2text_simple . . . 61
2.5.6 scripts/MatlabPostProcessing . . . 61
2.5.7 scripts/Matlab_SPb . . . 61
2.6 data . . . 62

3 Running the standalone version of B2.5 63
3.1 b2ag, setting up the geometry . . . 67
3.1.1 b2ag.dat . . . 67
3.1.2 Geometry for linear device . . . 68
3.2 b2ah, setting up the default physics parameters . . . 71
3.2.1 b2ah.dat . . . 71
3.3 b2ar, setting up the atomic physics rates . . . 76
3.3.1 b2ar.dat . . . 76
3.4 b2ai, the default initial plasma state . . . 78
3.4.1 b2ai.dat . . . 78
3.5 b2mn, the main program . . . 80
3.5.1 b2mn.dat . . . 80
3.5.2 b2.boundary.parameters . . . 81
3.5.3 b2.transport.parameters . . . 90
3.5.4 b2.numerics.parameters . . . 92
3.5.5 b2.transport.inputfile . . . 94
3.5.6 b2.sources.profile . . . 97
3.5.7 b2.neutrals.parameters . . . 102
3.5.8 b2.atomic_physics_rescale.parameters . . . 102
3.5.9 b2.user.parameters . . . 102
3.5.10 b2.XXX.parameters . . . 102
3.5.11 While the code is running . . . 102
3.6 b2pl, the plotting program . . . 104
3.7 b2yg, plotting the geometry . . . 121
3.8 b2yh, plotting the run parameters . . . 123
3.9 b2yi & b2yp, plotting the plasma state . . . 124
3.10 b2yr, plotting the atomic physics rates . . . 127
3.11 b2ym, producing movie output . . . 128
3.12 b2yn, display the progress of inner B2.5 iterations . . . 129
3.13 b2yq, display the evolution data . . . 130
3.14 b2yt, changing from one grid size and species set to another . . . 132

3.15 b2co, recovering from a crash . . . 135
3.16 b2uf & b2fu, converting the plasma output file . . . 136
3.17 b2rd, fetching an archived MDSPLUS run . . . 137
3.18 b2ymb, post processing for monoblock heat transfer model . . . 138
3.19 b2yrp, post processing for reflector plates heat transfer model . . . 139
3.20 b2ydm, post processing of dome wall loading . . . 141
3.21 b2_ual_write, writing last state to IDS . . . 142
3.21.1 b2_ual_rewrite, rewriting an existing IDS . . . 143
3.22 b2xx, the others . . . 144

4 Running the standalone version of Eirene **145**
4.1 Specific SOLPS-ITER blocks in the Eirene input file . . . 145
4.1.1 Block 14. Data for interfacing routine INFUSR . . . 145
4.1.2 Block 15. Data for interfacing routine GEOUSR . . . 147
4.1.3 Information for MPI . . . 148
4.2 Pressure Feedback Loop . . . 149
4.3 Modelling for Beryllium migration . . . 150

5 Running the coupled version of B2.5-Eirene **152**
5.1 Input files . . . 152
5.1.1 b2ag.dat . . . 152
5.1.2 b2ai.dat . . . 152
5.1.3 b2ar.dat . . . 153
5.1.4 b2ah.dat . . . 153
5.1.5 b2mn.dat . . . 161
5.1.6 b2.boundary.parameters . . . 163
5.1.7 b2.numerics.parameters . . . 164
5.1.8 b2.transport.parameters . . . 164
5.1.9 b2.neutrals.parameters . . . 164
5.2 Output files . . . 165
5.3 Changing dt . . . 170
5.4 Dynamic (”Time-dependent”) mode . . . 171
5.5 Time-dependent puff functions . . . 176
5.6 Averaging scheme . . . 177
5.7 b2ye, statistical error assessment . . . 178

6 Using MDSPLUS **179**
6.1 SOLPS tree structure . . . 180
6.1.1 SOLPS . . . 180
6.1.2 IDENT . . . 180
6.1.3 SNAPSHOT . . . 181
6.1.4 TALLIES . . . 185
6.1.5 TIMEDEP . . . 187
6.1.6 MOVIES . . . 191
6.1.7 DIAG . . . 191
6.2 Accessing data from the MDSPLUS server . . . 192
6.2.1 Using standard experimental data viewing programs . . . 192
6.2.2 Using IDL . . . 194
6.2.3 Using MATLAB . . . 195
6.2.4 Using Python . . . 195
6.3 Querying the database . . . 202
6.4 Coupling to other codes via MDSplus . . . 205
6.5 Some information about the server . . . 208

7 Running DivGeo **209**

8 Running Carre **211**

9 Running agg **215**

A Internal B2.5 documentation **219**
A.1 b2cdca.F . . . 219
A.2 b2cdcr.F . . . 222
A.3 b2cdcv.F . . . 239
A.4 b2cdci.F . . . 266
A.5 b2cdcn.F . . . 321

B The physics of B2.5 **353**
B.1 Derivation of parallel viscosity term in momentum equation . . . 360
B.2 Boundary conditions with drifts and currents . . . 361
B.3 Problems for a fluid description of the edge plasma . . . 362
B.4 Neutral models . . . 364
B.4.1 Fluid neutrals . . . 364
B.4.2 First-flight model . . . 366

B.4.3 Kinetic neutrals: B2.5-Eirene coupling . . . 366
B.5 Classical transport: Braginskii, Balescu and Zhdanov formulations . . . 369
B.6 Classical ion transport: Zhdanov-Grad module . . . 369

C Model equations for B2.5 **372**
C.1 Sign conventions . . . 372
C.2 Continuity equations for particle conservation . . . 373
C.2.1 Source terms in density equations . . . 375
C.2.2 Velocities . . . 376
C.2.3 Particle transport coefficients . . . 377
C.3 Parallel momentum conservation equations . . . 379
C.3.1 Sources for parallel momentum equations . . . 379
C.3.2 Parallel momentum fluxes . . . 384
C.3.3 Parallel momentum transport coefficients . . . 384
C.4 Electron energy equations . . . 386
C.4.1 Electron energy sources . . . 386
C.4.2 Poloidal electron energy fluxes . . . 387
C.4.3 Electron energy transport . . . 388
C.4.4 Radial electron energy fluxes . . . 391
C.5 Ion energy equations . . . 392
C.5.1 Ion energy sources . . . 392
C.5.2 Poloidal ion energy fluxes . . . 394
C.5.3 Poloidal ion energy transport coefficients . . . 394
C.5.4 Radial ion energy fluxes . . . 396
C.5.5 Radial ion energy transport coefficients . . . 397
C.6 Current continuity equation . . . 397
C.6.1 Poloidal currents . . . 398
C.6.2 Radial currents . . . 400
C.6.3 Conductivities . . . 402
C.7 Typical boundary conditions . . . 402
C.7.1 Particle balance equations . . . 403
C.7.2 Parallel momentum balance for ions . . . 408
C.7.3 Current continuity equation . . . 414
C.7.4 Energy balance equation for electrons . . . 415
C.7.5 Energy balance equation for ions . . . 419
C.8 Total energy balance . . . 424

C.9 Parallel viscosity . . . 428
C.10 Scaling of residuals . . . 428
C.11 Zhdanov-Grad closure . . . 429
C.11.1 Introduction . . . 429
C.11.2 Notation . . . 430
C.11.3 Ion heat conductivity . . . 430
C.11.4 Velocity-dependent ion heat flux . . . 433
C.11.5 Thermal force . . . 434
C.11.6 Friction force . . . 437
C.11.7 Viscous stress . . . 439
C.11.8 A-coefficients, G-objects and Z-variables . . . 446
C.11.9 Heat source . . . 449
C.11.10Currents . . . 450
C.11.11The thermal and friction forces (legacy) . . . 453
C.11.12The distribution function . . . 453

D Boundary conditions in B2.5 **455**
D.1 Introduction . . . 455
D.2 Analytical Expressions for the Standard ("Old") Boundary Sources in the B2.5 Code . . . 456
D.2.1 Definitions of Physical Parameters Used in the Analytical Expressions for the Standard Boundary Sources . . . 456
D.2.2 Standard Boundary Sources . . . 458

E Quantities stored in b2time.nc **465**

F Quantities stored in b2movies.nc and eirenemovies.nc **472**

G Quantities stored in b2tallies.nc **475**

H Quantities printed by `'b2wdat_iout' = '4'` **478**

I Region definitions **491**
I.1 Regions for 1-D and linear cases . . . 492
I.2 Regions for limiter cases . . . 493
I.3 Regions for SN . . . 494
I.4 Regions for CDN . . . 496
I.5 Regions for DDN . . . 498
I.6 Regions for periodic BC stellarator island case . . . 500
I.7 Regions for low field side snowflakes . . . 502

J b2plot manual **504**

J.1 Introduction . . . 504
J.2 Environment variables . . . 505
J.3 Commands . . . 505
J.4 Common errors . . . 531
J.5 Available Functions and Operators in **b2plot** . . . 531
J.5.1 Data plotting commands . . . 532
J.5.2 Data manipulating functions . . . 536
J.5.3 Arithmetic commands . . . 542
J.5.4 Stack manipulating commands . . . 550
J.5.5 Page format parameters . . . 556
J.5.6 Graph format commands . . . 556
J.5.7 Graph appearance commands . . . 558
J.5.8 Overlays to contour and line plots . . . 561
J.5.9 Changing contour parameters . . . 565
J.5.10 miscellaneous . . . 568
J.5.11 Still need to do . . . 573
J.6 Commands taking a string parameter . . . 574
J.6.1 Manipulating text appearing on graphs . . . 574
J.6.2 Commands using external files . . . 575
J.7 Examples . . . 580
J.7.1 Mesh plot . . . 580
J.7.2 Plotting plasma solution data . . . 581
J.7.3 Chord plots . . . 582
J.7.4 Wall plots . . . 582
J.7.5 Target plots . . . 583
J.7.6 Radiation plots . . . 584
J.7.7 Fluxes and values plot . . . 585

K Miscellaneous **588**

K.1 b2mod_dimensions.F . . . 588
K.2 Making movies . . . 590
K.3 Compiling B2.5 . . . 590
K.4 Defines . . . 615
K.4.1 SOLPS-ITER . . . 615
K.4.2 Carre . . . 617

K.4.3 DivGeo . . . 618
K.4.4 B2.5-Eirene . . . 619
K.4.5 B2.5 . . . 619
K.4.6 Eirene . . . 632
K.5 NAG calls in SOLPS . . . 641
K.5.1 B2.5 . . . 641
K.5.2 EIRENE . . . 642
K.6 Performance monitoring . . . 642

L DG manual: Version 2.10 **643**
L.1 Introduction . . . 644
L.1.1 Purpose of DG . . . 644
L.1.2 DG objects . . . 644
L.2 Using DG . . . 646
L.2.1 Starting DG . . . 646
L.2.2 Main window . . . 646
L.2.3 Using tools . . . 647
L.2.4 Using Undo . . . 648
L.3 Working with models . . . 648
L.3.1 Opening an existing model . . . 648
L.3.2 Creating a new model . . . 648
L.3.3 Saving a model . . . 648
L.3.4 Quitting DG . . . 649
L.4 Working with elements . . . 649
L.4.1 Converting a template . . . 649
L.4.2 Creating elements manually . . . 649
L.4.3 Editing elements . . . 650
L.4.4 Changing external normals . . . 650
L.4.5 Renumbering elements . . . 650
L.5 Controlling how objects are displayed . . . 651
L.5.1 Changing the zoom factor and location . . . 651
L.5.2 Hiding and displaying objects . . . 651
L.5.3 Rotating the view . . . 651
L.5.4 Stretching and shrinking the view . . . 651
L.5.5 Top-down view . . . 652
L.6 Working with magnetic surfaces and grid points . . . 653

L.6.1 Equilibrium conversion and manipulation . . . 653
L.6.2 Importing equilibrium . . . 656
L.6.3 Output mode . . . 656
L.6.4 Creating and editing surfaces . . . 656
L.6.5 Defining the X-point . . . 660
L.6.6 Limiter and linear configurations . . . 660
L.6.7 Snowflake configurations . . . 661
L.6.8 Creating and editing grid points . . . 661
L.7 Working with chords . . . 662
L.7.1 Creating chords . . . 663
L.7.2 Modifying chords . . . 663
L.8 Working with sources . . . 663
L.8.1 Creating and editing sources . . . 663
L.9 Working with meshes . . . 664
L.9.1 Importing a SONNET format mesh . . . 664
L.9.2 Modifying a mesh . . . 664
L.9.3 Repairing mesh cells using rotated view and variable aspect ratio . . . 664
L.9.4 Defining cells . . . 664
L.10 Working with variables . . . 665
L.10.1 Basics . . . 665
L.10.2 Working with layers . . . 667
L.10.3 Setting variables . . . 667
L.11 Creating output files . . . 668
L.11.1 Types of output files . . . 668
L.11.2 Producing output files . . . 669
L.12 Customizing DG . . . 669
L.12.1 Installing DG . . . 669
L.12.2 Files needed by DG . . . 669
L.12.3 Specifying directory masks . . . 669
L.12.4 The configuration file . . . 669
L.12.5 Creating or modifying the configuration file . . . 670
L.13 File formats . . . 672
L.13.1 Simple template file format . . . 672
L.13.2 Structure file format . . . 672
L.13.3 Targets file format . . . 672
L.13.4 Output file format . . . 673

M Some information from the SOLPS 2002 Course Notes **675**

M.1 Breakdown of B2.5 programs . . . 675

M.2 B2.5 ".dat" and ".prt" files . . . 677

M.3 B2.5 communication files . . . 678

N Subversion repository logs **679**

N.1 SOLPS-ITER . . . 679

N.2 Changes in output for the AUG_16151_D test case . . . 679

O License **681**

O.1 European Union Public Licence v. 1.1 . . . 681

O.1.1 1. Definitions . . . 681

O.1.2 2. Scope of the rights granted by the Licence . . . 681

O.1.3 3. Communication of the Source Code . . . 682

O.1.4 4. Limitations on copyright . . . 682

O.1.5 5. Obligations of the Licensee . . . 682

O.1.6 6. Chain of Authorship . . . 683

O.1.7 7. Disclaimer of Warranty . . . 683

O.1.8 8. Disclaimer of Liability . . . 683

O.1.9 9. Additional agreements . . . 683

O.1.10 10. Acceptance of the Licence . . . 683

O.1.11 11. Information to the public . . . 684

O.1.12 12. Termination of the Licence . . . 684

O.1.13 13. Miscellaneous . . . 684

O.1.14 14. Jurisdiction . . . 684

O.1.15 15. Applicable Law . . . 684

O.1.16 Appendix . . . 685

P Contributing **686**

Bibliography **690**

Index **693**

List of Figures

3.1 Typical geometry in B2.5 for a linear device. This case is taken from a SOLPS-ITER case for the linear device Magnum-PSI. The starting point of the magnetic field lines is represented as a dashed green line. . . . 70

3.2 Region numbers for single-null geometries. . . . 94

3.3 Region numbers for double-null geometries. . . . 95

3.4 Residuals. . . . 103

3.5 Power crossing various surfaces. . . . 103

3.6 Upstream (outer midplane) electron temperature. . . . 104

3.7 Monoblock tile cross-section . . . 139

3.8 Reflector plate tile cross-section . . . 140

5.1 Upstream (outer midplane) electron temperature for a range of time-steps. . . . 171

5.2 Upstream (outer midplane) electron temperature for a range of time-steps with a core speed-up factor. 171

5.3 Upstream (outer midplane) electron temperature for a range of time-steps with and without a core speed-up factor. . . . 172

5.4 Downstream (outer target) electron temperature for a range of time-steps. . . . 172

5.5 Downstream (outer target) electron temperature for a range of time-steps with a core speed-up factor. . 173

5.6 Downstream (outer target) electron temperature for a range of time-steps with and without a core speed-up factor. . . . 173

5.7 Upstream (outer midplane) electron temperature. . . . 174

5.8 Upstream (outer midplane) electron temperature. . . . 174

5.9 Upstream (outer midplane) electron temperature. . . . 175

5.10 Upstream electron temperature for several timesteps after puff b.c. applied. . . . 175

5.11 Total electron content for several timesteps after puff b.c. applied. . . . 176

5.12 Examples of the time-dependent puff functions. . . . 176

6.1 Using CVIEW to access SOLPS MDSplus data. . . . 193

6.2 Using JETDSP to access SOLPS MDSplus data. . . . 194

6.3 Comparison of profiles for a range of shots. . . . 198

6.4 Scatter plot for separatrix temperature versus density. . . . 204
6.5 Histogram plot for separatrix temperature versus density. . . . 204
6.6 Log histogram plot for separatrix temperature versus density. . . . 205

B.1 Coordinate system: x is the poloidal coordinate, y is the radial coordinate orthogonal to the flux surfaces. The coordinate $\perp$ corresponds to the direction perpendicular both to the magnetic field and to the y-axis. The directions of magnetic field and plasma current correspond to normal operation conditions of ASDEX–Upgrade (ion ∇B drift directed towards the X-point). . . . 354
B.2 Impact of ion flux limits for a $10^{19}\ m^{-3}$ upstream density case. . . . 364
B.3 Impact of ion flux limits for a $2 \times 10^{19}\ m^{-3}$ upstream density case. . . . 365
B.4 Dependence of the various models for the $\kappa_{e\|}$ prefactor as a function of Z_{eff}. . . . 370
B.5 Dependence of the various models for the parallel electron conductivity $\sigma_{\|}$ prefactor as a function of Z_{eff}. . . . 370
B.6 Dependence of the various models for the thermo-electric coefficient $\alpha_{e\|}$ prefactor as a function of Z_{eff}. 371

J.1 Ion and atomic densities for DIII–D. . . . 532
J.2 Trajectory plot. . . . 534
J.3 Demonstration of various color palettes available in **b2plot** . . . 576
J.4 BLS, BOL, DCN and DIV chords. . . . 577
J.5 HAL chords. . . . 578
J.6 Mesh plots. . . . 580
J.7 Electron temperature plots. . . . 581
J.8 Other solution data plots. . . . 581
J.9 Chord plots applied to JET. . . . 582
J.10 Wall plots. . . . 582
J.11 Target plots. . . . 583
J.12 C and Ne radiation. . . . 584
J.13 Total radiation for a D+He+C example case. . . . 585
J.14 Line plots of fluxes and other quantities. . . . 587

L.1 Demonstration of a rotated view with associated dialog box and tear-off menu. . . . 652
L.2 Demonstration of the variable aspect ratio capability along with the associated tear-off menu. . . . 653
L.3 `Create Surface(s)...` dialog with highlighted `Structures`. . . . 657
L.4 Zoom of the PFR showing the highlighted "bounding element". . . . 659
L.5 Zoom of the twisted mesh cells near the inner divertor target of an NSTX geometry. . . . 665
L.6 After stretching and rotation, the overlapping of the cells in the upper part of the region cells depicted in Fig. L.5 becomes clear. . . . 666

N.1 Residuals of the AUG_16151_D standard case for different code versions and options. . . . 680

N.2 Relative changes for the considered options. 680

List of Tables

3.1 Changes of quantity names from **SOLPS 4.x b2plot** to **SOLPS 5.x b2pl.exe**. 120

5.1 Input switch settings for time-dependent puff functions. 177

B.1 Comparison of classical transport coefficients according to the Braginskii, Balescu, and Zhdanov formulations 369

E.1 Scalar time-dependent quantities stored in b2time.nc 467

E.2 Vector time-dependent quantities stored in b2time.nc. Note that the conversion to "per unit area" for fluxes is done in the script "2d_profiles"; the actual data stored in b2time.nc for fluxes are the fluxes per cell. There is a similar conversion from J to eV for temperatures. 468

E.3 Double-null vector time-dependent quantities stored in b2time.nc 469

E.4 Multi-dimensional time-dependent quantities stored in b2time.nc 470

E.5 Scalar batch-averaged time-dependent quantities stored in b2time.nc 471

F.1 Multi-dimensional time-dependent quantities stored in b2movies.nc 473

F.2 Multi-dimensional time-dependent quantities stored in eirenemovies.nc 474

G.1 Quantities stored in b2tallies.nc 477

H.1 Files produced by means of `'b2wdat_iout = 4'`, the variable they contain, and the equation to which they are related 488

J.1 Available quantities 512

J.2 Commands acting on the "matrix" stack together with their stack requirements. 515

J.3 Commands acting on the "matrix" stack plotting together with their stack requirements. 518

J.4 Commands assigning real variables from stack for use in plotting 519

J.5 Commands taking character strings as parameters. Note that the string should be delimited by single quotes. 521

J.6 Quantities that can be stored in the "Eirene" stack. 522

J.7 Commands related to the "Eirene" stack and triangular grid data plots. 523

J.8 Quantities that can be stored in the "wall" stack. 525

J.9 Commands related to the ”wall” stack and wall plots. . . . 527
J.10 Available target quantities for plotting. . . . 528
J.11 Commands related to the ”target” stack and target plots. . . . 529
J.12 Available **b2plot** constants. . . . 530
J.13 Additional commands. . . . 530

K.1 Defines for Carre and where they are used. . . . 618
K.2 Defines for DivGeo and where they are used. . . . 619
K.3 Defines for B2.5 and where they are used. . . . 632
K.4 Defines for Eirene and where they are used. . . . 640
K.5 NAG calls from B2.5 642
K.6 NAG calls from Eirene 642

L.1 DG external objects 645
L.2 DG internal objects 646

Chapter 1

Introduction

This document describes the **SOLPS-ITER** code package and its components. Following this introduction is a chapter (2) describing the structure of the code. Then chapter 3 describes the standalone version of `B2.5`. This is followed by a description of the standalone version of `Eirene` in chapter 4 and then of the coupled `B2.5-Eirene` in chapter 5. Then follow chapters on `DivGeo` (7), `Carre` (8) and `agg` (9). Appendix A contains copies of the `B2.5` internal documentation files, B describes the physics in `B2.5`, C gives the full model equations used in `B2.5`, D gives details about some `B2.5` boundary conditions, E information about the quantities stored in `b2time.nc`, G information about the quantities stored in `b2tallies.nc`, H information about the quantities that can be output using `b2wdat_iout`, I gives the region definitions used in the code, J is an updated version of the **b2plot** manual, K contains information about array dimensions and compilation options, L is the `DG` manual, M includes a listing of the programs and files included in the `B2.5` part of the **SOLPS-ITER** package, followed by the repository logs, license document, and contribution rules.

Historically, there have been many implementations of the `SOLPS` (Scrape-Off Layer Plasma Simulation) code package, according to which version of the fluid plasma code (`B2` or `B2.5`) and of the `Eirene` Monte Carlo neutral transport code are used. We distinguish between **SOLPS 4.x** which uses `B2` and **SOLPS 5.x** which uses `B2.5`. This document is devoted solely to **SOLPS-ITER** although several features are common to multiple versions. **SOLPS-ITER** contains the `B2.5` plasma kernel and a recent version of `Eirene`. Most of the current development and support is done exclusively on **SOLPS 5.x** and **SOLPS-ITER** versions (with the important exception of the Forschungszentrum Jülich and their collaborators where **SOLPS 4.x** is still in use). The main differences of note between `B2` and `B2.5` are:

1. For ions, `B2` solves a *total* energy equation, while `B2.5` solves an *internal* energy equation. See section B.4.3 for an example of the consequences.
2. The primary momentum variable is the *poloidal* velocity in `B2` and the *parallel* velocity in `B2.5`.
3. The velocities are computed on cell faces in `B2` and on cell centres in `B2.5`.
4. The current continuity equation is solved by default in `B2.5`, but not in `B2`.
5. The treatment of drifts and currents is much more complete in `B2.5` than in `B2`.
6. Classical transport in `B2.5` follows the Zhdanov and Balescu treatments by default, while it uses Braginskii in `B2` (see B.5 for a comparison).
7. Some variables names and indexing have changed. See section 3.6 for a partial list.
8. The run parameters to `B2.5` are split into several input files. See sections 3.5, A.5 and M.3 for a detailed description.

Some effort has been undertaken for the user to be able to toggle between the **SOLPS 4.x** and **SOLPS 5.x** numerical treatments. The interested reader is referred to the listing of run switches in section A.4 for the complete list. Two

converters also exist to transfer run output from **SOLPS 4.x** or **SOLPS 5.x** to **SOLPS-ITER**:

From 4 to 5 and ITER `b2sxdr`

From 5 and ITER to 4 `b2xd`

From 5 to ITER `b2yt`

Additional documentation on other code versions can be found in the `${SOLPSTOP}/doc` directory. Namely, the original report by Bas Braams is:

`BRAAMS_-_A_Multi-Fluid_Code_for_the_Simulation_of_the_Edge_Plasma_in_Tokamaks_-_NET_Report_1987.pdf ,`

and an excellent review of **SOLPS 4.x** can be found in:

`Martine_Baelmans_-_Juelich_Report_1994.pdf .`

When referring to **SOLPS-ITER** in scientific papers, please use [1, 2] and references therein. You will also find these references in the `${SOLPSTOP}/doc` directory. For work involving `Eirene`, also quote reference [3].

Several of the **SOLPS4.2** features are detailed in `Kotov_SOLPS42_Report.pdf`.

The next section discusses obtaining and updating the code.

1.1 Obtaining and updating the code

To obtain the code a request should be sent to Xavier Bonnin (`Xavier.Bonnin@iter.org`) at ITER, the code Responsible Officer (RO).

The first formality will then be for prospective users to obtain an IDM (ITER Document Management) account, in case they do not have one already. This account request will be done by the RO, and the requesting user will receive an automated electronic message asking for confirmation of the provided e-mail address. This e-mail address should preferably be a professional one, otherwise requesting users need to provide a way for the IO to verify their belonging to an authorized ITER collaboration partner (academic or non-military state institutions within the ITER Member States, or that have an existing contractual relationship with the IO). This may also require the signature of an IMAS Access Request between the IO and the new user's institution, if one is not already in force. Another option is for the user to be sponsored nominatively by a third party institution through their own IMAS Access Request.

The next step, after the IDM account has been obtained, is for the user to agree to the ITER Integrated Modelling and Analysis Suite (IMAS) terms of use and request access to the needed repositories. The form to fill out for this is to be found at: https://jira.iter.org/servicedesk/customer/portal/1/create/257. For a full **SOLPS-ITER** installation, you should request access to the following repositories:

SOLPS-ITER The overarching **SOLPS-ITER** repository

B2.5 For the `B2.5` source code

EIRENE For the `Eirene` source code

CARRE For the `Carre` source code

DIVGEO For the `DG` source code and the equilibrium conversion programs

TIARA For the `TIARA` grid generator

SOLPS-GUI For the graphical environment in which **SOLPS-ITER** can be run

GR Software The graphics library needed by `Eirene` and the triangulation program

MSCL A utility library to parse the `DG` output files

AMNS ADAS A repository of atomic data used by the codes

Users may request access to other repositories as well according to their needs or interest. If not intending to participate in code development, one may ask for `REPO_READ` permissions only, instead of `REPO_WRITE`.

Note that access to the `Eirene` repository is also subject to the user agreeing to the terms of the `Eirene` license, which is not managed by the IO, but rather by Forschungszentrum Jülich. For this, go to the `Eirene` website at http://www.eirene.de and follow the registration process there.

After access is granted, the user will be able to consult the **SOLPS-ITER** GIT repository (https://git.iter.org/projects /BND/repos/solps-iter/browse) maintained at the IO, and the **SOLPS-ITER** SharePoint web site hosted by ITER (https://sharepoint.iter.org/departments/POP/CM/IMAS/SOLPS-ITER). The manual to the SOLPS-GUI can be found online at https://static.iter.org/imas/assets/solps-iter/html/. Links to the IDM-controlled documents related to **SOLPS-ITER** can be found in a centralized folder (https://user.iter.org/?uid=Q92BAQ). The account request will also serve for registration of the user to the **SOLPS-ITER** mailing list (`SOLPS-ITER@iter.org`). Additionally, there is a Confluence Wiki page located at https://confluence.iter.org/display/IMP/SOLPS-ITER for the purpose of collecting user contributions to the body of knowledge about the code and which contains minutes of the **SOLPS-ITER** User Forums and Release Notes for the various code versions. Lastly, a Slack discussion forum has been set up. To join, please send a request to the **SOLPS-ITER** RO at ITER: `xavier.bonnin@iter.org`.

The **SOLPS-ITER** Git repository includes the repositories for `B2.5`, `EIRENE`, `Carre` and `DivGeo` individually as submodules. Some service libraries (such as the `GGD`, `MSCL` and `GR` service libraries), can be found in the repository in the `IMAS Extra` and `Libraries` section. Users can also have access to the standalone `SOLPS4.3` repository.

One can generally browse the ITER Git repositories here: https://git.iter.org and **SOLPS-ITER** specifically here: https://git.iter.org/projects/BND/repos/solps-iter/browse, but since these repositories are restricted users will first need to authenticate with their ITER (IDM) username and password - possibly twice, once for the intranet, and once for the Git web front end in the top right hand corner in the web browser window.

To do this, one must upload a copy of their public ssh key here: https://git.iter.org/plugins/servlet/ssh/account/keys or create a token at https://git.iter.org/plugins/servlet/access-tokens/manage (by following links under "Manage Account"). Then one should be able to simply clone a copy of **SOLPS-ITER** to a local machine by doing:

```
> git clone ssh://git@git.iter.org/bnd/solps-iter.git
> cd solps-iter
> git submodule init
> git submodule update
```

If using the SSH method, make sure that there is no `PreferredAuthentications` setting in your ssh configuration file, as it can conflict with the IO GIT repository access protocol. Please also note that the SSH key needs to be 1024 bits.

It is best afterwards to position oneself on the `master` branches, which is achieved by the `switch_to_master` script (which can be used after the SOLPS-ITER session environment is established, see Section 1.4) or, alternatively, by hand, with the commands:

```
> git checkout master
> cd modules/B2.5
> git checkout master
> cd ../Eirene
> git checkout master
> cd ../DivGeo
> git checkout master
> cd ../Carre
> git checkout master
> cd ../adas
> git checkout master
```

For later updates, it is recommended you use the `solps-iter_update` script. If you are planning to do code development, your starting point should be the `develop` branches, which you can switch to by using the

`solps-iter_update_develop` script instead.

IMPORTANT: *When updating the code, some routines or other configuration files may have been moved, renamed, added, or deleted, and the updating script itself may have changed. In that case, compilation will likely fail on the first update attempt. The user should then start afresh with a new session, and run the update script a second time. If compilation still fails, then the user should remove old compiled files by means of the appropriate* `gmake clean` *instruction and attempt another compilation, from scratch.*

Please note that because of the way some of the **SOLPS-ITER** utility scripts function, the path to the **SOLPS-ITER** top directory must not contain any of the strings `.debug`, `.mpi`, `.openmp`, or `.ig`.

Since the user-id is immutably encoded into any changes, it is always a good idea to properly configure a local installation of Git on all the systems on which one plans on making code revisions with, e.g.:

```
> git config --global user.name "Your Name"
> git config --global user.email your.name@anywhere.you.are
```

Users who have obtained `REPO_WRITE` permissions can, for their own development, create `feature/[topic]`, `bugfix/[topic]`, or `hotfix/[topic]` branches and raise a "Pull Request" (https://git.iter.org/projects/BND/repos/solps-iter/pull-requests) to have them merged into the `develop` branch after pushing them back to the ITER repository. The `develop`, `master` and `svn/*` branches are protected and can only be manipulated by the owners of the repository.

Issues are handled using JIRA (https://jira.iter.org/browse/IMAS) and users should feel encouraged to create new issues associated with **SOLPS-ITER** (selected from the drop-down list of components) if they have any problems or suggestions. (Please ignore the options associated with resource estimates and sprints.)

For more general help with Git, we can recommend the documentation here (http://git-scm.com/) and the Pro Git book (linked from there) in particular. Another useful resource for novice Git users is the tutorial from Codecademy available at https://www.codecademy.com/learn/learn-git.

When using Git, the command

```
> git status
```

will check if there are any local changes to your local copies of the version-controlled files from the repository, using as a reference the state of the repository from when you last cloned or updated it. To update these references to the current state of the `origin` repository, please use

```
> git fetch
```

If that command returns no output, then your references are up-to-date. Otherwise, if someone else has been committing changes to the repository, Git will indicate the tags of the updated repository. You can then import these latest changes with the command

```
> git pull
```

followed by

```
> git submodule update
```

which will also update the files contained in the various submodules **SOLPS-ITER** is linking to. It is best to do this with your local directory up-to-date and your local changes committed, otherwise Git will ask you to resolve any conflicts before you may proceed. If you wish to update your entire **SOLPS-ITER** distribution, the recommended method is to use the automated script:

```
> solps-iter_update
```

which will fetch and pull the master branches for the main **SOLPS-ITER** repository and all of its submodules, provided you are in a clean state, i.e. with no locally modified files under version control, unless these are stashed or committed to local branches. After the distribution files have been updated, the script will attempt to recompile the code.

SOLPS-ITER is a complicated code package and it would be difficult to learn how to install, compile and run it from this document and the internal documentation alone. In the past we have found that an initial visit of at least two weeks to a research team already proficient with **SOLPS-ITER** use with subsequent visits, or about two months and no further visits, are desirable.

The Git **SOLPS-ITER** repository is structured as a self-contained whole, but some effort was also devoted to be able to check out, as individual standalone modules, the main pieces of the SOLPS suite, i.e. `DivGeo`, `Carre`, `Eirene` and `B2.5`, although this is not recommended.

1.2 Requirements

To best use the **SOLPS-ITER** code suite, the user will need, locally installed:

Fortran 90/95 Compiler the main programs are written in `FORTRAN` (see Section 1.3 for a list of supported compilers)

C Compiler the `DivGeo` code and a few utility routines are written in `C`

MPI Parallelization framework (optional)

NAG mathematical library (optional) (available from http://www.nag.com/content/nag-library)

BLAS/LAPACK mathematical libraries (available from http://www.netlib.org/)

NCAR plotting library (available from https://www.ncl.ucar.edu/Download/)

GR plotting library (optional, developed at FZ-Jülich, available from the IO Git repository at https://git.iter.org/projects/LIB/repos/gr-software/browse)

GKS graphical kernel library (available from http://ngwww.ucar.edu)

HDF5 data storage library (available from http://support.hdfgroup.org/HDF5/)

JSON data storage library (Fortran version, available from https://jacobwilliams.github.io/json-fortran/)

NetCDF storage/archival library (optional) (available from http://www.unidata.ucar.edu/netcdf/) (**SOLPS-ITER** needs at least version 4)

MSCL utility library (available from the IO Git repository at https://git.iter.org/projects/LIB/repos/mscl/browse)

makedepend to build the dependencies needed by the `Makefiles`

gnuplot for many analysis scripts

nc2text for obtaining time-dependent data traces (optional)

plotmtv for obtaining graphs of the time evolution of various profiles (optional)

LaTeX interpreter to produce this manual (optional)

GIT version control software

python scripting language (**SOLPS-ITER** needs at least version 2.7, but version at least 3.5 is required for many additional features)

/bin/bash used by some scripts

/bin/csh used by some scripts

/bin/ksh used by some scripts

/bin/sh used by some scripts

/bin/tcsh used by some scripts

Most of the required libraries should be, in principle, already available in the various HPC clusters as modules. It is recommended for these to be loaded as modules rather than be locally re-built, so that they may be shared amongst multiple users on the same machine, so as not to duplicate large files and to centralize their management.

If one uses local copies of these libraries, a copy or link to them should be put in the `${SOLPSTOP}/lib/${HOST_NAME}.${COMPILER}` directory.

Those libraries which are not available in a cluster must be necessarily locally built for the code to be used. By running the script:

```
> install_dependencies
```

it is possible to automatically obtain several libraries which might be not already available in some HPC clusters. In particular, the Blas/Lapack, NetCDF and NCARG libraries (using a local Conda installation based on `miniforge`), and the Gnuplot, JSON-Fortran, MSCL, GKS, and GR libraries (which are locally built from source). Both the local Conda installation and the locally compiled libraries will reside in the `${SOLPSTOP}/lib/${HOST_NAME}.${COMPILER}` directory. Note the NCARG library can be obtained only for cluster installations using a `gfortran` compiler. The paths and the compiling flags needed to correctly find and link these libraries are also automatically set.

For fuller code capability, it is also recommended to load the IMAS (Integrated Modelling and Analysis Suite) environment, provided by the IO, whenever possible.

If the `nc2text` utility program is not present on the system, but the `NetCDF` library is available, the code will then automatically build an alternative skeleton version and add it to the scripts path.

If there is no LaTeX interpreter found on the system, this manual will not be built. For other systems on which one may not wish to build the manual, the environment variable `NO_MANUAL` can be set to any non-empty string, for instance as part of the `$SOLPSTOP/SETUP/setup.csh.${HOST_NAME}.${COMPILER}(.local)` files.

1.3 Initial set-up

For each local implementation, environment variables are used to designate the location where the various libraries are to be found, as well as the preferred compilers to be used. These are propagated throughout the **SOLPS-ITER** code structure and used by the Makefiles and the various scripts. It is therefore important to carefully assign their values.

These environment variables are found in files `config.${HOST_NAME}.${COMPILER}`. Shared settings that are independent of host location are located in files `config.common.${COMPILER}`.

The current list of recognized hostnames, deduced by means of the `whereami` script, includes:

ASIPP Academica Sinica, Hefei: *.ipp.cas.cn

CCFE Culham Centre for Fusion Energy: *.ccfe.ac.uk

DARWIN Stand-in for a MacOS Darwin operating system on a stand-alone laptop

FZJ Forschungszentrum Jülich: *.fz-juelich.de, *.ipp.kfa-juelich.de

GA General Atomics: *.gat.com

IAEA Internal Atomic Energy Agency: *.iaea.org

IFERC IFERC supercomputer: *.iferc-csc.org

IN-DA ITER-India: *.iter-india.org

IPP IPP-Garching: *.ipp.mpg.de, *.ipp-garching.mpg.de, *.rzg.mpg.de

IPP-HGW IPP-Greifswald: *.ipp-hgw.mpg.de

IPPCR Institute for Plasma Physics - Czech Republic: *.tok.ipp.cas.cz

IPR Institute for Plasma Research, Gandhinagar: *.ipr.res.in

ITER ITER domain: *.iter.org

ITM ITM Gateway machine: *.efgw.local, *.eufus.eu, *.efda-itm.eu

JET JET domain: *.jet.uk

KEIO Keio University of Yokohama: *.keio.ac.jp

LEUVEN KU-Leuven: *.mech.kuleuven.be, *.hpc.kuleuven.be

LINUX Stand-in for a Linux-based operating system on a stand-alone laptop

MARCONI MARCONI supercomputer: *.marconi.cineca.it

MEPHI Moscow Engineering Physics Institute: *.mephi.ru

MPCDF Max Planck Computing and Data Facility: *.mpcdf.mpg.de

NERSC DoE supercomputer center: *.nersc.gov

NSCC National Supercomputing Center: *.nscc-hf.cn

ORNL Oak Ridge National Laboratory: *.ornl.gov

PITAGORA PITAGORA supercomputer: *.pitagora.cineca.it

PPPL PPPL domain: *.pppl.gov

SPBSTU St. Petersburg State Polytechnical University: *.spbstu.ru

SURFSARA Snellius National Supercomputer: *.snellius.surf.nl

SWIP SouthWestern Institute of Plasma Physics, Chengdu: *.swip.ac.cn

UL University of Ljubljana, *.uni-lj.si

WM College of William and Mary: *.wm.edu

AU_UNKNOWN All other domain names from Australia and associated territories

CA_UNKNOWN All other domain names from Canada

CH_UNKNOWN All other domain names from Switzerland

CN_UNKNOWN All other domain names from China and associated territories

EU_UNKNOWN All other domain names from EURATOM Member countries and their associated territories

IN_UNKNOWN All other domain names from India

JP_UNKNOWN All other domain names from Japan

KR_UNKNOWN All other domain names from the Republic of Korea

KZ_UNKNOWN All other domain names from Kazakhstan

RF_UNKNOWN All other domain names from the Russian Federation

UK_UNKNOWN All other domain names from the United Kingdom and associated territories

US_UNKNOWN All other domain names from the United States and associated territories, including *.edu and *.gov

UNKNOWN everywhere else

If your specific domain name is not on this list, please add it to `whereami`. If necessary, for example when the `hostname` and `nslookup` commands used by `whereami` do not provide the desired `HOST_NAME` value, you bypass the default domain name recognition in one of two ways, listed in their order of precedence:

1. set the SOLPS_HOST_NAME_FORCE environment variable in your current shell session, i.e.:

```
setenv SOLPS_HOST_NAME_FORCE <your_desired_HOST_NAME_value>
```

2. create a ${SOLPSTOP}/SETUP/setup.csh.HOST_NAME.local file to bypass the whereami script permanently, with contents like:

```
setenv HOST_NAME <your_desired_HOST_NAME_value>
```

At the moment the following compiler platforms are supported, assuming a Linux work environment:

cray The Cray compiler, in use at the JFRS supercomputing center

pgf90 Portland Group 32-bit Fortran compiler, in use at JET and ORNL

ifort64 Intel 64-bit Fortran compiler, in use at ITER

gfortran The gfortran compiler, in use at IPP, EFGW (EUROfusion Gateway), FZ-Jülich, and KU-Leuven

g95 The g95 compiler

nag_f90 The NAG Fortran compiler

For each hostname domain, a default compiler can be defined by means of the default_compiler file. If your domain name does not yet have a default compiler associated with it or you want to change it, please edit the default_compiler script accordingly. If necessary, the default compiler assignment can be bypassed in one of two ways, listed in their order of precedence:

1. specify an argument to the source setup.csh command, i.e.:

```
source setup.csh ifort64
```

2. set the SOLPS_COMPILER_FORCE environment variable in your current shell session, i.e.:

```
setenv SOLPS_COMPILER_FORCE ifort64
```

Then, if this is the first installation in your file space, please do:

```
first_setup
```

This will, if they are not already present in the distribution, create the various config.${HOST_NAME}.${COMPILER} files for each of the **SOLPS-ITER** submodules. Feel free to modify them afterwards according to your local requirements. As with whereami and default_compiler, if those modifications are worth sharing with the other users, please commit them back to the repository using a pull request.

For each combination of hostname and compiler, the set of corresponding environment variables are loaded from the ${SOLPSTOP}/SETUP/setup.csh.${HOST_NAME}.${COMPILER} file and, subsequently, the ${SOLPSTOP}/SETUP/setup.csh.${HOST_NAME}.${COMPILER}.local file. The compiler switches are given through the configuration files ${SOLPSTOP}/SETUP/config.${HOST_NAME}.${COMPILER} and ${SOLPSTOP}/SETUP/config.common.${COMPILER}. The former file should contain the location of all the main libraries with which the **SOLPS-ITER** executables will be linked and their linking instructions. The latter file provides shared instructions for that compiler, independent of location. These files also contain the shared compiler directives for all the **SOLPS-ITER** object files (see Section K.4.1 for a full listing) and the name of the debugger program to be used, if any. Users should make sure the settings within these files fit their run environment and modify them as needed. In particular, if the default python command on the system points to version 2.7 or similar, and a newer version 3 executable is available, it is recommended to define the PYTHON environment variable to point to that executable in the appropriate local ${SOLPSTOP}/SETUP/setup.csh file. If changes are needed, please forward them to the **SOLPS-ITER** RO so that they may be included in the set of distributed files to help other future users.

For the case where single submodules are cloned from the **SOLPS-ITER** repository, without the full regalia of programs, additional sets of configuration files can be found in the `config` directory in each module. The compiler settings specific to this module are included there. Users are encouraged to modify these as needed. If local changes are required that should not be archived in the repository, then users should create a file named `config.${HOST_NAME}.${COMPILER}.local` which the `Makefile` will load and use to overwrite the more general settings from the distributed `config` files.

As a general rule, a `config.${HOST_NAME}.${COMPILER}.local` file should not be a modified copy of the original `config.${HOST_NAME}.${COMPILER}` file, but instead *only* contain those variable definitions that need to be locally modified.

For easier building of the `gfortran` and `ifort64` toolchains, including the IMAS modules and tools for **SOLPS-ITER**, a `SETUP/easybuild-local.sh` script is provided. The script includes a complete list of modules as they appear on the ITER SDCC cluster and is convenient for standalone/local "builders" and for site administrators that want to build an ITER-equivalent installation on their machine. Example `setup.csh.UL.gfortran` and `setup.csh.UL.ifort64` are thus very similar to `setup.csh.ITER.gfortran` and `setup.csh.ITER.ifort64`, respectively. The configuration files `config.UL.gfortran` and `config.UL.ifort64` can then simply be symlinked to the corresponding `config.ITER` versions.

Essentially, this script exploits the functionality of the EasyBuild tool by adding search paths to ITER-specific easyconfig `.eb` sources from the ITER Git repository. The only requirement for this script is a recent version of Python 3 and `modulesfiles` or `Lmod` support. The rest is being downloaded from the internet and the ITER Git website. A quick start for building may be by creating a Personal Access Token from the ITER Git website and entering:

```
export EASYBUILD_MODULES_TOOL=Lmod # EnvironmentModules are default
export HTTP_AUTH_BEARER="ReplaceWithPersonalAccessToken"
SETUP/easybuild-local.sh --help | more
SETUP/easybuild-local.sh
SETUP/easybuild-local.sh --imas-foss install
SETUP/easybuild-local.sh --imas-apps
```

Note that the above minimal example requires Python 3.8+, `Lmod` and Korn shell to be installed on the system and functional. The rest is installed by the script in the `easybuild.local` directory or elsewhere if desired.

Some Windows implementations have been attempted, and require an additional `-DWINDOWS` compiler switch, but are not curently being supported. Windows subsystem for Linux (WSL2) on Windows 10 and 11 can be a convenient replacement for standalone builds using the `SETUP/easybuild-local.sh` script.

If you wish your domain name to be included in the list of hostnames above, please contact the **SOLPS-ITER** RO and send the hostname you would like to have and the Internet address range to which it would apply. Please also send the relevant compiler configuration files so they may be distributed to future users from the same domain or serve as examples for other users using the same compiler.

If the NAG library is not available (it requires a pay licence), then the BLAS and LAPACK packages should be used instead, and links to them added to the corresponding `config` files.

To compile the coupled version of the code you should begin by checking that the default array dimensions provided fit your problem size. These are stored in the file `${SOLPSTOP}/modules/B2.5/src/modules/b2mod_dimensions.F`. If the dimensions are not right for you, then do the following:

```
sbb
cd src
mkdir -p modules.local
cp modules/b2mod_dimensions.F modules.local
touch modules/.new_modules
```

and edit the file `modules.local/b2mod_dimensions.F` so that the dimensions (mainly used for the namelists in `B2.5` and the `Uinp` input file build-up program) are big enough (see the description of the relevant variables in

section K.1). Once you have edited that file, you will need to recompile with:

```
stop
gmake listobj
gmake tags
gmake depend
gmake
```

so that your local file will be used. The last `gmake` command may of course be replaced by `gmake solps(_debug)`, `gmake solps_mpi(_debug)`, `gmake all(_debug)` or `gmake all_mpi(_debug)`, etc... as needed. Conversely, if you want to revert back to the `b2mod_dimensions.F` provided with your code distribution, use the commands:

```
sbb
rm src/modules.local/b2mod_dimensions.F
touch src/modules/.new_modules
touch src/modules/b2mod_dimensions.F
```

and proceed with the recompilation as above.

It is possible to pass down Makefile options with the syntax

```
gmake MAKE_OPTIONS=-j8
```

for instance to multi-thread on the compilation. It is not recommended to give compilation options directly in the compilation line (say, with `gmake -j8 solps`) because the **SOLPS-ITER** compilation includes many parts which must be done in order and sometimes serially.

Once the configuration files and dimensions have been filled in with the appropriate information for the local implementation desired, one proceeds with the compilation of the code. For this purpose, a master **SOLPS-ITER** `Makefile` is provided, located at `${SOLPSTOP}/Makefile`. The options supported by this `Makefile` are:

solps (Default) Compiles the necessary components for a full **SOLPS-ITER** implementation, including:

- **libs** Installs the needed libraries according to configuration files
- **carre** The `Carre` grid generator and associated translators
- **divgeo** The `DivGeo` graphical input file builder and accompanying equilibrium and template converters
- **b25eirene** The `B2.5-Eirene` main code and accompanying programs
- **uinp** The `Uinp` translator from `DivGeo` output to `B2.5-Eirene` input
- **sonnet-light** The `Sonnet` library
- **triang** The `Tria` and `Triageom` triangulation utilities
- **amds** The `AMDS` utility for building consistent reaction sets
- **manual** Builds this manual

all The same as **solps** above, but in addition:

- **b25** The `B2.5` standalone plasma fluid solver and accompanying programs
- **eirene** The `Eirene` standalone Monte Carlo neutral transport code

nox The same as **solps** above, but without linking to any graphical libraries:

- **carre_nox** The `Carre` grid generator will not generate graphs
- **divgeo_nox** Equilibrium and template converters for `DivGeo`
- **b25eirene_nox** Only the `B2.5-Eirene` executable will be built, without graphics

triang_nox The `tria.eps` and `triageom.eps` files will not be produced

uinp_nox The `Uinp` translator from `DivGeo` output to `B2.5-Eirene` input

manual Builds this manual

libs, divgeo, amds, sonnet-light Will not be built

all_nox The same as **nox** above, but in addition:

b25_nox The `B2.5` standalone plasma fluid solver and accompanying programs

eirene_nox The `Eirene` standalone Monte Carlo neutral transport code

nox_build The same as **nox** above and used for compilation tests on the Continous Integration server:

carre_nox The `Carre` grid generator will not generate graphs

divgeo_nox Equilibrium and template converters for `DivGeo`

b25eirene_nox Only the `B2.5-Eirene` executable will be built without graphics

triang_nox The `tria.eps` and `triageom.eps` files will not be produced

uinp_nox The `Uinp` translator from `DivGeo` output to `B2.5-Eirene` input

libs, divgeo, amds, sonnet-light, manual Will not be built

carre The `Carre` grid generator and associated translators only

divgeo The `DivGeo` graphical input file builder and accompanying equilibrium and template converters only

divgeo_nox Equilibrium and template converters for `DivGeo`

b25 The `B2.5` standalone plasma fluid solver and accompanying programs only

b25_nox The `B2.5` standalone plasma fluid solver and accompanying programs, not using NCAR Graphics

b25_all The `B2.5` standalone plasma fluid solver and accompanying programs only, plus `b2xd` the **SOLPS 5.x** to **SOLPS 4.x** converter

eirene The `Eirene` standalone Monte Carlo neutral transport code only

eirene_nox The `Eirene` standalone Monte Carlo neutral transport code only, without graphics

b25eirene The `B2.5-Eirene` main code and accompanying programs only

b25eirene_all The `B2.5-Eirene` main code and accompanying programs only, plus `b2xd` the **SOLPS 5.x** to **SOLPS 4.x** converter

uinp The `Uinp` translator from `DivGeo` output to `B2.5-Eirene` input only (requires **b25eirene** previous compilation)

sonnet-light The `Sonnet` library only

triang The `Tria` and `Triageom` triangulation utilities only

nc2text_simple The `nc2text_simple` replacement program for systems where `nc2text` is not available (requires the `NetCDF` library)

nc_reduce The `nc_reduce` utility program to make large NetCDF files smaller (requires the `NetCDF` library)

b2sxdr The `b2sxdr` converter from **SOLPS 4.x** to **SOLPS-ITER**

amds The `AMDS` utility for building consistent reaction sets only

manual The **SOLPS-ITER** and `Eirene` manuals only

local Creates the local directories in `B2.5` and `Eirene` used for local code modifications

listobj Builds the list of object files for the various code components

depend Builds the dependencies files for the various code components

depend_nox Builds the dependencies files for the various code components when not linking to graphics libraries

tags Builds `etags` files for easy source code searched using `emacs`

VERSION Updates the GIT version number included in the source code and output

clean Alias to **clean_solps**

solps_nox Alias to **nox**

help Lists the available targets that can be built

echo Provides debugging information about internal `Makefile` variables

Adding the `_mpi` suffix at the end of any build target will execute the command with MPI parallelization options, when relevant, and put the executables in a separate `builds/${TOOL_CHAIN}.mpi` directory. This can also be done automatically by use of the `set_mpi` script before the compilation.

Adding the `_openmp` suffix at the end of any build target will execute the command with OpenMP parallelization options, when relevant, and put the executables in a separate `builds/${TOOL_CHAIN}.openmp` directory. This can also be done automatically by use of the `set_openmp` script before the compilation.

Adding both the `_openmp` and `_mpi` suffixes, in either order, i.e. `_openmp_mpi` or `_mpi_openmp`, at the end of any build target will execute the command with both OpenMP and MPI parallelization options, when relevant, and put the executables in a separate `builds/${TOOL_CHAIN}.openmp.mpi` directory.

Adding the `_debug` suffix at the end of any build target will execute the command in debug mode, and put the executables in a separate `builds/${TOOL_CHAIN}.debug` directory. This can also be done automatically by use of the `set_debug` script before the compilation.

Adding the `clean_` prefix at the beginning of any build target will remove that build target from its directory, as well as all associated pre-processed source files, object files, module files, dependencies and libraries.

For compilation of the `Sonnet` library, the user should first do:

```
fs sysname
```

and copy the result in `${SOLPSTOP}/modules/Sonnet-light/Makefile`.

1.3.1 Installation on local machines

The code can be compiled also on local machines for full offline usage, provided that the user attempting the installation has admin rights. Different Unix-based environments are supported, namely Linux (e.g. for Ubuntu computers) and Darwin (for Apple MacOS computers), and different CPU architectures (x86_64 and arm64). The corresponding hostnames are **LINUX** and **DARWIN**, respectively, and the used compiler is `gfortran`. The various setup and configuration files for both the main code and the submodules are already provided.
On Linux systems a Conda-based package manager needs to be installed, which is automatically done while building the various libraries (see next paragraph). On Darwin systems, instead, as a prerequirement the user will need to install the `XQuartz` package (https://www.xquartz.org/), the `Homebrew` package manager (https://brew.sh/) and the `MacPorts` package manager (https://ports.macports.org/). A python installation also needs to be present.
The various libraries required for the code to be fully operational need to be installed, before attempting to compile the code. All required libraries, for both Linux and Darwin systems, can be completely automatically downloaded and/or compiled by running the script

```
> install_dependencies
```

which will guide the user through all the required installation steps. The provided setup and configuration files already recognize the paths to the compiled libraries as automatically provided by the above script.
The compilation of B2.5 and Eirene (both standalone and coupled versions) has been successfully tested. Both serial and parallelized versions of the code (OpenMP and MPI) can be built. All components (standalone B2.5 code, standalone Eirene code, coupled B2.5-Eirene code, DivGeo, Carre, Triang and Uinp) can be successfully compiled and used.

1.4 Before every SOLPS-ITER session

At the beginning of every **SOLPS-ITER** session, the user must load up the **SOLPS-ITER** environment, which will contain the aliases and environment variables needed for smooth code operation (See also Section 2.1 for full details). **SOLPS-ITER** is expected to be run within the TC-shell. This is achieved by the command:

```
tcsh
source setup.csh
```

Among other things, these setup files establish some aliases that are useful for navigating the **SOLPS-ITER** directory structure (see details in Chapter 2). Those aliases are:

stop Goes to `${SOLPSTOP}`, the **SOLPS-ITER** top directory

sb2, sbb Go to `${SOLPSTOP}/modules/B2.5`, the `B2.5` top directory

sei Goes to `${SOLPSTOP}/modules/Eirene`, the `Eirene` top directory

ssc Goes to `${SOLPSTOP}/modules/Carre`, the `Carre` top directory

ssd Goes to `${SOLPSTOP}/modules/DivGeo`, the `DivGeo` top directory

sst Goes to `${SOLPSTOP}/modules/Triang`, the `Tria` top directory

ssu Goes to `${SOLPSTOP}/modules/Uinp`, the `Uinp` top directory

ssw Goes to `${SOLPSTOP}/modules/Sonnet`, the `Sonnet` top directory

slib Goes to `${SOLPSTOP}/lib/${HOST_NAME}.${COMPILER}`, the directory where libraries are stored

sbr Go to `${SOLPSTOP}/runs/`, the top run directory

scr Goes to `${SOLPSTOP}/scripts`, the directory where the scripts are stored

$DG The location of the `DivGeo` top directory

It is also possible to toggle between versions of the code compiled with and without debug options (array bounds checking, traceback, etc...) by use of the `set_debug` and `unset_debug` commands. Similarly, the `set_mpi` and `unset_mpi` will turn on or off, respectively, compilation with MPI parallelization flags. Same for `set_openmp` and `unset_openmp`, for OpenMP parallelization. By default, neither debug nor parallelization (MPI and OpenMP) flags are turned on.

After activating one of these flags and upon succesful compilation, it is recommended to do:

```
rehash
```

to refresh the list of executables to be found with the search path.

1.5 Examples

On the **SOLPS-ITER** web site (https://sharepoint.iter.org/departments/POP/CM/IMAS/SOLPS-ITER), in the folder *Examples*, one can find a few example cases for **SOLPS-ITER** runs, in both `B2.5` standalone or `B2.5-Eirene`

coupled modes.

There is also a dedicated directory `${SOLPSTOP}/runs/examples` which contains links to a series of sample cases. These are actually stored in the `/work/imas/shared/external/assets/solps-iter/MD5` directory hosted at the IO, and the means to retrieve these cases is detailed in the `${SOLPSTOP}/runs/examples/README.md` file. Following the instructions therein will extract an archived file in gzipped tar format containing a `baserun` and one or more run directories and associated files. In some cases, a `b2mod_dimensions.F` file is also provided, which shows the dimensions necessary to run those cases (see Appendix K.1 for full details about the contents of such files).

The cases stored for retrieval include:

test_slab_ortho_standalone An orthogonal slab test case for `B2.5` standalone

AUG_16151_D ASDEX-Upgrade D-only benchmark cases, including `B2.5` standalone, coupled and single `Eirene` call run directories

AUG_16151_D+C+He ASDEX-Upgrade multi-fluid benchmark cases, including `B2.5` standalone, coupled and single `Eirene` call run directories

AUG_28903_D+N_drifts ASDEX-Upgrade D+N case including drifts

AUG_28903_D+Ne_drifts ASDEX-Upgrade D+Ne case including drifts

DIIID_3106_DN_D+C DIII-D disconnected double-null case including Carbon chemical sputtering

ITER_2171_D+He+Be+Ne ITER full performance case with Ne injection and Be sputtering

ITER_2264_0.6Ne_100MW Another ITER full perfomance case with Ne injection

ITER_2275_Donly_20MW ITER low power case at low density (D-fuelled)

ITER_2292_Honly_20MW ITER low power case at low density (H-fuelled)

ITER_2294_Honly_20MW Same as above, but increasing density

ITER_2296_Honly_20MW Same as above, but continuing density scan

ITER_2297_Honly_20MW Same as above, continuing scan

ITER_2298_Honly_20MW Same as above, continuing scan

ITER_2299_Honly_20MW Same as above, continuing scan, reaching detachment

ITER_2308_Honly_20MW Same as above, continuing scan, detached conditions

ITER_2360_0.95He+0.05H_40MW ITER Helium plasma case at mid-power (also contains an example with metastable-resolved Helium neutrals)

ITER_2588_D+He+N ITER full power case with N injection and gas puff feedback on SOL D particle content

ITER_2588_drifts ITER `B2.5` standalone D-only demonstration case with and without drifts

ITER_535_D+He+Ar ITER full power case with Ar injection

ITER_Be-W_D+T+He+Ne ITER full power case with 98 species

JTEXT_Limiter_H J-TEXT limiter case (H-only, full drifts)

MAST_U_SFminus_Donly Lower snowflake-minus case at low power with D-only

as well as additional partial cases used in tutorials when learning how to build a case using the `DivGeo` and `SOLPS-GUI` utilities. Other cases can be added at the suggestion of users, if available.

After the cases have been downloaded to your system, make sure that the timestamps of the various input files are in the proper chronological order (using the `correct_baserun_timestamps` and `touch bf2tstati` instructions as

necessary) before attempting to launch any actual runs. Also, the links to database files are not provided within these examples, and they can be set up using the `setup_baserun_eirene_links` script.

1.6 Using a central installation

For systems with several users it is advisable to set up a central **SOLPS-ITER** installation that can be maintained by an administrator. The scripts have been set up to allow this by specifying a `${SOLPSWORK}` directory that is different from the default `${SOLPSTOP}/runs` and setting the environment variable `SOLPS_CENTRAL` to `'yes'`. Each user will then have a directory for their runs, `$SOLPSWORK`, which should contain a file named `SOLPSTOP` which contains the value for `${SOLPSTOP}`. Using an example where each user has their runs in `${HOME}/solps-runs` then the central installation could have a `SETUP/setup.csh.local` with the following lines:

```
setenv SOLPSWORK /home/$USER/solps-runs
alias sbr 'cd ${SOLPSWORK}'
echo $SOLPSTOP >! $SOLPSWORK/SOLPSTOP
if (-e $SOLPSWORK/setup.csh.local) then
  source $SOLPSWORK/setup.csh.local
endif
setenv SOLPS_CENTRAL "yes"
```

This will set `SOLPSWORK`, create the appropriate `SOLPSTOP` file, and load a `setup.csh.local` for each user if it exists. The scripts such as `dg`, `carre`, and `triang` then check if `SOLPS_CENTRAL` is set and will create the meshes, etc under `${DEVICE}/${USER}` instead of just `${DEVICE}`. It is therefore important that the `DivGeo/device`, `Carre/meshes` and `Triang/triangles` and each `${DEVICE}` subdirectory have the permissions set such that each user can write to it. This can be achieved by having a `solps` group, for example, or by providing write access for all users. Users should also have read and execute access throughout the arboresence emanating from the central installation directory. To avoid issues where the first user to create a `${DEVICE}` directory will do so without the correct permissions, an `add_central_device` script has been provided which will create the appropriate directories with write access for all users. The administrator should run this and add each `DEVICE` that will be used by the group.

Before each session, each user will then set up the **SOLPS-ITER** environment by running

```
tcsh; source $(SOLPS_central_dir)/setup.csh [optional_compiler_name]
```

from their `$SOLPSWORK` directory, and the rest of the usage would be standard, where `$(SOLPS_central_dir)` is the location of the central installation. Please be aware that the capability to source the `setup.csh` file from elsewhere than the directory where it is located may not be supported by all systems.

Chapter 2

The structure

The **SOLPS-ITER** directory has the following global structure:

```
SOLPSTOP
+---doc
|   +---SOLPS_2002_Course
|   +---solps
+---scripts
|   +---Balance
|   +---commands
|   +---matlab
|   +---MatlabPostProcessing
|   +---Matlab_SPb
|   +---nc2text_simple
|   +---palettes
|   +---ITER.ifort64
+---scripts.local
+---modules
|   +---Sonnet-light
|   +---fxdr
|   +---Triang
|   +---Uinp
|   +---Carre
|   +---amds
|   +---adas
|   +---B2.5
|   +---solps4-5
|   +---DivGeo
|   +---Eirene
+---lib
|   +---ITER.ifort64
+---runs
+---SETUP
```

In order to set up the appropriate environment variables and aliases, the command

```
source setup.csh
```

should be executed in the top level directory.

Three important environment variables are `SOLPSTOP` which points to the top level directory, `HOST_NAME` which

tells **SOLPS-ITER** in which environment it is running and COMPILER which identifies the chosen Fortran compiler (*e.g.* ifort64).

2.1 Setup and configuration files

The setup.csh file and its subsidiaries ${SOLPSTOP}/SETUP/setup.csh.${HOST_NAME}.${COMPILER}, ${SOLPSTOP}/SETUP/config.common.${COMPILER}, and ${SOLPSTOP}/SETUP/config.${HOST_NAME}.${COMPILER} are used to load any modules packages useful for the local **SOLPS-ITER** run environment and the following environment variables:

1. B25_DEFINES, an environment variable containing the additional "defines" to use during compilation of B2.5
2. DBX, the name of the debugger to be used
3. DEVICE the name of the fusion device most likely to be simulated for this HOST_NAME (see Chapter 7 for available choices)
4. EIRENE_DEFINES, an environment variable containing the additional "defines" to use during compilation of Eirene
5. FFLAGSEXTRA, for providing system-wide Fortran compilation flags
6. GKS_ROOT, the root of the GKS tree, and LD_GKS, the appropriate loader instruction to load the GKS library
7. GLI_HOME, the location of the gksfont.dat file (this definition must appear in both the setup.csh.*.* and config.*.* files as it is used both during compilation and at runtime)
8. GR_ROOT, the root of the GR tree, and LD_GR, the appropriate loader instruction to load the GR library (a graphics package produced by FZ-Jülich and used by Eirene)
9. H5DIR, the root directory for the HDF5 utility library
10. LD_CATALYST, the appropriate loader instruction to load the Paraview Catalyst utility
11. LD_MSCL, the appropriate loader instruction to load the MSCL library
12. LD_NETCDF, the appropriate loader instruction to load the NetCDF library
13. LD_SONNET, the appropriate loader instruction to load the Sonnet-light library
14. LD_UAL, the appropriate loader instruction to load the Integrated Modelling and Analysis Suite (IMAS) Universal Access Layer (UAL) library
15. MDSPLUS_DIR, the root of the MDSplus tree, and LD_MDSPLUS, the appropriate loader instruction to load MDSplus
16. MSCL_ROOT, the root of the mscl tree, and LD_MSCL, the appropriate loader instruction to load the mscl library
17. NAGDIR, the root of the NAG tree, and LD_NAG, the appropriate loader instruction to load the NAG library
18. NCARG_ROOT, the root of the NCAR tree, and LD_NCARG, the appropriate loader instruction to load NCAR (It is best if the definition of NCARG_ROOT appears in both the setup.csh.*.* and config.*.* files as some implementations of NCAR use it both during compilation and at runtime)
19. NCDIR, the root of the NetCDF tree, and LD_NETCDF, the appropriate loader instruction to load NetCDF
20. PERFMON, an environment variable used by the compiler to know whether to enable the performance monitoring libraries
21. SOLPS_CPP, an environment variable containing the global "defines" to use during compilation of SOLPS code components

22. some aliases to move around the tree (`sbb`, `sei`, *etc.*)
23. some aliases for plotting (`xyplot`, etc...)
24. two aliases to swap between the debug and no_debug versions of the code (`set_debug` and `unset_debug`)
25. two aliases to swap between the MPI and no_MPI versions of the code (`set_mpi` and `unset_mpi`)
26. two aliases to swap between the OpenMP and no_OpenMP versions of the code (`set_openmp` and `unset_openmp`)

Of course, any additional desired definitions can be added. The preferred place for those is, for those variables that need not be shared by other users, the `${SOLPSTOP}/SETUP/setup.csh.${HOST_NAME}.${COMPILER}.local` and `${SOLPSTOP}/SETUP/config.${HOST_NAME}.${COMPILER}.local` files. Variables that are needed at compilation time need to appear in the `config.*.*(.local)` file, while variables needed at runtime must appear in `setup.csh.*.*(.local)` file.

Moreover, the **SOLPS-ITER** GIT server holds several examples in the `${SOLPSTOP}/SETUP/` directory.

The command

```
stop
```

will take you to `${SOLPSTOP}`.

2.2 modules

```
+---modules
|   +---Sonnet-light
|   +---fxdr
|   +---Triang
|   +---Uinp
|   +---Carre
|   +---adas
|   +---amds
|   +---B2.5
|   +---solps4-5
|   +---DivGeo
|   +---Eirene
```

The `modules/B2.5` subtree contains the source directories for `B2.5`, `modules/Eirene` for the Monte Carlo code, `modules/Carre` and `modules/fxdr` the grid generator and associated converter, `modules/DivGeo` for the `DG` graphical interface to the grid preprocessor, `modules/Triang` for the `Eirene` triangular mesh generator, `modules/Uinp` for the `Uinp` input file generator, `modules/adas` for the ADAS atomic data files, `modules/amds` for the automated generator of atomic physics blocks for `Eirene`, `modules/solps4-5` for the `b2sxdr` converter from **SOLPS 4.x** to **SOLPS-ITER**, and `modules/Sonnet-light` for the `Sonnet` library in use for **SOLPS 4.x** grids.

2.2.1 modules/B2.5

The different versions of `B2.5` are shown below:

```
SOLPSTOP
+---modules
|   +---B2.5
|   |   +---config
|   |   +---src
|   |   +---src.local (optional)
|   |   +---builds
```

config contains compiler option files

src.local is an optional location which can contain locally modified source files (not version-controlled)

src contains the `B2.5` source code files

builds contains the compiled object files for building the `B2.5` executables.

A more varied selection of `B2.5` historical versions is available within the **SOLPS5.0** distribution.

The commands

```
ssb
```

or

```
sb2
```

will take you to `${SOLPSTOP}/modules/B2.5`.

Local modifications to the B2.5 source

With the installation instructions from Section 1.1, users get a local clone of the `B2.5` GIT submodule, and **this is the only supported option!** The files in `src` will be version-controlled and updated whenever the user will pull in new code version. If, however, one wishes to make local changes to a file and protect those from GIT updates, the procedure for that purpose is to copy the file you want to change to `src.local` and then edit that file. You will then need to do

```
sbb
gmake local
```

to create the `src.local` directory. Once you have copied and edited the file(s) you need to change in `src.local`, you should recompile with:

```
stop
gmake listobj
gmake tags
gmake depend
gmake
```

so that the local file is used. The last `gmake` command may of course be replaced by `gmake solps(_debug)`, `gmake solps_mpi(_debug)`, `gmake all(_debug)` or `gmake all_mpi(_debug)` as needed.

If this local change should no longer be needed, then it should be removed from `src.local` and the following performed:

```
stop
gmake clean
gmake listobj
```

```
gmake tags
gmake depend
gmake
```

If you want to avoid a long recompilation, you need to remove from the `builds/*/` directories, the object and module files corresponding to your locally changed file(s), and then ask for a normal recompilation with a single `gmake` command.

If the user wishes to modify the compiling options used, the creation of a `config.${HOST_NAME}.${COMPILER}.local` file in the `config` subdirectory is recommended. Since the `Makefile` reads those after the `config.${HOST_NAME}.${COMPILER}` and `config.common.${COMPILER}` files, they need only contain the modified definitions.

It is also possible to add additional dependencies by means of a `config/dependencies.local` file, which should then contain lines with the format:

```
${OBJDIR}/<calling_program>.o: ${OBJDIR}/<called_program>.o
```

This can be particularly useful when compiling with options such as `warn interfaces` to make sure that the files containing the called routines are compiled before the files containing the calling routines.

If you just want to compile `b2mn.exe`, you can do

```
gmake MAIN
```

or to compile all but the graphics programs

```
gmake NOPLOT
```

To compile everything, including the converters from **SOLPS 5.x** to and from **SOLPS 4.x** (which might not function as they require a complicated combination of options), you can

```
gmake ALL
```

The default compiles all but the **SOLPS 5.x** to **SOLPS 4.x** converters.

If you wish to submit code changes for a pull request so that they may be inserted in the general **SOLPS-ITER** distribution or to have them version-controlled within your own topical branches, the locally modified files need to remain in the `src` version-controlled directories.

modules/B2.5/src

Within the `modules/B2.5/src` source directory, we have:

```
SOLPSTOP
+---modules
|   +---B2.5
|   |   +---src.local
|   |   +---src
|   |   |   +---b2aux
|   |   |   +---solvers
|   |   |   +---sources
|   |   |   +---include
|   |   |   +---include.local (optional)
|   |   |   +---documentation
|   |   |   +---postprocessing
|   |   |   +---user
|   |   |   +---driver
|   |   |   +---utility
|   |   |   +---output
|   |   |   +---input
|   |   |   +---preprocessing
|   |   |   +---b2plot
|   |   |   +---modules
|   |   |   +---modules.local (optional)
|   |   |   +---convert
|   |   |   +---transport
|   |   |   +---common
|   |   |   +---equations
|   |   |   +---catalyst
|   |   |   +---ids
|   |   |   +---mpi_dummy
|   |   +---Database
|   |   |   +---Bulk_properties
|   |   |   +---Surface_properties
|   |   |   +---trim.data
|   |   |   |   +---range.data
|   |   |   |   +---refl.data
|   |   |   |   +---sputter.data
|   |   +---builds
```

Similarly to the `src.local` directory, two additional local directories are provided `src/include.local` and `src/modules.local` for locally modified include files and modules, respectively, that the user does not wish to have updated by the version control software. The same compilation instructions as in previous Section 2.2.1 apply.

The user wanting to delve deeper into the source code is directed to the detailed routine listing in Appendix A.2.

modules/B2.5/Database

The `Database` directory contain the data files which `B2.5` relies on to compute things such as reaction rates, wall element surface temperatures, etc...

These data files include:

ionization_potentials is a `B2` and `B2.5` input file containing ionisation potentials

incatm is a `B2.5` input file containing data for atomic mass, surface binding energy and density of all elements

ratadas.filelist is a `B2.5` input file containing information used to index into the ADAS files when the `'adas'` option is used in `b2ar` (see section 3.3)

stra.dat is a file containing `STRAHL` atomic data results for use by `b2ar` when building `b2frates` files where the `'strahl'` data option is chosen. Must be copied to the local `baserun` directory.

weis.dat is a file containing atomic data results for hydrogen by Weisheit *et al.* for use by `b2ar` when building `b2frates` files. Must be copied to the local `baserun` directory.

modules/B2.5/Database/Bulk_properties

Contains data for bulk properties of different target materials. A sample file to be filled by the user for additional materials is contained in `blank`.

modules/B2.5/Database/Surface_properties

Contains data for surface properties of different target materials. A sample file to be filled by the user for additional materials is contained in `blank`.

modules/B2.5/Database/trim.data

Subdivided into `range.data`, `refl.data` and `sputter.data` directories. They contain the `TRIM` range, reflection and sputtering data used by `B2.5`. Files ending in `e` are for energy, in `n` are for particles.

2.2.2 modules/Eirene

```
SOLPSTOP
+---modules
|   +---Eirene
|   |   +---builds
|   |   +---config
|   |   +---Database
|   |   |   +---AMdata
|   |   |   +---SurfaceData
|   |   |   |   +---TRIM
|   |   |   |   |   +---10_bins
|   |   +---Manual
|   |   |   +---figures
|   |   |   +---tex
|   |   +---scripts
|   |   +---src.local
|   |   +---src
|   |   |   +---assistant
|   |   |   +---broadcast
|   |   |   +---broadcast_dummy
|   |   |   +---cmake
|   |   |   +---diagno
|   |   |   +---file-handling
|   |   |   +---geometry
|   |   |   |   +---time-routines
|   |   |   +---grsoft-dummy
|   |   |   +---interfaces
|   |   |   |   +---couple_B2
|   |   |   |   +---couple_B2.5
|   |   |   |   +---couple_Dummy
|   |   |   |   +---couple_SOLPS-ITER
|   |   |   |   +---couple_Tria
|   |   |   +---iterate
|   |   |   +---main-routines
|   |   |   +---mathematics
|   |   |   +---modules
|   |   |   +---output
|   |   |   +---particle-tracing
|   |   |   +---photons
|   |   |   +---plot_dummy
|   |   |   +---plotting
|   |   |   +---sampling
|   |   |   +---scoring
|   |   |   +---startup-routines
|   |   |   +---surface-processes
|   |   |   +---tetrahedra
|   |   |   +---user-routines
|   |   |   |   +---user_default
|   |   |   |   +---user_EDGE2D
|   |   |   |   +---user_generalized_interface
|   |   |   |   +---user_iter
|   |   |   |   +---user_solps4.3
```

```
|   |   |   |   +---user_solps5.0
|   |   |   +---volume-processes
```

The command

```
sei
```

will take you to `${SOLPSTOP}/modules/Eirene`.

Because `Eirene` is distributed as a standalone submodule and is being maintained by FZ-Jülich, some of the directories found above correspond to `Eirene` implementations other than that used in **SOLPS-ITER**. These are the subdirectories within `interfaces` and `user-routines`. **SOLPS-ITER** uses only the code found in the `interfaces/couple_SOLPS-ITER` and `user-routines/user_iter` subdirectories. The other are included for compatibility with the FZ-Jülich master `Eirene` repository.

Additionally, the `Database` directory contain the data files which `Eirene` relies on to compute things such as backscattering coefficients, atomic reactions, etc...

As with `B2.5`, a `local` directory for the user that wishes to make code changes sheltered from GIT updates. The same procedures apply.

If routines have been added or removed, if the dependencies have been altered or if this is the first compilation in a new tree, and you are not compiling using the **SOLPS-ITER** master `Makefile`, you should do

```
gmake depend
gmake listobj
gmake tags
gmake depend
gmake
```

The first `gmake depend` is only needed for a brand new compilation. The `gmake listobj` command is needed for a first-time compilation or if a new source file has been added. The `gmake tags` command is needed if this is the first compilation or new files are added or a new routine has been added within an existing source file. The second `gmake depend` is needed if this is the first compilation or if the dependencies have been altered.

If the user wishes to modify the compiling options used, the creation of a `config.${HOST_NAME}.${COMPILER}.local` file in the `config` subdirectory is recommended. Since the `Makefile` reads those after the `config.${HOST_NAME}.${COMPILER}` and `config.common.${COMPILER}` files, they need only contain the modified definitions.

If you are having problems (you have tried to be clever with altering files, or the `SOLPS` administrator has), you might need to do

```
gmake clean
rm builds/*/LISTOBJ
gmake depend
gmake listobj
gmake tags
gmake depend
gmake
```

modules/Eirene/Database

Data files needed by `Eirene` are stroed in this directory. The most important of these data files are:

AMJUEL is an `Eirene` input file containing atomic physics rates

H2VIBR is an `Eirene` input file containing atomic physics rates for the vibrational states of hydrogen molecules

HYDHEL is an `Eirene` input file containing atomic physics rates for hydrogen and helium

METHANE is an `Eirene` input file containing atomic physics rate for methane

SPUTER is an `Eirene` input file containing sputtering data

TRIM is an `Eirene` input file containing sputtering data

modules/Eirene/Database/Surfacedata/TRIM

Contains `TRIM` data used by `Eirene`, pointed to by a `cpath` instruction in the `Eirene` input file.

This directory is now located in `modules/Eirene/Database/Surfacedata`.

2.2.3 modules/amds

Allows for the provision of chemistry set files to be used by `Eirene`, through selection in `DG` and pre-processing by `Uinp`. See the dedicated manual `${SOLPSTOP}/modules/amds/doc/manual.pdf` for full details.

The `AMDS` program can be called by use of the `amds` script.

Several `AMDS` files are provided by default as part of the **SOLPS-ITER** distribution. Users are encouraged to browse through those files to see which ones could fit their needs on a case-by-case basis. Of particular note is the `Uinp_defaults.amd` file, which provides the same set of reactions as the `Uinp` default recommended set.

2.2.4 modules/adas

The `modules/adas` subdirectory contains files which are used to tell various programs which `ADAS` files to use.

`source.location` indicates which class of lines to use (`pjr` currently) and the name of the other index file (`ratadas.filelist` currently).

`ratadas.filelist` indicates the global source of the `ADAS` files, and which ”years” to use.

2.3 lib

The `lib` subtree contains libraries used during compilation.

```
SOLPSTOP
+---lib
|   +---ITER.ifort64
```

Since most of the libraries are now expected to be provided as modules, only the `Sonnet` library, `libsonnet.a`, needs to be stored here. However, local copies of the required libraries can also be put there at the user's convenience.

The command

```
slib
```

will take you to `${SOLPSTOP}/lib/${HOST_NAME}.${COMPILER}`.

2.4 runs

The runs for **SOLPS-ITER** are done in `runs`.

```
SOLPSTOP
+---runs
```

The command

```
sbr
```

will take you to `${SOLPSTOP}/runs`.

Usually a "case series" would be run in a sub-directory of `${SOLPSTOP}/runs`. Sometimes, further subdirectories are created to separate categories of runs — the depth of sub-directories is a matter of taste! In each of the directories, a `baserun` sub-directory is created, and then, at the same level, run directories. The `baserun` directory contains the defaults for all of the other directories at its same level, and typically the grid generation steps and the input build programs `b2ag`, `b2ah`, `b2ar` and `b2ai` are run in this directory (see chapter 3).

2.5 scripts

The command

```
scr
```

will take you to `${SOLPSTOP}/scripts`.

The number of scripts developed for various **SOLPS** versions is very large. Efforts have been made to translate all the relevant ones for use within **SOLPS-ITER**. Any of these scripts can be copied to the `${SOLPSTOP}/scripts.local` directory for local user modifications as desired. There also are `${HOST_NAME}.${COMPILER}` subdirectories where compiler-dependent script programs will reside. A non-exhaustive listing of these scripts, with some explanation as to their purpose, follows:

- Job submission scripts

  b2plot Shortcut to **b2plot**. A `-n` option allows for dry-run tests. A `-d` option allows to run within the chosen debugger. A `-s` option allows to run **b2plot** without coupling to `Eirene` data. A `-h` option prints a help description.

  b2run Shortcut to the `Makefile` launching `B2.5`. A `-n` option allows for dry-run tests. A `-d` option allows to run within the chosen debugger. A `-s` option allows to run `B2.5` in standalone mode. A `-m` option allows to run with MPI options. A `-o` option adds any other desired run options. The options can be grouped together inside quotes or given individually with multiple `-o` instances. A `-t #` option allows to request an OpenMP run with # threads. A `-b` option can be used when invoking the `b2co` program to obtain a provisional `b2fstate` file while the main code is running. A `-h` option prints a help description. (See 3 for fuller details)

  b2run_hgw_cluster Shortcut to the `Makefile` launching `B2.5` for the IPP–Greifswald cluster. A `-n` option allows for dry-run tests.

  b2sxdr Shortcut to the **SOLPS 4.x** to **SOLPS 5.x** converter. A `-n` option allows for dry-run tests. A `-d` option allows to run within the chosen debugger. A `-s` option allows to specify that the run to convert was not coupled to `Eirene`. A `-o` option adds any other desired run options. The options can be grouped together inside quotes or given individually with multiple `-o` instances. A `-h` option prints a help description.

  convert_from_solps4.3 #1 Converts run number #1 from `DEVICE` from the SOLPS4.3 ITER archive to a **SOLPS-ITER** run in directory `${SOLPSTOP}/runs/${DEVICE}/#1`.

  csd3submit Command file to launch runs on the CCFE CSD3 cluster. See Section 3 for details.

  cumulussubmit Command file to launch runs on the CCFE Cumulus cluster. See Section 3 for details.

  darwinsubmit Command file to launch runs on MacOS devices (DarwinOS). See Section 3 for details.

  efgwmake Command file to launch a batch compilation script on the EFGW machine. A `-o` option allows to pass Makefile options. Other options are as per the submission scripts. The list of objects to be compiled should form the end of the list of arguments.

  eirobj Alias to run `Eirene` standalone with graphics.

  eirobjx Alias to run `Eirene` standalone without graphics.

  ippsubmit Command file to launch runs on the new (2019) IPP cluster. See Section 3 for details.

  ITER_divertor_HTM Performs the thermal calculation of the divertor target monoblocks and reflector plate hypervapotron channels. It requires `b2ymb.dat` and `b2yrp.dat` to be present in the run directory. It runs **b2plot** to produce the heat load profile data of the targets and reflector plates. Then it runs `b2ymb` to produce the input files (see 3.18) for the `FreeFem++`[4] heat transfer model of the monoblock `Monoblock_HTM.edp` for the outer target. Afterwards, `b2ymb` is run again to produce the input files to run `Monoblock_HTM.edp` for the inner target. When the inner target is completed,

`b2yrp` (see 3.19) is run to produce the input files for the inner reflector plate heat transfer model `ReflectorPlate_HTM.edp`. Finally, `b2yrp` is executed one last time to produce the input files for the thermal analysis of the outer reflector plate.

itermake Command file to launch a batch compilation script on the ITER SDCC cluster. A `-o` option allows to pass Makefile options. Other options are as per the submission scripts. The list of objects to be compiled should form the end of the list of arguments.

itersubmit Command file to launch runs on the ITER SDCC cluster. See Section 3 for details.

itmsubmit Command file to launch runs on the EFGW machine. See Section 3 for details.

jetsubmit Command file to launch runs on the JET Analysis Cluster (JAC). See Section 3 for details.

jfrssubmit Command file to launch runs on the JFRS IFERC-CSC supercomputer. See Section 3 for details.

localsubmit Command file to launch runs on the local machine. See Section 3 for details.

marconisubmit Command file to launch runs on the Marconi supercomputer. See Section 3 for details.

mpcdfsubmit Command file to launch runs on the MPCDF clusters Raven and Viper. See Section 3 for details.

pitagorasubmit Command file to launch runs on the Pitagora supercomputer. See Section 3 for details.

resave_mds Update an archived `MDSplus` case. A `-n` option allows for dry-run tests. A `-d` option allows to run within the chosen debugger. A `-s` option allows to update a `B2.5` standalone case. A `-r` option allows to pass variable redefinitions to the underlying Makefile.

save_mds Archive an `MDSplus` case. A `-n` option allows for dry-run tests. A `-d` option allows to run within the chosen debugger. A `-s` option allows to archive a `B2.5` standalone case. A `-r` option allows to pass variable redefinitions to the underlying Makefile.

sgesubmit Command file to launch runs on an SGE system. See Section 3 for details.

ustcsubmit Command file to launch runs on the USTC computer cluster. See Section 3 for details.

- The residuals can be plotted with

resall_D all residuals for a D-only case. A python version of this script also exists (Python 2.7 and 3.5 compatible).

resall_D+_only all residuals for a D^+-only case

resall_D+D all residuals for test cases where we treat two separate D sequences

resall_D+T all residuals for a $D + T$ case

resall_H all residuals for a H only case

resrest residuals for non-species dependent equations

res_D+C non-species dependent residuals for a $D + C$ case

res_D+O non-species dependent residuals for a $D + O$ case

res_D+C+He non-species dependent residuals for a $D + C + He$ case

res_D+C+Ar+Ne+He non-species dependent residuals for a $D + C + Ar + Ne + He$ case

resco continuity residuals

resco_reg continuity residuals per region

resco_reg_region #1 continuity residuals per region for species #1 (default 0)

resco_D+C continuity residuals for a $D + C$ case

resco_D+O continuity residuals for a $D + O$ case

resco_D+C+He continuity residuals for a $D + C + He$ case

resmo parallel momentum residuals per species

resmo_reg #1 parallel momentum residuals per species for region #1 (default 0)

resmo_reg_region #1 parallel momentum residuals per region for species #1 (default 0)

resmo_D+C parallel momentum residuals for a $D + C$ case

resmo_D+O parallel momentum residuals for a $D + O$ case

resmo_D+C+He parallel momentum residuals for a $D + C + He$ case

resmt_reg residuals of the total parallel momentum equation

reshe_reg residuals of the electron heat equation per region

reshi_reg residuals of the ion heat equation per region

respo residuals of the potential equation, including the internal iterations

respo_num the number of internal potential equation iterations

respo_reg residuals of the potential equation per region

resopt Internal script to apply options to above residuals scripts (see below)

resmod #1 Selects residuals *modulo* #1 (default all)

all of the above can take `-m #` to plot the residuals *modulo* `#` occurrences and/or `-l #` to plot the last `#` residuals and/or `-a #` to plot the average residual over `#` occurrences. The options are applied in that order independently of the order in which they were typed.

- Power/particle losses (see section I and Appendix 3.5.4 for the region definitions)

b2stbc_sna_reg #1 Boundary particle sources for region #1 (default 0)

b2stbr_sna_reg #1 Recycling particle sources for region #1 (default 0)

b2stbm_sna_reg #1 Additional particle sources for region #1 (default 0)

b2sext_sna_reg #1 External particle sources for region #1 (default 0)

b2stbc_smo_reg #1 Boundary parallel momentum sources for region #1 (default 0)

b2stbr_smo_reg #1 Recycling parallel momentum sources for region #1 (default 0)

b2stbm_smo_reg #1 Additional parallel momentum sources for region #1 (default 0)

b2sext_smo_reg #1 External parallel momentum sources for region #1 (default 0)

rcxhireg #1 charge exchange energy for region #1 (default 0)

rcxmoreg #1 charge exchange parallel momentum for region #1 (default 0)

rcxnareg #1 charge exchange rate for region #1 (default 0)

rdneureg #1 radiation rate from `Eirene` neutrals for region #1 (default 0)

rqahereg #1 electron heat loss for region #1 (default 0)

rqbrmreg #1 bremsstrahlung losses for region #1 (default 0)

rqradreg #1 total radiation rate for region #1 (default 0)

rrahireg #1 recombination energy for region #1 (default 0)

rramoreg #1 recombination parallel momentum for region #1 (default 0)

rranareg #1 recombination rate for region #1 (default 0)

rsahireg #1 ionisation energy for region #1 (default 0)

rsamoreg #1 ionisation parallel momentum for region #1 (default 0)

rsanareg #1 ionisation rate for region #1 (default 0)

b2stbc_she_reg_region #1 Boundary electron heat sources, by region, for species #1 (default 0)

b2stbc_shi_reg_region #1 Boundary ion heat sources, by region, for species #1 (default 0)

b2stbc_sne_reg_region #1 Boundary electron particle sources, by region, for species #1 (default 0)

b2stbc_sch_reg_region #1 Boundary charge sources, by region, for species #1 (default 0)

b2stbc_sna_reg_region #1 Boundary particle sources, by region, for species #1 (default 0)

b2stbc_smo_reg_region #1 Boundary parallel momentum sources, by region, for species #1 (default 0)

b2stbr_she_reg_region #1 Recycling electron heat sources, by region, for species #1 (default 0)

b2stbr_shi_reg_region #1 Recycling ion heat sources, by region, for species #1 (default 0)

b2stbr_sne_reg_region #1 Recycling electron particle sources, by region, for species #1 (default 0)

b2stbr_sch_reg_region #1 Recycling charge sources, by region, for species #1 (default 0)

b2stbr_sna_reg_region #1 Recycling particle sources, by region, for species #1 (default 0)

b2stbr_smo_reg_region #1 Recycling parallel momentum sources, by region, for species #1 (default 0)

b2stbm_she_reg_region #1 Additional electron heat sources, by region, for species #1 (default 0)

b2stbm_shi_reg_region #1 Additional ion heat sources, by region, for species #1 (default 0)

b2stbm_sne_reg_region #1 Additional electron particle sources, by region, for species #1 (default 0)

b2stbm_sch_reg_region #1 Additional charge sources, by region, for species #1 (default 0)

b2stbm_sna_reg_region #1 Additional particle sources, by region, for species #1 (default 0)

b2stbm_smo_reg_region #1 Additional parallel momentum sources, by region, for species #1 (default 0)

b2sext_she_reg_region #1 External electron heat sources, by region, for species #1 (default 0)

b2sext_shi_reg_region #1 External ion heat sources, by region, for species #1 (default 0)

b2sext_sne_reg_region #1 External electron particle sources, by region, for species #1 (default 0)

b2sext_sch_reg_region #1 External charge sources, by region, for species #1 (default 0)

b2sext_sna_reg_region #1 External particle sources, by region, for species #1 (default 0)

b2sext_smo_reg_region #1 External parallel momentum sources, by region, for species #1 (default 0)

rcxhireg_region #1 charge exchange energy, by region, for species #1 (default 0)

rcxmoreg_region #1 charge exchange parallel momentum, by region, for species #1 (default 0)

rcxnareg_region #1 charge exchange rate, by region, for species #1 (default 0)

rqahereg_region #1 electron heat loss, by region, for species #1 (default 0)

rqbrmreg_region #1 bremsstrahlung radiation loss, by region, for species #1 (default 0)

rqradreg_region #1 total radiation rate, by region, for species #1 (default 0)

rrahireg_region #1 recombination energy, by region, for species #1 (default 0)

rramoreg_region #1 recombination parallel momentum, by region, for species #1 (default 0)

rranareg_region #1 recombination rate, by region, for species #1 (default 0)

rsahireg_region #1 ionisation energy, by region, for species #1 (default 0)

rsamoreg_region #1 ionisation parallel momentum, by region, for species #1 (default 0)

rsanareg_region #1 ionisation rate, by region, for species #1 (default 0)

these require the SOLPS_FLUIDS and NREG variables to be set correctly (see section 3.5.4 and Appendix I for the definitions of the regions). If the SOLPS_FLUIDS or NREG files are not found, the necessary information is extracted from the run.log file.

- Plotting

plot The various xyplot#, lxyplot#, xlyplot# and lxlyplot# call plot passing their name as the 1^{st} parameter; plot parses the 1^{st} parameter to determine if $\log x$ (lx) or $\log y$ (ly) and the number of curves (#); args 1..# specify labels for curves; the next two arguments specify x- and y-axis labels, the last specifies a header. The system normally uses gnuplot.cmd for the gnuplot commands, and gnuplot.data for the data. A number of environment variables can be used to modify the behaviour of plot:

GNUPLOT_BATCH used at the start of the process of specifying a batch of plots that should all be processed together into one PostScript file. The basic structure for using this feature would be of the following form:

```
setenv GNUPLOT_BATCH yes
setenv GNUPLOT_POSTSCRIPT_OPTIONS 'color solid'
setenv PRINT_PLOT b2time2d.ps
rm -f GNUPLOT_COUNT gnuplot.cmd
echo 0 > GNUPLOT_COUNT

2dt nesepm
2dt tesepm tisepm
2dt nesepi nemxip nesepa nemxap
...
2dt tmne
2dt tmte tmti

gnuplot gnuplot.cmd
rm GNUPLOT_COUNT gnuplot.*data
```

where the 2dt could be any series of commands that use plot for their graphics. See the script 2d_plots (2.5 on page 52) for an example

GNUPLOT_CMD changes the command file used from gnuplot.cmd to the filename specified by this environment variable

GNUPLOT_COUNT internal environment variable used for batch plots

GNUPLOT_DATA changes the data file used from gnuplot.data to the filename specified by this environment variable

GNUPLOT_DELAY allows the user to change the delay set at the bottom of gnuplot.cmd

GNUPLOT_EXTRA adds the contents of the file whose name is stored in GNUPLOT_EXTRA to the gnuplot.cmd file

GNUPLOT_LINEWIDTH changes the line width being used

GNUPLOT_MAX_POINTS sets the maximum number of points that will be plotted; if more points are present, the data will be averaged so that the limit will not be exceeded

GNUPLOT_MAX_POINTS_FACTOR integer factor by which the number of points is reduced to be below that set by `GNUPLOT_MAX_POINTS` (defaults to 2)

GNUPLOT_NOTIMESTAMP removes the timestamp from the plots

GNUPLOT_POINTSIZE changes the size of points plotted

GNUPLOT_POSTSCRIPT_OPTIONS `gnuplot` options for the PostScript driver

GNUPLOT_TERM provides the "term" information for `gnuplot`

GNUPLOT_TMP causes the `gnuplot.cmd`, `gnuplot.data` and other temporary files to be written to the directory specified by this environment variable. Works for most scripts. Also moves the location of the `GNUPLOT_EXTRA` and `GNUPLOT_COUNT` files to being located in `GNUPLOT_TMP`.

PLOTPOINTS just plot points

PRINT_PLOT if set to a filename, then PostScript is written to that file; else if set but empty, PostScript is sent to the default printer; otherwise the plot is done to the X terminal

If the file `GNUPLOT_COUNT` exists, then an internal counter `GNUPLOT_COUNT` is set by reading the number from the file; incremented; and then saved to the file; instead of using `gnuplot.data` for the data, the file `gnuplot.${GNUPLOT_COUNT}.data` is used and the commands for plotting are concatenated to `gnuplot.cmd` allowing for multiple pages of plots.

plot_cps #1 Copies `gnuplot.cmd` to `gnuplot_cps.cmd`, modifying it so that colour plotting occurs to #1 or by default `out.ps`; executes `gnuplot` on new file. The location of these files can be modified by use of the `GNUPLOT_TMP` variable.

plot_it Utility script used to produce complex plots.

plot_mesh #1 #2 Plots the mesh specified as #1 accompanied by the `mesh.extra` file specified as #2 as #1.ps.

plot_png produces a PNG output file instead

plot_ps #1 Copies `gnuplot.cmd` to `gnuplot_ps.cmd`, modifying it so that black-and-white plotting occurs to #1 or by default `out.ps`; executes `gnuplot` on new file. The location of these files can be modified by use of the `GNUPLOT_TMP` variable.

plot_rzpsi.dat Utility script using `plotmtv` to obtain a plot of the equilibrium data from the `rzpsi.dat` file used by `Carre` to build the computational grid.

plot_trouble Plots `Eirene` problematic trajectories over the `B2.5` mesh and `mesh.extra`.

plot_set Utility script preparing files for complex plots.

xyplot alias for simple "$x\ y$" plot

xyplot2 alias for simple "$x\ y_1\ y_2$" plot

xyplot3 alias for simple "$x\ y_1\ y_2\ y_3$" plot

xyplot4 alias for simple "$x\ y_1\ y_2\ y_3\ y_4$" plot

xyplot5 alias for simple "$x\ y_1\ y_2\ y_3\ y_4\ y_5$" plot

xyplot6 alias for simple "$x\ y_1\ y_2\ y_3\ y_4\ y_5\ y_6$" plot

xyplot7 alias for simple "$x\ y_1\ y_2\ y_3\ y_4\ y_5\ y_6\ y_7$" plot

xyplot8 alias for simple "$x\ y_1\ y_2\ y_3\ y_4\ y_5\ y_6\ y_7\ y_8$" plot

xyplot9 alias for simple "$x\ y_1\ y_2\ y_3\ y_4\ y_5\ y_6\ y_7\ y_8\ y_9$" plot

xlyplot alias for simple "$x\ \log(y)$" plot

xlyplot2 alias for simple "$x\ \log(y_1)\ \log(y_2)$" plot

xlyplot3 alias for simple "$x\ \log(y_1)\ \log(y_2)\ \log(y_3)$" plot

xlyplot4 alias for simple "$x\ \log(y_1)\ \log(y_2)\ \log(y_3)\ \log(y_4)$" plot

xlyplot5 alias for simple "$x\ \log(y_1)\ \log(y_2)\ \log(y_3)\ \log(y_4)\ \log(y_5)$" plot

xlyplot6 alias for simple "$x\ \log(y_1)\ \log(y_2)\ \log(y_3)\ \log(y_4)\ \log(y_5)\ \log(y_6)$" plot

xlyplot7 alias for simple "$x\ \log(y_1)\ \log(y_2)\ \log(y_3)\ \log(y_4)\ \log(y_5)\ \log(y_6)\ \log(y_7)$" plot

xlyplot8 alias for simple "$x\ \log(y_1)\ \log(y_2)\ \log(y_3)\ \log(y_4)\ \log(y_5)\ \log(y_6)\ \log(y_7)\ \log(y_8)$" plot

xlyplot9 alias for simple "$x\ \log(y_1)\ \log(y_2)\ \log(y_3)\ \log(y_4)\ \log(y_5)\ \log(y_6)\ \log(y_7)\ \log(y_8)\ \log(y_9)$" plot

lxyplot alias for simple "$\log(x)\ y$" plot

lxyplot2 alias for simple "$\log(x)\ y_1\ y_2$" plot

lxyplot3 alias for simple "$\log(x)\ y_1\ y_2\ y_3$" plot

lxyplot4 alias for simple "$\log(x)\ y_1\ y_2\ y_3\ y_4$" plot

lxyplot5 alias for simple "$\log(x)\ y_1\ y_2\ y_3\ y_4\ y_5$" plot

lxyplot6 alias for simple "$\log(x)\ y_1\ y_2\ y_3\ y_4\ y_5\ y_6$" plot

lxyplot7 alias for simple "$\log(x)\ y_1\ y_2\ y_3\ y_4\ y_5\ y_6\ y_7$" plot

lxyplot8 alias for simple "$\log(x)\ y_1\ y_2\ y_3\ y_4\ y_5\ y_6\ y_7\ y_8$" plot

lxyplot9 alias for simple "$\log(x)\ y_1\ y_2\ y_3\ y_4\ y_5\ y_6\ y_7\ y_8\ y_9$" plot

lxlyplot alias for simple "$\log(x)\ \log(y)$" plot

lxlyplot2 alias for simple "$\log(x)\ \log(y_1)\ \log(y_2)$" plot

lxlyplot3 alias for simple "$\log(x)\ \log(y_1)\ \log(y_2)\ \log(y_3)$" plot

lxlyplot4 alias for simple "$\log(x)\ \log(y_1)\ \log(y_2)\ \log(y_3)\ \log(y_4)$" plot

lxlyplot5 alias for simple "$\log(x)\ \log(y_1)\ \log(y_2)\ \log(y_3)\ \log(y_4)\ \log(y_5)$" plot

lxlyplot6 alias for simple "$\log(x)\ \log(y_1)\ \log(y_2)\ \log(y_3)\ \log(y_4)\ \log(y_5)\ \log(y_6)$" plot

lxlyplot7 alias for simple "$\log(x)\ \log(y_1)\ \log(y_2)\ \log(y_3)\ \log(y_4)\ \log(y_5)\ \log(y_6)\ \log(y_7)$" plot

lxlyplot8 alias for simple "$\log(x)\ \log(y_1)\ \log(y_2)\ \log(y_3)\ \log(y_4)\ \log(y_5)\ \log(y_6)\ \log(y_7)\ \log(y_8)$" plot

lxlyplot9 alias for simple "$\log(x)\ \log(y_1)\ \log(y_2)\ \log(y_3)\ \log(y_4)\ \log(y_5)\ \log(y_6)\ \log(y_7)\ \log(y_8)\ \log(y_9)$" plot

- Detailed run diagnostics (Many of these invoke the `plot` command described above and can be toggled to use a logarithmic y scale by use of an additional `l` argument while `L` also plots the absolute values of negative quantities on the same scale and `M` the absolute values of negative quantities in a separate plot)

  2d plots time-dependent quantities from `b2time.nc` (Appendix E), 2^{nd}, 3^{rd} *etc. versus* the 1^{st} argument.

  2da prints time-dependent quantities from `b2time.nc` (Appendix E), 2^{nd}, 3^{rd} *etc. versus* the 1^{st} argument.

  2dt plots time-dependent quantities from `b2time.nc` (Appendix E) versus time. A python version of this script also exists (Python 2.7 and 3.5 compatible).

  2dt_tallies.py plots time-dependent quantities from `b2tallies.nc` (Appendix G) versus time (Python 3.5 compatible).

2dt_av plots averaged time-dependent quantities from `b2time.nc` (Appendix E) versus time.

2dvi prepares a **b2plot** command file `TDV/contr.b2p` for production of input files to start a `DIVIMP` run.

2d_plots prepares `b2time2d.ps` from `b2time.nc` (Appendix E). This uses the batch feature of the `plot` command.

2d_plots_av prepares `b2time2d_av.ps` from averaged values in `b2time.nc` (Appendix E). This uses the batch feature of the `plot` command.

2d_profiles creates `*.last10` files for plotting. When processing fluxes, divides them by the area of contact to the neighbouring cell. By default, the profiles will be averaged over the last 10 snapshots. If the number of available snapshots is less than 10, the default will be to average over all the snapshots available. The averaging limits can be adjusted with the #1 and #2 arguments (default values are 10 and 0 respectively).

2d_profiles_extended creates `*.last10` files for plotting. When processing fluxes, divides them by the area of contact to the neighbouring cell. The flux files ending in `P.last10` are divided by the perpendicular area of the cell face. By default, the profiles will be averaged over the last 10 snapshots. If the number of available snapshots is less than 10, the default will be to average over all the snapshots available. The averaging number can be adjusted with the #1 argument (default value is 10).

2d.summarize Outputs the time-dependent quantities from the last time point in `b2time.nc`.

b2favere_OK Checks that the `b2favere` file is of correct size and content.

b2fstate_OK Checks that the `b2fstate` file is of correct size and content.

b2fstate_OK_bool Checks that the `b2fstate` file is of correct size and content and returns 0 or 1 for use in scripting.

D+_fluid_fluxes Plots the fluxes of D^+ ions across the grid boundaries for a `B2.5` standalone case

D0_fluid_fluxes Plots the fluxes of D^0 neutrals across the grid boundaries for a `B2.5` standalone case

D_coupled_conservation Plots the particle balance of D for coupled runs

D_coupled_conservation_core Plots the particle balance of D in the core plasma for coupled runs

D_fluid_conservation Plots the particle balance of D for `B2.5` standalone runs

D_fluid_conservation_core Plots the particle balance of D in the core plasma for `B2.5` standalone runs

D_fluid_fluxes Plots the fluxes of D across the grid boundaries for a `B2.5` standalone case

He_coupled_conservation_core Plots the particle balance of He in the core plasma for coupled runs

He_fluid_conservation Plots the particle balance of He for `B2.5` standalone runs

He_fluid_fluxes Plots the fluxes of He across the grid boundaries for a `B2.5` standalone case

bremsstrahlung Obsolete. Plots bremsstrahlung losses in **SOLPS 4.x** cases. Use `rqbrmreg` instead.

check_b2_output Compares the two `b2fstate` files provided as arguments.

cpu Lists the CPU time per `B2.5` timestep.

currents.py Plots the electric current across various surfaces. Currents into the domain are shown as negative values, fluxes out of the domain as positive values.

delta Plots the maximum fractional change of the primary variables per `B2.5` timestep.

density Plots the time evolution of the density in a `B2.5` run.

display_tallies Displays the `B2.5` tallies in an easy-to-read format (argument `H` gives the first instance, and `L` the last instance)

energy_analysis Plots the energy flows across certain important surfaces in `B2.5`.

energy_analysis_fht Plots the total energy flows across certain important surfaces in `B2.5`.

energy_analysis_total Plots the total energy flows across certain important surfaces in `B2.5`.

energy_balance Plots the energy balance.

energy_balance_core Plots the energy balance for the core plasma.

energy_balance_core_extended Plots an extended version including volumetric sources of the energy balance for the core plasma.

energy_balance_coupled Plots the energy balance, including sources from `Eirene`.

energy_balance_extended Plots an extended version including volumetric sources of the energy balance.

fchxreg Plots the electric current flow across important x-directed surfaces in `B2.5`.

fchyreg Plots the electric current flow across important y-directed surfaces in `B2.5`.

fhexreg Plots the electron energy flow across important x-directed surfaces in `B2.5`.

fheyreg Plots the electron energy flow across important y-directed surfaces in `B2.5`.

fhixreg Plots the ion energy flow across important x-directed surfaces in `B2.5`.

fhiyreg Plots the ion energy flow across important y-directed surfaces in `B2.5`.

fhmxreg Plots the parallel kinetic energy flow across important x-directed surfaces in `B2.5`.

fhmyreg Plots the parallel kinetic energy flow across important y-directed surfaces in `B2.5`.

fhpxreg Plots the ionisation energy flow across important x-directed surfaces in `B2.5`.

fhpyreg Plots the ionisation energy flow across important y-directed surfaces in `B2.5`.

fhjxreg Plots the electrostatic energy flow across important x-directed surfaces in `B2.5`.

fhjyreg Plots the electrostatic energy flow across important y-directed surfaces in `B2.5`.

fhtxreg Plots the total energy flow across important x-directed surfaces in `B2.5`.

fhtyreg Plots the total energy flow across important y-directed surfaces in `B2.5`.

fluid_fluxes #1 #2 Plots the sum of fluid fluxes for species numbers varying from #1 to #2 (respective defaults are 0 and 1). Also exists as a Python script. Fluxes into the domain are shown as negative values, fluxes out of the domain as positive values.

fluxt #1 Gives the flux strength for the first #1 `Eirene` strata. The default value is all the strata, as read from the `b2.neutrals.parameters` file, or if not present, the `${SOLPSTOP}/modules/B2.5/src/modules.local/b2mod_dimensions.F` file, or if not present, the `${SOLPSTOP}/modules/B2.5/src/modules/b2mod_dimensions.F` file.

get_solps_fluids A utility script to obtain the list of fluid species labels from the `run.log` file.

internal_energy_balance Outputs the energy balance tallies.

internal_energy_balance_coupled Outputs the energy balance tallies for a coupled run.

joule_heating Plots the amount of Joule heating per region.

mtv_3d Plots the time evolution of a `CDF` quantity (Appendix E) profile.

mtv_3d_f Plots the time evolution of the profile of the flux of a `CDF` quantity (Appendix E) along the targets.

mtv_3d_nf Plots the time evolution of the profile of a `CDF` quantity (Appendix E) along both targets side by side.

mtv_3d_t Plots the time evolution of the profile of the a `CDF` quantity (Appendix E) along the targets.

nareg #1 Shows averaged density per region for species #1.

nareg_species #1 Shows averaged density per species for region #1.

nereg Shows averaged electron density per region.

nesepm_feedback Plots in `feedback.ps` the parameters of the *nesepm_sol* feedback scheme.

nesepm_feedback_by_core Plots in `feedback.ps` the parameters of the *nesepm* feedback scheme.

nesepm_feedback_by_pfr Plots in `feedback.ps` the parameters of the *nesepm_pfr* feedback scheme.

nesepm_feedback_fluid Plots in `feedback.ps` the parameters of the *nesepm_sol* feedback scheme for fluid runs.

nireg Shows averaged total ion density per region.

particle_conservation_coupled Plots the particle balance for multi-species coupled runs

print_species Prints the list of species included in the run.

reflection Shows reflected particle fluxes (for `B2.5` standalone runs).

reflection_energy Shows reflected energy fluxes (for `B2.5` standalone runs).

species Obtains the list of species from `SOLPS_FLUIDS`.

sputter Shows sputtering particle fluxes (for `B2.5` standalone runs).

sputter_energy Shows sputtering energy fluxes (for `B2.5` standalone runs).

sputter_RES Shows RES sputtering particle fluxes (for `B2.5` standalone runs).

sputter_RES_energy Shows RES sputtering energy fluxes (for `B2.5` standalone runs).

sputter_chemical Shows chemical sputtering particle fluxes (for `B2.5` standalone runs).

sputter_chemical_energy Shows chemical sputtering energy fluxes (for `B2.5` standalone runs).

sputter_physical Shows physical sputtering particle fluxes (for `B2.5` standalone runs).

sputter_physical_energy Shows physical sputtering energy fluxes (for `B2.5` standalone runs).

tflux #1 Gives the flux strength for the first #1 strata from `b2.neutrals.parameters`. The default value is all the strata, as read from the `b2.neutrals.parameters` file, or if not present, the `${SOLPSTOP}/modules/B2.5/src/modules.local/b2mod_dimensions.F` file, or if not present, the `${SOLPSTOP}/modules/B2.5/src/modules/b2mod_dimensions.F` file.

thermal_evaporation Shows thermal evaporation particle fluxes (for `B2.5` standalone runs).

thermal_evaporation_energy Shows thermal evaporation energy fluxes (for `B2.5` standalone runs).

tereg Shows averaged electron temperature per region.

time_dep Script to obtain time traces of many useful quantities.

tireg Shows averaged ion temperature per region.

viscous_heating Shows the viscous heating term per region.

zeffreg Shows Z_{eff} per region.

many of the above require the `SOLPS_FLUIDS` and `NREG` variables to be set correctly (see section 3.5.4 and Appendix I for the definitions of the regions and surfaces). If the `SOLPS_FLUIDS` or `NREG` files is not found, the necessary information is extracted from the `run.log` file.

- Utility

add_central_device DEVICE Creates writeable directories for a given `DEVICE` for a centralized **SOLPS-ITER** installation.

add_b2plot_to_series Appends b2plot data generated according to command file `add_b2plot_to_series.com` to multi-column file `Series.dat` ganerated by `get_series` (default example of this file can be found in `$SOLPSTOP/scripts/commands`). Read the script header for further details.

ave.awk #1 Performs an average over the #1 rows (default all).

awk.first+even Selects the first and every even column from an array of data.

awk.integrate Integrates an array of data.

awk.mesh Outputs the `mesh.extra` file data in a form usable for `xyplot`.

awk.notfirst Removes the first column from an array of data.

awk.plot_mesh #1 Outputs the mesh specified as #1 in a form usable for `xyplot`.

awk.reverse Outputs a file in reverse order.

awk.transpose Transposes an array of data.

blank Dummy script that does nothing.

canonical #1 Provides the full absolute path to the filename given as argument.

chop_pwd #1 Truncates the name of the current directory to not exceed #1 characters (default 60).

count Basic counting with advanced options (`count -h` for help).

datestamp Gets a date stamp from the machine.

define_compilers This script sets the compiling variables `${CC}` and `${FC}` to the proper C and Fortran compilers, respectively, if not already set, according to the value of the main `${COMPILER}` variable.

env_var_length Lists the length of each environment variable in the shell. When executed as in `'env_var_length | sort -n | tail'`, prints the 10 longest environment variables.

env_var_prune Prunes an environment variable by removing duplicated entries. Usage is `setenv X 'env_var_prune X'`.

fixbb A utility script to properly define the bounding box of PostScript files.

get_series Gets the data from `last_2d_out` files of the runs associated with the series described in `series_file` and prints them out in a plottable multi-column file `Series.dat` according to `get_series.cmd` file (default example of this file can be found in `$SOLPSTOP/scripts/commands`). Read the script header for further details.

gtdr Script to make a directory tree local.

gtfl Replaces the specified links by the files to which they are linked.

grf A more effective version of `grep` when searching for use of a variable in the source code.

inspect_plasmastate #1 This script will return the plasma time stamp at which the `plasmastate` savefile #1 was written.

install_dependencies This script locally downloads and installs all external dependencies needed to correctly compile and run **SOLPS-ITER** , if not already present, in a fully automatized way. Additionally, if needed, it automatically appends the linking flags to the corresponding compiled binaries in a `${SOLPSTOP}/SETUP/config.${HOST_NAME}.${COMPILER}.local` file and in the corresponding configuration files of the various submodules. For local **SOLPS-ITER** installations, all required external dependencies can be locally provided. For installations on HPC clusters, only the dependencies

which may be missing are provided, namely the Blas/Lapack, NetCDF, NCARG, JSON-Fortran, MSCL, GKS, and GR libraries. Note that the NCARG library can be built only for cluster installations using a `gfortran` compiler.

install_miniforge This script installs a **SOLPS-ITER** -specific local Conda installation, based on `miniforge`, located in the `${SOLPSTOP}/lib/${HOST_NAME}.${COMPILER}` directory. Such installation is required to obtain some external packages needed by the compilation of the code, namely Blas/Lapack, NetCDF and NCARG, using the `install_dependencies` script.

last_2d Averaging script for `tracing/*.trc`, `b2time.nc` and `b2tallies.nc` files. Averaging can be perfromed over the last `N` iterations or last `n` seconds. Writes raw output to `last_2d_out` and formatted output to `last_2d_out_f`, the latter is governed by the `last_2d.cmd` file (default example of this file can be found in `$SOLPSTOP/scripts/commands`). Read the script header for further details.

lln Lists the chain of symbolic links leading to a file.

lndr routine to generate shadow trees.

noquote_pwd Returns the current working directory as a string without quotes (c.f. `quote_pwd`).

plot_series Produces `Series.pdf` file (or `Series.ps` if `PLOT_TRC_FMT` environment variable is set to `ps`) containing desired figures based on the data from `Series.dat` file obtained by `get_series` script according to `plot_series.cmd` file (default example of this file can be found in `$SOLPSTOP/scripts/commands`). Read the script header for further details.

plot_trc Produces `trc.pdf` file (or `trc.ps` if `PLOT_TRC_FMT` environment variable is set to `ps`) containing desired figures based on the data from `tracing/*.trc` and `b2time.nc` (potentially other similarly formatted files as well) according to `plot_trc.cmd` file (default example of this file can be found in `$SOLPSTOP/scripts/commands`). Read the script header for further details.

plot_trg Produces `trg.pdf` file (or `trg.pdf` if `PLOT_TRC_FMT` environment variable is set to `ps`) containing desired figures based on the data from `ld_tg_*.dat` and `fp_tg_*.dat` (potentially other similarly formatted files as well) according to `plot_trg.cmd` file (default example of this file can be found in `$SOLPSTOP/scripts/commands`). Also produces `trg_sum.dat` file containing some integral, peak and average quantities at the targets. Read the script header for further details.

quote_pwd Returns the current working directory as a string with quotes (c.f. `noquote_pwd`).

rlndr Replaces a link address by another.

saveb2 Workflow script used to backup and save current run data. Intended usage would be running `saveb2 -l` whenever the run is stopped for the diagnostic purpose. This will create `backup.date` directory, copy all the important files there and move tracing files there as well (this allows to keep the whole run history and simultaneously limit the size of current tracing files for faster plotting and better space management). After that `b2mndr_stime` switch should be set to the current timestamp (can be done by running `STIME -l` script) and run can be continued. After the run is converged `saveb2 -s run_name` can be used to copy all the necessary files to `run_name.save` directory and the current `run_name` directory can be safely deleted to clean-up the space. The script can also be used to setup a workspace on scratch, copy results back and clean-up the workspace.

STIME Auxiliary script to handle `b2mndr_stime` with `saveb2` workflow, run `STIME -h` for details.

wavelength_to_energy Converts a wavelength (in m) to an energy (in eV) (and the reverse).

- Others

b2 Shortcut to `B2`.

b2p Script to obtain a standard series of **b2plot** graphs.

b2pc Script to obtain a standard series of **b2plot** graphs.

b2pl Script to obtain a standard series of **b2plot** graphs, `box.b2p` can be used to define one or more bounding boxes for the PostScript output file (example can be found in `$SOLPSTOP/scripts/commands`).

b2pp Script to obtain a standard series of **b2plot** graphs.

b2pr Script to obtain a standard series of **b2plot** graphs and 1D profiles along X-direction in plottable multi-column format.

b2pt Script to obtain a standard series of **b2plot** graphs.

carre Script used to launch the `Carre` grid generator in the current working directory

climb_and_look_for Script used to find a file in a directory above the current one.

copy_tracing_files Script to bring tracing files to temporary run directory, if present.

correct_baserun_timestamps Makes sure the timestamps for the start-up files in the `baserun` directory are in the right chronological order.

correct_b2yt_timestamps Makes sure the timestamps for the start-up files in the current directory are in the right chronological order for the `b2yt` converter.

d2d Doubles the resolution of the `DG` equilibrium file (see Sec. L.6.1).

d2e Converts `DG` equilibrium files to `EFIT` format (see Sec. L.6.1).

d2v Converts `DG` equilibrium files to `Carre` format (see Sec. L.6.1).

determine_SOLPSTOP Assigns the SOLPSTOP variable inside submission scripts according to the environment variable `SOLPSTOP_FORCE`, if present and pointing to a valid directory, or, next, the content of the local `SOLPSTOP` file or, last, the environment variable `$SOLPSTOP`.

e2d Converts `EFIT` equilibrium files to `DG` format (see Sec. L.6.1).

first_setup Creates the `config.${HOST_NAME}.${COMPILER}` files, if missing, in each of the **SOLPS-ITER** submodules.

get_git_email Finds the user's email address from the `.gitconfig` file.

get_git_name Finds the user's name from the `.gitconfig` file.

get_parm.bpl extracts certain useful quantities from a run input and output files for use in later scripts, particularly those building **b2plot** command files.

IDS_yaml_replace #pulse #old_run #new_run Script invoked by the `b2_ual_rewrite` program (see 3.21.1) to update the `.yaml` IMAS database files to indicate that the `#old_run` run number of pulse `#pulse` is now replaced by run number `#new_run`.

import_files_from_convert Imports and renames the files produced by `b2yt` in a `convert` directory at the same level as the current directory to prepare the new converted case for a run. Also updates some of the files in the `baserun` directory, so it is often prudent to use `correct_baserun_timestamps`, from the parent directory, afterwards.

import_files_from_convert_wg Imports and renames the files produced by `b2us` in a `convert` directory at the same level as the current directory to prepare the new converted wide-grid case for a run.

input.dat_to_mesh.extra Converts standard input (usually directed from a `input.dat` file) into a `mesh.extra` file.

last10 Truncates the `run.log` file to contain only the output from the last 10 written code iterations and stores them in `run.log.last10`.

mds_id Obtains certain data necessary for run identification when saving to the `MDSplus` database.

move_optional_files Script used by the run Makefile to check for the existence of optional output files and, upon succesful termination of a run, move these files to the run directory.

move_save_files Script used by the crash recovery program `b2co` to replace optional run output files with their recovered versions.

n2d Converts "Naka" format equilibrium files to `DG` format (see Sec. L.6.1).

nc_reduce Utility program to reduce the size of very large NetCDF files by keeping only a fraction of the timeslices. Uses `-m #` and `-l #` options to obtain every m-th time slice and/or the last l time slices.

nc2text_simple Replacement program providing the basic capabilities of the `nc2text` utility when not available on the system. A symbolic link to `nc2text` will be created for other scripts to be able to use it.

nn2d Converts new "Naka" format equilibrium files to `DG` format (see Sec. L.6.1).

obtain_tracing_files Script to grab the last version of the tracing files for run recovery.

parse_b2pw Transforms `b2plot.write` file into a set of multi-column files more suitable for plotting, used by `b2pr` script.

pipe_b2.log Prepares the `B2` log file for use by other scripts.

pipe_run.log Prepares the `B2.5` log file for use by other scripts.

preserve_scratch_to_work List of instructions to move run results from temporary scratch space to a backed-up filespace

print_versions Prints the GIT repository versions of the various code executables

ps_GV Used to modify the multi-page `*.ps` output file created via `b2pl` script.

ps_mac Used to modify the multi-page `*.ps` output file created via `b2pl` script.

QSUB.postprocess List of instructions appended to the `B2.5` standalone submission scripts for run post-processing

QSUB.postprocess_coupled List of instructions appended to the **SOLPS-ITER** submission scripts for run post-processing

recover_initial_files For use to copy back the stored initial case files (`.i` extension) before a run restart, or when use of `b2co` is not possible because the run crash happened too early for `plasmastate` files to be written.

regression_test Utility script to run regression tests.

regxy_sum_over_species Intermediate utility script to extract a face-directed region quantity and sum it over species.

res_av Special script to reduce residuals in the finalization step of the `Eirene` sources averaging scheme (see Section 5.6 for details).

rps Shifts Ψ values by a given amount within a `DG` equilibrium file (see Sec. L.6.1).

run_av Special script to prepare the averaging step of the `Eirene` sources averaging scheme (see Section 5.6 for details).

save_initial_files Script used by runs Makefile at start of a run to save initial state files for eventual recovery.

setup_baserun_B25_links Prepares the run directory before a `B2.5` stand-alone run by setting up links to the proper input and data files.

setup_baserun_eirene_links Prepares the run directory before a `B2.5-Eirene` coupled run by setting up links to the proper input and data files. Uses the formatted variant of the main `Eirene` input file (`input.dat`).

setup_baserun_eirene_json_links Prepares the run directory before a `B2.5-Eirene` coupled run by setting up links to the proper input and data files. Uses the JSON variant of the main `Eirene` input file (`eirene.input.json`).

SLURM.efgw_compile Template file for batch compilation on the EFGW machine.

SLURM.iter_compile Template file for batch compilation on the ITER SDCC cluster.

SLURM.postprocess Alias to `QSUB.postprocess`.

SLURM.postprocess_coupled Alias to `QSUB.postprocess_coupled`.

SOLPS4to5 Script to convert a **SOLPS 4.x** run directory into a **SOLPS 5.x** one.

SOLPS-ITER_CI_env Script used to set up the common environment for all continuous integration (CI) tests at ITER.

SOLPS-ITER_CI_test Script used by the continuous integration (CI) server to test **SOLPS-ITER** in standard mode.

SOLPS-ITER_debug_CI_test Script used by the continuous integration (CI) server to test **SOLPS-ITER** in debug mode.

SOLPS-ITER_MPI_CI_test Script used by the continuous integration (CI) server to test **SOLPS-ITER** with MPI options turned on.

SOLPS-ITER_OpenMP_CI_test Script used by the continuous integration (CI) server to test **SOLPS-ITER** with OpenMP options turned on.

solps-iter_update Script to update to the latest **SOLPS-ITER** master distribution.

solps-iter_update_develop Script to switch to the up-to-date **SOLPS-ITER** develop branch.

solps-iter_update_3.0.9_release Script to switch to the **SOLPS-ITER** 3.0.9 release candidate code branch.

solps-iter_update_3.1.0_release Script to switch to the up-to-date **SOLPS-ITER** 9-point-stencil 3.1.0 code branch.

solps-iter_update_3.1.1_release Script to switch to the **SOLPS-ITER** 9-point-stencil 3.1.1 release candidate code branch.

solps-iter_update_wg_release Script to switch to the up-to-date **SOLPS-ITER** wide grids `wg-release` branch (slow-moving reference branch).

solps-iter_update_wg_workflow Script to switch to the up-to-date **SOLPS-ITER** wide grids `wg_workflow` branch (fast-moving development branch).

str2ogr Converts a `.str` structure file into a `DG .ogr` template file.

summarize_run List of instructions provided during job submission for obtaining a standard set of run diagnostics.

switch_to_master Script to switch to the up-to-date **SOLPS-ITER** master branch.

triang Main script used to prepare run input files after the `DivGeo` case has been defined and the `Carre` grid has been built.

triang.conf Script containing utility functions used by the `triang` script.

transfer_solps-iter_runs Script for transferring a **SOLPS-ITER** case. If a run directory from machine `X` is to be transferred to machine `Y` then the script is to be run from machine `X` and just one level up from the directory which is to be transferred. The script will trace back and update the soft links in the run directory according to the remote environment. Running the script requires knowing the IP address of the remote machine and have a username and password to log on to it. This script makes use of the `sshpass` utility.

truncate_tracing_files Script used by the crash recovery program `b2co` to truncate the optional tracing files so that they stay synchronized with the recovered plasma state.

v2d Converts `Carre` format equilibrium files to `DG` format (see Sec. L.6.1).

- Some new Python based scripts have also been added:

calculate_sputter_inner_target.py Calculates the physical sputtering to be expected at the inner target in a simulation saved in MDSplus (chapter 6) on the solps-mdsplus.ipp.mpg.de server (`-h` help is provided).

calculate_sputter_outer_target.py Calculates the physical sputtering to be expected at the outer target in a simulation saved in MDSplus (chapter 6) on the solps-mdsplus.ipp.mpg.de server (`-h` help is provided).

compare_2_solps.py Compares two SOLPS shots with same dimensions saved in MDSplus (chapter 6) on the solps-mdsplus.ipp.mpg.de server (`-h` help is provided).

energy_analysis.py Energy passing through various surfaces.

fluid_core_fluxes.py #1 #2 Total fluid flux entering the core region summed from species #1 to #2.

fluid_fluxes.py #1 #2 Fluid fluxes through various surfaces summed from species #1 to #2.

get_series.py Driver for the `get_series` script.

last_2d.py Driver for the `last_2d` script.

nareg.py #1 Average ion densities for region #1.

nareg_species.py #1 Average ion densities for region #1 summed over charge states for each species.

nereg.py Average electron densities.

plot_trc.py Driver for the `plot_trc`, `plot_trg` and `plot_series` scripts.

rqradreg.py #1 Line radiation summed over each species' charge states for region #1.

rranareg.py #1 Recombination rates for region #1.

rranareg_per_rsanareg.py #1 Recombination rates divided by ionisation rates for region #1 (summed over charge states).

rsanareg.py #1 Ionisation rates for region #1.

sum_trg.py Produces output file `trg_sum.dat` containing some integral, averaged and peaked data at the target using `ld_t_*.dat` and `fp_tg_*.dat` files.

tereg.py Average electron temperatures.

tireg.py Average ion temperatures.

zeffreg.py Average Z_{eff}.

Note that the Python scripts above may require Python versions of at least 3.7 and the availability of the `netCDF4` Python package. At ITER, because some of the underlying service modules are built with `iimpi`, it is even necessary to launch them as interactive MPI jobs, such as:

```
mpirun -np 1 tereg.py
```

Still need to do : The following have not yet been documented — user contributions are welcome!

awk.derivative

cnvtria

min_ave_max.awk

plot_bc_conpar13

plotdg

2.5.1 scripts/commands

Contains various **b2plot** command files.

2.5.2 scripts/matlab

Contains various `Matlab` routines provided by users for data analysis and post-processing.

2.5.3 scripts/palettes

Contains the palettes that can be used by **b2plot** and `mtv_3d` scripts.

2.5.4 scripts/Balance

Contains the Matlab routines used for balance diagnostics, provided by David Moulton of CCFE. A detailed description of their use can be found in the document `${SOLPSTOP}/doc/Balance_routines_in_SOLPS-ITER.pdf`.

2.5.5 scripts/nc2text_simple

Contains the replacement skeleton program for the `nc2text` and `nc_reduce` utilities.

2.5.6 scripts/MatlabPostProcessing

Contains various Matlab routines used for postprocessing run results.

2.5.7 scripts/Matlab_SPb

Contains various Matlab routines used for postprocessing run results when activating the SPb code output switches (`b2wdat_iout` and the like).

2.6 data

Contrary to previous **SOLPS** versions, there is no more centralized version-controlled `${SOLPSTOP}/data` directory. The user may create one if needed, and use it a similar structure as in the past.

For data files that are still experimental in nature, the option to create a parallel `data.local` directory, with the same structure (only putting in the needed new files and subdirectories), at the level `${SOLPSTOP}/data.local` is also still available. The code routinely checks for the existence of local data before relying on the presence of Master copies.

Because the **SOLPS-ITER** distribution is split into submodules, the data files that are accessed exclusively by either `Eirene` or `B2.5` are now placed in directories named `Database` and placed in the corresponding `${SOLPSTOP}/modules` directory. Files that were needed by the post-processing or analysis scripts are now placed within the `${SOLPSTOP}/scripts` directory. Please refer to the appropriate sections above for more details.

Chapter 3

Running the standalone version of B2.5

The compilation of B2.5 is described in section 2.2.1 on page 37.

The run directory is described in section 2.4 on page 44. It is accessible with the sbr command.

The main code is b2mn.exe but it needs input files written by b2ag.exe, b2ah.exe, b2ar.exe and b2ai.exe. As described in section 2.4, these last four are usually run in the baserun directory, and then b2mn.exe is run in the actual run directory.

If one is running B2.5 in standalone mode, it is recommended to set the environment variable:

```
setenv STAND_ALONE yes
```

The general way of running any of these programs is by

```
gmake -f $SOLPSTOP/runs/Makefile b2XX.prt
```

or

```
gmake -f $SOLPSTOP/runs/Makefile b2fYYYYY
```

where b2XX.exe would be the program run, and b2fYYYYY would be one of the output files created by the program. The Makefile is best accessed via ${SOLPSTOP}/runs/Makefile in the above examples and in the following sections. Alternatively, if you are running within the ${SOLPSTOP}/runs tree, the Makefile can be found as ../../Makefile where the number of ../s would reflect how many levels the actual run directory is below the ${SOLPSTOP}/runs directory.

A useful shortcut is to use

```
b2run b2XX
```

This automatically does the gmake for you, finding the Makefile for itself by climbing up the tree. You can add options to the b2run command:

-h Print a help message.

-n Will run the Makefile in dry-run mode. This is useful to check that all the needed files for the runs are in the right order of precedence. If they are not, the correct_baserun_timestamps script can be used to re-order the pre-requisite files found in the associated baserun directory

-m \"MPI-launcher + options\" Allows to launch b2XX in MPI mode with the provided options (example: [the syntax is important, both the slashes and the quotes are needed] b2run -m \"mpiexec -np 2\" b2mn)

-s Specifies that one wishes to run B2.5 in standalone mode, i.e. without any coupling to Eirene.

-o "run options" Allows to launch b2XX with additional run options. The options can be grouped together inside quotes or given individually with multiple -o instances.

-t # Requests an OpenMP run using # threads.

-b Specifies, when calling the `b2co` program, that only the `b2fstate` file should be produced. To be used when invoking `b2co` while the main code is running.

-d "debugger_name" Allows to run `b2XX` with the named debugger (example: `b2run -d totalview b2mn`). If no debugger name is specified, the debugger defined with the `DBX` environment variable and specified in the `${SOLPSTOP}/SETUP/setup.csh.${HOST_NAME}.${COMPILER}.local` or `${SOLPSTOP}/SETUP/setup.csh.${HOST_NAME}.${COMPILER}` files will be used.

The use of `b2run` allows for the run to be launched in a file space that is outside the `${SOLPSTOP}` tree, provided there is a file named `SOLPSTOP` in the run directory or above that contains the path to the `${SOLPSTOP}` tree. If the file named `SOLPSTOP` is missing, the submission script (`QSUB.*` or `SLURM.*`) will construct `${SOLPSTOP}` by truncating the path to actual run directory `${SOLPSTOP}/runs/<RUN_DIRECTORY>` to `${SOLPSTOP}`.

To stop an on-going run in a clean way, before it has reached the initially requested number of iterations, one can use, from the run directory:

```
touch b2mn.exe.dir/.quit
```

This will force the code to stop cleanly at the end of the current iteration, and produce all the output files in an orderly fashion, which then can be post-processed with the usual tools.

When submitting jobs in a cluster environment, one will make use of the job submission commands provided in the `${SOLPSTOP}/runs` directory (`QSUB.*`, `SBATCH.*`, or `SLURM.*` files). Currently, the following commands are provided:

cumulussubmit for submissions on the CCFE Cumulus cluster

csd3submit for submissions on the CCFE CSD3 cluster

darwinsubmit for submissions on MacOS devices (DarwinOS)

ippsubmit for submissions on the new (2019) IPP cluster

itersubmit for submissions on the ITER SDCC cluster

itmsubmit for submissions on the EFGW machine

jetsubmit for submissions on the JET Analysis Cluster (JAC)

jfrssubmit for submissions on the JFRS IFERC-CSC supercomputer

localsubmit for local submission

marconisubmit for submissions on the Marconi supercomputer

mpcdfsubmit for submissions on the MPCDF clusters Raven and Viper

pitagorasubmit for submissions on the Pitagora supercomputer

sgesubmit for submissions on a system running SGE

spbstusubmit for submissions on the SPbSPU HPC cluster in Saint Petersburg, Russia

ustcsubmit for submissions on the USTC computer cluster

Users on other systems are encouraged to recopy these command scripts (found in `${SOLPSTOP}/scripts`) and adapt them to their local environment, along with the needed modifications to the `${SOLPSTOP}/runs/QSUB.*` or `${SOLPSTOP}/runs/SLURM.*` actual submission scripts. Here, `*` stands for `csd3`, `cumulus`, `ipp`, `iter`, `itm`, `jet`, `jfrs`, `lsf`, `marconi`, `mpcdf`, `pitagora`, `sge`, or `spbstu`, referring to the machine system on which the submission script is to be used. Any otherwise named file specified by the user can also be given as an argument to the submission command (in which case the `-z` option is ineffective). The specified file must be located in the `${SOLPSTOP}/runs` directory, or in the actual run directory, or one above it.

Looping scripts, with a _loop extension, are also available for some systems. These will automatically relaunch a case at the end of the time allotted to an individual job submission, provided the run appears to have finished cleanly. If the run appears to have failed, the looping script will stop. It may happen on some systems that some environment variables are appended to or duplicated on job resubmission, which leads to the loop breaking when the maximum allowed length for the environment variable is exceeded — in this case the recommended solution is to save the value of the environment variable at its initialization within the setup.csh scripts and reload it at the beginning of the SLURM.*_loop script. Please refer to the solution suggested for the ITM Gateway concerning environment variables $PATH_FOR_LOOP and $MANPATH_FOR_LOOP, see scripts ${SOLPSTOP}/setup.csh and ${SOLPSTOP}/runs/SLURM.itm*_loop.

All the submission scripts make use of b2run so the same possibility of submitting runs from outside the ${SOLPSTOP} file space is available.

If, for instance, one is running on the ITER cluster, then, the instructions to follow would be to run the script

```
itersubmit
```

or

```
itersubmit -z
```

or

```
itersubmit <<Submission_file>>
```

from your current run directory. This script climbs up the tree looking for <<Submission_file>>, and submits the job to the cluster by the command

```
sbatch <<Job_file>>
```

where <<Job_file>> is a temporary modified copy of <<Submission_file>> with changes corresponding to the particular job submission at hand (built automatically by itersubmit), or stops if the submission file is not found.

If <<Submission_file>> is not specified, itersubmit looks for SLURM.iter. To request a B2.5 standalone run explicitly, the user should use the -s option. By default, the script will assume a run using the coupled code.

In the case one wishes to run the code with MPI and needs to pass MPI options to the launching program, this is achieved by means of the -m option, followed by the desired options inside double quotes, as follows:

```
itersubmit -m "<my MPI options>" <my_optional_submission_file_name>
```

The launching program is identified by the variable MPI_EXEC inside the submission file, and options can be by hand as well using the MPI_OPTS variable.

If the user wishes to submit an OpenMP job, this is done by use of the -t option, followed by the number of desired CPU threads.

On some environments (ITER and SPBSTU, for instance), it is also possible to specify directly the number of nodes on which the job should run, by means of the

```
itersubmit -N <number_of_nodes>
```

command. The default value for the number of nodes is 1. If running on multiple nodes, then the number of processors requested among the MPI options must be the *total* number of processors required.

The default job name is SOLPS-ITER for coupled jobs and B2.5 for standalone jobs. It can be changed by means of the

```
itersubmit -j <my_job_name>
```

command. Standalone jobs are requested with the -s option.

The number of CPU hours dedicated to the job (with possible consequences for the queue on which the submission will take place) can be modified by means of the

```
itersubmit -T 24
```

command, here to ask for a 24-hour long job. The various submission environments will have different default values for the job duration. Similarly, the requested memory size can be modified from the queue default with the

```
itersubmit -M 8GB
```

command, to obtain here a run using 8GB of memory. The `B` (for "bytes") is necessary in this syntax for SLURM submission, but should be omitted when using QSUB submission.

The partition on which the job will be run can be specified by means of the `-Q` option. For example:

```
itersubmit -Q sirius
```

chooses the `sirius` partition, which is the default on the ITER SDCC cluster. Another usable partition on the cluster is `sun`.

When using the SLURM job submission engine, it is also possible to submit a chain of jobs simultaneously (or at least in very quick succession), in which jobs only start after the previous one has completed successfully. Chains can be of arbitrary length and each job in the chain will be added individually to the queue. On oversubscribed clusters, this chain submission option tends to launch the next job in the chain earlier than if resubmitted only after the preceding job has finished. In this context, a successful job completion means that the `b2fstate_OK_bool` returns `0`, i.e. the `b2fstate` end state file and `b2fstati` initial state file are of the same size and format. The relevant option has syntax:

```
itersubmit -c <<JOB_ID>>
```

where `<<JOB_ID>>` is the job number of the previous job in the submission chain. Before continuing with the next job submission, the `run.log.gz` and `b2mn.prt` output files will be stored as `run.log.$JOB_ID.gz` and `b2mn.prt.$JOB_ID`, respectively. The job submission script will be stored as `SLURM.$HOST.after_$JOB_ID`. This option is not available when using the loop submission scripts.

A listing of the available options for each submission script can be obtained by the

```
itersubmit -h
```

command. All options (except `-h`) can be used simultaneously and in any order, provided the name of the (optional) submission file is given last.

Some submission scripts (currently only `ippsubmit` and `mpcdfsubmit`) also allow to launch several runs within one single SLURM job, by means of the `-b` option, if the runs share the same `baserun`. This is done executing the script from the directory above the run and `baserun` directories, specifying the names of the run directories from which the jobs should be launched through a comma-separated list. For example:

```
ippsubmit -b 'run1,run2'
```

This option should be used with careful consideration, in order to request CPU resources in an appropriate way. The ideal situation in which such option could be used is in case of reasonably similar runs, and with all launched runs asked to be stopped after the same wallclock time, so that no allocated CPUs end up being in idle state. Users are encouraged to refer to the possible options, within the above mentioned submission scripts, for the specification of the requested job properties (e.g. requested number of CPUs per run and memory).

The `SLURM.iter` script first of all changes directory to `${SOLPSTOP}` with the help of the `SOLPSTOP` file and executes `setup.csh` there. This is necessary to set up environment variables for the cluster node which would actually execute the job, since that machine does not necessarily coincide with one from which the job was submitted. Then the `SLURM.iter` script returns back to the current run directory and executes the script `b2run`.

The examples presented below are for a pure D plasma.

3.1 b2ag, setting up the geometry

`b2ag` prepares the geometry file, `b2fgmtry`, reading the input file `b2ag.dat`

It is run by doing

```
gmake -f $SOLPSTOP/runs/Makefile b2ag.prt
```

or

```
gmake -f $SOLPSTOP/runs/Makefile b2fgmtry
```

or

```
gmake -f $SOLPSTOP/runs/Makefile fort.30
```

or, best:

```
b2run b2ag
```

3.1.1 b2ag.dat

In the following example of `b2ag.dat`, we identify the mesh file as `jet.50401.58000.geo`, identify it as being of size 96×24 and require a 96×24 `b2fgmtry` file to be prepared.

```
*dimens          (nx, ny, nx1, ny1; free format)
  96  24   96  24
*param           (param(0:99); free format)
  -1.0, 0.0, 0.75, 1.0, -0.010, 0.0, 0.040,
  5.0, 1.0, 0.02, 0.05, 0.05, 0.05,
  0.06, 1.0,
  100*0.0
'b2agfs_geometry'     'jet.50401.58000.geo'
```

***dimens** specifies the size of the grid

- first pair is `NX` & `NY` of the grid you want to produce
- second pair is the size of the grid that was originally created
  - each needs to be an integer multiple of the corresponding entry of the first pair. Note that for double-null cases, the interior guard cells corresponding to the top divertor boundaries should not be multiplied.

***param** at least 100 additional numbers, of which only the first is relevant for us

-1.0 read the mesh data using the "simplified" `Carre` format

-2.0 read the mesh data using the `Sonnet` format

-3.0 read the geometry information for a plasma linear device

b2agfs_geometry specifies the mesh file to be read

- searched for in (in order)
  1. the `b2ag.exe.dir` sub-directory of the current run directory
  2. the current run directory
  3. `${SOLPSTOP}/data.local/meshes`
  4. `${SOLPSTOP}/modules/Carre/meshes/${DEVICE}`
  5. `${SOLPSTOP}/modules/Carre/meshes/${DEVICE}/${USER}`

b2agfs_lineardevice specifies the geometry file for a linear device. The default value is `linear.geometry`. An example can be found in Section 3.1.2.

- searched for in (in order)
  1. the `b2ag.exe.dir` sub-directory of the current run directory
  2. the current run directory
  3. `${SOLPSTOP}/data.local/meshes`
  4. `${SOLPSTOP}/modules/Carre/meshes/${DEVICE}`
  5. `${SOLPSTOP}/modules/Carre/meshes/${DEVICE}/${USER}`

... other options can be used to modify the mesh and magnetic field information that are read in from the mesh file (see Appendix A.4)

`b2ag` will also prepare the grid information in the format usable by `Eirene`, i.e. a `fort.30` file.

3.1.2 Geometry for linear device

The following example contains the required parameters to generate a geometry for a linear device in `B2.5`. These variables must be contained in the filename defined in `b2agfs_lineardevice`.

NCOIL number of magnetic coils in the device

R1 array of size `NCOIL` with the inner radius in units of m of the coils. Default value is 0.

Z1 array of size `NCOIL` with the axial start position in units of m of the coils. Default value is 0.

R2 array of size `NCOIL` with the outer radius in units of m of the coils. Default value is 0.

Z2 array of size `NCOIL` with the axial end position in units of m of the coils. Default value is 0.

CURRENT array of size `NCOIL` with the current per coil in units of A. Default value is 0.

windings array of size `NCOIL` with the number of windings per coil. Default value is 0.

nlayers array of size `NCOIL` with the number of layers to divide each coil in the radial direction. Default value is 5.

stepsize Initial spatial step size used in calculating the field lines in units of m. Default value is 10^{-4}.

bTor Toroidal magnetic field in units of T. Default value is 0.

fn_z name of two column file with the profile of axial cell size in the numerical space. First line in the file determines the number of rows to read.

fn_r name of two column file with the profile of radial cell size in the numerical space. First line in the file determines the number of rows to read.

r_ne radial position in m of the northeastmost point of the plasma region. Default value is 0.

z_ne axial position in m of the northeastmost point of the plasma region. Default value is 0.

r_se radial position in m of the southeastmost point of the plasma region. Default value is 0.

z_se axial position in m of the southeastmost point of the plasma region. Default value is 0.

r_sw radial position in m of the southwestmost point of the plasma region. Default value is 0.

z_sw axial position in m of the southwestmost point of the plasma region. Default value is 0.

r_nw radial position in m of the northwestmost point of the plasma region. Default value is 0.

z_nw axial position in m of the northwestmost point of the plasma region. Default value is 0

r_start Starting radial point in m for the distribution of cells provided by `fn_r`. This is used as the starting point to propagate the field lines. Default value is 0.

r_end End radial point in m for the distribution of cells provided by `fn_r`. This is used as the end point to propagate the field lines. Default value is 0.

z_start Starting axial point in m for the distribution of cells provided by `fn_z`. This is used as the starting point to propagate the field lines. Default value is 0.

z_end End axial point in m for the distribution of cells provided by `fn_z`. This is used as the end point to propagate the field lines. Default value is 0.

gc_n Guard cell size for the North boundary in m. Default value is 10^{-4}.

gc_s Guard cell size for the South boundary in m. Default value is 10^{-4}.

gc_w Guard cell size for the West boundary in m. Default value is 10^{-4}.

gc_e Guard cell size for the East boundary in m. Default value is 10^{-4}.

nExternalRegions Number of external regions to be used in the grid. Default value is 10.

nNodesInRegion Number of nodes per region. Minimum of 3 is required. Default value is 0.

regionR array of size `(nNodesInRegion,nExternalRegions)`. Radial position in m of the nodes defining the external region. Default value is 0.

regionZ array of size `(nNodesInRegion,nExternalRegions)`. Axial position in m of the nodes defining the external region. Default value is 0.

startCellR array of size `nExternalRegions`. Radial index of the first node in the region. Default value is -1.

startCellZ array of size `nExternalRegions`. Axial index of the first node in the region. Default value is -1.

endCellR array of size `nExternalRegions`. Radial index of the last node in the region. Default value is -1.

endCellZ array of size `nExternalRegions`. Axial index of the last node in the region. Default value is -1.

write_nml_ld logical. If `.true.`, writes the contents of the namelist after it has been read. Defaults to `.false.`.

Once parameters are read, the magnetic field lines starting from `r_start`, `r_end`, `z_start`, `z_end` and propagating towards the E and W boundaries are calculated. The files `fn_r` and `fn_z` distribute the cells in the numerical space in the radial and axial directions, respectively. For typical cases in Magnum-PSI, see Fig. 3.1, these field lines start from the target, which has a known geometry and propagate towards the source. The coordinates of each node are calculated to match the field lines computed, thus the final corner coordinates provided in the input file might vary as they are moved to match positions along field lines.

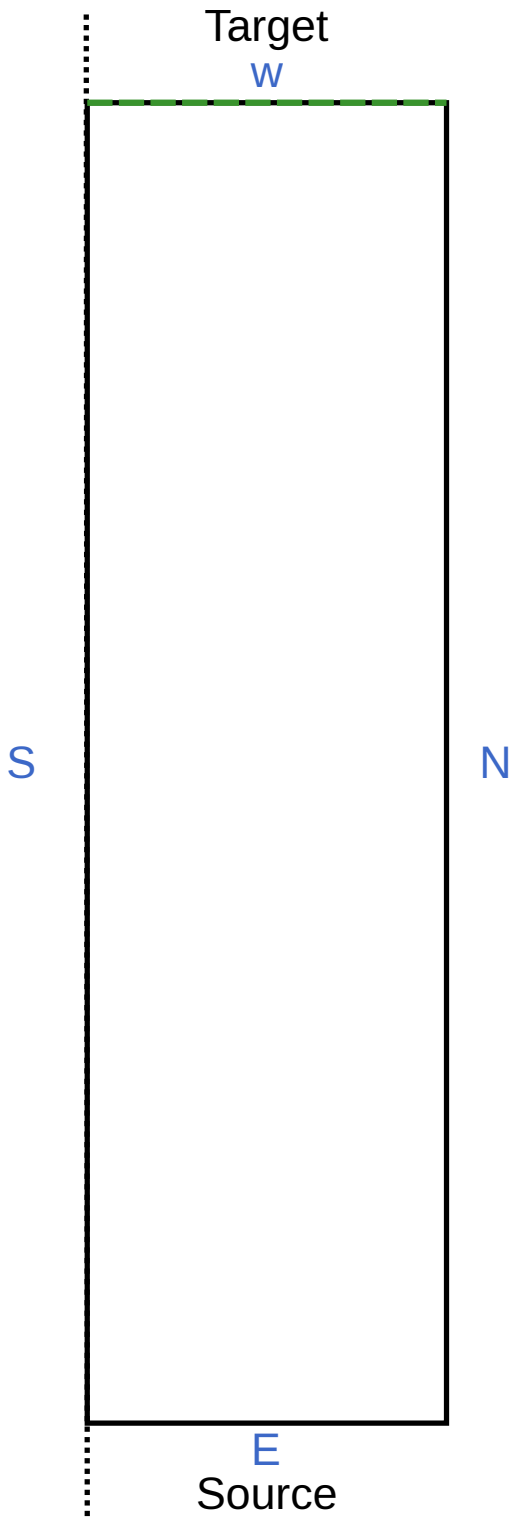


Figure 3.1: Typical geometry in B2.5 for a linear device. This case is taken from a SOLPS-ITER case for the linear device Magnum-PSI. The starting point of the magnetic field lines is represented as a dashed green line.

3.2 b2ah, setting up the default physics parameters

`b2ah` prepares the default physics file, `b2fpardf`, reading the input file `b2ah.dat`

```
gmake -f $SOLPSTOP/runs/Makefile b2ah.prt
```

or

```
gmake -f $SOLPSTOP/runs/Makefile b2fpardf
```

or, best:

```
b2run b2ah
```

3.2.1 b2ah.dat

The recommended way of specifying the actual values used for boundary conditions and transport coefficients is specified later.

```
*dimens          (ns; free format)
   2
*label           (lblps, character*60; free format)
  'Species D'
*b2cmpa          basic parameters
*specs       za        zn        am       za2
   '0'      0.0       1.0       2.0       0.0
   '1'      1.0       1.0       2.0       1.0
*b2cmpb          boundary conditions
*cbregs
 'south'     3        0.25      0.75      2.00
 'north'     1        2.0
 'west'      1        2.0
 'east'      1        2.0
*region
 0  'south, private flux'
*cbsna      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0    -1.0e-02       0         0         0         0         0
   '1'       0         0    -1.0e-03       0         0         0         0         0
*cbsmo      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0    -1.0e+10       0         0      -1.0         0
   '1'       0         0         0         0         0         0      -1.0         0
*cbshi      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'  1.00e-18  -100.0  -1.0e-01       0         0         0         0      -2.5
   '1'  1.00e-18  -100.0  -1.0e-02       0         0         0         0      -2.5
*cbshe      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'  1.00e-18  -100.0  -1.0e-04       0         0         0         0      -2.5
*cbsch      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'       0         0         0         0         0         0         0         0
*cbrec      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0         0         0         0         0         0
   '1'     1.0       0.3         0         0         0         0         0         0
*cbmsa      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'       0         0         0         0         0         0         0         0
*cbmsb      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0         0         0         0         0         0
   '1'       0         0         0         0         0         0         0         0
*region
 1  'south, core plasma'
*cbsna      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0    -1.0e+10       0         0         0         0         0         0
   '1'  4.00e+29 -1.0e+10       0         0         0         0         0         0
*cbsmo      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0    -1.0e+10       0         0         0         0
   '1'       0         0         0    -1.0e+10       0         0         0         0
*cbshi      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
```

```
   '0'  1.00e+04     0         0         0         0         0         0      -2.5
   '1'  1.00e+04     0         0         0         0         0         0      -2.5
*cbshe      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'  1.00e+04     0         0         0         0         0         0      -2.5
*cbsch      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'       0         0         0         0         0         0         0         0
*cbrec      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0         0         0         0         0         0
   '1'       0         0         0         0         0         0         0         0
*cbmsa      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'       0         0         0         0         0         0         0         0
*cbmsb      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0         0         0         0         0         0
   '1'       0         0         0         0         0         0         0         0
*region
 2  'south, private flux'
*cbsna      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0    -1.0e-02       0         0         0         0         0
   '1'       0         0    -1.0e-03       0         0         0         0         0
*cbsmo      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0    -1.0e+10       0         0      -1.0         0
   '1'       0         0         0         0         0         0      -1.0         0
*cbshi      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'  1.00e-18  -100.0  -1.0e-01       0         0         0         0      -2.5
   '1'  1.00e-18  -100.0  -1.0e-02       0         0         0         0      -2.5
*cbshe      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'  1.00e-18  -100.0  -1.0e-04       0         0         0         0      -2.5
*cbsch      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'       0         0         0         0         0         0         0         0
*cbrec      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0         0         0         0         0         0
   '1'     1.0       0.3         0         0         0         0         0         0
*cbmsa      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'       0         0         0         0         0         0         0         0
*cbmsb      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0         0         0         0         0         0
   '1'       0         0         0         0         0         0         0         0
*region
 3  'north, SOL/DIV outer wall'
*cbsna      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0    -1.0e-02       0         0         0         0         0
   '1'       0         0    -1.0e-03       0         0         0         0         0
*cbsmo      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0    -1.0e+10       0         0      -1.0         0
   '1'       0         0         0         0         0         0      -1.0         0
*cbshi      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'  1.00e-18  -100.0  -1.0e-01       0         0         0         0      -2.5
   '1'  1.00e-18  -100.0  -1.0e-02       0         0         0         0      -2.5
*cbshe      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'  1.00e-18  -100.0  -1.0e-04       0         0         0         0      -2.5
*cbsch      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'       0         0         0         0         0         0         0         0
*cbrec      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0         0         0         0         0         0
   '1'     1.0       0.3         0         0         0         0         0         0
*cbmsa      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'       0         0         0         0         0         0         0         0
*cbmsb      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0         0         0         0         0         0
   '1'       0         0         0         0         0         0         0         0
*region
 4  'west, target plate'
*cbsna      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0         0         0         0         0      -1.0
   '1'       0         0         0         0         0         0         0      -1.0
*cbsmo      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0         0    -1.0e+10       0         0         0
```

```
   '1'       0         0      1.0e+10      0     -1.0e+10      0         0         0
*cbshi      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'   1.00e-18   -100.0       0         0         0         0         0       -1.5
   '1'   1.00e-18   -100.0       0       -0.1        0         0         0       -1.5
*cbshe      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'   1.00e-18   -100.0       0       -0.1        0         0         0       -1.5
*cbsch      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'       0         0         0         0         0         0       1.0       0.5
*cbrec      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0         0         0         0         0         0
   '1'     1.0       0.3         0         0         0         0         0         0
*cbmsa      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'       0         0         0         0         0         0         0         0
*cbmsb      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0         0         0         0         0         0
   '1'       0         0         0         0         0         0         0         0
*region
 5  'east, target plate'
*cbsna      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0         0         0         0         0      -1.0
   '1'       0         0         0         0         0         0         0      -1.0
*cbsmo      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0         0     -1.0e+10      0         0         0
   '1'       0         0      1.0e+10      0     -1.0e+10      0         0         0
*cbshi      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'   1.00e-18   -100.0       0         0         0         0         0       -1.5
   '1'   1.00e-18   -100.0       0       -0.1        0         0         0       -1.5
*cbshe      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'   1.00e-18   -100.0       0       -0.1        0         0         0       -1.5
*cbsch      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'       0         0         0         0         0         0       1.0       0.5
*cbrec      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0         0         0         0         0         0
   '1'     1.0       0.3         0         0         0         0         0         0
*cbmsa      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'       0         0         0         0         0         0         0         0
*cbmsb      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0         0         0         0         0         0
   '1'       0         0         0         0         0         0         0         0
*b2cmpt  transport coefficients
*cfdf0      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'    1.0e+02      0        1.0     5.0e-20   1.5e-18      0         0         0
   '1'      0.5        0         0         0         0         0         0         0
*cfdna      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0         0         0         0         0         0
   '1'       0         0       1.0         0         0         0         0         0
*cfdpa      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0       1.0         0         0         0         0         0
   '1'       0         0         0         0         0         0         0         0
*cfvla      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0         0         0         0         0         0
   '1'       0         0         0         0         0         0         0         0
*cfvsa      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0       1.0         0         0         0         0         0
   '1'       0         0       1.0         0         0         0         0         0
*cfhci      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0       1.4         0         0         0         0         0
   '1'       0         0       1.4         0         0         0         0         0
*cfhce      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'     0.7         0         0         0         0         0         0         0
*cfsig      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'   1.0e-07       0         0         0         0         0         0         0
*cfalf      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'       0         0         0         0         0         0         0         0
*cflim    cflme     cflmi     cflmv     cflal     cflab      ...
  '-1'     0.15      0.15      0.5       0.5       0.5        0         0         0
*endphy
```

The basic structure is

***dimens** the number of charge states

***label** specifies, on the next line, a label for the run

***b2cmpa** specifies a block of basic parameters

***b2cmpb** specifies a block of boundary conditions

***b2cmpt** specifies a block of transport coefficients

Taking each of the `b2cmp` blocks in turn:

b2cmpa, basic parameters

The `b2cmpa` block contains

***specs** atomic charge, nuclear charge, atomic mass and atomic charge squared; this data should match that given in `b2ai.dat`

Starting with code version `01.001.024`, an alternative means of describing the plasma species and filling out the `b2cmpa` block is provided, in order to allow for bundling of charge states, when running cases with high-Z species. The relevant description is then

***specs** minimum atomic charge of the stage, maximum atomic charge of the stage, nuclear charge, and atomic mass; this data should match that given in `b2ai.dat`

The code can accept indifferently both types of input and makes the appropriate self-consistency checks. It is not permitted to bundle neutral and ionised species together.

b2cmpb, boundary conditions

The `b2cmpb` block contains

***cbregs** specifies the number of regions where boundary conditions will be specified

- the "South" boundary is broken into three pieces with one quarter of the cells in the first region, half of the cells in the second, and the remaining quarter in the third region (West target private flux, core, East target private flux). This is geometry-dependent information the code will check against the mesh connectivity and return an error if the two do not match
- the "North", "West" and "East" boundaries are not broken up, and correspond to the SOL boundary, inner (outer) target and outer (inner) target, respectively, for lower (upper) single-null configurations. For double-null cases, the north boundary should be split into two sections

***region** one block for each of the regions containing; the immediately following line being the region identifier

***cbsna** boundary conditions for density (1 line per species)

***cbsmo** boundary conditions for parallel momentum (1 line per species)

***cbshi** ion/neutral (or "atomic") temperature/energy boundary condition (1 line per species)

***cbshe** electron temperature/heat flux boundary condition

***cbsch** boundary condition for the electric potential equation

***cbrec** recycling coefficients (1 line per species)

***cbmsa** [unused]

***cbmsb** [unused] (1 line per species)

The exact form of the specification is somewhat arcane: users are referred to the code (b2stbc_bas) for the definitive meaning, or to the write-up in Appendix D.

Users are encouraged to use the "physics" formulation of the boundary conditions specified later (see section 3.5.2), rather than the "Bas" formulation listed here. For double-null and snowflake cases, the "South" boundary must be split in at least six pieces, and the "North" boundary in at least two. However, the b2ah format does not allow for the specification of additional "East" or "West" boundaries, so double-null and snowflake cases **must** use the "physics" formulation in addition. The code enforces this requirement.

The boundary condition implemented arises from numerical combinations of the eight parameters with various quantities from the code.

Anything from this block can be overridden in b2mn.dat and, depending on switches in b2mn.dat, by b2.boundary.parameters (the recycling coefficients can be overridden by b2.neutrals.parameters (see section 3.5.7)).

b2cmpt, transport coefficients

The b2cmpt block contains

***cfdf0** auxiliary transport quantity (1 line per species)

***cfdna** density driven particle diffusion coefficient (1 line per species)

***cfdpa** pressure driven particle diffusion coefficient (1 line per species)

***cfvla** anomalous "pinch" velocity (1 line per species)

***cfvsa** anomalous transverse momentum diffusivity (1 line per species)

***cfhci** anomalous ion/neutral heat diffusivity (1 line per species)

***cfhce** anomalous electron heat diffusivity

***cfsig** diffusivity corresponding to the anomalous electrical conductivity

***cfalf** diffusivity corresponding to the anomalous thermo-electric coefficient

***cflim** coefficients for flux limiting

Users are encouraged to use the "physics" formulation of the transport coefficients using the namelist formulation in b2.transport.parameters or b2.transport.inputfile as described in sections 3.5.3 and 3.5.5.

The transport coefficients implemented arise from numerical combinations of the seven parameters with various quantities from the code.

Anything from this block can be overridden in b2mn.dat and, depending on switches in b2mn.dat, by use of the b2.transport.parameters (section 3.5.3) and b2.transport.inputfile (section 3.5.5) files.

Fuller descriptions are found in b2cdcv.F in the documentation sub-directory of the B2.5 source code (see Appendix A.3), and the relevant subroutine is b2tqna.

It is important, for numerical stability reasons, to provide mechanisms by which some particle transport and radial current (if the electric potential equation is being solved) can be established. If the transport is given by b2cmpt, this means that, first, the cfdna and cfdpa arrays cannot be simultaneously zero when solving density equations. Second, the arrays cfsig and cfalf cannot be simultaneously zero when solving the potential equations without drift current terms on. The code will fail a transport coefficients test in b2tral otherwise.

3.3 b2ar, setting up the atomic physics rates

`b2ar` prepares the default atomic physics rates file, `b2frates`, reading the input file `b2ar.dat`

```
gmake -f $SOLPSTOP/runs/Makefile b2ar.prt
```

or

```
gmake -f $SOLPSTOP/runs/Makefile b2frates
```

or, best:

```
b2run b2ar
```

3.3.1 b2ar.dat

```
*tlohi          (tlo, thi; real, free format)
  1.0e-1  1.0e4
*nlohi          (nlo, nhi; real, free format)
  1.0e18  1.0e22
*numnuc         (nnuc; integer, free format)
   1
*nucspec        (nz, izlo, izhi; integers, one triplet per line)
   1  0  1
*flag           (either 'adpak' or 'strahl', free format)
  'adas'
*tailep         (real, free format)
  0.08
```

tlohi range of temperatures for atomic physics table in eV

nlohi range of densities for atomic physics table in m^{-3}

numnuc number of distinct isonuclear sequences

nucspec nuclear charge, lowest charge state, highest charge state

flag which atomic physics database to use

tailep smoothing parameter for the temperature dependence of some rate coefficients

switches optional, described in `b2cdci`, (see Appendix A.4)

The allowable choices for the **flag** switch are:

'adas' (Atomic Data and Analysis Structure) A complete collisional-radiative atomic physics database, actively being maintained and upgraded [5, 6]. Recommended. Available at http://adas.phys.strath.ac.uk/

'adpak' A complete atomic physics package with scaling law rates applicable to all charge states, but not always of high accuracy [7]

'amns' (Atomic, Molecular, Nuclear, and Surface data) For IMAS and ITM environment runs only, uses access to the AMNS tools and database

'strahl' A collisional-radiative package developed at IPP–Garching [8], stored in the `stra.dat` file

Unless the user chooses to activate the `b2ardr_no_weisheit` switch (see section A.4 for description), the hydrogen rates come from [9] and [10] and overwrite those provided by the data packages above. Those data are stored in the `weis.dat` file which should be present in the `baserun` directory where `b2ar` is run. When running `b2ar.dat` using the dedicated `Makefile`, this is automatically done.

Supplemental information on atomic physics rates can be found in the file `${SOLPSTOP}/doc/SOLPS_2002_Course/atomic_physics.pdf`.

The 'adas' option allows for charge state bundling, provided one has pre-computed the bundled atomic rates. A set of such bundled datasets can be found in the $SOLPSTOP/modules/adas/adas_bundled directory. The adas_DPC_3 directory contains datasets in which isonuclear sequences are bundled in the most aggressive way, i.e. in 3 bundles only (neutral, all partially ionized species, and fully stripped ion). The adas_W directory contains datasets for bundled tungsten (W) using a variety of bundling schemes (see [11] and the contents of the adas_W/Documentation directory for a full description).

As an example, let us assume that one wishes to use the 'jett' bundling scheme for tungsten. The files for this scheme have the naming convention adf11/xxx89/xxx89_w_02_jett.dat where xxx stands for one of acd (recombination), ccd (charge exchange), ecd (average ionisation potential), plt (line radiation rate), prb (energy loss rate), scd (ionisation), ycd (average square charge), or zcd (average charge). Then, the header for the ratadas.filelist should be modified to read:

```
'SOLPSTOP/modules/adas/adas_bundled/adas_W'
```

and the line for the tungsten data should be:

```
'8902' '8902' '8902' '8902' '8902' '8902' '8902' 'w '  'jett'
```

The syntax of the header line understands SOLPSTOP to mean the current value of the $SOLPSTOP environment variable. The SOLPSTOP/modules part of the path can be skipped and will be added automatically by the code when searching for the ADAS datasets if the path provided was not absolute. Only the file path up to but not including the adf11/ directory should be included. If a dataset is not found in the provided path, the code will revert to the one stored in the default directory $SOLPSTOP/modules/adas. The syntax of the tungsten data line contains seven four-digit numbers, one for each process (ycd and zcd must point to the same calculation, so only the latter appears in the file), followed by the name of the element and the label of the bundling scheme, all within single quotes. Each of the seven numbers is made up of the first two digits being the ADAS 'year' and the second set of two digits giving the 'partition level' of the bundling scheme being used for that process. A dataset containing all ions, unbundled, has partition level 01. A bundle of ions will have partition level 02. A bundle of bundles will have partition level 03, and so on. The partition level appears in the dataset filename after the element name and before the bundling scheme label.

The modified ratadas.filelist file should be present in the baserun directory from which b2ar will be run. If this file is prepared before Uinp is run, then the case build-up will assume that the indicated bundling scheme is to be used and create all appropriate input files accordingly. If one is introducing bundling to an existing case, then the recommended procedure is to use the b2yt conversion program to modify all input files as needed. See the corresponding section 3.14 for the full details.

If the ADAS bundled rates are also used in the Eirene input file, then the syntax for the reaction cards in block 4 of the Eirene input file is as follows:

```
 54 ADAS   H.4  scd89    EI    0 184
    w    1       02_jett
```

in which the extension of the ADAS dataset filename (partition level and bundling scheme label) has been added starting on the 18^{th} character of the second line.

3.4 b2ai, the default initial plasma state

`b2ai` prepares the default initial plasma state file, `b2fstati`, reading the input file `b2ai.dat`

```
gmake -f $SOLPSTOP/runs/Makefile b2ai.prt
```

or

```
gmake -f $SOLPSTOP/runs/Makefile b2fstati
```

or, best:

```
b2run b2ai
```

3.4.1 b2ai.dat

```
*dimens          (ns; free format)
   2
*label           (lblps, character*60; free format)
  'Flat initial state for D, all charge states'
*specs       za        zn        am       za2
   '0'      0.0       1.0       2.0       0.0
   '1'      1.0       1.0       2.0       1.0
*naini           (na0(0:ns-1); free format)
  1.0e12  1.0e19  100*1.0e0
*ttini           (tt0; free format)
   1.0e2
```

The basic structure is

***dimens** the number of charge states

***label** a label

***specs** atomic charge, nuclear charge, atomic mass and atomic charge squared for each charge state

***naini** initial densities for each of the charge states, in m^{-3}

***ttini** initial ion and electron temperature, in eV

The initial potential is set to 3.1 times the initial temperature.

Starting with code version `01.001.024`, an alternative means of describing the plasma species and filling out the `b2cmpa` block is provided, in order to allow for bundling of charge states, when running cases with high-Z species. The relevant description is then

***specs** minimum atomic charge of the stage, maximum atomic charge of the stage, nuclear charge, and atomic mass; this data should match that given in `b2ah.dat`

The code can accept indifferently both types of input and makes the appropriate self-consistency checks. It is not permitted to bundle neutral and ionised species together.

The initial state being created by `b2ai.dat` will be a homogeneous isothermal static plasma, which is very far from a realistic plasma edge solution. It is therefore almost always better ans strongly recommended to start a case using an already converged case of the same topology and containing the same main species, even it does not have the same grid size, or corresponds to a different device, or if the species list is somewhat different. To convert such a solution to the parameters of the case at hand, the `b2yt` converter is provided. See Section 3.14 for details on how to use it.

It is possible, by adding the switch

```
'b2aidr_read_b2fstate'    '1'
```

at the end of the `b2ai.dat` file, to obtain a new starting point file which will include the data from the already computed solution in the locally present `b2fstate` file, and to which are added fields corresponding to a flat state for new species declared in the species list from `b2ai.dat`.

3.5 b2mn, the main program

The program can be run with one of the following

```
gmake -f $SOLPSTOP/runs/Makefile b2mn.prt
gmake -f $SOLPSTOP/runs/Makefile b2fparam
gmake -f $SOLPSTOP/runs/Makefile b2fstate
gmake -f $SOLPSTOP/runs/Makefile b2fmovie
gmake -f $SOLPSTOP/runs/Makefile b2ftrace
gmake -f $SOLPSTOP/runs/Makefile b2ftrack
gmake -f $SOLPSTOP/runs/Makefile b2fplasma
```

or, best:

```
b2run b2mn
```

With all of these the code output will come to the screen. It is recommended that it be re-directed to the file `run.log` and the program run in the background. Thus

```
b2run b2mn > run.log &
```

The code reads its data from `b2mn.dat`, and optionally from a number of other files, as described below. Options for monitoring the code run are described in section 3.5.11.

3.5.1 b2mn.dat

```
*label          (lblmn: character*60)
  'JET : no drifts : D only : no drifts'
*b2cmpa  basic parameters
*b2cmpb  boundary conditions
*cbregs
 'south'    3       0.25     0.75     2.00
 'north'    1       2.0
 'west'     1       2.0
 'east'     1       2.0
*region
 4  'west, target plate'
*cbrec     (0)      (1)      (2)      (3)      (4)      (5)      (6)      (7)
   '0'      0        0        0        0        0        0        0        0
   '1'     1.0      0.3       0        0        0        0        0        0
*region
 5  'east, target plate'
*cbrec     (0)      (1)      (2)      (3)      (4)      (5)      (6)      (7)
   '0'      0        0        0        0        0        0        0        0
   '1'     1.0      0.3       0        0        0        0        0        0
*b2cmpt  transport coefficients
*cfvsa     (0)      (1)      (2)      (3)      (4)      (5)      (6)      (7)
   '0'      0        0       1.0       0        0        0        0        0
   '1'      0        0       1.0       0        0        0        0        0
*cfsig     (0)      (1)      (2)      (3)      (4)      (5)      (6)      (7)
  '-1'   1.00E-07 0.0e+00  0.0e+00  0.0e+00  0.0e+00  0.0e+00  0.0e+00  0.0e+00
*cfalf     (0)      (1)      (2)      (3)      (4)      (5)      (6)      (7)
  '-1'   0.00E+00 0.0e+00  0.0e+00  0.0e+00  0.0e+00  0.0e+00  0.0e+00  0.0e+00
*endphy
'b2mndr_ntim'      '1000'
'b2mndr_dtim'      '1.0e-3'
'b2mndt_nstg0'     '1'
'b2mndt_nstg1'     '1'
'b2mndt_nstg2'     '1'
'b2mndt_rxf'       '0.5'
'b2mndr_mvinc'     '1'
'b2mndr_mvnum'     '0'
'b2mndr_b2time'    '10'
```

```
'b2mndr_cpu'       '182800.0'
'b2mndr_savecpu'   '3600.0'
'b2ux5p_nltrsol'   '0'
'b2ux5p_style'     '2'
'b2stbc_sna0ep'    '1.0e-10'
'b2trcl_lluciani'  '1'
'b2news_potit'     '100'
'b2news_potok'     '1e-2'
'b2trcl_lvis21'    '0'
'b2trcl_lthf21'    '0'
'b2tfhe_neutral'         '0.0'
'b2tfhe_vis_par'         '0.0'
'b2news_facdrift_target' '0.00'
'b2news_facdrift_start'  '0.00'
'b2news_facdrift_inc'    '1.01'
'b2news_facdrift_dec'    '0.95'
'b2news_facExB_target'   '0.00'
'b2news_facExB_start'    '0.00'
'b2news_facExB_inc'      '1.01'
'b2news_facExB_dec'      '0.95'
'b2stbc_fchy_dia_coreonly'  '1'
'b2stbc_boundary_namelist'  '1'
'b2srdt_numerics_namelist'  '1'
'b2tqna_transport_namelist' '1'
'b2mwti_jxi'     '40'
'b2mwti_jxa'     '54'
'b2tlh0_alpha'  '1.0'
'b2tfnb_alpha'  '1.0'
'b2news_potitmin' '5'
'b2mndr_eirene'   '0'
'b2stbc_feedback'     '1'
'b2stbc_nesepm_sol'   '1e19'
'b2stbc_diagno'       '1'
'b2stbc_solregno'     '3'
```

The basic structure of `b2mn.dat` is

***label** specifies, on the next line, a label for the run

***b2cmpa** specifies a block of basic parameters, overriding those specified in the block of the same name in `b2ah.dat`

***b2cmpb** specifies a block of boundary conditions, overriding those specified in the block of the same name in `b2ah.dat`

***b2cmpt** specifies a block of transport coefficients, overriding those specified in the block of the same name in `b2ah.dat`

switches described in `b2cdci`, (see Appendix A.4)

3.5.2 b2.boundary.parameters

See also Appendix A.5 for a more up-to-date description.

If `'b2stbc_boundary_namelist'` is set to `'1'` in `b2mn.dat` then the boundary conditions are read in from `b2.boundary.parameters` instead of being taken from the file created by `b2ah` and possibly modified in `b2mn.dat`. For cases with a double-null topology, use of `b2.boundary.parameters` is required, and will be enforced irrespective of the value of the `'b2stbc_boundary_namelist'` switch.

The boundary conditions are specified by the `boundary` namelist which contains

NBC `integer`, number of boundary segments

BCCHAR `character array, length NBC`, specifying the nature of the boundary segment

N "North" boundary

S "South" boundary

W "West" boundary

E "East" boundary

X "X" boundary used for specifying a fixed value on a row of cells

Y "Y" boundary used for specifying a fixed value on a column of cells

BCPOS `integer array, length NBC`, for North, South or X boundary conditions, it specifies the row index; for West, East and Y boundary conditions, it specifies the column index.

BCSTART `integer array, length NBC`, for North, South or X boundary conditions, it specifies the start column index; for West, East and Y boundary conditions, it specifies the start row index.

BCEND `integer array, length NBC`, for North, South or X boundary conditions, it specifies the end column index; for West, East and Y boundary conditions, it specifies the end row index.

BCCON `integer array, length NS * NBC`, specifying the type of density boundary condition for each segment and species (fastest varying index is species); makes use of `CONPAR` to specify additional information, as indicated.

0 no boundary condition is applied (default).

1 prescribe the value of the density, `CONPAR(,,1)` specifies the required density in m^{-3}.

2 prescribe the gradient of the density, `CONPAR(,,1)` specifies the required density gradient in m^{-4}.

3 sheath conditions, `CONPAR(,,1)` not used (zero gradient is used).

4 prescribe the value of the density, weakly a mixed boundary condition, `CONPAR(,,1)` specifies the required density in m^{-3} and `CONPAR(,,2)` specifies the "strength" of the boundary condition.

5 prescribe the particle flux per unit area, `CONPAR(,,1)` specifies the required particle flux density in $m^{-2}.s^{-1}$.

6 prescribe the total particle flux for a constant density, `CONPAR(,,1)` specifies the particle flux in s^{-1}.

7 prescribe the density as a function of other plasma parameters [not yet available].

8 prescribe the total particle flux with constant flux density, `CONPAR(,,1)` specifies the particle flux in s^{-1}.

9 prescribe the decay length for the density, `CONPAR(,,1)` specifies the gradient length in m.

10 leakage option for density, recommended for cases with drifts, `CONPAR(,,1)` specifies the leakage factor, α in $\Gamma_{\text{loss}} = \alpha C_{s,a} n_a$. `CONPAR(,,2)` provides an optional additional stabilization parameter to help with the linearization (must be positive if used).

11 particle flux feedback boundary condition, `CONPAR(,,1)` not used, derived from `CBSNA(0,IS,IREG)`. The species used must be declared using the `b2stbc_isfeedback` switch.

12 particle density feedback boundary condition, as above, `CONPAR(,,1)` not used, derived from `CBSNA(0,IS,IREG)`. The species used must be declared using the `b2stbc_isfeedback` switch.

13 particle density to achieve specified total flux, `CONPAR(,,1)` is the specified radial flux crossing the surface `b2stbc_type13_ref` steps away from the boundary, to which is added the returning flux of ionised neutrals that crossed into the plasma core (*via* `CONPAR(,,3)`) when running with `Eirene` and the `'ionising_core'` switch is used, `CONPAR(,,2)` is the strength of the feedback.

14 if `MOMPAR(,,2)` < 0.5, collective sound speed velocity flux, and marginal imposition of the Bohm-Chodura sheath entrance condition; `CONPAR(,,1)` is a multiplier to the outgoing collective sound speed C_s; to be used in conjunction with `BCENE/I=15`, `BCPOT=11`, and `BCMOM=13`. If `MOMPAR(,,2)`

≥ 0.5, single species sound speed velocity flux, non-marginal imposition of the Bohm-Chodura sheath entrance condition; `MOMPAR(,,1)` is a multiplier to the outgoing specific species sound speed $C_{s,a}$; to be used in conjunction with `BCMOM=13`, `BCENE=15` with `bcene_15_style=1`, `BCENI=15`, `BCPOT=3` or `BCPOT=11` with `istyle_fchi=0`. Recommended for cases with drifts.

15 prescribe a radial leakage velocity, `CONPAR(,,1)` specifies the leakage velocity in units of the local thermal velocity.

16 particle density to achieve specified total flux, used with `ASTRA` coupling. The total desired flux is summed over all `BCCON=16` core boundaries. `CONPAR(,,1)` specifies the desired particle flux in s^{-1}.

18 prescribe total main ion particle flux, used with `ASTRA` coupling.

19 particle flux feedback boundary condition, flux is summed over neutrals and ions, for coupling with `ASTRA`. The total desired flux is summed over all `BCCON=19` core boundaries. This boundary condition applies to ions in their highest ionisation stage. `CONPAR(,,1)` specifies the desired particle flux in s^{-1}.

20 constant density feedback condition, `CONPAR(,,1)` specifies the desired density in m^{-3}.

21 prescribe the value of the density and add a poloidal density variation to get a solution which is as close as possible to neoclassical theory. It is recommended to use this boundary condition together with corresponding condition on ion temperature (`BCENI=24`). The total desired density is summed over all `BCCON=21` core boundaries. `CONPAR(,,1)` specifies the desired density in m^{-3}.

22 feedback boundary condition: given total particle flux with constant average density. The poloidal density variation is added to make the solution as close as possible to the neoclassical value derived for pure H/D/T plasma. It is recommended to use this boundary condition with corresponding boundary condition on ion temperature (`BCENI=23,24`). The total desired flux is summed over all `BCCON=22` core boundaries. `CONPAR(,,1)` specifies the desired particle flux in s^{-1}.

23 feedback boundary condition: given sum of integrated neutrals and main ion particle fluxes with constant average density. The poloidal density variation is added to make the solution as close as possible to the neoclassical value derived for pure H/D/T plasma. It is recommended to use this boundary condition with corresponding boundary condition on ion temperature (`BCENI=23,24`). The total desired flux is summed over all `BCCON=23` core boundaries. `CONPAR(,,1)` specifies the desired particle flux in s^{-1}.

24 constant density feedback scaled by density on the ring `bc_type21_ref` away `CONPAR(,,1)` specifies the desired density in m^{-3}. `CONPAR(,,2)` is the strength of the feedback.

25 feedback boundary condition: prescribe the average value of the density and add the poloidal density variation from neighbouring radial ring. This boundary condition is suitable for any species of a multi-species plasma (i.e. in the case when the condition used in `BCCON=21` fails). It is recommended to use this boundary condition together with the corresponding condition on ion temperature (`BCENI=26,27`). `CONPAR(,,1)` specifies the desired average density in m^{-3}.

26 feedback boundary condition: prescribe the total ion flux and find the average density. The poloidal density variation from the neighbouring radial ring is applied on top of the average density. This boundary condition is suitable for any species of a multi-species plasma (i.e. in the case when the condition used in `BCCON=22` fails). It is recommended to use this boundary condition together with a corresponding condition on the ion temperature (`BCENI=26,27`). The total desired flux is summed over all `BCCON=26` core boundaries. `CONPAR(,,1)` specifies the desired particle flux in s^{-1}.

27 feedback boundary condition: prescribe the particle flux sum for all neutrals and ions belonging to a given isonuclear sequence and find the average density of the ionisation state to which the `BCCON=27` condition is applied. The poloidal density variation from the neighbouring radial ring is applied on top of the average density. It is recommended to use this boundary condition together with corresponding condition on ion temperature (`BCENI=25,26`). The total desired flux is summed over all `BCCON=27` core boundaries. For main ions and low Z impurities, it is recommended to set `BCCON=27` for the highest ionisation state. For high Z impurities, if the ionisation potential of the highest ionisation state of the given isonuclear

sequence is much larger than the expected core plasma temperature, the BCCON=27 condition should be set for the ionisation state for which the density is expected to be maximal. For ionisation states higher than that one, it is recommended to set BCCON=2 with CONPAR=0.0, since their densities are expected to be negligible, and BCCON=2 with CONPAR=0.0 seems to be the safest boundary condition for low density species. CONPAR(,,1) specifies the desired particle flux in s^{-1}.

28 density profile is read from file specified in CON_FN. If CONPAR(IS,IB,2) is positive, the profile is scaled such that CONPAR(IS,IB,1) gives the maximum density of the profile.

29 constant pressure prescribed by CONPAR(,,1) in units of Pa. Density is calculated by dividing the pressure by the temperature.

30 the particle flux per unit area profile is read from the file CON_FN. If CONPAR(IS,IB,2) is positive, the profile is scaled such that CONPAR(IS,IB,1) gives the maximum value of the profile. Only applicable to E and W boundaries.

BCMOM integer array, length NS * NBC, specifying the type of parallel momentum or velocity boundary condition for each segment and species (fastest varying index is species); makes use of MOMPAR to specify additional information, as indicated

0 no boundary condition is applied (default).

1 prescribe the value of the parallel velocity, MOMPAR(,,1) specifies the parallel velocity in $m.s^{-1}$.

2 prescribe the gradient of the parallel velocity, MOMPAR(,,1) specifies the parallel velocity gradient in s^{-1}.

3 sheath conditions, Mach number as input, if MOMPAR(,,2) $< \frac{1}{2}$, then the velocity is set to exactly MOMPAR(,,1) $\times C_{s,\text{collective}}$, otherwise the velocity is set to be at least MOMPAR(,,1) $\times C_{s,\text{species}}$.

4 prescribe the value of the velocity, weakly a mixed boundary condition, MOMPAR(,,1) specifies the parallel velocity in $m.s^{-1}$ and MOMPAR(,,2) specifies the "strength" of the boundary condition.

5 prescribe the parallel momentum flux per unit area, MOMPAR(,,1) specifies the parallel momentum flux density in $N.m^{-2}$.

6 prescribe the total parallel momentum flux for a constant parallel velocity [not yet available].

7 prescribe the parallel momentum as a function of other plasma parameters [not yet available].

8 special : limited shear, imposes zero gradient for the Mach number. [[[Eventually intended to have MOMPAR(,,1) specify the gradient of the Mach number in m^{-1}]]].

9 prescribe the total parallel momentum flux with constant flux density, MOMPAR(,,1) specifies the parallel momentum flux in N.

10 prescribe the decay length for the parallel momentum, MOMPAR(,,1) specifies the decay length in m.

11 Rozhansky viscosity condition for the parallel momentum, MOMPAR(,,1) is not used.

12 condition from b2stbc_spb for the parallel momentum.

13 if MOMPAR(,,2) < 0.5, collective sound speed velocity flux, and marginal imposition of the Bohm-Chodura sheath entrance condition; MOMPAR(,,1) is a multiplier to the outgoing collective sound speed C_s; to be used in conjunction with BCENE/I=15, BCPOT=11, and BCCON=14. If MOMPAR(,,2) ≥ 0.5, single species sound speed velocity flux, non-marginal imposition of the Bohm-Chodura sheath entrance condition; MOMPAR(,,1) is a multiplier to the outgoing specific species sound speed $C_{s,a}$; to be used in conjunction with BCCON=14, BCENE=15 with bcene_15_style=1, BCENI=15, BCPOT=3 or BCPOT=11 with istyle_fchi=0. Recommended for cases with drifts.

14 condition from b2stbc_spb for the parallel momentum.

15 prescribe the value of the parallel velocity as a function of poloidal coordinate, scaling with $\frac{\langle B \rangle}{B}$.

16 prescribe the average value of the parallel velocity, MOMPAR(,,1) specifies the parallel velocity in $m.s^{-1}$.

17 leakage option for parallel momentum, `MOMPAR(,,1)` specifies the leakage factor, α in $\Gamma_{\text{loss}} = \alpha C_{s,a} m_a n_a u_a$.

18 prescribe the parallel velocity profile in units of $m.s^{-1}$, read from file specified in `MOM_FN`. If `MOMPAR(IS,IB,2) > 0.0`, the profile is scaled such that the maximum parallel velocity is given by `MOMPAR(IS,IB,1)` in $m.s^{-1}$. Only available for E and W boundaries.

19 prescribe the momentum flux profile in units of $m^{-2}.s^{-1}$, read from `MOM_FN`. This has to be used in combination with `BCCON = 1`, `28` , or `30` and `BCENI = 15`. Only available for E and W boundaries.

20 the parallel velocity is based on the temperature and the Mach number read from the `MOM_FN` file. It is assumed that $\gamma_i = 1$. Only available for E and W boundaries.

21 prescribe the radial parallel velocity gradient profile in units of s^{-1}, read from file specified in `MOM_FN`. If `MOMPAR(IS,IB,2) > 0.0`, the profile is scaled such that the maximum parallel velocity gradient is given by `MOMPAR(IS,IB,1)` in s^{-1}. Only available for E and W boundaries.

BCENE `integer array, length NBC`, specifying the type of electron energy or temperature boundary condition for each segment; makes use of `ENEPAR` to specify additional information, as indicated

0 no boundary condition is applied (default).

1 prescribe the value of the electron temperature, `ENEPAR(,1)` specifies the temperature in eV.

2 prescribe the gradient of the electron temperature, `ENEPAR(,1)` specifies the temperature gradient in $eV.m^{-1}$.

3 sheath conditions, electron energy transmission, `ENEPAR(,1)` specifies an additional contribution to the energy transmission coefficient in addition to that of the potential difference [the sound speed used depends on settings of `MOMPAR(,ISMAIN,2)`].

4 prescribe the value of the electron temperature, weakly a mixed boundary condition, `ENEPAR(,1)` specifies the temperature in eV and `ENEPAR(,2)` specifies the "strength" of the boundary condition.

5 prescribe the electron energy flux per unit area, `ENEPAR(,1)` specifies the energy flux density in $W.m^{-2}$.

6 prescribe the total electron energy flux for a constant electron temperature, `ENEPAR(,1)` specifies the energy flux in W (no longer supported, use types [16] or [17] instead).

7 prescribe the electron temperature as a function of other plasma parameters [not yet available].

8 prescribe the total electron heat flux with constant flux density, `ENEPAR(,1)` specifies the energy flux in W.

9 prescribe the decay length for the electron temperature, `ENEPAR(,1)` specifies the decay length in m (can also use type [19] instead).

10 feedback option for core, `ENEPAR(,1)` not used, derived from `cbshe(0,coreregno)`.

11 not used

12 sheath conditions, electron energy transmission coefficient, `ENEPAR(,1)` specifies an energy transmission factor, δ_e in $Q_e = \delta_e \Gamma_e T_e$.

13 prescribe the electron energy flux per unit area proportional to temperature, `ENEPAR(,1)` specifies the energy flux density per temperature in $W.m^{-2}.J^{-1}$ (the temperature here in J).

14 leakage option for electron energy, `ENEPAR(,1)` specifies the leakage factor, α in $\Gamma_{\text{loss}} = \alpha C_{s,\text{collective}} n_e T_e$

15 sheath boundary condition, recommended when using drifts. Linked to using `BCCON=14` and `BCMOM=13` for all ion species, `BCENI=15`, and `BCPOT=11`. When `bcene_15_style=1`, the electron flux is computed via the exponential of the sheath potential drop, and `ENEPAR(:,1)` is used to artificially increase or decrease the sheath transmission coefficient for electrons (default = `0`).

16 feedback boundary condition with constant temperature, `ENEPAR(,1)` specifies the power flux in W across the flux surface with index `b2stbc_type16_ref` (default is `-1`), `ENEPAR(,2)` should be something like `0.1` and specifies the strength of the feedback. Also see type [17] below. Available if `bcene_16_style=0` (default). If `bcene_16_style=1`, integrated electron heat flux with constant T_e, summed over all core boundaries with `BCENE=16`.

17 feedback boundary condition with constant shared temperature for both electrons and ions, with `ENEPAR(,1) + ENIPAR(,1)` giving the total power flux in W across the flux surface with index `b2stbc_type16_ref` (default=-1), `ENEPAR(,2)` should be something like `0.1` and specifies the strength of the feedback. Replaces [16] for high densities and large values of `b2stbc_type16_ref`.

18 fractional drop condition. Not yet working.

19 same as [9] but to be used when simultaneously setting `BCCON=1` on the same boundary.

20 feedback boundary condition with constant temperature, as per type [16] but with `type20_` switches and a different feedback scheme.

21 constant temperature feedback scaled by temperature on the ring `bc_type21_ref` away. `ENEPAR(,1)` specifies the desired electron temperature in eV. `ENEPAR(,2)` is the strength of the feedback.

22 radial leakage condition for the electron temperature. `ENEPAR(,1)` specifies the leakage velocity in units of the electron thermal velocity. A temperature gradient such that the diffusive flux is set to match this leakage is imposed.

23 prescribe electron temperature profile in eV, read from file `ENE_FN`. If `ENEPAR(IB,2)` is positive, the profile is scaled such that `ENEPAR(IB,1)` is the maximum electron temperature of the profile in eV. Only available for E and W boundaries.

BCENI `integer array, length NBC`, specifying the type of ion energy or temperature boundary condition for each segment; makes use of `ENIPAR` to specify additional information, as indicated

0 no boundary condition is applied (default).

1 prescribe the value of the ion temperature, `ENIPAR(,1)` specifies the temperature in eV.

2 prescribe the gradient of the ion temperature, `ENIPAR(,1)` specifies the temperature gradient in $eV.m^{-1}$.

3 sheath conditions, ion energy transmission, `ENIPAR(,1)` specifies the contribution to the energy transmission coefficient [the sound speed used depends on settings of `MOMPAR(,ISMAIN,2)`].

4 prescribe the value of the ion temperature, weakly a mixed boundary condition, `ENIPAR(,1)` specifies the temperature in eV and `ENIPAR(,2)` specifies the "strength" of the boundary condition.

5 prescribe the ion energy flux per unit area, `ENIPAR(,1)` specifies the energy flux density in $W.m^{-2}$.

6 prescribe the total ion energy flux for a constant ion temperature, `ENIPAR(,1)` specifies the energy flux in W (no longer supported, use types [16] or [17] instead).

7 prescribe the ion temperature as a function of other plasma parameters [not yet available]

8 prescribe the total ion heat flux with constant flux density, `ENIPAR(,1)` specifies the energy flux in W.

9 prescribe the decay length for the ion temperature, `ENIPAR(,1)` specifies the decay length in m (can also use type [19] instead).

10 feedback option for core, `ENIPAR(,1)` not used, derived from `cbshi(0,ISMAIN,coreregno)`.

11 sheath conditions, ion energy transmission coefficient, `ENIPAR(,1)` specifies an energy transmission factor, δ_i in $Q_i = \delta_i T_i \sum_a n_a C_{s,a}$.

12 sheath conditions, ion energy transmission coefficient, `ENIPAR(,1)` specifies an energy transmission factor, δ_i in $Q_i = \delta_i T_i \sum_a \Gamma_a$.

13 prescribe the ion energy flux per unit area proportional to temperature, `ENIPAR(,1)` specifies the energy flux density per temperature in $W.m^{-2}.J^{-1}$ (the temperature here in J).

14 leakage option for ion energy, `ENIPAR(,1)` specifies the leakage factor, α in $\Gamma_{\text{loss}} = \alpha C_{s,\text{collective}} n_i T_i$

15 sheath boundary condition. If `MOMPAR(ismain,,2)` < 0.5, collective sound speed velocity flux, and marginal imposition of the Bohm-Chodura sheath entrance condition; the ion heat flux is computed using the collective sound speed and `ENIPAR(:,1)` sets the ion sheath transmission coefficient (default `1.5`); to be used in conjunction with `BCENE=15`, `BCPOT=11`, `BCCON=14` and `BCMOM=13` for all ion species. If `MOMPAR(,,2)` ≥ 0.5, single species sound speed velocity flux, non-marginal imposition of the Bohm-Chodura sheath entrance condition; the ion heat flux is computed using the target particle flux and `ENIPAR(:,1)` sets the ion sheath transmission coefficient (default `1.5`); to be used in conjunction with `BCMOM=13`, `BCENE=15` with `bcene_15_style=1`, `BCCON=14`, `BCPOT=3` or `BCPOT=11` with `istyle_fchi=0`. Recommended for cases with drifts.

16 Feedback boundary condition with constant temperature, `ENIPAR(,1)` specifies the power flux in W across the flux surface with index `b2stbc_type16_ref` (default is -1), `ENIPAR(,2)` should be something like `0.1` and specifies the strength of the feedback. Also see type [17] below. Available if `bceni_16_style=0` (default). If `bceni_16_style=1`, integrated ion heat flux with constant T_i, summed over all core boundaries with `BCENI=16`.

17 feedback boundary condition with constant shared temperature for both electrons and ions, see `BCENE=17` above for description.

18 fractional drop condition. Not yet working.

19 same as [9] but to be used when simultaneously setting `BCCON=1` on the same boundary.

20 feedback boundary condition with constant temperature, as per type [16] but with `type20_` switches and a different feedback scheme.

21 from `b2stbc_spb`

22 radial leakage velocity for the ion temperature. `ENIPAR(,1)` specifies the leakage velocity in units of the collective ion thermal velocity. A temperature gradient such that the diffusive flux is set to match this leakage is imposed.

23 prescribe the poloidally averaged value of the ion temperature and introduce a poloidal variation as close as possible to neoclassical solution. It is recommended to use this boundary condition together with corresponding condition on ion density (`BCCON=21,22,23`). The average is taken over all core boundaries with `BCENI=23`. `ENIPAR(,1)` specifies the temperature in eV.

24 feedback boundary condition with prescribed total ion flux, constant poloidally averaged ion temperature and a poloidal variation as close as possible to neoclassical solution. It is recommended to use this boundary condition together with corresponding condition on ion density (`BCCON=21,22,23`). The flux is summed over all core boundaries with `BCENI=24`. `ENIPAR(,1)` specifies the energy flux in W.

25 constant temperature feedback scaled by temperature on the ring `bc_type21_ref` away `ENIPAR(,1)` specifies the desired ion temperature in eV. `ENIPAR(,2)` is the strength of the feedback.

26 prescribe the poloidally averaged value of the ion temperature and introduce a poloidal variation in a simplified manner. This boundary condition is suitable for any plasma composition (i.e. when `BCENI=23` fails). It is recommended to use this boundary condition together with corresponding condition on ion density (`BCCON = 25,26,27`). The average is taken over all core boundaries with `BCENI=26`. `ENIPAR(,1)` specifies the temperature in eV.

27 feedback boundary condition with prescribed total ion heat flux, constant poloidally averaged ion temperature and a poloidal variation in a simplified manner. This boundary condition is suitable for any plasma composition (i.e. when `BCENI=24` fails). It is recommended to use this boundary condition together

with corresponding condition on ion density (`BCCON = 25,26,27`). The flux is summed over all core boundaries with `BCENI=27`. `ENIPAR(,1)` specifies the energy flux in W.

28 prescribe ion temperature profile in eV, read from file `ENI_FN`. If `ENIPAR(IB,2)` is positive, the profile is scaled such that `ENIPAR(IB,1)` is the maximum ion temperature of the profile in eV. Only available for E and W boundaries.

BCPOT `integer array, length NBC`, specifying the type of electric potential or current boundary condition for each segment; makes use of `POTPAR` to specify additional information, as indicated

0 no boundary conditon is applied (default).

1 prescribe the value of the potential, `POTPAR(,1)` specifies the potential in V.

2 prescribe the gradient of the potential, `POTPAR(,1)` specifies the potential gradient in $V.m^{-1}$.

3 sheath conditions, `POTPAR(,2)` used for biasing [see code for details].

4 prescribe the value of the potential weakly a mixed boundary condition, `POTPAR(,1)` specifies the potential in V and `POTPAR(,2)` specifies the "strength" of the boundary condition.

5 prescribe the current flux density per unit area, `POTPAR(,1)` specifies the electric current flux density in $A.m^{-2}$.

6 prescribe the total current flux density for a constant potential [not yet available].

7 prescribe the potential as a function of other plasma parameters [not yet available].

8 prescribe the total electric current with constant flux density, `POTPAR(,1)` specifies the electric current in A.

9 prescribe the decay length for the potential, `POTPAR(,1)` specifies the decay length in m.

10 feedback option for core [not yet tested!!!!!!!!!] (based on using `cbsch(0,coreregno)`).

11 sheath conditions, electron energy transmission from `b2stbc_spb`, recommended when using drifts. Linked to using `BCCON=14` and `BCMOM=13` for all ion species, and `BCENE/I=15`. `POTPAR(,2)` specifies the bias potential in V.

12 imposes the currents due to drifts for the South core boundary. Must be used in conjunction with the switch setting `istyle_cur_contr_on_S_and_N=2`.

13 imposes the currents due to drifts for the South private flux and North boundaries. Must be used in conjunction with `istyle_cur_contr_on_S_and_N=2`.

14 prescribe the value of the radial sheath potential, `POTPAR(,1)`, in V.

15 prescribe potential profile in V, read from file defined in `POT_FN`. Only available in E and W boundaries.

16 feedback condition of constant electric potential for a given prescribed current. The prescribed current (in A) is given by the sum of the `POTPAR(,1)` of all the boundary conditions with `BCPOT=16`. Still experimental, will not work for drift cases.

17 sheath condition for floating target. This BC provides a zero net current through the target. First, the total current through the surface is computed. Then, the potential is corrected by a factor of `POTPAR(,1) * current` to reduce the current. `POTPAR(,2)` used for storing the potential value, in V, during the iterative process and to provide an initial guess.

21 constant potential feedback scaled by potential on the ring `bc_type21_ref` away, `POTPAR(,1)` specifies the desired electric potential in V, `POTPAR(,2)` is the strength of the feedback.

CONPAR `real array, length NS * NBC * 3`, supplies the additional parameters needed for the option specified by `BCCON`

MOMPAR `real array, length NS * NBC * 2`, supplies the additional parameters needed for the option specified by `BCMOM`

ENEPAR `real array, length NBC * 2`, supplies the additional parameters needed for the option specified by `BCENE`

ENIPAR `real array, length NBC * 2`, supplies the additional parameters needed for the option specified by `BCENI`

POTPAR `real array, length NBC * 2`, supplies the additional parameters needed for the option specified by `BCPOT`

GAMMAI `real`, ratio of specific heats (adiabatic coefficient)

GAMMAE `real`, secondary electron emission coefficient

LBNDUSR `logical`, if true will also call Braams boundary condition routine after the end of the physics boundary condition routine

LFEEDBACK `logical`, if true enables various feedback mechanisms (needed for any feedback mechanisms)

NNISO `integer`, number of dead (or isolated) regions

NXISO1 `integer array, length NNISO`, column number of bottom left corner of the dead regions

NXISO2 `integer array, length NNISO`, column number of top right corner of the dead regions

NYISO1 `integer array, length NNISO`, row number of bottom left corner of the dead regions

NYISO2 `integer array, length NNISO`, row number of top right corner of the dead regions

NIISO `real array, length NS`, ion densities of the dead regions in m^{-3}

TEISO `real`, electron temperature of the dead regions in eV

TIISO `real`, ion temperature of the dead regions in eV

PHIISO `real`, electric potential of the dead regions in V

BOUNDARY_FILENAME `filename`, the name of a new boundary input file that will be read based on a condition determined by mod (`TIM`,`BOUNDARY_TIME_MOD`) >`BOUNDARY_TIME_SWITCH`

BOUNDARY_TIME_MOD `real`, the basic time periodicity used to set time varying boundary conditions, in s

BOUNDARY_TIME_SWITCH `real`, the basic time at which a switch is to be made to a new boundary input file, in s

LCBS `integer array, length NBC`, indices of the core boundary segments in `B2` and `B2.5` for passing to `Eirene`

CON_FN `character*256 array of length NS * NBC`, list of files for specifying the profile for `BCCON = 28` and `30` or `BCMOM = 19`. The format is a plain text file in which the first line is the number of data points and then two columns with information about the coordinate (in m) and the profile value. The number of data values is unrelated with the number of grid elements as the profile is interpolated to the boundary.

MOM_FN `character*256 array of length NS * NBC`, list of files for specifying the profile for `BCMOM = 18`, `19`, or `21`. Same format as `CON_FN`.

ENE_FN `character*256 array of length NBC`, list of files for specifying the profile for `BCENE = 23`. Same format as `CON_FN`.

ENI_FN `character*256 array of length NBC`, list of files for specifying the profile for `BCENI = 28`. Same format as `CON_FN`.

POT_FN `character*256 array of length NBC`, list of files for specifying the profile for `BCPOT = 15`. Same format as `CON_FN`.

Additional boundary conditions modifications due to drifts are automatically included if drifts are turned on (the core current is adjusted to be its neo-classical value, and the Bohm criterion at the plates is adjusted to include drift velocities).

3.5.3 b2.transport.parameters

See also Appendix A.5 for a more up-to-date description.

If `'b2tqna_transport_namelist'` is set to `'1'` in `b2mn.dat` then the transport coefficients are read in from the namelist found in `b2.transport.parameters` instead of being taken from the file created by `b2ah` and possibly modified in `b2mn.dat`. The transport coefficients are specified by the `transport` namelist which contains

FLAG_DNA `integer array, length NS`, density driven particle diffusion model

- **0** use value specified in `b2cmpt` block in `b2ah.dat`, perhaps overridden in `b2mn.dat`
- **1** constant value specified by `PARM_DNA(IS)`
- **2** constant value specified by `PARM_DNA(IS)` scaled by $\frac{10^{20}}{n_e[m^{-3}]}$
- **3** constant value specified by `PARM_DNA(IS)` scaled by Bohm ($\frac{T_e[eV]}{16B[T]}$)
- **4** constant value specified by `PARM_DNA(IS)` (flux scaled)

FLAG_DPA `integer array, length NS`, pressure driven particle diffusion model

- **0** use value specified in `b2cmpt` block in `b2ah.dat`, perhaps overridden in `b2mn.dat`
- **1** constant value specified by `PARM_DPA(IS)`
- **2** constant value specified by `PARM_DPA(IS)` scaled by $\frac{10^{20}}{n_e[m^{-3}]}$
- **3** constant value specified by `PARM_DPA(IS)` scaled by Bohm ($\frac{T_e[eV]}{16B[T]}$)
- **4** constant value specified by `PARM_DPA(IS)` (flux scaled)

FLAG_VLA `integer array, length NS`, radial pinch velocity model

- **0** use value specified in `b2cmpt` block in `b2ah.dat`, perhaps overridden in `b2mn.dat`
- **1** constant value specified by `PARM_VLA(IS)`
- **2** constant value specified by `PARM_VLA(IS)` scaled by $\frac{10^{20}}{n_e[m^{-3}]}$
- **3** constant value specified by `PARM_VLA(IS)` scaled by Bohm ($\frac{T_e[eV]}{16B[T]}$)
- **4** constant value specified by `PARM_VLA(IS)` (flux scaled)

FLAG_VSA `integer array, length NS`, anomalous viscosity model

- **0** use value specified in `b2cmpt` block in `b2ah.dat`, perhaps overridden in `b2mn.dat`
- **1** constant value specified by `PARM_VSA(IS)`
- **2** constant value specified by `PARM_VSA(IS)` scaled by $\frac{10^{20}}{n_e[m^{-3}]}$
- **3** constant value specified by `PARM_VSA(IS)` scaled by Bohm ($\frac{T_e[eV]}{16B[T]}$)
- **4** constant value specified by `PARM_VSA(IS)` (flux scaled)

FLAG_HCI `integer array, length NS`, anomalous ion thermal diffusivity model

- **0** use value specified in `b2cmpt` block in `b2ah.dat`, perhaps overridden in `b2mn.dat`

1 constant value specified by `PARM_HCI(IS)`

2 constant value specified by `PARM_HCI(IS)` scaled by $\frac{10^{20}}{n_e[m^{-3}]}$

3 constant value specified by `PARM_HCI(IS)` scaled by Bohm ($\frac{T_e[eV]}{16B[T]}$)

4 constant value specified by `PARM_HCI(IS)` (flux scaled)

FLAG_HCE `integer`, anomalous electron thermal diffusivity model

0 use value specified in `b2cmpt` block in `b2ah.dat`, perhaps overridden in `b2mn.dat`

1 constant value specified by `PARM_HCE`

2 constant value specified by `PARM_HCE` scaled by $\frac{10^{20}}{n_e[m^{-3}]}$

3 constant value specified by `PARM_HCE` scaled by Bohm ($\frac{T_e[eV]}{16B[T]}$)

4 constant value specified by `PARM_HCE` (flux scaled)

FLAG_SIG `integer`, anomalous current conductivity model

0 use value specified in `b2cmpt` block in `b2ah.dat`, perhaps overridden in `b2mn.dat`

1 constant value specified by `PARM_SIG`

2 constant value specified by `PARM_SIG` scaled by $\frac{10^{20}}{n_e[m^{-3}]}$

3 constant value specified by `PARM_SIG` scaled by Bohm ($\frac{T_e[eV]}{16B[T]}$)

4 constant value specified by `PARM_SIG` (flux scaled)

FLAG_ALF `integer`, anomalous thermo-electric current model

0 use value specified in `b2cmpt` block in `b2ah.dat`, perhaps overridden in `b2mn.dat`

1 constant value specified by `PARM_ALF`

2 constant value specified by `PARM_ALF` scaled by $\frac{10^{20}}{n_e[m^{-3}]}$

3 constant value specified by `PARM_ALF` scaled by Bohm ($\frac{T_e[eV]}{16B[T]}$)

4 constant value specified by `PARM_ALF` (flux scaled)

PARM_DNA `real array, length NS`, provides parameters for `FLAG_DNA`

PARM_DPA `real array, length NS`, provides parameters for `FLAG_DPA`

PARM_VLA `real array, length NS`, provides parameters for `FLAG_VLA`

PARM_VSA `real array, length NS`, provides parameters for `FLAG_VSA`

PARM_HCI `real array, length NS`, provides parameters for `FLAG_HCI`

PARM_HCE `real`, provides parameters for `FLAG_HCE`

PARM_SIG `real`, provides parameters for `FLAG_SIG`

PARM_ALF `real`, provides parameters for `FLAG_ALF`

CFLME `real`, main multiplier to the electron flux limit

CFLMI `real`, main multiplier to the ion flux limit

CFLMV `real`, main multiplier to the viscosity flux limit

CFLAL `real`, multiplier to the thermo-electric coefficient flux limit

CFLAB `real`, multiplier to the friction force flux limit

CFLME_CORE `real`, limit to multiplier to the electron heat flux limit in the deep core

CFLME_SOL `real`, limit to multiplier to the electron heat flux limit in the far SOL

CFLME_TANH_A `real`, radial location, measured (in m) as a radial distance along the OMP away from the separatrix, for the center of the tanh profile multiplying the electron heat flux limit

CFLME_TANH_B `real`, radial width, measured (in m) as a radial distance along the OMP, for the tanh profile multiplying the electron heat flux limit

CFLMI_CORE `real`, limit to multiplier to the ion heat flux limit in the deep core

CFLMI_SOL `real`, limit to multiplier to the ion heat flux limit in the far SOL

CFLMI_TANH_A `real`, radial location, measured (in m) as a radial distance along the OMP away from the separatrix, for the center of the tanh profile multiplying the ion heat flux limit

CFLMI_TANH_B `real`, radial width, measured (in m) as a radial distance along the OMP, for the tanh profile multiplying the ion heat flux limit

CFLMV_CORE `real`, limit to multiplier to the viscosity heat flux limit in the deep core

CFLMV_SOL `real`, limit to multiplier to the viscosity heat flux limit in the far SOL

CFLMV_TANH_A `real`, radial location, measured (in m) as a radial distance along the OMP away from the separatrix, for the center of the tanh profile multiplying the viscosity heat flux limit

CFLMV_TANH_B `real`, radial width, measured (in m) as a radial distance along the OMP, for the tanh profile multiplying the viscosity heat flux limit

TRANSPORT_FILENAME `filename`, the name of a new transport input file that will be read based on a condition determined by mod (TIM,TRANSPORT_TIME_MOD) >TRANSPORT_TIME_SWITCH

TRANSPORT_TIME_MOD `real`, the basic time periodicity used to set time varying transport coefficients, in s

TRANSPORT_TIME_SWITCH `real`, the basic time at which a switch is to be made to a new transport input file, in s

See also section 3.5.5 for an alternative description of the transport coefficients which allows for radial variation.

It is important to note that the settings from `b2.transport.parameters` will **NOT** be applied to neutral species!

For the electron, ion, and viscous heat flux limit coefficients, if the main multiplier is not specified but `CFLM*_CORE` and `CFLM*_SOL` are set to equal non-zero values, their value will be used. If `CFLM*_CORE` and `CFLM*_SOL` are different and non-zero, then a radial tanh profile, centered at radial OMP position `CFLM*_TANH_A` (in m) and of width `CFLM*_TANH_B` is applied. The OMP reference is set to the poloidal index given by `b2news_fac_ref`, or, if not specified, by `b2mwti_jxa`. The flux limit coefficients in the PFR are set to a constant value equal to that at the separatrix. For all flux limit multipliers, setting them to zero turns off the corresponding flux limiter. Otherwise, the multipliers must be positive.

As stated in the `b2cmpt` block description (Section 3.2.1), for numerical stability reasons, mechanisms must exist to provide some particle transport and establish radial currents. Therefore, the user must not, when determining the transport coefficients by use of the `b2.transport.parameters` file, set both `parm_dna` and `parm_dpa` simultaneously to zero when solving density equations. Similarly, `parm_sig` and `parm_alf` must not simultaneously be set to zero when solving the potential equations without drift current terms on. The code will fail a transport coefficients test in `b2tral` otherwise.

3.5.4 b2.numerics.parameters

See also Appendix A.5 for a more up-to-date description.

DTCO `real array, length NS * (NREG+1)`, the timestep multiplier for the continuity equation of species `IS` in region `IREG`

DTMO `real array, length NS * (NREG+1)`, the timestep multiplier for the parallel momentum equation of species `IS` in region `IREG`

DTEE `real array, length (NREG+1)`, the timestep multiplier for the electron energy equation in region `IREG`

DTEI `real array, length (NREG+1)`, the timestep multiplier for the ion energy equation in region `IREG`

DTCO_SOL `real array, length NS`, additional timestep multiplier for the continuity equation of species `IS` in the far SOL ($iy \geq iy_SOL_dtXX$)

DTMO_SOL `real array, length NS`, additional timestep multiplier for the parallel momentum equation of species `IS` in the far SOL ($iy \geq iy_SOL_dtXX$)

DTEE_SOL `real`, additional timestep multiplier for the electron energy equation in the far SOL ($iy \geq iy_SOL_dtXX$)

DTEI_SOL `real`, additional timestep multiplier for the ion energy equation in the far SOL ($iy \geq iy_SOL_dtXX$)

IY_SOL_DTXX `integer`, index of the first radial ring where the far SOL starts

SOLVECO `logical array, length NS * (NREG+1)`, indicates whether the continuity equations are to be solved in each region

SOLVEMO `logical array, length NS * (NREG+1)`, indicates whether the parallel momentum equations are to be solved in each region

SOLVEMT `logical array, length (NREG+1)`, indicates whether the total parallel momentum equation is to be solved in each region

SOLVEPO `logical array, length (NREG+1)`, indicates whether the potential equation is to be solved in each region

SOLVEEE `logical array, length (NREG+1)`, indicates whether the electron heat equation is to be solved in each region

SOLVEEI `logical array, length (NREG+1)`, indicates whether the ion heat equation is to be solved in each region

SOLVEET `logical array, length (NREG+1)`, indicates whether the total heat equation is to be solved in each region

TIME_FACTOR_REQUIRED `real`, related to local source re-scaling (see routine `b2srsm.F`)

CORR_CORE_DN `real array, length NS`, damping parameter for speed-up of the density equations in the core when drifts are activated (does not apply to neutral species)

CORR_CORE_DT `real`, damping parameter for speed-up of the temperature equations in the core when drifts are activated

CORE_DT_SUPPRESSION `real`, multiplier to the timestep for the innermost core ring of cells

CORE_DT_FACTOR `real`, multiplier to the timestep for successive core rings of cells

ADD_TE_CORR_TO_PO `logical array, length (NREG+1)`, indicates whether the electron temperature correction is added to the electric potential correction (`.true.` by default)

When referring to array length `(NREG+1)` above, we mean range `[0:NREG]` in the code, while `NS` refers to `[0:NS-1]`.

When running in dynamic mode (`b2mndt_style.eq.2`), the total momentum and energy equations are not solved, and the timestep multipliers should not be used, unless their product with the under-relaxation parameters is 1 (as the code defaults provide).

The region indices are defined as follows for a SN case:

0 Dead regions (unused cells)

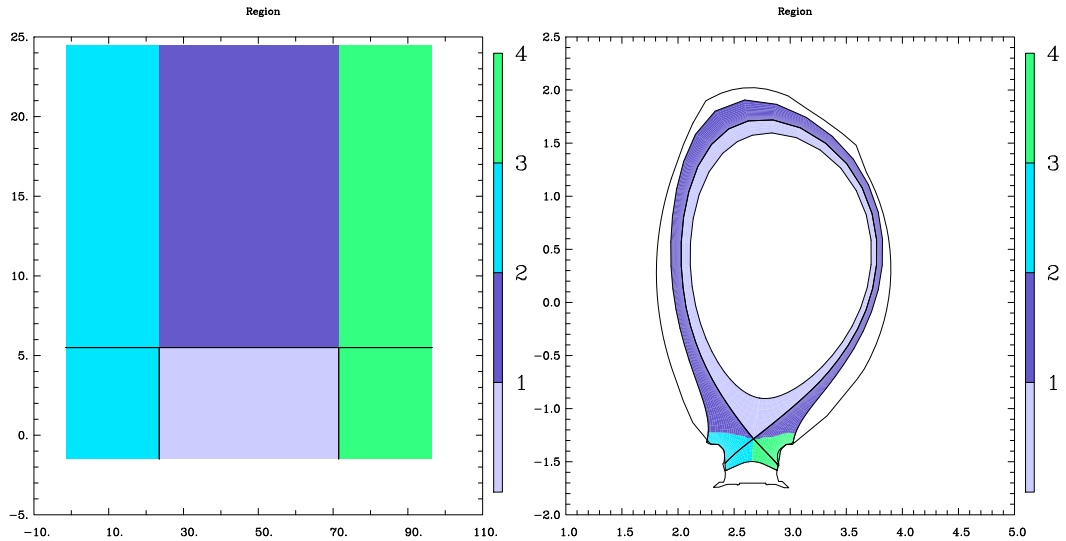
1 Core

2 SOL

3 Western Divertor (Inner for LSN, Outer for USN)

4 Eastern Divertor (Outer for LSN, Inner for USN)

The extent of those regions in both computational and physical space is shown in Fig. 3.2. For double-null configurations, the regions follow a similar numbering scheme (modulo 4) as shown in Fig. 3.3. Details for other geometries can be found in the `init_region` routine of `b2agfs.F` and in Appendix I.



(a) Computational domain (b) Physics domain

Figure 3.2: Region numbers for single-null geometries.

3.5.5 b2.transport.inputfile

See also Appendix A.5 for a more up-to-date description.

This allows for a radial variation ... for a simpler option see section 3.5.3.

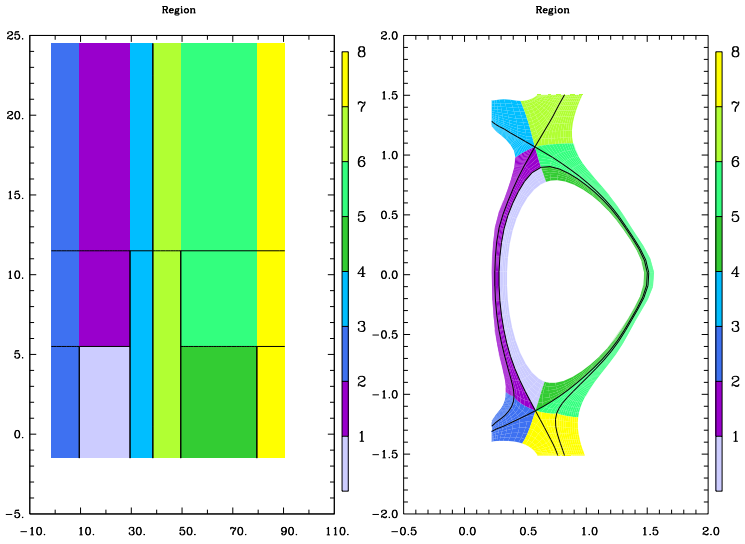
If `'b2tqna_inputfile'` is set to `'1'` in `b2mn.dat` then the transport coefficients are read in from the namelist in `b2.transport.inputfile` instead of being taken from the file created by `b2ah` and possibly modified in `b2mn.dat`. This then allows one to set radially varying profiles of the transport coefficients, and to achieve time dependence in these transport coefficients by triggering a read of a new file (as for `b2.transport.parameters`). The transport coefficients are specified by the `transport` namelist as described below.

The input format for these files is somewhat arcane (the relevant subroutine is `transport_input`, to be found in `b2mod_input_profile.F`) and an example, without the modification for reading in a new file triggered by time, which is the same as for the `b2.transport.parameters`, is included below.

The basic idea is to specify pairs of numbers which give the position and value of the relevant transport coefficients.

`tdata` has four indices:

The value of the first index $= 1$ means position in space, $= 2$ means the value.



(a) Computational domain (b) Physics domain

Figure 3.3: Region numbers for double-null geometries.

The second is used to index the various positions (6 positions in the example below, specified by `ndata`). The total number of positions should not exceed $ny + 2$, as having transport coefficient profiles finer than the plasma grid resolution is overkill.

The third index of `tdata` relates to the kind of transport coefficient being specified, where:

kind_coeff.eq.1 ⇒ **dna0** particle density-driven diffusivity $D_{\perp}^{na}$

kind_coeff.eq.2 ⇒ **dpa0** particle pressure-driven diffusivity $D_{\perp}^{pa}$

kind_coeff.eq.3 ⇒ **hcib** ion thermal anomalous diffusivity χ_i

kind_coeff.eq.4 ⇒ **hce0** electron thermal anomalous diffusivity χ_e

kind_coeff.eq.5 ⇒ **vla0_x** x-component of the anomalous "pinch" velocity

kind_coeff.eq.6 ⇒ **vla0_y** y-component of the anomalous "pinch" velocity

kind_coeff.eq.7 ⇒ **vsa0** anomalous viscosity

kind_coeff.eq.8 ⇒ **sig0** anomalous radial electrical conductivity

kind_coeff.eq.9 ⇒ **alf0** anomalous radial thermo-electric coefficient

The fourth index is the species index.

Again, as in Section 3.2.1, it is important, for numerical stability reasons, to provide mechanisms by which some particle transport and radial current can be established. If the transport coefficients are given here, this means that, first, the `dna0` and `dpa0` arrays cannot be simultaneously zero when solving density equations. Second, the arrays `sig0` and `alf0` cannot be simultaneously zero when solving the potential equations without drift current terms on. The code will fail a transport coefficients test in `b2tral` otherwise.

The relevant routine was written by

```
!**********************************************************************
!*     subroutine for reading in experimental transport coefficients from *
!*     the midplane and calculating transport coefficients               *
!*     for B2.5 (written by Heimo Buerbaumer 1999)                       *
```

```
!*      Inputfile:  b2.transport.inputfile                                  *
!*************************************************************************
```

Some additional description of the relevant variables is as follows:

ndata(1,i,j) where i is an index for the type of transport coefficient being defined (see table above) and j is a species index (0 for atoms and 1 for ions in a D-only case, for instance, and irrelevant (use 1) for electron coefficients). `ndata` is the number of data points where the value of the transport coefficients is given.

addspec(k,i,j) are the indices of any additional species whose transport coefficients are also described by the same set of numbers. k is a dummy that runs from 1 to the number of additional species described.

tdata(1,m,i,j) are the radial positions (measured in metres) of the `ndata(1,i,j)` points, where m runs from 1 to `ndata(1,i,j)`. By default, the radial positions are measured from the separatrix outward, along the outer midplane, but this can be changed by making use of the $ixref$ and $iyref$ switches, respectively, to modify the poloidal location of the profile and the radial position from which the distances are measured.

tdata(2,m,i,j) are the corresponding values of the coefficients.

no_pflux is a logical that, if `.true.`, does not apply these transport coefficients to the private flux region(s).

no_div is a logical that, if `.true.`, does not apply these transport coefficients to the divertor regions.

So in the example file below, three transport coefficients profiles are given, i.e. in order, $D_\perp^{na}$, χ_i and χ_e, defined with 6 points each, and setting a transport barrier around the separatrix. Note that the radial locations of those points are quite independent of where the mesh is. The code interpolates the transport values onto the mesh, so one needs not worry about making sure that a point is at an exact cell location. It is recommended that the first and last points be a large distance away from the separatrix, typically well outside the computational domain. This ensures that the code is still interpolating and not using (possibly different) default values when computing coefficients for cells near the edges of the mesh. The interpolation the code uses is a monotonicity-preserving cubic Hermite polynomial spline from the NAG library, or a simple linear interpolation if the NAG library is unavailable at compilation time. For computational cells located outside the range provided by the `tdata(1,m,i,j)` positions, the code will use whatever 'default' values were declared previously. As a reminder, the order of precedence of the transport coefficients settings is:

`b2ah.dat < b2mn.dat < b2.tranport.parameters < b2.transport.inputfile`

The following contains an example where the flag `no_pflux` is used to switch off the setting of transport coefficients in the private flux region.

```
&TRANSPORT
ndata(1, 1 , 1 )= 6 ,
tdata(1, 1 , 1 , 1 )= -4.00E-02 , tdata(2, 1 , 1 , 1 )= 0.50 ,
tdata(1, 2 , 1 , 1 )= -1.50E-02 , tdata(2, 2 , 1 , 1 )= 0.50 ,
tdata(1, 3 , 1 , 1 )= -1.40E-02 , tdata(2, 3 , 1 , 1 )= 0.05 ,
tdata(1, 4 , 1 , 1 )=  5.00E-03 , tdata(2, 4 , 1 , 1 )= 0.05 ,
tdata(1, 5 , 1 , 1 )=  5.50E-03 , tdata(2, 5 , 1 , 1 )= 0.50 ,
tdata(1, 6 , 1 , 1 )=  2.50E-02 , tdata(2, 6 , 1 , 1 )= 0.50 ,
ndata(1, 3 , 1 )= 6 ,
tdata(1, 1 , 3 , 1 )= -4.00E-02 , tdata(2, 1 , 3 , 1 )= 0.70 ,
tdata(1, 2 , 3 , 1 )= -1.50E-02 , tdata(2, 2 , 3 , 1 )= 0.70 ,
tdata(1, 3 , 3 , 1 )= -1.40E-02 , tdata(2, 3 , 3 , 1 )= 0.07 ,
tdata(1, 4 , 3 , 1 )=  5.00E-03 , tdata(2, 4 , 3 , 1 )= 0.07 ,
tdata(1, 5 , 3 , 1 )=  5.50E-03 , tdata(2, 5 , 3 , 1 )= 0.70 ,
tdata(1, 6 , 3 , 1 )=  2.50E-02 , tdata(2, 6 , 3 , 1 )= 0.70 ,
ndata(1, 4 , 1 )= 6 ,
tdata(1, 1 , 4 , 1 )= -4.00E-02 , tdata(2, 1 , 4 , 1 )= 0.70 ,
tdata(1, 2 , 4 , 1 )= -1.50E-02 , tdata(2, 2 , 4 , 1 )= 0.70 ,
tdata(1, 3 , 4 , 1 )= -1.40E-02 , tdata(2, 3 , 4 , 1 )= 0.07 ,
tdata(1, 4 , 4 , 1 )=  5.00E-03 , tdata(2, 4 , 4 , 1 )= 0.07 ,
tdata(1, 5 , 4 , 1 )=  5.50E-03 , tdata(2, 5 , 4 , 1 )= 0.70 ,
tdata(1, 6 , 4 , 1 )=  2.50E-02 , tdata(2, 6 , 4 , 1 )= 0.70 ,
no_pflux=.true.
/
```

3.5.6 b2.sources.profile

See also Appendix A.5 for a more up-to-date description.

One can also set radially dependent source profiles, in much the same way as with `b2.transport.inputfile`, with the `b2.sources.profile` input file, accessed by setting `'b2sral_inputfile'` to `'1'`. The logic is the same in terms of fixing radial positions, applicable species, etc...

The coefficient types go as:

1. Particle sources `sna`
2. Parallel momentum sources `smo`
3. Electron heat sources `she`
4. Ion heat sources `shi`
5. Electric charge sources `sch`
6. Electric particle source (non-ambipolar) `sne`

It is also possible to multiply these radial profiles by an axial form factor. This axial multiplier is set in the same way as the poloidal profile.

The description of the variables found in `b2.sources.profile` is as follows:

nsdata(1,i,j) where `i` is an index for the type of source being defined (see table above) and `j` is a species index (`0` for atoms and `1` for ions in a D-only case, for instance, and irrelevant (use `1`) for electron sources). `nsdata` is the number of data points where the strength of the source is given.

sdata(1,m,i,j) are the radial positions (measured in metres) of the `nsdata(1,i,j)` points, where `m` runs from `1` to `nsdata(1,i,j)`. By default, the radial positions are measured from the separatrix outward, along the outer midplane, but this can be changed by making use of the `ixref` and `iyref` switches, respectively, to modify the poloidal location of the profile and the radial position from which the distances are measured.

sdata(2,m,i,j) are the corresponding strengths of the sources.

nxdata(k,i,j) where `i` is an index for the type of source being defined (see table above) and `j` is a species index (`0` for atoms and `1` for ions in a D-only case, for instance, and irrelevant (use `1`) for electron sources). `nxdata` is the number of data points where the value of the axial multiplier to the source is given. If `k` is equal to `1`, the data is to be understood as function of normalized connection length. If `k` is equal to `2`, then the axial profile is given in terms of the normalized computational coordinate (running from `0` to `1` as `ix` runs from `-1` to `nx`).

xdata(1,m,i,j) are the axial positions (normalised to the total length) of the `nxdata(1,i,j)` points, where `m` runs from `1` to `nxdata(1,i,j)`. By default, the connection length used is the one measured just inside the separatrix, with the origin located at the outer midplane, but this can be changed by making use of the `ixref` and `iyref` switches, respectively, to modify the poloidal location of the origin and the radial position along which the distances are measured. In case the reference location is chosen inside a closed field line region, the axial multiplier is set to zero everywhere on the open field lines.

xdata(2,m,i,j) are the corresponding values of the source strengths.

divheat is the total volumetric ion heat source (in W/m^3) spread equally throughout the divertor volume.

ixref_profile `integer`, the poloidal location along which the sources radial profile is measured. Defaults to `-2`, in which case the value from `set_transport_ixref` is used.

iyref_profile `integer`, the radial location along which the sources axial profile is measured. Defaults to `-2`, in which case the value from `set_transport_iyref` is used.

sources_filename `filename`, the name of a new sources input file that will be read based on a condition determined by mod (`tim`,`sources_time_mod`) >`sources_time_switch`.

sources_time_mod `real`, the basic time periodicity used to set time varying sources, in s.

sources_time_switch `real`, the basic time at which a switch is to be made to a new sources input file, in s.

The results of these imported source profiles can be visualised in **b2plot** with the **exna**, **exmo**, **exhe**, **exhi**, **exch**, and **exne** commands. An example of input file is given below:

```
&profile
nsdata(1,1,0)=42,
sdata(1,1,1,0)=-0.4165, sdata(2,1,1,0)=0.9895E+20,
sdata(1,2,1,0)=-0.4029, sdata(2,2,1,0)=0.9582E+20,
sdata(1,3,1,0)=-0.3900, sdata(2,3,1,0)=0.9000E+20,
sdata(1,4,1,0)=-0.3775, sdata(2,4,1,0)=0.8314E+20,
sdata(1,5,1,0)=-0.3652, sdata(2,5,1,0)=0.7621E+20,
sdata(1,6,1,0)=-0.3532, sdata(2,6,1,0)=0.6930E+20,
sdata(1,7,1,0)=-0.3416, sdata(2,7,1,0)=0.6253E+20,
sdata(1,8,1,0)=-0.3300, sdata(2,8,1,0)=0.5615E+20,
sdata(1,9,1,0)=-0.3185, sdata(2,9,1,0)=0.5057E+20,
sdata(1,10,1,0)=-0.3071, sdata(2,10,1,0)=0.4570E+20,
sdata(1,11,1,0)=-0.2955, sdata(2,11,1,0)=0.4189E+20,
sdata(1,12,1,0)=-0.2835, sdata(2,12,1,0)=0.3876E+20,
sdata(1,13,1,0)=-0.2715, sdata(2,13,1,0)=0.3642E+20,
sdata(1,14,1,0)=-0.2594, sdata(2,14,1,0)=0.3455E+20,
sdata(1,15,1,0)=-0.2473, sdata(2,15,1,0)=0.3316E+20,
sdata(1,16,1,0)=-0.2352, sdata(2,16,1,0)=0.3205E+20,
sdata(1,17,1,0)=-0.2232, sdata(2,17,1,0)=0.3122E+20,
sdata(1,18,1,0)=-0.2112, sdata(2,18,1,0)=0.3054E+20,
sdata(1,19,1,0)=-0.1996, sdata(2,19,1,0)=0.3002E+20,
sdata(1,20,1,0)=-0.1881, sdata(2,20,1,0)=0.2960E+20,
sdata(1,21,1,0)=-0.1768, sdata(2,21,1,0)=0.2925E+20,
sdata(1,22,1,0)=-0.1658, sdata(2,22,1,0)=0.2895E+20,
sdata(1,23,1,0)=-0.1550, sdata(2,23,1,0)=0.2867E+20,
sdata(1,24,1,0)=-0.1445, sdata(2,24,1,0)=0.2838E+20,
sdata(1,25,1,0)=-0.1342, sdata(2,25,1,0)=0.2805E+20,
sdata(1,26,1,0)=-0.1243, sdata(2,26,1,0)=0.2769E+20,
sdata(1,27,1,0)=-0.1145, sdata(2,27,1,0)=0.2728E+20,
sdata(1,28,1,0)=-0.1050, sdata(2,28,1,0)=0.2683E+20,
sdata(1,29,1,0)=-0.0958, sdata(2,29,1,0)=0.2632E+20,
sdata(1,30,1,0)=-0.0868, sdata(2,30,1,0)=0.2576E+20,
sdata(1,31,1,0)=-0.0780, sdata(2,31,1,0)=0.2511E+20,
sdata(1,32,1,0)=-0.0694, sdata(2,32,1,0)=0.2438E+20,
sdata(1,33,1,0)=-0.0612, sdata(2,33,1,0)=0.2354E+20,
sdata(1,34,1,0)=-0.0530, sdata(2,34,1,0)=0.2258E+20,
sdata(1,35,1,0)=-0.0451, sdata(2,35,1,0)=0.2144E+20,
sdata(1,36,1,0)=-0.0374, sdata(2,36,1,0)=0.2007E+20,
sdata(1,37,1,0)=-0.0298, sdata(2,37,1,0)=0.1831E+20,
sdata(1,38,1,0)=-0.0224, sdata(2,38,1,0)=0.1605E+20,
sdata(1,39,1,0)=-0.0153, sdata(2,39,1,0)=0.1302E+20,
sdata(1,40,1,0)=-0.0082, sdata(2,40,1,0)=0.9238E+19,
sdata(1,41,1,0)=-0.0014, sdata(2,41,1,0)=0.4474E+19,
sdata(1,42,1,0)= 0.0035, sdata(2,42,1,0)=0.0000E+00,
nsdata(1,1,1)=42,
sdata(1,1,1,1)=-0.4165, sdata(2,1,1,1)=0.7723E+20,
sdata(1,2,1,1)=-0.4029, sdata(2,2,1,1)=0.7472E+20,
sdata(1,3,1,1)=-0.3900, sdata(2,3,1,1)=0.7003E+20,
sdata(1,4,1,1)=-0.3775, sdata(2,4,1,1)=0.6447E+20,
sdata(1,5,1,1)=-0.3652, sdata(2,5,1,1)=0.5872E+20,
sdata(1,6,1,1)=-0.3532, sdata(2,6,1,1)=0.5292E+20,
sdata(1,7,1,1)=-0.3416, sdata(2,7,1,1)=0.4712E+20,
sdata(1,8,1,1)=-0.3300, sdata(2,8,1,1)=0.4158E+20,
sdata(1,9,1,1)=-0.3185, sdata(2,9,1,1)=0.3667E+20,
sdata(1,10,1,1)=-0.3071, sdata(2,10,1,1)=0.3234E+20,
sdata(1,11,1,1)=-0.2955, sdata(2,11,1,1)=0.2894E+20,
sdata(1,12,1,1)=-0.2835, sdata(2,12,1,1)=0.2612E+20,
sdata(1,13,1,1)=-0.2715, sdata(2,13,1,1)=0.2400E+20,
sdata(1,14,1,1)=-0.2594, sdata(2,14,1,1)=0.2227E+20,
sdata(1,15,1,1)=-0.2473, sdata(2,15,1,1)=0.2094E+20,
```

```
sdata(1,16,1,1)=-0.2352, sdata(2,16,1,1)=0.1983E+20,
sdata(1,17,1,1)=-0.2232, sdata(2,17,1,1)=0.1898E+20,
sdata(1,18,1,1)=-0.2112, sdata(2,18,1,1)=0.1828E+20,
sdata(1,19,1,1)=-0.1996, sdata(2,19,1,1)=0.1772E+20,
sdata(1,20,1,1)=-0.1881, sdata(2,20,1,1)=0.1725E+20,
sdata(1,21,1,1)=-0.1768, sdata(2,21,1,1)=0.1684E+20,
sdata(1,22,1,1)=-0.1658, sdata(2,22,1,1)=0.1646E+20,
sdata(1,23,1,1)=-0.1550, sdata(2,23,1,1)=0.1609E+20,
sdata(1,24,1,1)=-0.1445, sdata(2,24,1,1)=0.1574E+20,
sdata(1,25,1,1)=-0.1342, sdata(2,25,1,1)=0.1536E+20,
sdata(1,26,1,1)=-0.1243, sdata(2,26,1,1)=0.1499E+20,
sdata(1,27,1,1)=-0.1145, sdata(2,27,1,1)=0.1460E+20,
sdata(1,28,1,1)=-0.1050, sdata(2,28,1,1)=0.1421E+20,
sdata(1,29,1,1)=-0.0958, sdata(2,29,1,1)=0.1381E+20,
sdata(1,30,1,1)=-0.0868, sdata(2,30,1,1)=0.1339E+20,
sdata(1,31,1,1)=-0.0780, sdata(2,31,1,1)=0.1294E+20,
sdata(1,32,1,1)=-0.0694, sdata(2,32,1,1)=0.1245E+20,
sdata(1,33,1,1)=-0.0612, sdata(2,33,1,1)=0.1191E+20,
sdata(1,34,1,1)=-0.0530, sdata(2,34,1,1)=0.1133E+20,
sdata(1,35,1,1)=-0.0451, sdata(2,35,1,1)=0.1071E+20,
sdata(1,36,1,1)=-0.0374, sdata(2,36,1,1)=0.9985E+19,
sdata(1,37,1,1)=-0.0298, sdata(2,37,1,1)=0.9087E+19,
sdata(1,38,1,1)=-0.0224, sdata(2,38,1,1)=0.7958E+19,
sdata(1,39,1,1)=-0.0153, sdata(2,39,1,1)=0.6455E+19,
sdata(1,40,1,1)=-0.0082, sdata(2,40,1,1)=0.4582E+19,
sdata(1,41,1,1)=-0.0014, sdata(2,41,1,1)=0.2220E+19,
sdata(1,42,1,1)= 0.0035, sdata(2,42,1,1)=0.0000E+00,
nsdata(1,2,0)=42,
sdata(1,1,2,0)=-0.4165, sdata(2,1,2,0)=0.7080E+00,
sdata(1,2,2,0)=-0.4029, sdata(2,2,2,0)=0.6405E+00,
sdata(1,3,2,0)=-0.3900, sdata(2,3,2,0)=0.5226E+00,
sdata(1,4,2,0)=-0.3775, sdata(2,4,2,0)=0.4044E+00,
sdata(1,5,2,0)=-0.3652, sdata(2,5,2,0)=0.3377E+00,
sdata(1,6,2,0)=-0.3532, sdata(2,6,2,0)=0.2951E+00,
sdata(1,7,2,0)=-0.3416, sdata(2,7,2,0)=0.2676E+00,
sdata(1,8,2,0)=-0.3300, sdata(2,8,2,0)=0.2466E+00,
sdata(1,9,2,0)=-0.3185, sdata(2,9,2,0)=0.2285E+00,
sdata(1,10,2,0)=-0.3071, sdata(2,10,2,0)=0.2115E+00,
sdata(1,11,2,0)=-0.2955, sdata(2,11,2,0)=0.1942E+00,
sdata(1,12,2,0)=-0.2835, sdata(2,12,2,0)=0.1772E+00,
sdata(1,13,2,0)=-0.2715, sdata(2,13,2,0)=0.1614E+00,
sdata(1,14,2,0)=-0.2594, sdata(2,14,2,0)=0.1464E+00,
sdata(1,15,2,0)=-0.2473, sdata(2,15,2,0)=0.1327E+00,
sdata(1,16,2,0)=-0.2352, sdata(2,16,2,0)=0.1205E+00,
sdata(1,17,2,0)=-0.2232, sdata(2,17,2,0)=0.1110E+00,
sdata(1,18,2,0)=-0.2112, sdata(2,18,2,0)=0.1023E+00,
sdata(1,19,2,0)=-0.1996, sdata(2,19,2,0)=0.9300E-01,
sdata(1,20,2,0)=-0.1881, sdata(2,20,2,0)=0.8348E-01,
sdata(1,21,2,0)=-0.1768, sdata(2,21,2,0)=0.7411E-01,
sdata(1,22,2,0)=-0.1658, sdata(2,22,2,0)=0.6554E-01,
sdata(1,23,2,0)=-0.1550, sdata(2,23,2,0)=0.5882E-01,
sdata(1,24,2,0)=-0.1445, sdata(2,24,2,0)=0.5295E-01,
sdata(1,25,2,0)=-0.1342, sdata(2,25,2,0)=0.4763E-01,
sdata(1,26,2,0)=-0.1243, sdata(2,26,2,0)=0.4266E-01,
sdata(1,27,2,0)=-0.1145, sdata(2,27,2,0)=0.3789E-01,
sdata(1,28,2,0)=-0.1050, sdata(2,28,2,0)=0.3327E-01,
sdata(1,29,2,0)=-0.0958, sdata(2,29,2,0)=0.2855E-01,
sdata(1,30,2,0)=-0.0868, sdata(2,30,2,0)=0.2438E-01,
sdata(1,31,2,0)=-0.0780, sdata(2,31,2,0)=0.2137E-01,
sdata(1,32,2,0)=-0.0694, sdata(2,32,2,0)=0.1900E-01,
sdata(1,33,2,0)=-0.0612, sdata(2,33,2,0)=0.1699E-01,
sdata(1,34,2,0)=-0.0530, sdata(2,34,2,0)=0.1498E-01,
sdata(1,35,2,0)=-0.0451, sdata(2,35,2,0)=0.1257E-01,
sdata(1,36,2,0)=-0.0374, sdata(2,36,2,0)=0.1022E-01,
sdata(1,37,2,0)=-0.0298, sdata(2,37,2,0)=0.8162E-02,
sdata(1,38,2,0)=-0.0224, sdata(2,38,2,0)=0.6258E-02,
sdata(1,39,2,0)=-0.0153, sdata(2,39,2,0)=0.4355E-02,
```

```
sdata(1,40,2,0)=-0.0082, sdata(2,40,2,0)=0.2603E-02,
sdata(1,41,2,0)=-0.0014, sdata(2,41,2,0)=0.1151E-02,
sdata(1,42,2,0)= 0.0035, sdata(2,42,2,0)=0.0000E+00,
nsdata(1,2,1)=42,
sdata(1,1,2,1)=-0.4165, sdata(2,1,2,1)=0.7080E+00,
sdata(1,2,2,1)=-0.4029, sdata(2,2,2,1)=0.6405E+00,
sdata(1,3,2,1)=-0.3900, sdata(2,3,2,1)=0.5226E+00,
sdata(1,4,2,1)=-0.3775, sdata(2,4,2,1)=0.4044E+00,
sdata(1,5,2,1)=-0.3652, sdata(2,5,2,1)=0.3377E+00,
sdata(1,6,2,1)=-0.3532, sdata(2,6,2,1)=0.2951E+00,
sdata(1,7,2,1)=-0.3416, sdata(2,7,2,1)=0.2676E+00,
sdata(1,8,2,1)=-0.3300, sdata(2,8,2,1)=0.2466E+00,
sdata(1,9,2,1)=-0.3185, sdata(2,9,2,1)=0.2285E+00,
sdata(1,10,2,1)=-0.3071, sdata(2,10,2,1)=0.2115E+00,
sdata(1,11,2,1)=-0.2955, sdata(2,11,2,1)=0.1942E+00,
sdata(1,12,2,1)=-0.2835, sdata(2,12,2,1)=0.1772E+00,
sdata(1,13,2,1)=-0.2715, sdata(2,13,2,1)=0.1614E+00,
sdata(1,14,2,1)=-0.2594, sdata(2,14,2,1)=0.1464E+00,
sdata(1,15,2,1)=-0.2473, sdata(2,15,2,1)=0.1327E+00,
sdata(1,16,2,1)=-0.2352, sdata(2,16,2,1)=0.1205E+00,
sdata(1,17,2,1)=-0.2232, sdata(2,17,2,1)=0.1110E+00,
sdata(1,18,2,1)=-0.2112, sdata(2,18,2,1)=0.1023E+00,
sdata(1,19,2,1)=-0.1996, sdata(2,19,2,1)=0.9300E-01,
sdata(1,20,2,1)=-0.1881, sdata(2,20,2,1)=0.8348E-01,
sdata(1,21,2,1)=-0.1768, sdata(2,21,2,1)=0.7411E-01,
sdata(1,22,2,1)=-0.1658, sdata(2,22,2,1)=0.6554E-01,
sdata(1,23,2,1)=-0.1550, sdata(2,23,2,1)=0.5882E-01,
sdata(1,24,2,1)=-0.1445, sdata(2,24,2,1)=0.5295E-01,
sdata(1,25,2,1)=-0.1342, sdata(2,25,2,1)=0.4763E-01,
sdata(1,26,2,1)=-0.1243, sdata(2,26,2,1)=0.4266E-01,
sdata(1,27,2,1)=-0.1145, sdata(2,27,2,1)=0.3789E-01,
sdata(1,28,2,1)=-0.1050, sdata(2,28,2,1)=0.3327E-01,
sdata(1,29,2,1)=-0.0958, sdata(2,29,2,1)=0.2855E-01,
sdata(1,30,2,1)=-0.0868, sdata(2,30,2,1)=0.2438E-01,
sdata(1,31,2,1)=-0.0780, sdata(2,31,2,1)=0.2137E-01,
sdata(1,32,2,1)=-0.0694, sdata(2,32,2,1)=0.1900E-01,
sdata(1,33,2,1)=-0.0612, sdata(2,33,2,1)=0.1699E-01,
sdata(1,34,2,1)=-0.0530, sdata(2,34,2,1)=0.1498E-01,
sdata(1,35,2,1)=-0.0451, sdata(2,35,2,1)=0.1257E-01,
sdata(1,36,2,1)=-0.0374, sdata(2,36,2,1)=0.1022E-01,
sdata(1,37,2,1)=-0.0298, sdata(2,37,2,1)=0.8162E-02,
sdata(1,38,2,1)=-0.0224, sdata(2,38,2,1)=0.6258E-02,
sdata(1,39,2,1)=-0.0153, sdata(2,39,2,1)=0.4355E-02,
sdata(1,40,2,1)=-0.0082, sdata(2,40,2,1)=0.2603E-02,
sdata(1,41,2,1)=-0.0014, sdata(2,41,2,1)=0.1151E-02,
sdata(1,42,2,1)= 0.0035, sdata(2,42,2,1)=0.0000E+00,
nsdata(1,3,1)=42,
sdata(1,1,3,1)=-0.4165, sdata(2,1,3,1)=0.1570E+07,
sdata(1,2,3,1)=-0.4029, sdata(2,2,3,1)=0.1418E+07,
sdata(1,3,3,1)=-0.3900, sdata(2,3,3,1)=0.1152E+07,
sdata(1,4,3,1)=-0.3775, sdata(2,4,3,1)=0.8862E+06,
sdata(1,5,3,1)=-0.3652, sdata(2,5,3,1)=0.7358E+06,
sdata(1,6,3,1)=-0.3532, sdata(2,6,3,1)=0.6396E+06,
sdata(1,7,3,1)=-0.3416, sdata(2,7,3,1)=0.5760E+06,
sdata(1,8,3,1)=-0.3300, sdata(2,8,3,1)=0.5254E+06,
sdata(1,9,3,1)=-0.3185, sdata(2,9,3,1)=0.4774E+06,
sdata(1,10,3,1)=-0.3071, sdata(2,10,3,1)=0.4298E+06,
sdata(1,11,3,1)=-0.2955, sdata(2,11,3,1)=0.3804E+06,
sdata(1,12,3,1)=-0.2835, sdata(2,12,3,1)=0.3331E+06,
sdata(1,13,3,1)=-0.2715, sdata(2,13,3,1)=0.2943E+06,
sdata(1,14,3,1)=-0.2594, sdata(2,14,3,1)=0.2616E+06,
sdata(1,15,3,1)=-0.2473, sdata(2,15,3,1)=0.2369E+06,
sdata(1,16,3,1)=-0.2352, sdata(2,16,3,1)=0.2167E+06,
sdata(1,17,3,1)=-0.2232, sdata(2,17,3,1)=0.2008E+06,
sdata(1,18,3,1)=-0.2112, sdata(2,18,3,1)=0.1871E+06,
sdata(1,19,3,1)=-0.1996, sdata(2,19,3,1)=0.1757E+06,
sdata(1,20,3,1)=-0.1881, sdata(2,20,3,1)=0.1651E+06,
```

```
sdata(1,21,3,1)=-0.1768, sdata(2,21,3,1)=0.1547E+06,
sdata(1,22,3,1)=-0.1658, sdata(2,22,3,1)=0.1450E+06,
sdata(1,23,3,1)=-0.1550, sdata(2,23,3,1)=0.1362E+06,
sdata(1,24,3,1)=-0.1445, sdata(2,24,3,1)=0.1278E+06,
sdata(1,25,3,1)=-0.1342, sdata(2,25,3,1)=0.1192E+06,
sdata(1,26,3,1)=-0.1243, sdata(2,26,3,1)=0.1107E+06,
sdata(1,27,3,1)=-0.1145, sdata(2,27,3,1)=0.1022E+06,
sdata(1,28,3,1)=-0.1050, sdata(2,28,3,1)=0.9360E+05,
sdata(1,29,3,1)=-0.0958, sdata(2,29,3,1)=0.8460E+05,
sdata(1,30,3,1)=-0.0868, sdata(2,30,3,1)=0.7630E+05,
sdata(1,31,3,1)=-0.0780, sdata(2,31,3,1)=0.6990E+05,
sdata(1,32,3,1)=-0.0694, sdata(2,32,3,1)=0.6460E+05,
sdata(1,33,3,1)=-0.0612, sdata(2,33,3,1)=0.6010E+05,
sdata(1,34,3,1)=-0.0530, sdata(2,34,3,1)=0.5490E+05,
sdata(1,35,3,1)=-0.0451, sdata(2,35,3,1)=0.4780E+05,
sdata(1,36,3,1)=-0.0374, sdata(2,36,3,1)=0.4030E+05,
sdata(1,37,3,1)=-0.0298, sdata(2,37,3,1)=0.3350E+05,
sdata(1,38,3,1)=-0.0224, sdata(2,38,3,1)=0.2690E+05,
sdata(1,39,3,1)=-0.0153, sdata(2,39,3,1)=0.1950E+05,
sdata(1,40,3,1)=-0.0082, sdata(2,40,3,1)=0.1210E+05,
sdata(1,41,3,1)=-0.0014, sdata(2,41,3,1)=0.5500E+04,
sdata(1,42,3,1)= 0.0035, sdata(2,42,3,1)=0.0000E+00,
nsdata(1,4,1)=42,
sdata(1,1,4,1)=-0.4165, sdata(2,1,4,1)=0.1300E+07,
sdata(1,2,4,1)=-0.4029, sdata(2,2,4,1)=0.1179E+07,
sdata(1,3,4,1)=-0.3900, sdata(2,3,4,1)=0.9645E+06,
sdata(1,4,4,1)=-0.3775, sdata(2,4,4,1)=0.7452E+06,
sdata(1,5,4,1)=-0.3652, sdata(2,5,4,1)=0.6223E+06,
sdata(1,6,4,1)=-0.3532, sdata(2,6,4,1)=0.5476E+06,
sdata(1,7,4,1)=-0.3416, sdata(2,7,4,1)=0.5071E+06,
sdata(1,8,4,1)=-0.3300, sdata(2,8,4,1)=0.4841E+06,
sdata(1,9,4,1)=-0.3185, sdata(2,9,4,1)=0.4703E+06,
sdata(1,10,4,1)=-0.3071, sdata(2,10,4,1)=0.4604E+06,
sdata(1,11,4,1)=-0.2955, sdata(2,11,4,1)=0.4513E+06,
sdata(1,12,4,1)=-0.2835, sdata(2,12,4,1)=0.4421E+06,
sdata(1,13,4,1)=-0.2715, sdata(2,13,4,1)=0.4286E+06,
sdata(1,14,4,1)=-0.2594, sdata(2,14,4,1)=0.4104E+06,
sdata(1,15,4,1)=-0.2473, sdata(2,15,4,1)=0.3869E+06,
sdata(1,16,4,1)=-0.2352, sdata(2,16,4,1)=0.3636E+06,
sdata(1,17,4,1)=-0.2232, sdata(2,17,4,1)=0.3458E+06,
sdata(1,18,4,1)=-0.2112, sdata(2,18,4,1)=0.3279E+06,
sdata(1,19,4,1)=-0.1996, sdata(2,19,4,1)=0.3047E+06,
sdata(1,20,4,1)=-0.1881, sdata(2,20,4,1)=0.2791E+06,
sdata(1,21,4,1)=-0.1768, sdata(2,21,4,1)=0.2524E+06,
sdata(1,22,4,1)=-0.1658, sdata(2,22,4,1)=0.2272E+06,
sdata(1,23,4,1)=-0.1550, sdata(2,23,4,1)=0.2064E+06,
sdata(1,24,4,1)=-0.1445, sdata(2,24,4,1)=0.1877E+06,
sdata(1,25,4,1)=-0.1342, sdata(2,25,4,1)=0.1699E+06,
sdata(1,26,4,1)=-0.1243, sdata(2,26,4,1)=0.1526E+06,
sdata(1,27,4,1)=-0.1145, sdata(2,27,4,1)=0.1358E+06,
sdata(1,28,4,1)=-0.1050, sdata(2,28,4,1)=0.1195E+06,
sdata(1,29,4,1)=-0.0958, sdata(2,29,4,1)=0.1037E+06,
sdata(1,30,4,1)=-0.0868, sdata(2,30,4,1)=0.8930E+05,
sdata(1,31,4,1)=-0.0780, sdata(2,31,4,1)=0.7750E+05,
sdata(1,32,4,1)=-0.0694, sdata(2,32,4,1)=0.6740E+05,
sdata(1,33,4,1)=-0.0612, sdata(2,33,4,1)=0.5880E+05,
sdata(1,34,4,1)=-0.0530, sdata(2,34,4,1)=0.5050E+05,
sdata(1,35,4,1)=-0.0451, sdata(2,35,4,1)=0.4170E+05,
sdata(1,36,4,1)=-0.0374, sdata(2,36,4,1)=0.3320E+05,
sdata(1,37,4,1)=-0.0298, sdata(2,37,4,1)=0.2590E+05,
sdata(1,38,4,1)=-0.0224, sdata(2,38,4,1)=0.1930E+05,
sdata(1,39,4,1)=-0.0153, sdata(2,39,4,1)=0.1300E+05,
sdata(1,40,4,1)=-0.0082, sdata(2,40,4,1)=0.7400E+04,
sdata(1,41,4,1)=-0.0014, sdata(2,41,4,1)=0.3200E+04,
sdata(1,42,4,1)= 0.0035, sdata(2,42,4,1)=0.0000E+00,
/
```

3.5.7 b2.neutrals.parameters

See descriptions in Coupled `Eirene` chapter 5 and Appendix A.5.

3.5.8 b2.atomic_physics_rescale.parameters

Contains a series of multipliers to the atomic physics rates provided by `b2ar` in `b2frates`. These are:

RESCALE_SA `real*8 array, length NS`, contains the scaling factor per species to be applied to the ionisation rates

RESCALE_RA `real*8 array, length NS`, contains the scaling factor per species to be applied to the recombination rates

RESCALE_QA `real*8 array, length NS`, contains the scaling factor per species to be applied to the electron cooling rates

RESCALE_CX `real*8 array, length NS`, contains the scaling factor per species to be applied to the charge exchange rates

RESCALE_RD `real*8 array, length NS`, contains the scaling factor per species to be applied to the line radiation rates

RESCALE_BR `real*8 array, length NS`, contains the scaling factor per species to be applied to the bremsstrahlung radiation rates

See Appendix A.5 for a full description.

3.5.9 b2.user.parameters

See descriptions in Coupled `Eirene` chapter 5 and Appendix A.5.

3.5.10 b2.XXX.parameters

See descriptions in Appendix A.5.

3.5.11 While the code is running

While the code is running and writing into `run.log`, a number of the scripts described in section 2.5 can be run.

`cpu` shows how much CPU time each iteration has taken and is a quick way of monitoring the progress of a run.

```
18:22> cpu
# /scratch/dpc/solps5.0_test_xpb/src/Braams/b2/runs/JET_B2_comparison/50401/57_coarse/2e19_8MW
1 2.129999935626984
2 1.940000295639038
3 1.730000019073486
4 1.719999313354492
5 1.710000038146973
6 1.720000267028809
7 1.690000534057617
8 1.710000038146973
9 1.679999351501465
10 1.700000762939453
11 1.699998855590820
12 1.710000991821289
```

```
13 1.699998855590820
18:22>
```

All of the residuals of a pure deuterium run can be plotted with `resall_D`, figure 3.4.

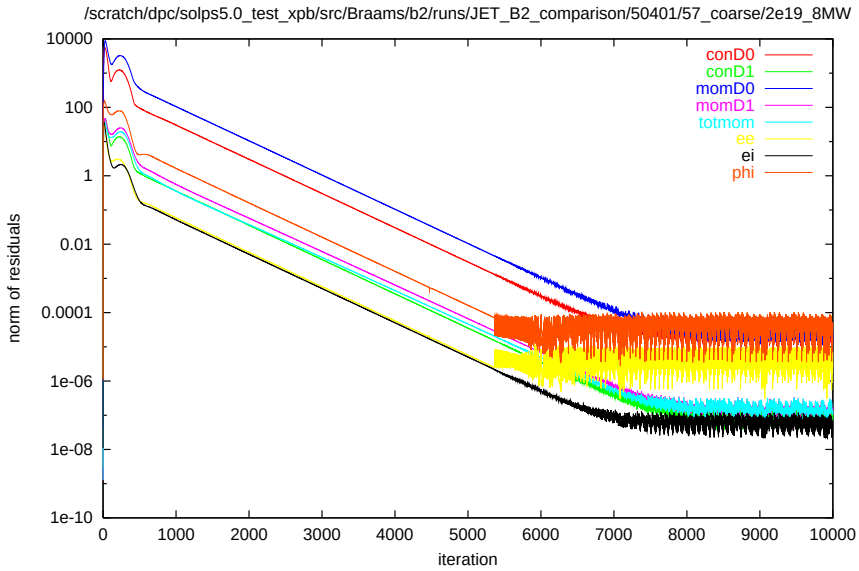


Figure 3.4: Residuals.

The power crossing various surfaces can be seen with `energy_analysis` or `energy_analysis.py`, figure 3.5. The detailed definition of these surfaces can be found in Appendix I or the commented header to the `init_region` subroutine in `b2agfs.F`.

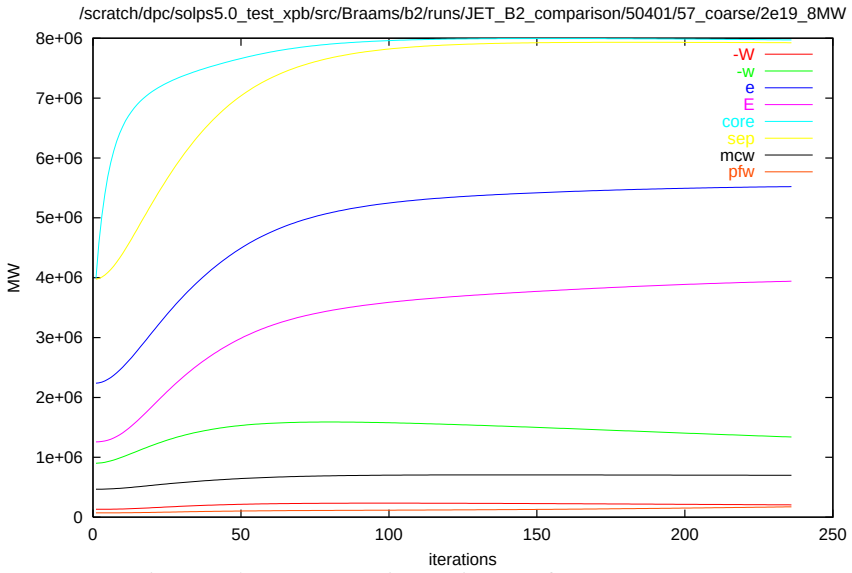


Figure 3.5: Power crossing various surfaces.

`2dt tesepm` can be used to plot the separatrix temperature as a function of time, figure 3.6. See table E.1 (page 467) for a list of quantities that can be plotted in this way.

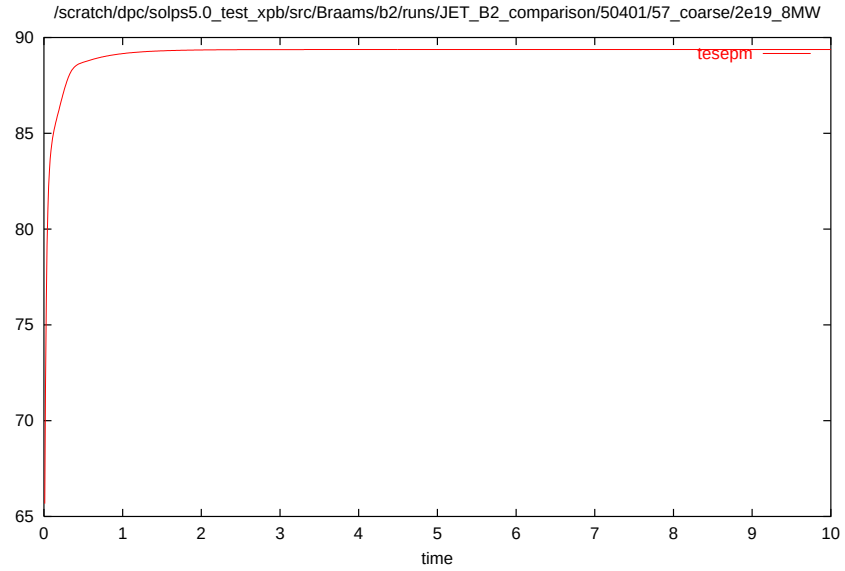


Figure 3.6: Upstream (outer midplane) electron temperature.

3.6 b2pl, the plotting program

The following commands apply:

```
pit,pitch          Pitch
bx,bpol            b_{x} (T) (poloidal)
by,brad            b_{y} (T)
bz,btor            b_{z} (T) (toroidal)
bb          bb (T)
b           B (T) (components)
ffbz        Toroidal magnetic flux function (Wb/m)
psi         Poloidal magnetic flux function (Wb)
na          Ion density (m^{-3})
nh          Hydrogenic atom density (m^{-3})
ni          Total density (m^{-3})
ua          Parallel velocity (u_{a}) (m.s^{-1})
ue          Parallel electron velocity (u_{e}) (m.s^{-1})
vv          Radial velocity (vv) (m.s^{-1})
ww          Toroidal velocity (ww) (m.s^{-1})
te          Electron temperature (eV)
ti          Ion temperature (eV)
pr          Static pressure (Pa)
pz          Plasma pressure (Pa)
ne          Electron density (m^{-3})
ne2         ne2 (m^{-3})
po          Electric potential (V)
dfhe        \mtD\dfhe (W)
dfhi        \mtD\dfhi (W)
dhet,dfhet         \mtD\dfhet (W)
dhit,dfhit         \mtD\dfhit (W)
dfhj        \mtD\dfhj (W)
dfch        \mtD\dfch (A)
dfna        \mtD\dfna (particles/s)
dfmo        \mtD\dfmo (N.s)
dfhm        \mtD\dfhm (W)
dfhp        \mtD\dfhp (W)
zeff        Z_{eff}
up          Poloidal velocity (up) (m.s^{-1})
```

```
fnax      Poloidal particle flux (s^{-1})
fnay      Radial particle flux (s^{-1})
fnex      Poloidal electron flux (s^{-1})
fney      Radial electron flux (s^{-1})
fmox      Poloidal flux of parallel momentum (N)
fmoy      Radial flux of parallel momentum (N)
fhex      Poloidal electron energy flux (W)
fhey      Radial electron energy flux (W)
fhetx     Poloidal electron energy flux [fhetx] (W)
fhety     Radial electron energy flux [fhety] (W)
fhix      Poloidal ion energy flux (W)
fhiy      Radial ion energy flux (W)
fhitx     Poloidal ion energy flux [fhitx] (W)
fhity     Radial ion energy flux [fhity] (W)
fhmx      Poloidal parallel kinetic energy flux (W)
fhmy      Radial parallel kinetic energy flux (W)
fhpx      Poloidal potential energy flux (W)
fhpy      Radial potential energy flux (W)
fhjx      Poloidal electrostatic energy flux (W)
fhjy      Radial electrostatic energy flux (W)
fhtx      Poloidal total energy flux (W)
fhty      Radial total energy flux (W)
fle       Electron heat flow flux limit factor
fli       Ion heat flow flux limit factor
nflfnax           Poloidal neutral particle flux limit factor
nflfnay           Radial neutral particle flux limit factor
nflfhix           Poloidal neutral ion energy flux limit factor
nflfhiy           Radial neutral ion energy flux limit factor
flvisc    Viscosity flux limit factor
fluce     Electron heat diffusivity Luciani flux limit factor
fluci     Ion heat diffusivity Luciani flux limit factor
flucv     Viscosity Luciani flux limit factor
flucs     Electric conductivity Luciani flux limit factor
fluca     Thermo-electric Luciani flux limit factor
flke      Electron heat diffusivity classical flux limit factor
flki      Ion heat diffusivity classical flux limit factor
flal      Thermo-electric classical flux limit factor
flalc     Thermo-electric coefficient limit factor
cvsx      cvsa_cl_x
cvsy      cvsa_cl_y
ceqp      ceqp
ebld      Energy balance (-divergence of fluxes) (W)
ebls      Energy balance (sources) (W)
ebal      Energy balance (W)
pbal      Particle balance (particles)
hx        hx (m)
hy        hy (m)
hy1       hy1 (m) (hy corrected)
sx        sx (m^{2})
sxperp    sxperp (m^{2})
sy        sy (m^{2})
sz        sz (m^{2})
vol       vol (m^{3})
qz0       qz0
qz1       qz1
qc        qc
mach      Mach number
mach4     Mach number (SOLPS4.0)
ehx       ehx
ehy       ehy
dab2      Atomic density (m^{-3})
dmb2      Molecular density (m^{-3})
ahal      Atomic H-\Ga\R emissivity (photons.m^{-3}.s^{-1})
mhal      Molecular H-\Ga\R emissivity (photons.m^{-3}.s^{-1})
srcml     Molecule dissociation rate (#.s^{-1})
edissml           Power to atoms due to molecule dissociation (W)
neutrad           Power loss due to atom radiation (W)
```

```
molrad    Power loss due to molecule radiation (W)
ionrad    Power loss due to molecular ion radiation (W)
rqae      Electron cooling rate (W) [rqahe]
b2ra      Line radiation rate (W)
b2br      Bremsstrahlung radiation rate (W)
rsai      Ionisation ion energy losses (W) [rsahi]
rsan      Ionisation particle losses (s^{-1}) [rsana]
rsam      Ionisation parallel momentum losses (N}) [rsamo]
rran      Recombination particle losses (s^{-1}) [rrana]
rrai      Recombination ion energy losses (W) [rrahi]
rram      Recombination parallel momentum losses (N) [rramo]
rcxn      Particle neutral losses (s^{-1}) [rcxna]
rcxi      Ion energy neutral losses (W) [rcxhi]
rcxm      Ion parallel momentum neutral losses (N) [rcxmo]
fetx      Total poloidal internal energy flux (W)
fety      Total radial internal energy flux (W)
tab2      Neutral atom temperature (eV)
tmb2      Neutral molecule temperature (eV)
rfla      Radial atomic flux (s^{-1}.m^{-2})
rflm      Radial molecular flux (s^{-1}.m^{-2})
refa      Radial atomic energy flux (W.m^{-2})
refm      Radial molecular energy flux (W.m^{-2})
pfla      Poloidal atomic flux (s^{-1}.m^{-2})
pflm      Poloidal molecular flux (s^{-1}.m^{-2})
pefa      Poloidal atomic energy flux (W.m^{-2})
pefm      Poloidal molecular energy flux (W.m^{-2})
mch2      Normalised parallel velocity
ro        Mass density (kg.m^{-3})
b2espcr           Nuclear sequence index
b2eatcr           Eirene atomic species index
mi        Ion mass (kg)
zi,za     Ion charge (e)
za2       Squared ion charge (e^{2})
zn        Nuclear charge (e)
fmx       Poloidal mass flux (kg/s)
fmy       Radial mass flux (kg/s)
p*        Dynamic plasma pressure (Pa)
conn      Connection length (m)
wpe       \Gx\R_{pe} (rad/s)
wce       \Gx\R_{ce} (rad/s)
wci       \Gx\R_{ci} (rad/s)
vthe      v_{th,e} (m/s)
vthi      v_{th,i} (m/s)
valfven           v_{Alfven} (m/s)
cr        CR (m)
cz        CZ (m)
na0       na at previous time step (m^{-3})
ne0       ne at previous time step (m^-3)
ni0       ni at previous time step (m^-3)
te0       te at previous time step (eV)
ti0       ti at previous time step (eV)
ua0       ua at previous time step (m.s^{-1})
po0       po at previous time step (V)
smth      Thermal force (N)
smpr      Thermal force (proportional to exhp) (N)
smpt      Thermal force (proportional to T_{i} and T_{e} gradients) (N)
smfr      Friction force (N)
smcf      Centrifugal force (N)
pot       Potential energy (eV) (cum. ionisation energy)
poti      Ionisation energy (eV)
pot*      Ionisation energy (eV) [shifted down one]
smp       Self-pressure force
nreg      Volume region index
xreg      X-directed flux region
yreg      Y-directed flux region
sna       Particle source (#.s^{-1})
sne       Electron particle source (e.s^{-1})
```

```
smo       Parallel momentum source (N)
smq       smq (momentum sources from atomic physics) (N)
she       Electron heat source (W)
shi       Ion heat source (W)
sch       Charge source (A)
smaf      smaf (extended friction term) (N)
smag      smag (pressure gradient term) (N)
smav      smav (additional viscosity term) (N)
rco       RESCO
rmo       RESMO
rmt       RESMT
ree       RESEE
rei       RESEI
rpo       RESPO
cpa       CORPA
cua       CORUA
cut       CORUT
cte       CORTE
cti       CORTI
cpo       CORPO
ixlt      ixlt
ixle      ixle
ixeq      ixeq
ixne      ixne
ixge      ixge
ixgt      ixgt
iylt      iylt
iyle      iyle
iyeq      iyeq
iyne      iyne
iyge      iyge
iygt      iygt
islt      islt
isle      isle
iseq      iseq
isne      isne
isge      isge
isgt      isgt
lnlam     ln \GK\R
q         q
s         Shear
alph      \Ga\R (ratio or pressure gradient to critical ballooning value)
d         D (m^{2}.s^{-1})
dp        D_{pa} (m^{2}.s^{-1})
kye       \Gv\R_{e} (m^{2}.s^{-1})
kyi       \Gv\R_{i} (m^{2}.s^{-1})
kyi0      \Gv\R_{i}^{0} (m^{2}.s^{-1})
vlax      Poloidal pinch velocity (m.s^{-1})
vlay      Radial pinch velocity (m.s^{-1})
visc      Viscosity (m.kg^{-1}.s^{-1})
bvis      Braams viscosity piece (m.kg^{-1}.s^{-1})
sig       Anomalous conductivity (mho.m^{-1})
alf       Thermo-electric coefficient (mho.m^{-1})
fchp      Parallel current (A)
fchx      Poloidal current (A)
fchy      Radial current (A)
ud        pol comp of the drift velocity [uadia.x] (m.s^{-1})
vd        rad comp of the drift velocity [uadia.y] (m.s^{-1})
vadia.x           pol comp of the divergent part of the ion diamagnetic velocity [vadia.x] (m.s
    ^{-1})
vadia.y           rad comp of the divergent part of the ion diamagnetic velocity [vadia.y] (m.s
    ^{-1})
wadia.x           pol comp of the total ion diamagnetic velocity [wadia.x] (m.s^{-1})
wadia.y           rad comp of the total ion diamagnetic velocity [wadia.y] (m.s^{-1})
vexb.x    pol comp of the ion E\mU\dB velocity [vaecrb.x] (m.s^{-1})
vexb.y    rad comp of the ion E\mU\dB velocity [vaecrb.y] (m.s^{-1})
```

```
vedia.x          pol comp of the divergent part of the electron diamagnetic velocity [vedia.x] (
    m.s^{-1})
vedia.y          rad comp of the divergent part of the electron diamagnetic velocity [vedia.y] (
    m.s^{-1})
veexb.x          pol comp of the electron E\mU\dB velocity [veecrb.x] (m.s^{-1})
veexb.y          rad comp of the electron E\mU\dB velocity [veecrb.y] (m.s^{-1})
cd       fchdia.x
dd       fchdia.y
niso     Isolating region
bc       Boundary conditions
rc       Recycling conditions
ix       ix (normalised)
iy       iy (normalised)
iix      ix
iiy      iy
lix      leftix
liy      leftiy
rix      rightix
riy      rightiy
bix      bottomix
biy      bottomiy
tix      topix
tiy      topiy
rlcl     Real cells
gdcl     Guard cells
dib2     Test ion density (m^{-3})
tib2     Test ion temperature (eV)
brna     Recycling particle source (#.s^{-1})
brne     Recycling electron source (e.s^{-1})
brmo     Recycling parallel momentum source (N)
brhe     Recycling electron heat source (W)
brhi     Recycling ion heat source (W)
brch     Recycling charge source (A)
bcna     Boundary particle source (#.s^{-1})
bcne     Boundary electron source (e.s^{-1})
bcmo     Boundary parallel momentum source (N)
bche     Boundary electron heat source (W)
bchi     Boundary ion heat source (W)
bcch     Boundary charge source (A)
bmna     Additional particle source (#.s^{-1})
bmne     Additional electron source (e.s^{-1})
bmmo     Additional parallel momentum source (N)
bmhe     Additional electron heat source (W)
bmhi     Additional ion heat source (W)
bmch     Additional charge source (A)
exna     External particle source (#.s^{-1})
exne     External electron source (e.s^{-1})
exmo     External parallel momentum source (N)
exhe     External electron heat source (W)
exhi     External ion heat source (W)
exch     External charge source (A)
dvue     b2sihs_divue (phm0)
dvua     b2sihs_divua (phm1)
exbe     b2sihs_exbe (phm5)
exba     b2sihs_exba (phm6)
visa     b2sihs_visa (phm2)
joul     b2sihs_joule (phm3)
fraa     b2sihs_fraa (phm4)
str      b2sihs_str (style)
ke       Parallel kinetic energy (J)
facdrift         facdrift
fac_exb          fac_ExB
fac_vis          fac_vis
xymap    xymap
fnax32   Convective poloidal particle flux (s^{-1})
fnay32   Convective radial particle flux (s^{-1})
fnax52   Conductive poloidal particle flux (s^{-1})
```

```
fnay52    Conductive radial particle flux (s^{-1})
fnaxnodr          Poloidal particle flux without drifts (s^{-1})
fnaynodr          Radial particle flux without drifts (s^{-1})
fnix32    Convective poloidal total ion flux (s^{-1})
fniy32    Convective radial total ion flux (s^{-1})
fnix52    Conductive poloidal total ion flux (s^{-1})
fniy52    Conductive radial total ion flux (s^{-1})
fnex32    Convective poloidal electron flux (s^{-1})
fney32    Convective radial electron flux (s^{-1})
fnex52    Conductive poloidal electron flux (s^{-1})
fney52    Conductive radial electron flux (s^{-1})
fchx32    Convective poloidal current (A)
fchy32    Convective radial current (A)
fchx52    Conductive poloidal current (A)
fchy52    Conductive radial current (A)
fchxvspa          Pol comp of fchvispar (A)
fchyvspa          Rad comp of fchvispar (A)
fchxvspe          Pol comp of fchvisper (A)
fchyvspe          Rad comp of fchvisper (A)
fchxn     Pol comp of fchin (A)
fchyn     Rad comp of fchin (A)
fnaxmdf           Modified poloidal particle flux (s^{-1})
fnaymdf           Modified radial particle flux (s^{-1})
fhexmdf           Modified poloidal electron heat flux (W)
fheymdf           Modified radial electron heat flux (W)
fhixmdf           Modified poloidal ion heat flux (W)
fhiymdf           Modified radial ion heat flux (W)
fhexps    Pfirsch-Schlueter poloidal electron heat flux (W)
fheyps    Pfirsch-Schlueter radial electron heat flux (W)
fhixps    Pfirsch-Schlueter poloidal ion heat flux (W)
fhiyps    Pfirsch-Schlueter radial ion heat flux (W)
fnaxps    Pfirsch-Schlueter poloidal particle flux (s^{-1})
fnayps    Pfirsch-Schlueter radial particle flux (s^{-1})
fnaxfcor          Poloidal particle flux with Coriolis term (s^{-1})
fnayfcor          Radial particle flux with Coriolis term (s^{-1})
fnaxhe    Poloidal particle flux for heat equations (s^{-1})
fnayhe    Radial particle flux for heat equations (s^{-1})
fchxvisq          Poloidal current due to heat flux viscosity tensor (A)
fchyvisq          Radial current due to heat flux viscosity tensor (A)
fchxinrt          Poloidal current due to inertia and gyroviscosity (A)
fchyinrt          Radial current due to inertia and gyroviscosity (A)
dnneo     Neoclassical particle diffusivity D_{n} (m^{2}.s^{-1})
chineo    Neoclassical heat conductivity \Gv\R (m^{2}.s^{-1})
vnneo     Neoclassical particle pinch v_{n} (m.s^{-1})
vwneo     Neoclassical heat pinch vw (m.s^{-1})
lnn       Density gradient length (m)
ltt       Temperature gradient length (m)
nuab      Two species collision frequency (s^{-1})
nua       Total collision frequency (s^{-1})
nustar    Normalized collisionality
vth       Thermal velocity (m.s^{-1})
rhol      Full-B Larmor radius (m)
deltal    Ratio of Larmor radius to minimum gradient length
wb        Width of the banana orbit (m)
deltaw    Ratio of banana width to minimum gradient length
deltawn           Ratio of banana width to density gradient length
deltawt           Ratio of banana width to temperature gradient length
omega     Gyro-frequency (rad.s^{-1})
deltab    Ratio of collision frequency to gyro-frequency
deltapar          Ratio of radial to parallel dynamics
chhichii          Chang-Hinton \Gv\R_{i} (m^{2}/s)
crx       Corner radial coordinate (m)
cry       Corner vertical coordinate (m)
zn_ext    External species nuclear charge (e)
za_ext    External species charge (e)
za2_ext           Squared external species charge (e^{2})
na_ext    External species density (m^{-3})
```

```
mi_ext    External species mass (kg)
ua_ext    External species parallel velocity (m.s^{-1})
ta_ext    External species temperature (eV)
fax_ext          External species poloidal flux (#.s^{-1})
fay_ext          External species radial flux (#.s^{-1})
ne_ext    External species associated electron density (m^{-3})
mdims     print out matrix stack information
sumx      sum over x
sumy      sum over y
f.x       line plot versus x
f.y       line plot versus y
intx      integrate over x
inty      integrate over y
f.lx      line plot versus log x
f.ly      line plot versus log y
cont      plot contours
surf      plot patch plot
line      !
psx       per sx
psxperp   per sxperp
psy       per sy
abs       abs()
ddx       d/dx
ddy       d/dy
sumz      sum over ns
pvol      per vol
pne       per ne
m+        matrix addition
m-        matrix subtraction
m*        matrix multiplication
m/        matrix division
rm+       matrix + real
rm-       matrix - real
rm*       matrix * real
rm/       matrix / real
rm**      matrix ** real
drop      drop top of matrix stack
dup       duplicate top of matrix stack
swap      swap top two elements of matrix stack
over      m2 m1 --> m2 m1 m2
list      list contents of matrix stack
xsel      select column
ysel      select row
zsel      select species
mmin      min(m1,m2)
mmax      max(m1,m2)
mlt       m<r
mle       m<=r
meq       m=r
mne       m!=r
mgt       m>r
mge       m>=r
rmmi,rmmin     min(r1,m1)
rmma,rmmax     max(r1,m1)
rtom      real --> matrix stack
rtoe      real --> Eirene stack
log       log10()
ln        ln()
sqrt      sqrt()
exp       exp()
zrng      select species range
chs       change sign
bala      balance ???
traj      trajectories
stoe      matrix --> Eirene array
stow      matrix --> wall array
xtow      x dimension of matrix --> wall array
```

```
ytow      y dimension of matrix —> wall array
rtow      real —> wall stack
rtot      real —> target stack
eirc      add the Eirene atomic quantities to the B2 values
nese      set ne from matrix stack
tese      set te from matrix stack
dase      set dab2 from matrix stack
uuse      set uutraj from matrix stack
vvse      set vvtraj from matrix stack
arox      set arrow field x-component from matrix
aroy      set arrow field y-component from matrix
pos       matrix with 1 if element > 0, else 0
target    add target elements plot to surface plot
trgoff    turns off target elements plot
idrop     drop top of integer stack
idup      duplicate top of integer stack
iswap     swap top two elements of integer stack
iover     i2 i1 —> i2 i1 i2
rdrop     drop top of real stack
rdup      duplicate top of real stack
rswap     swap top two elements of real stack
rover     r2 r1 —> r2 r1 r2
xmin      Minimum x for plotting in the computational domain
xmax      Maximum x for plotting in the computational domain
ymin      Minimum y for plotting in the computational domain
ymax      Maximum y for plotting in the computational domain
rmin      Minimum r for plotting in the physical domain
rmax      Maximum r for plotting in the physical domain
zmin      Minimum z for plotting in the physical domain
zmax      Maximum z for plotting in the physical domain
fmin      Minimum function value for plotting
fmax      Maximum function value for plotting
omin      Minimum ordinate for 2-d plots
omax      Maximum ordinate for 2-d plots
xref      Reference position in x
wmin      wmin in pltwall
wmax      wmax in pltwall
t_pt      total flight time of particles in caltraj
rshi      relative R shift to be applied to chords
zshi      relative Z shift to be applied to chords
r0        used to specify the R position of an ellipse in chord.F
z0        used to specify the Z position of an ellipse in chord.F
a0        used to specify the R radius of an ellipse in chord.F
b0        used to specify the Z radius of an ellipse in chord.F
tt0       used to specify the value within the ellipse in chord.F
trgxmin   Minimum x for target plotting in the computational domain
trgxmax   Maximum x for target plotting in the computational domain
trgymin   Minimum y for target plotting in the computational domain
trgymax   Maximum y for target plotting in the computational domain
trgrmin   Minimum r for target plotting in the physical domain
trgrmax   Maximum r for target plotting in the physical domain
trgzmin   Minimum z for target plotting in the physical domain
trgzmax   Maximum z for target plotting in the physical domain
trgfmin   Minimum function value for target plotting
trgfmax   Maximum function value for target plotting
trgomin   Minimum ordinate for 2-d target plots
trgomax   Maximum ordinate for 2-d target plots
smin      Minimum ordinate for wall plots
smax      Maximum ordinate for wall plots
emin      Minimum function value for Eirene data plots
emax      Maximum function value for Eirene data plots
neo_ic    Neoclassical model number
extl      extra label
xlab      extra label
ylab      extra label
cltb      colour table
chor      chord file
```

```
chpl      plot chords
cmd       command file
cmdcdf    read te, ti, ne, na, dab2, dmb2 from the b2movies cdf file and loop calling a command
    file for each frame
chbx      boxcar average of chords
chvl      value at positions
flux      call WOSlib flux routine
chmp      plots a contour plot of the chord contribution to the mesh element
cam       camera plot
movi      movie
sonn      something to do with Sonnet (no longer working)
saw       reads external data onto the matrix stack
saws      reads external data about plasma species onto the matrix stack
sawx      reads external data about external species onto the matrix stack
oson      output geometry in Sonnet format
uedgegrd  output geometry in UEDGE format (string + real arg)
expf      add experimental data to plot
outp      prints the value of a specific element
pdev      PLPLOT device name
mprt      output the top of matrix stack to a file
mtab      output a table from the top of matrix stack to a file
amju,amjuel    accesses Eirene type atomic physics data
adas      accesses ADAS atomic physics data
scat      concatenate two strings on top of string stack
sdrop     drop top of string stack
sdup      duplicate top of string stack
sswap     swap top two elements of string stack
sover     str2 str1 --> str2 str1 str2
sprt      output the string stack
omovie    old movie option
movie,nmovie    new movie option
wallblbl  set wall bar label
trgblbl   set target bar label
glbl      set graph label
elbl      set Eirene graph label
wlbl      set wall graph label
tlbl      set target graph label
slbl      set species label
eslbl     set Eirene species label
wslbl     set wall species label
tranprof  output transport profile
pdena     Atoms particle density (m^{-3})
pdenm     Molecules particle density (m^{-3})
pdeni     Test ions particle density (m^{-3})
edena     Atoms energy density (eV.m^{-3})
edenm     Molecules energy density (eV.m^{-3})
edeni     Test ions energy density (eV.m^{-3})
massa     Atom mass (kg)
massm     Molecule mass (kg)
massi     Molecular ion mass (kg)
vxa       Atoms Cartesian X-directed velocity (m.s^{-1})
vxm       Molecules Cartesian X-directed velocity (m.s^{-1})
vxi       Test ions Cartesian X-directed velocity (m.s^{-1})
vya       Atoms Cartesian Y-directed velocity (m.s^{-1})
vym       Molecules Cartesian Y-directed velocity (m.s^{-1})
vyi       Test ions Cartesian Y-directed velocity (m.s^{-1})
vza       Atoms Cartesian Z-directed velocity (m.s^{-1})
vzm       Molecules Cartesian Z-directed velocity (m.s^{-1})
vzi       Test ions Cartesian Z-directed velocity (m.s^{-1})
vxdena    Atoms Cartesian X-momentum density (kg.m^{-2}.s^{-1})
vxdenm    Molecules Cartesian X-momentum density (kg.m^{-2}.s^{-1})
vxdeni    Test ions Cartesian X-momentum density (kg.m^{-2}.s^{-1})
vydena    Atoms Cartesian Y-momentum density (kg.m^{-2}.s^{-1})
vydenm    Molecules Cartesian Y-momentum density (kg.m^{-2}.s^{-1})
vydeni    Test ions Cartesian Y-momentum density (kg.m^{-2}.s^{-1})
vzdena    Atoms Cartesian Z-momentum density (kg.m^{-2}.s^{-1})
vzdenm    Molecules Cartesian Z-momentum density (kg.m^{-2}.s^{-1})
```

```
vzdeni   Test ions Cartesian Z-momentum density (kg.m^{-2}.s^{-1})
triarea          Triangle area (m^{2})
triavol          Triangle volume (m^{3})
tr       Triangle centroid R coordinate (m)
tz       Triangle centroid Z coordinate (m)
pux      PUX
puy      PUY
pvx      PVX
pvy      PVY
tindex   Triangle index
edims    print out Eirene stack information
psta     per triangle area
pstv     per triangle volume
sumt     sum over triangles
esurf    plot triangle patch plot
eclvs    customise Eirene contour levels
eclvd    customise Eirene contour level delta
eclvn    customise Eirene contour level range and number
eclvp    customise Eirene contour level by proportion
eabs     abs() on Eirene stack
esumz    sum over ns for Eirene stack data
em+      Eirene stack addition
em-      Eirene stack subtraction
em*      Eirene matrix multiplication
em/      Eirene matrix division
rem+     Eirene stack + real
rem-     Eirene stack - real
rem*     Eirene stack * real
rem/     Eirene stack / real
rem**    Eirene stack ** real
edrop    drop top of Eirene stack
edup     duplicate top of Eirene stack
eswap    swap top two elements of Eirene stack
eover    e2 e1 --> e2 e1 e2
elist    list contents of Eirene stack
esel     select Eirene element
ezsel    select species in Eirene stack
emmin    min(e1,e2)
emmax    max(e1,e2)
remmi    min(r1,e1)
remma    max(r1,e1)
emlt     e<r
emle     e<=r
emeq     e=r
emne     e!=r
emgt     e>r
emge     e=>r
elog     log10(Eirene)
eln      ln(Eirene)
esqrt    sqrt(Eirene)
eexp     exp(Eirene)
ezrng    select species range in Eirene stack
echs     change sign in Eirene stack
etos     Eirene array -> m
epos     Eirene stack with 1 if element > 0, else 0
evac     fill vacuum region of Eirene stack with a value
earox    set arrow field x-component from Eirene data
earoy    set arrow field y-component from Eirene data
etal     read Eirene tally output file onto the Eirene stack
echor    chord file for Eirene stack data
echbx    boxcar average of chords for Eirene stack data
echvl    Eirene stack values at positions
echmp    plots a contour plot of the chord contribution to the Eirene element
ecam     camera plot of Eirene stack data
eoutp    prints the value of a specific Eirene element
eprt     output the top of Eirene stack to a file
eavg
```

```
wallarea        Wall area (m^{2})
ei      Incident energy (eV)
wallflux        Wall particle flux (s^{-1})
wallhflx        Wall heat flux (W/m^{2})
r1      Wall start R position (m)
r2      Wall end R position (m)
z1      Wall start Z position (m)
z2      Wall end Z position (m)
walltemp        Wall surface temperature (K)
plathick        Plate thickness (m)
coathick        Coating thickness (m)
cooltemp        Backplate temperature (K)
vaprtemp        Vaporisation temperature (K)
north   North boundary
south   South boundary
west    West boundary
east    East boundary
nseg    Wall segment
iwlt    iwlt
iwle    iwle
iweq    iweq
iwne    iwne
iwgt    iwgt
iwge    iwge
iswlt   iswlt
iswle   iswle
isweq   isweq
iswne   iswne
iswgt   iswgt
iswge   iswge
ibnd    ibnd
walldist        Distance along wall (m)
wallngth        Wall length (m)
sput    Physical sputtering yield
sput2   Physical sputtering energy yield
sputchem        Chemical sputtering yield
thermev         Thermal evaporation fluxes (/m^{2}/s)
sputres         RES sputtering yield
backscat        Backscattering fraction
backscae        Backscattering energy fraction
erosion         Gross erosion
deposit         Gross deposition
ero_ml  Monolayer erosion
depo_ml         Monolayer deposition
time    Plate time (s)
imapx   imapx
imapy   imapy
kappa_a         Bulk Kappa A factor (1/AT+B) (m/W)
kappa_b         Bulk Kappa B factor (1/AT+B) (m.K/W)
kappa_a1        Bulk Kappa A1 polynomial factor (W/m/K)
kappa_a2        Bulk Kappa A2 polynomial factor (W/m/K^{2})
kappa_a3        Bulk Kappa A3 polynomial factor (W/m/K^{3})
kappa_a4        Bulk Kappa A4 polynomial factor (W/m/K^{4})
kappa_a5        Bulk Kappa A5 polynomial factor (W/m/K^{5})
kappapa1        Bulk Kappa_perp A1 polynomial factor (W/m/K)
kappapa2        Bulk Kappa_perp A2 polynomial factor (W/m/K^{2})
kappapa3        Bulk Kappa_perp A3 polynomial factor (W/m/K^{3})
kappapa4        Bulk Kappa_perp A4 polynomial factor (W/m/K^{4})
kappapa5        Bulk Kappa_perp A5 polynomial factor (W/m/K^{5})
tgdns   Bulk mass density (kg/m^{3})
cp_b1   Bulk heat capacity B1 factor (J/kg/K)
cp_b2   Bulk heat capacity B2 factor (J/kg/K^{2})
cp_b3   Bulk heat capacity B3 factor
cp_b4   Bulk heat capacity B4 factor (J/kg/K^{4})
ckappa_a        Coating Kappa A factor (1/AT+B) (m/W)
ckappa_b        Coating Kappa B factor (1/AT+B) (m.K/W)
ckappaa1        Coating Kappa A1 polynomial factor (W/m/K)
```

```
ckappaa2          Coating Kappa A2 polynomial factor (W/m/K^{2})
ckappaa3          Coating Kappa A3 polynomial factor (W/m/K^{3})
ckappaa4          Coating Kappa A4 polynomial factor (W/m/K^{4})
ckappaa5          Coating Kappa A5 polynomial factor (W/m/K^{5})
ckapppa1          Coating Kappa_perp A1 polynomial factor (W/m/K)
ckapppa2          Coating Kappa_perp A2 polynomial factor (W/m/K^{2})
ckapppa3          Coating Kappa_perp A3 polynomial factor (W/m/K^{3})
ckapppa4          Coating Kappa_perp A4 polynomial factor (W/m/K^{4})
ckapppa5          Coating Kappa_perp A5 polynomial factor (W/m/K^{5})
ctgdns    Coating mass density (kg/m^{3})
c_cp_b1           Coating heat capacity B1 factor (J/kg/K)
c_cp_b2           Coating heat capacity B2 factor (J/kg/K^{2})
c_cp_b3           Coating heat capacity B3 factor
c_cp_b4           Coating heat capacity B4 factor (J/kg/K^{4})
erel      Chemical sputtering factor erel
bbdyemis          Blackbody emissivity
vapprs_a          Vapor pressure A factor (K)
vapprs_b          Vapor pressure B factor
vapprs    Vapor pressure (Pa)
qevap     Energy lost by vaporisation (J/m^{2}/s)
qrad      Energy lost by blackbody radiation (J/m^{2}/s)
stckcoef          Sticking coefficient
sublheat          Heat of sublimation (J/kg)
lkappa    Layer heat conductivity (W/m/K)
lrho      Layer mass density (kg/m^{3})
lcp       Layer heat capacity (J/kg/K)
lemis     Layer blackbody emissivity
lthick    Layer thickness (m)
ldepth    Layer depth (m)
sputfrac          Layer fraction
wall      adds wall stack data to contour plot
wall_ank  add Kukushkin wall data to contour plot
logw      log wall elements
linw      linear wall elements
logxw     log x wall elements plot
linxw     linear x wall elements plot
plwl      plot in wall domain
sumw      sum over walls
intw      integrate over walls
intw*     integrate over walls (no account of segments)
pswa      per wall area
wclvs     customise wall contour levels
wclvd     customise wall contour level delta
wclvn     customise wall contour level range and number
wclvp     customise wall contour level by proportion
wwid      wall line width factor
wabs      abs() on wall stack
ddw       d/dw derivative along wall length
wsumz     sum over ns for wall stack data
pwne      wall data per ne
wm+       wall stack addition
wm-       wall stack subtraction
wm*       wall stack multiplication
wm/       wall stack division
rwm+      wall stack + real
rwm-      wall stack - real
rwm*      wall stack * real
rwm/      wall stack / real
rwm**     wall stack ** real
wdrop     drop top of wall stack
wdup      duplicate top of wall stack
wswap     swap top two elements of wall stack
wover     w2 w1 --> w2 w1 w2
wlist     list contents of wall stack
wsel      select wall element
wzsel     select species in wall stack
wmmin     min(w1,w2)
```

```
wmmax     max(w1,w2)
rwmmi     min(r1,w1)
rwmma     max(r1,w1)
wmlt      w<r
wmle      w<=r
wmeq      w=r
wmne      w!=r
wmgt      w>r
wmge      w=>r
wlog      log10(wall)
wln       ln(wall)
wsqrt     sqrt(wall)
wexp      exp(wall)
wzrng     select species range in wall stack
wchs      change sign in wall stack
wtos      wall array -> m
wtot      wall array --> target
wpos      wall stack with 1 if element > 0, else 0
wsaw      reads external data onto the wall stack
wallf     add experimental wall data to plot
outpw     prints the value of a specific wall element
wprt      output the top of wall stack to a file
wtab      output a table from the top of wall stack to a file
wlld      ANK wall loading
woff      turn off wall plots
logtrg    log target elements
lintrg    linear target elements
logxtrg   log x target elements plot
linxtrg   linear x target elements plot
tclvs     customize target contour levels
tclvd     customise target contour level delta
tclvn     customise target contour level range and number
tclvp     customise target contour level by proportion
trg.w     line plot versus target wall
trg.z     line plot versus target depth
sumtrgw   sum over targets along walls
sumtrgz   sum over targets along z
inttrgw   integrate over targets along walls
inttrgw*  integrate over targets along walls (no account of segments)
inttrgz   integrate over targets along z
trgabs    absolute value of target data
trgddw    target d/dw derivative along wall length
trgddp    target d/dp derivative along perpendicular direction
trgddz    target d/dz derivative along target depth
rtrgm+    target stack + real
rtrgm-    target stack - real
rtrgm*    target stack * real
rtrgm/    target stack / real
rtrgm**   target stack ** real
trgtemp   target temperature (K)
trgdrop   drop top of target stack
trgwsel   select wall element in target stack
trgzsel   select depth element in target stack
rtrgmmi   min(r1,trg)
rtrgmma   max (r1,trg)
trglt     trg<r
trgle     trg<=r
trgeq     trg=r
trgne     trg!=r
trggt     trg>r
trgge     trg>=r
trglog    log10(target)
trgln     ln(target)
trgsqrt   sqrt(target)
trgexp    exp(target)
trgchs    change sign in target stack
trgtow    target array --> wall
```

```
trgpos    target stack with 1 if element > 0, else 0
outptrg   prints the value of a specific target element
trgprt    output the target stack to a file
trgtab    output a table from the top of target stack to a file
trgsurf   plot target plot
kappa_ab  target heat conductivity computed as 1/(AT+B)
kappa     target heat conductivity computed from polynomial fit
dkappadt  target heat conductivity derivative
kappap    perp. target heat conductivity computed from polynomial fit
kappapdt  perp. target heat conductivity derivative
cp        target specific heat capacity
rho       target mass density
trgwidth  target element width
trgdepth  target element depth position
trgvol    target element volume
trgdup    duplicate top of target stack
trgswap   swap top two elements of target stack
trgover   t2 t1 —> t2 t1 t2
trglist   list contents of target stack
trgmmin   min(t1,t2)
trgmmax   max(t1,t2)
trgr1     target element R1 position (m)
trgr2     target element R2 position (m)
trgz1     target element Z1 position (m)
trgz2     target element Z2 position (m)
trgnorth  North boundary
trgsouth  South boundary
trgwest   West boundary
trgeast   East boundary
trgseg    target segment
trgibnd   trgibnd
trgiz     trgiz
itrglt    itrglt
itrgle    itrgle
itrgeq    itrgeq
itrgne    itrgne
itrggt    itrggt
itrgge    itrgge
ztrglt    ztrglt
ztrgle    ztrgle
ztrgeq    ztrgeq
ztrgne    ztrgne
ztrggt    ztrggt
ztrgge    ztrgge
trgm+     target stack addition
trgm-     target stack subtraction
trgm*     target stack multiplication
trgm/     target stack division
me        electron mass
mp        proton mass
ev        electron charge
qe        electron charge
mu0       magnetic permittivity of the vacuum
eps0      electric permeability of the vacuum
pi        pi
c         speed of light
avogr     Avogadro's number
kbolt     Boltzmann constant
jxi       jxi
jxa       jxa
jsep      jsep
ixref     ixref
iyref     iyref
nx        nx
ny        ny
ns        ns
nspecies  nspecies
```

```
natm       natm
nmol       nmol
nion       nion
nxtl       nxtl
nxtr       nxtr
nbnd       nbnd
ndepth     ndepth_nml
ismain     ismain
ismain0    ismain0
ns_ext     ns_ext
f30        write fort.30
f31        write fort.31
glob       transform an extra label into a global header
both       both phys and comp
phys       physical domain only
comp       computational domain only
trigmesh   overlay triangular grid
mesh       overlay mesh
expl       toggle value of expline
page       set number of graphs per page
logf       log f plot
linf       linear f plot
logx       log x plot
linx       linear x plot
clvs       customise contour levels
clvd       customise contour level delta
clvn       customise contour level range and number
clvp       customise contour level by proportion
debu,debug  set debugging level
exps       set symbol for experimental data
eclv       do not fill contour Levels
fclv       do fill contour Levels
font       set font
lacl       label orientation along contour levels
lhor       label orientation horizontal
lbon       labeling on / off
lbof       labeling on / off
lwid       line width factor
zwid       axis line width factor
lfsz       line label font size
grid       grid on
trig       triangular grid on
clvl       contour level lines off
spln       switch on interpolation
arow       change default arrow length
earow      change default arrow length for Eirene stack data
arww       change default arrow width
acol       colour index for arrows
sep        plot separatrix
vesl       plot vessel walls from mesh.extra file
mmid       map to midplane
fpsi       plot versus full psi
mpsi       plot versus normalised psi
rpsi       plot versus root(normalised psi)
nocp       do not plot contour plot
nice       toggle whether axes will have nice values
bold       toggle whether to bold the axis labels
sepw       change separatrix line width
ewid       change experimental data line width
owid       change outline line width
vwid       change vessel line width
sq         square aspect ratio
xpol       use poloidal distance for f.x
xpar       use parallel distance for f.x
a4p        A4 portrait
a4l        A4 landscape
same       next graph on same axes
```

```
filo      fill option
i+        i1 + i2
i-        i1 - i2
i*        i1 * i2
i/        i1 / i2
i**       i1 ** i2
itor      integer --> real stack
itostr    integer --> string stack
itom      integer --> matrix stack
itoe      integer --> Eirene stack
itow      integer --> wall stack
itot      integer --> target stack
iprt      output the integer stack
r+        r1 + r2
r-        r1 - r2
r*        r1 * r2
r/        r1 / r2
r**       r1 ** r2
rtostr    real --> string stack
rprt      output the real stack
new       new page
lbsz      label size
glsz      globalheader size
ndec      number of decades
aspc      preserve aspect ratio flag
col       colour
bw        black and white
logt      log time for trajectories
lint      linear time for trajectories
zero      plot origin
outl      outline of plasma grid
stdl      "standard" label
mvsk      movie skip
ave       number of things to average over
t_#p      number of trajectories
reso      reset output
swid      part of tcl/tk interface
xyrz      part of tcl/tk interface
t_r+      ratio true
t_r-      ratio false
mcol      colour index of mesh
scol      colour index of separatrix
vcol      colour index of vessel
ocol      colour index of grid outline
zcol      colour index of zero lines
hcol      colour index of text header
gcol      colour index of graph title
bcol      colour index of bar label title
xcol      colour index of extralabel
dcol      colour index for matrix data line plots
wcol      colour index for wall data line plots
tcol      colour index for target data line plots
rigz      toggle set\_resignore\_regions\_to\_zero
noax      no axes
part      partition colour map
lbsa      toggle label lines
ytra      transform the "y" coordinate for comparison to experiment
ecol      colour index of experimental data
fcol      colour index for graph frame
nspe,nspec    new spectrometer
spec      old spectrometer
coro      print coronal rates
cora      coronal radiation rate
coni      coronal densities
shco      part of tcl/tk
user      call a user function "b2paux"
writ,write    set writing to file
```

```
omp       set jxa (OMP)
imp       set jxi (IMP)
yref      set yref
```

A revised copy of the **b2plot** manual can be found in Appendix J, page 504. Note that some of the quantities have changed name between **SOLPS 4.x** and **SOLPS 5.x** versions, table 3.1, and that whereas `ix` ran from `0` to `nx+1` it now runs from `-1` to `nx`, `iy` which ran from `0` to `ny+1` now runs from `-1` to `ny` and `is` which ran from `1` to `ns` now runs from `0` to `ns-1`. **SOLPS-ITER** uses the same indexing and naming conventions as **SOLPS 5.x**.

SOLPS 4.x	**SOLPS 5.x**
ni	na
zi	za
up	ua
dx	hx
dy	hy1
feex	fhex
feey	fhey
feix	fhix
feiy	fhiy
fnix	fnax
fniy	fnay

Table 3.1: Changes of quantity names from **SOLPS 4.x b2plot** to **SOLPS 5.x b2pl.exe**.

3.7 b2yg, plotting the geometry

The commands

```
b2run b2yg
```

or

```
gmake -f $SOLPSTOP/runs/Makefile b2yg.prt
```

or

```
gmake -f $SOLPSTOP/runs/Makefile b2yg.plt
```

will produce a `b2yg.plt` file containing a series of plots giving some details about the geometry, as well as a `b2yg.prt` ASCII file containing tables of the geometry quantities.

The `b2yg.plt` file is an NCAR GKS file. The presence in the run directory of a file called `ncar.cfg`, with the first character of the first line non-blank, will cause the program to use 'publication-ready' settings for the graphs, namely thicker and more visible lines and an easier-to-read font. The file can be converted into a PostScript file by means of the command:

```
ctrans -d ps.mono b2yg.plt > b2yg.ps
```

The input file `b2yg.dat` looks like:

```
*idout          (int, char*80, free format)
  0, '....'
*nmbpcc
  1
  0.5 0.5
*nmbpdm
  1
  0.5 3.0 -1.25 1.25
```

The meaning of the variables is:

iftp Formatting index for the plot contour lines

idout String specifying the type of data tables requested

nmbpcc Number of computational domain plots requested

cwaxr, cwayr For each computational plot requested, a doublet of numbers indicating the x and y relative positions, in computational space, of the reference location for these plots

nmbpdm Number of physical domain plots requested

cwxmin, cwxmax, cwymin, cwymax For each physical domain plot requested, a quadruplet of coordinates (x_{min}, x_{max}, y_{min}, y_{max}) that give the bounds of the domain to be plotted

The data fields that can be requested through **idout** are:

'pgmm' Print min/max values of the geometry data fields

'pgcc' Print geometry data on 5×5 corners

'pgnl' Print singular regions data

'pgmx' Print x-faced data

'pgmy' Print y-faced data

'pgso' Print South boundary data

'pgno' Print North boundary data

'pgwe' Print West boundary data

'pgea' Print East boundary data

'mpcr' Plot the geometry and mesh

'mpff' Plot the flux functions **fpsi** and **ffbz**

'mpbb' Plot the magnetic field

A `'.'` can be used as a wild card when specifying **idout**. Therefore, the string `'....'` will request all data fields. Several fields can be requested through the same **idout** string by listing them in sequence with or without some separating character(s).

3.8 b2yh, plotting the run parameters

The commands

```
b2run b2yh
```

or

```
gmake -f $SOLPSTOP/runs/Makefile b2yh.prt
```

will produce a `b2yh.prt` file containing a description of the run parameters: a species list, default boundary conditions parameters, and transport coefficients. These parameters will be extracted from `b2fparam` if it is present (thereby giving the run parameters as they were used by `b2mn`), or from `b2fpardf` which contains the default set of parameters built from `b2ah`.

The input file `b2yh.dat` looks like:

```
*style          (integer, free format)
  1
```

The only number in that file is a style switch that changes the type of output printed out in `b2yh.prt`. Current accepted values are 0 and 1.

3.9 b2yi & b2yp, plotting the plasma state

The commands

```
b2run b2yi
```

or

```
gmake -f $SOLPSTOP/runs/Makefile.plot b2yi.prt
```

or

```
gmake -f $SOLPSTOP/runs/Makefile.plot b2yi.plt
```

will produce a `b2yi.plt` file containing a series of plots describing the plasma state, as well as a `b2yi.prt` ASCII file containing tables of the plasma quantities. The plasma state will be extracted from `b2fstate` if it is present (thereby giving the plasma state as it was computed by `b2mn`), or from `b2fstati` which contains the initial plasma state built from `b2ai`.

Similarly, the commands

```
b2run b2yp
```

or any of

```
gmake -f $SOLPSTOP/runs/Makefile b2yp.prt
gmake -f $SOLPSTOP/runs/Makefile b2yp.plt
gmake -f $SOLPSTOP/runs/Makefile b2specp
```

will produce a `b2yp.plt` file with plots describing the plasma state computed by `b2mn` and stored in the `b2fstate` file, as well as a `b2yp.prt` ASCII file with tables of plasma quantities. `b2yp` also produces a `b2specp` file.

The `b2yi.plt` and `b2yp.plt` files are NCAR GKS files. The presence in the run directory of a file called `ncar.cfg`, with the first character of the first line non-blank, will cause the program to use 'publication-ready' settings for the graphs, namely thicker and more visible lines and an easier-to-read font. The files can be converted into PostScript by means of the commands:

```
ctrans -d ps.mono b2yi.plt > b2yi.ps
ctrans -d ps.mono b2yp.plt > b2yp.ps
```

The input file `b2yi.dat` looks like:

```
*idout          (int, char*80, free format)
  0, 'cpne,cpni,cpte,cpti,fne.,fhe.,pf..,pz..'
```

The input file `b2yp.dat` looks like:

```
*idout          (int, char*80, free format)
  1, 'cpne,cpte,cpti,pf..,pz..'
*nmbpcc
  0
*nmbpdm
  1
  0.5 3.0 -1.25 1.25
```

The meaning of the variables is:

in both `b2yi.dat` and `b2yp.dat` files:

iftp Formatting index for the plot contour lines

idout String specifying the type of data requested

additionally, in `b2yp.dat`:

nmbpcc Number of computational domain plots requested

cwaxr, cwayr For each computational plot requested, a doublet of numbers indicating the x and y relative positions, in computational space, of the reference location for these plots

nmbpdm Number of physical domain plots requested

cwxmin, cwxmax, cwymin, cwymax For each physical domain plot requested, a quadruplet of coordinates $(x_{min}, x_{max}, y_{min}, y_{max})$ that give the bounds of the domain to be plotted

The data fields that can be requested through **idout** are:

'pzmm' Print min/max values of the plasma quantities

'pfnx' Print reduced x-face particle fluxes

'pfzx' Print reduced x-face electron and ion fluxes

'pfny' Print reduced y-face particle fluxes

'pfzy' Print reduced y-face electron and ion fluxes

'pzso' Print South boundary data

'pzno' Print North boundary data

'pzwe' Print West boundary data

'pzea' Print East boundary data

'cpne' Plot the electron density

'cpni' Plot the total ion density

'cpte' Plot the electron temperature

'cpti' Plot the ion temperature

'cppr' Plot the total pressure (including neutrals)

'cppz' Plot the plasma pressure

'cppo' Plot the electric potential

'cpfn' Plot the fractional atom density

'cppe' Plot the electron pressure

'cpua' Plot the parallel velocity

'cpmc' Plot the Mach number

'cphe' Plot the electron heat flux

'cphi' Plot the ion heat flux

'fnex' Plot the integrated total particle fluxes as function of x

'fhex' Plot the integrated heat fluxes as function of x

'fchx' Plot the integrated electric current as function of x

'fnax' Plot the integrated ion particle fluxes as function of x

'fney' Plot the integrated total particle fluxes as function of y

'fhey' Plot the integrated heat fluxes as function of y

'fchy' Plot the integrated electric current as function of y

'fnay' Plot the integrated ion particle fluxes as function of y

A `'.'` can be used as a wild card when specifying **idout**. Therefore, the string `'....'` will request all data fields. Several fields can be requested through the same **idout** string by listing them in sequence with or without some separating character(s).

3.10 b2yr, plotting the atomic physics rates

The commands

```
b2run b2yr
```

or

```
gmake -f $SOLPSTOP/runs/Makefile b2yr.prt
```

or

```
gmake -f $SOLPSTOP/runs/Makefile b2yr.plt
```

will produce a `b2yr.plt` file containing a series of plots detailing the dependencies of the atomic rates on density and temperature for each isonuclear sequence, as well as a `b2yr.prt` ASCII file containing tables of these rates. If the `fix_recomb` switch has been activated, then the electron cooling rate is plotted with and without the contribution due to the ionisation potential of the recombining ion, and additional graphs are included for the negative of that rate if it becomes a heating (instead of cooling) rate.

The `b2yr.plt` file is an NCAR GKS file. The presence in the run directory of a file called `ncar.cfg`, with the first character of the first line non-blank, will cause the program to use 'publication-ready' settings for the graphs, namely thicker and more visible lines and an easier-to-read font. The file can be converted into a PostScript file by means of the command:

```
ctrans -d ps.mono b2yr.plt > b2yr.ps
```

The input file for `b2yr` is `b2yr.dat`, and an example is provided below:

```
*nijdis        (integer, free format)
   4   4
*idis          (nidis*integer, free format)
   8  16  24  32
*jdis          (njdis*integer, free format)
   2   6  10  14
```

The first set of integer is the number of density and temperature slices, respectively, that will be plotted, and the next two sets of integers are the indices within the density and temperatures tables in `b2frates` that are being plotted.

The user has the option when running the main code to rescale the rates given by `b2ar` through the use of the `b2.atomic_physics_rescale.parameters` file. See 3.5.8 for details.

3.11 b2ym, producing movie output

The commands

```
b2run b2ym
```

or

```
gmake -f $SOLPSTOP/runs/Makefile b2ym.prt
```

or

```
gmake -f $SOLPSTOP/runs/Makefile b2ym.plt
```

will produce a `b2ym.plt` file containing a series of movie frames, as well as a `b2ym.prt` ASCII file containing some additional output. The data for the movie is extracted from the `b2fmovie` file.

The `b2ym.plt` file is an NCAR GKS file. The presence in the run directory of a file called `ncar.cfg`, with the first character of the first line non-blank, will cause the program to use 'publication-ready' settings for the graphs, namely thicker and more isible lines and an easier-to-read font. The file can be converted into a PostScript file by means of the command:

```
ctrans -d ps.mono b2ym.plt > b2ym.ps
```

The input file `b2ym.dat` looks like:

```
*idout          (int, char*80, free format)
  0, 'cpne,cpte,cpti'
```

The meaning of the variables is:

iftp Formatting index for the plot contour lines

idout String specifying the type of data requested

The data fields that can be requested through **idout** are:

'cpne' Make contour plots of the electron density change

'cpna' Make contour plots of the ion density changes

'cpni' Make contour plots of the total atom density change

'cpte' Make contour plots of the electron temperature change

'cpti' Make contour plots of the ion temperature change

'cppo' Make contour plots of the electric potential change

'cpua' Make contour plots of the ion parallel velocity changes

A `'.'` can be used as a wild card when specifying **idout**. Therefore, the string `'....'` will request all data fields. Several fields can be requested through the same **idout** string by listing them in sequence with or without some separating character(s).

3.12 b2yn, display the progress of inner B2.5 iterations

The commands

```
b2run b2yn
```

or

```
gmake -f $SOLPSTOP/runs/Makefile.plot b2yn.prt
```

or

```
gmake -f $SOLPSTOP/runs/Makefile.plot b2yn.plt
```

will produce a `b2yn.plt` file containing contour plots of the residuals and corrections from the `B2.5` equations, as well as a `b2yn.prt` ASCII file containing some additional output. The data from the `B2.5` inner iterations is extracted from the `b2ftrack` file.

The `b2yn.plt` file is an NCAR GKS file. The presence in the run directory of a file called `ncar.cfg`, with the first character of the first line non-blank, will cause the program to use 'publication-ready' settings for the graphs, namely thicker and more isible lines and an easier-to-read font. The file can be converted into a PostScript file by means of the command:

```
ctrans -d ps.mono b2yn.plt > b2yn.ps
```

The input file `b2yn.dat` looks like:

```
*idout          (int, char*80, free format)
  0, 'cpne,cpte,cpti'
```

The meaning of the variables is:

iftp Formatting index for the plot contour lines

idout String specifying the type of data requested

The data fields that can be requested through **idout** are:

'cpne' Make contour plots of electron density residuals and correction

'cpni' Make contour plots of total ion density residuals and correction

'cpte' Make contour plots of electron temperature residuals and correction

'cpti' Make contour plots of ion temperature residuals and correction

'cppo' Make contour plots of electric potential residuals and correction

'cpna' Make contour plots of ion density residuals and correction

'cpua' Make contour plots of ion parallel momentum residuals and correction

'cput' Make contour plots of total parallel momentum residuals and correction

'cppr' Make contour plots of total pressure correction

A `'.'` can be used as a wild card when specifying **idout**. Therefore, the string `'....'` will request all data fields. Several fields can be requested through the same **idout** string by listing them in sequence with or without some separating character(s).

3.13 b2yq, display the evolution data

The commands

```
b2run b2yq
```

or

```
gmake -f $SOLPSTOP/runs/Makefile b2yq.prt
```

or

```
gmake -f $SOLPSTOP/runs/Makefile b2yq.plt
```

will produce a `b2yq.plt` file containing a quick display of the `B2.5` evolution data, as well as a `b2yq.prt` ASCII file containing some additional output. The data from `B2.5` is extracted from the `b2ftrace` file.

The input file `b2yq.dat` looks like:

```
1   2
1   3
1   6
1   7
1   8
1   9
```

Each line of the `b2yq.dat` file contains a doublet of integers:

indx Index of the data field in the `b2ftrace` file to use as abscissa

indy Index of the data field in the `b2ftrace` file to use as ordinate

The number of quantities actually written to the `b2ftrace` file during a `B2.5` run depends on the setting of the `b2mwqt_style` switch in `b2mn.dat`. They are, according to `b2mwqt_style` value:

style=1 default value

1 plasma simulation time

2 iteration counter

3..ns+2 norms of ion continuity residuals

ns+3..2*ns+2 norms of ion momentum residuals

2*ns+3 norm of total momentum residual

2*ns+4 norm of electron energy equation

2*ns+5 norm of ion energy equation

2*ns+6 norm of electric potential equation

2*ns+7..3*ns+6 norms of ion continuity corrections

3*ns+7..4*ns+6 norms of ion momentum corrections

4*ns+7 norm of total momentum correction

4*ns+8 norm of electron temperature correction

4*ns+9 norm of ion temperature correction

4*ns+10 norm of electric potential equation

style=2 all of the above, plus data for 36 grid regions:

4*ns+11..40*ns+10 total ion particle content per region

40*ns+11..40*ns+46 total electron energy content per region

40*ns+47..40*ns+82 total ion energy content per region

40*ns+83..40*ns+118 total electric energy content per region

style=3 all of the above, plus 20 integrated cross-region fluxes:

40*ns+119..60*ns+118 x-directed particle fluxes

60*ns+119..80*ns+118 y-directed particle fluxes

80*ns+119..80*ns+138 x-directed electron energy fluxes

80*ns+139..80*ns+158 y-directed electron energy fluxes

80*ns+159..80*ns+178 x-directed ion energy fluxes

80*ns+179..80*ns+198 y-directed ion energy fluxes

80*ns+199..80*ns+218 x-directed electric current

80*ns+219..80*ns+238 y-directed electric current

style=4 all of the above, plus data for 36 grid regions

80*ns+239..152*ns+238 min/max values of ion densities per region

152*ns+239..152*ns+310 min/max values of electron temperature per region

152*ns+311..152*ns+382 min/max values of ion temperature per region

152*ns+383..152*ns+454 min/max values of electric potential per region

The grid regions are formed as computational rectangles where the first and last region correspond to the domain boundaries ($ix = -1$, $ix = nx$, $iy = -1$ and $iy = ny$) and the intermediate regions each cover one fourth of the extent of the computational domain ($nx/4$ and $ny/4$ respectively). The integrated fluxes are the fluxes connecting those regions, excluding the fluxes within boundary regions.

3.14 b2yt, changing from one grid size and species set to another

The converter program `b2yt` can be used for a variety of purposes, and prospective users are urged to read the *entire* section below carefully before proceeding to use it. As a summary, however, the sequence of steps to be followed for a succesful conversion is:

1 Create a new case directory: `$SOLPSWORK/<my_new_case>`.

2 In the new case directory, create subdirectories `baserun` and `convert` as well as a run subdirectory (`<my_run>`). `baserun` and `convert` are special names and should not be changed.

3 Copy from your old case to the new case's `convert` directory all input and output files you wish to have converted. You may copy `baserun` files that you know will not be converted to the new case's `baserun` directory instead.

4 Check and correct as necessary the relative ages of the input files, for which you can use the `correct_b2yt_timestamps` script.

5 Run the `b2yt` converter.

6 Displace the converted case input files to the converted case `baserun` and run directories, for which you can use the `import_files_from_convert` script.

The `b2yt` utility program allows a user to take a `b2fstate` solution file with size $nx \times ny$ and transform it into an $nx1 \times ny1$ `b2fstatt` file, provided the numbers of real cells in both x and y directions in the new solution are divisors of the ones in the original mesh. Other files necessary to proceed with the run under the new geometry will also be produced. The two numbers $nx1$ and $ny1$ are provided by the user in the `b2yt.dat` file. The `b2yt.dat` file should contain, for example:

```
*dimens          (nx1, ny1; free format)
  8  8
'solps_version'  'iter'
'eirene_format'  'iter'
```

with any optional switches mentioned afterwards, here to create a solution on an 8×8 grid.

A solution for non-commensurate grids is also available, requiring instead that the topology of the two solutions be the same, and works by 2-D interpolation of the plasma data in the computational domain, by region. This is activated with the `'b2ytdr_non_commensurate'` option set to `'1'` in `b2yt.dat`. This option should also be used when the overall grid size remains the same but the location and/or depth of the cuts has changed, or when mapping a lower single-null (LSN) case to an upper single-null (USN) case or vice versa (or between USN-like and LSN-like limiter geometries). When switching between USN and LSN, the solution will be mirror-reflected to keep the inboard and outboard divertor solutions in the correct geometrical locations.

The conversion of the `Eirene` input file additionally requires for the converter program to know the original version of `Eirene` with which the input file was used, and this is provided by means of the `'eirene_format'` switch. Its possible values (case-insensitive) are:

'old' for input files from **SOLPS4.0** and **SOLPS5.0** runs using "old" Eirene_96

'new' for input files from **SOLPS4.0** and **SOLPS5.0** runs using "new" Eirene_99

'facelift' for input files from **SOLPS 5.1** runs

'juelich' for input files from Juelich `Eirene` versions (2008 and younger)

'iter' (default) for input files from **SOLPS4.2/4.3** and **SOLPS-ITER** runs

'msv' for input files from the `Eirene` Milestone Version

The `input.dat.2` file that the converter will create will follow the Milestone Version format.

The conversion of the Eirene output file (fort.44) additionally requires for the converter program to know the original version of SOLPS with which the output file was created, and this is provided by means of the 'solps_version' switch. Its possible values (case-insensitive) are:

'4.3' for fort.44 output files from **SOLPS4.3** runs

'5.0' for fort.44 output files from **SOLPS5.0** runs

'5.1' for fort.44 output files from **SOLPS 5.1** runs

'5.2' for fort.44 output files from **SOLPS 5.2** runs

'iter' (default) for input files from **SOLPS-ITER** runs (no conversion necessary)

The fort.44.2 file that the converter will create will follow the **SOLPS-ITER** format.

One can also use b2yt to flip the sign of the poloidal and/or toroidal components of the magnetic field, and the code will make the appropriate adjustments in the input and data files. The switches to obtain this are:

'invert_bpol' '1' Inverts the poloidal field and related quantities

'invert_btor' '1' Inverts the toroidal field and related quantities

This is important because the parallel velocity is co-aligned with the magnetic field in **SOLPS-ITER**, which was not the case in earlier code versions (please consult Section C.1 for a full description of the sign conventions used in **SOLPS-ITER**). For instance, if you are converting from an older **SOLPS4.3** case (where the magnetic field components were always positive by default), you need to use these switches to adapt the plasma solution and the geometry files. This is done automatically for ITER cases when using the convert_from_solps4.3 script.

b2yt may also be used to change the order of species (for example changing from $D + C + He$ to $D + He + C$) or for bundling, de-bundling, or re-bundling species, or even removing species and complete isonuclear sequences. It is however not possible to use this converter to *add* a new isonuclear sequence. When modifying the species data, the user should run b2ar with the new species arrangement before running b2yt to obtain a temporary file b2frates. After b2yt has been run successfully, then the user should re-run b2ah (using the produced b2ah.dat.2 file, after renaming it b2ah.dat) and b2ar before re-launching the run.

If a new geometry is requested, the b2yt program should be run after the new geometry file ($nx1 \times ny1$) has been produced and is available in the convert or baserun directories as b2fgmtry. If converting a linear case, it is possible to move the location of the nominal separatrix on the new grid by means of the b2ytdr_jsep1 switch in b2yt.dat. The old geometry file should also remain available in the convert directory as b2fgmtry.old. If this is a coupled case and the number of triangles for the Eirene grid remains the same, the new fort.33, fort.34 and fort.35 files corresponding to the new geometry should also be provided. If, however, the number of triangles between the old and new geometries changes, there is currently no conversion possible of the data in the fort.46 file and it should not be made present in the convert directory.

The program also needs the b2mn.dat and any other used input files (such as input.dat, b2.*.parameters or b2.*.inputfile) from the original run.

It is important for proper b2yt execution that the necessary files have the expected seniority from the Makefile. Therefore, you must ensure that:

1. b2fstate is the *youngest* file.
2. b2frates (or ../baserun/b2frates) is *older* than b2fstate but *younger* than b2fgmtry.
3. b2fpardf (or ../baserun/b2fpardf) is *younger* than b2ah.dat and *older* than b2frates.
4. b2fgmtry must be *younger* than b2ag.dat and *older* than b2fpardf.

So, ALL of the following must be verified:

```
b2ag.dat < b2fgmtry < b2fpardf < b2frates < b2fstate
```

```
b2ah.dat < b2fpardf

b2ar.dat < b2frates
```

If this order is not met, use a series of `touch 'filename'` commands to give the files new dates. The best to check this is to run the command:

```
b2run -n b2yt
```

which will launch a dry-run of the Makefile invoking `b2yt`. Then, make sure that this dry-run only attempts to run `b2yt` and no other programs. If it does call, say `b2ag` and/or `b2mn`, then the age hierarchy above was not respected. In that case, you can run the script

```
correct_b2yt_timestamps
```

to get the correct ordering of files. Do check again afterwards with the dry-run command to make sure it worked.

Upon successful termination of `b2yt`, the user will find a `b2fstatt` file containing the transformed plasma state, as well as a `b2mn.dat.2` file which will contain the updated run switches appropriate for the new geometry and species selection. Similarly, if `b2.boundary.parameters` and `b2.neutrals.parameters` files were present in the working directory, files `b2.boundary.parameters.2` and `b2.neutrals.parameters.2`, respectively, will contain the updated namelist variables. The program will also create a new `b2.wall_save.parameters.2` file containing the new wall namelist settings and a new `b2favere.2` file containing the new averaged solution, when appropriate. Other `B2.5` input files are treated in the same fashion. The code will also attempt to create a new `SOLPS_FLUIDS.2` file appropriate for the new species list.

If an `input.dat` file is present in the run directory, the code will also attempt to produce a new `input.dat.2` file. Note that conversion of an `eirene.input.json` file is not supported. In order to ensure consistency between the `B2.5` and `Eirene` modified inputs, it is important that the order of the neutrals boundary conditions in `b2.neutrals.parameters` and the strata in `input.dat` match! If the `B2.5` grid has been changed, and a new `Eirene` triangular grid is needed, the triangulation will need to be redone. In order to do so, copy the converted `input.dat.2` file to `../baserun/triang.eir`, modify the copied file to set `NLTRIMESH` to `.true.`, and then launch the `triang` script (in the `baserun` directory, skipping the `(U)inp` and `(b)2ag` steps, i.e. starting at the `(E)irene` step.

When species are being re-bundled, possible conflicts may occur between their individual settings. In such cases, the code behaviour is to use the setting from the highest charge state within the bundle. If species are being removed, strata, tallies or diagnostics involving these species may either be removed from the input file or revert to their default settings. The user is encouraged to check that those choices are appropriate and modify them as needed.

Once the user has finished converting a case, a script exists, named `import_files_from_convert`, that will take the converted files from `b2yt` and place them, after proper renaming, in the current run directory, provided the `convert` directory where `b2yt` was run is located at the same level. Some of the files are put in the `baserun` directory if they would belong there, so it is always best, after the import, to execute `correct_baserun_timestamps` from the parent directory, followed by a `touch b2fstati` command in each of the run subdirectories.

Because of the complexity of the conversion process, for which not all possible cases have been tested, users are encouraged to inspect the `b2yt` output files visually after execution to make sure no mistakes remain. No attempt is made to adjust or modify the text labels pertaining to the various input files, so users should consider adapting those to the new case being built. Report of conversion errors or omissions to the developer team are always welcome.

3.15 b2co, recovering from a crash

In case of a computer crash during a run, it may be possible to recover the results and continue the simulation. The code writes, at regular intervals (default is 3600 CPU seconds, and can be modified with use of the `b2mndr_cpu`, `b2mndr_ntim_save`, or `b2mndr_plasmatim` switches, see Appendix A.4 for details), recovery files entitled `plasmastate.XXXX` and `b2.*_save.parameters.XXXX` in the `b2mn.exe.dir` run subdirectory. The `XXXX` are numbers indicating the order in which the files were written. The `plasmastate` files contain the same information as the `b2fstati` or `b2fstate` files, but in a more compact format. In order to recover from a crash, the user should then:

1. Check for the presence of `plasmastate.XXXX` files in the `b2mn.exe.dir` run subdirectory
2. If present, check that the most recent file is of the same size as the others (sometimes the crash happens while writing the `plasmastate` file!)
3. You can check the plasma time to which the `plasmastate` file corresponds by means of the `inspect_plasmastate #1` script where `#1` is the name of the plasmastate file you are considering to use
4. `setenv PLASMASTATE b2mn.exe.dir/plasmastate.XXXX` where `XXXX` is the number of the desired `plasmastate` file
5. `b2run b2co` or `gmake -f $SOLPSTOP/runs/Makefile b2co.prt` which will create a `b2fstate` file
6. `mv b2fstate b2fstati`
7. Re-launch the run

If the crash happened early on in the simulation and one wishes to return to the initial state of the simulation to re-launch the run cleanly, one may use the `recover_initial_files` script instead.

This procedure will produce the files necessary to re-start the run from the point at which the `plasmastate` file was written. In cases with feedback schemes active, the feedback values will be grabbed from the relevant `b2.*_save.parameters.XXXX` files and a new set of `b2.*_save.parameters` files will be created. Optional output files (i.e the NetCDF files `b2time.nc`, `b2tallies.nc`, `b2wall.nc`, `b2movies.nc` and `eirenemovies.nc` (the last one only for runs with `Eirene`) , as well as the `*.trc` files in the `tracing` directory), if present, will be truncated to be synchronized with the recovered plasma state file.

It is also possible to use the `b2co` utility *while* `SOLPS` is running, in order to obtain a `b2fstate` file in the run directory that one may use to plot certain plasma quantities and monitor the current status of the run. However, to do this and not overwrite the current tracing files, `b2co` *must* be invoked as follows:

1. `setenv PLASMASTATE b2mn.exe.dir/plasmastate.XXXX` where `XXXX` is the number of the desired `plasmastate` file
2. `b2run -b b2co` or `gmake -f $SOLPSTOP/runs/Makefile b2co.prt B2FSTATE_ONLY='yes'` which will create the provisional `b2fstate` file

3.16 b2uf & b2fu, converting the plasma output file

The commands

```
b2run b2uf
```

or

```
gmake -f $SOLPSTOP/runs/Makefile b2uf.prt
```

will convert an 'un*formatted' `b2fplasma` file into a 'formatted' `b2fplasmf` file. This file contains a full description of the plasma solution and derived quantities.

The reverse operation, converting back from `b2fplasmf` to `b2fplasma`, is obtained by means of the commands:

```
b2run b2fu
```

or

```
gmake -f $SOLPSTOP/runs/Makefile b2fu.prt
```

3.17 b2rd, fetching an archived MDSPLUS run

If a run has been archived into the `MDSplus` tree structure (see 6 for details) it is possible to extract from the tree the information necessary to obtain `b2fstate` and `b2fgmtry` files from which to continue the run. To do so, follow the procedure below:

1. Check for the presence of a `shotnumber.history` file in the run directory
2. If present, the fetched run will be the one with the run number specified last in `shotnumber.history`
3. If absent or incorrect, create or modify the `shotnumber.history` file to add an extra line at the end with the run number of the run you which to retrieve
4. `b2run b2rd` or `gmake -f $SOLPSTOP/runs/Makefile.plot b2rd.prt` which will create a `b2fstate` file
5. `mv b2fstate b2fstati`
6. `mv b2fgmtrt b2fgmtry`
7. Re-launch the run

This procedure will re-start the run from the point at which the `MDSplus` file was written. However, the input files and run settings are not retrieved along with `b2fstate` (these are not stored) so the user must check the input data present in the run directory before re-launching a case. In cases with feedback schemes active, for example, the feedback values will be the ones stored in the appropriate namelists from the `b2.*.parameters` files in the run directory. Concerning the retrieval of the geometry case, some quantities must be recomputed by `b2rd` as they are not saved in `MDSplus`, so the code assumes a tokamak geometry and that the mesh is a basis mesh.

3.18 b2ymb, post processing for monoblock heat transfer model

This routine first reads the formatted file `b2ymb.dat` and writes the unformatted file `mnb_in.dat` that is needed by the `FreeFem++`(http://www.freefem.org)[4] heat transfer model of the monoblock, see the details in the manual in `${SOLPSTOP}/doc/SOLPS-ITER_PFCs_heat_transfer_model.pdf`. Then, it reads the files `wlld_trg[i-o].dat` produced by **b2plot** with the `"1000 wlld"` command and writes the unformatted files `W_tg[i-o].dat`, `X_tg[i-o].dat` and `tg[i-o]_size.dat` that are needed by the `FreeFem++` heat transfer model of the monoblock. In `W_tg[i-o].dat` is stored the value of the heat flux for each tile of the monoblock in Watts. In `X_tg[i-o].dat` is stored the value of the poloidal coordinate of the center of each tile along the monoblock axis (in metres). In `tg[i-o]_size.dat` is stored the number of points (i.e. tiles) in which the inner/outer monoblocks have been subdivided according to the assigned tile poloidal length, which is read from the `b2ymb.dat` file. To use the routine:

1. Define in the `DG` model the plot zone variables for inner and outer targets with labels `trgi` and `trgo` respectively
2. Create the file `b2ymb.dat` in the run directory (see below for an example)
3. `echo "1000 wlld" | b2plot` will produce the `wlld_trg[i-o].dat` files
4. Copy the `wlld_trg[i-o].dat` files in the run directory
5. `b2plot b2ymb`

Example of the `b2ymb.dat` file:

```
&GEOMETRY

HEIGHT      =    28.0e-3,    ! Tile heigth [m]
T_LENGTH    =    28.0e-3,    ! Tile toroidal length [m]
CU_TH       =    1.0e-3,     ! Cu interlayer thickness [m]
CUCRZR_TH   =    1.5e-3,     ! CuCrZr tube thickness [m]
DIAMETER    =    12.0e-3,    ! CuCrZr tube internal diameter [m]
W_TH        =    6.0e-3,     ! W armour thickness (from plasma facing surface to Cu interlayer) [m]
P_LENGTH    =    1.2e-2,     ! Tile poloidal length [m]
T_GAP       =    0.5e-3,     ! Toroidal gap between monoblocks [m]
BETA        =    1.0,        ! Poloidal chamfer [degrees]
ALPHA       =    0.5,        ! Vertical target tilting [degrees]
THETA       =    4.2,        ! Toroidal strike angle of magnetic flux lines [degrees]
REVERSE     =    0,          ! Integer, if >0 reverse the heat flux profile from top to bottom
/

&MESH

NN          =    50,         ! \# of mesh points on longest edge
/

&COOLANT

T_COOLANT   =    70.0,       ! Coolant inlet temperature [C]
M_FLOW      =    0.734,      ! Coolant mass flow rate [kg/s]
CP_COOLANT  =    4.22e3,     ! Coolant specific heat capacity [J/kg/K]
/

&OUTPUT

NPOINTS     =    2,          ! Number of points for output surface temperature
LPOINTS     =    0.,14.0e-3, ! Location of output surface temperature
NPLOTS      =    10,         ! Plot the 2D profile each NPLOTS cuts
/
```

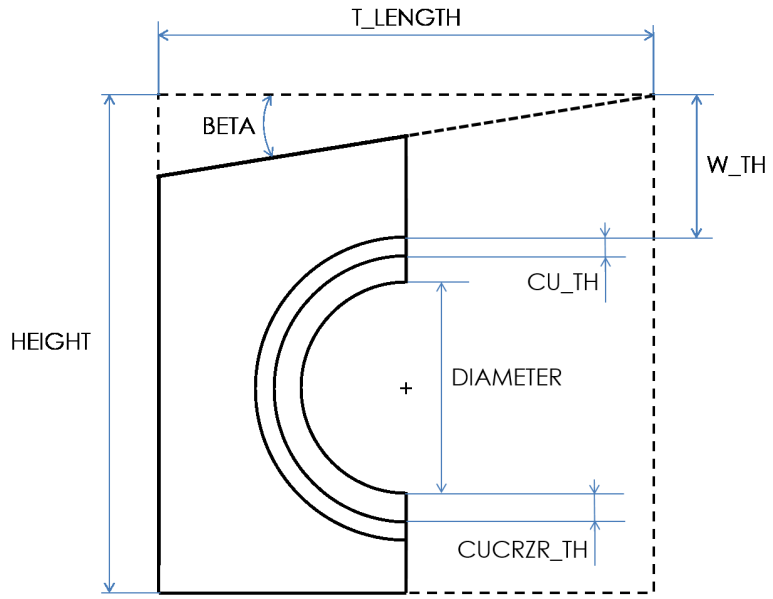


Figure 3.7: Monoblock tile cross-section

3.19 b2yrp, post processing for reflector plates heat transfer model

This routine first reads the formatted file `b2yrp.dat` and writes the unformatted file `rps_in.dat` that is needed by the `FreeFem++`(http://www.freefem.org)[4] heat transfer model of the reflector plates, see the details in the manual in `${SOLPSTOP}/doc/SOLPS-ITER_PFCs_heat_transfer_model.pdf`. Then, it reads the files `wlld_[i-o]rp.dat` produced by **b2plot** with the `"1000 wlld"` command and writes the unformatted files `W_[i-o]rp.dat`, `X_[i-o]rp.dat` and `[i-o]rp_size.dat` that are needed by the `FreeFem++` heat transfer model. In `W_[i-o]rp.dat` is stored the value of the heat flux for each tile of the RPs in Watts. In `X_[i-o]rp.dat` is stored the value of the poloidal coordinate of the center of each tile along the RPs axis (in metres). In `[i-o]rp_size.dat` is stored the number of points (i.e. tiles) in which the inner/outer RPs have been subdivided according to the assigned tile poloidal length, which is read from the `b2yrp.dat` file. To use the routine:

1. Define in the `DG` model the plot zone variables for inner and outer reflector plates with labels `irp` and `orp` respectively
2. Create the file `b2yrp.dat` in the run directory (see below for an example)
3. `echo "1000 wlld" | b2plot` will produce the `wlld_[i-o]rp.dat` files
4. Copy the `wlld_[i-o]rp.dat` files in the run directory
5. `b2plot b2yrp`

Example of the `b2yrp.dat` file:

```
&GEOMETRY
L1          =     25.0e-3,    ! [m]
L2          =     17.0e-3,    ! [m]
L3          =     0.25e-3,    ! [m]
H1          =      4.0e-3,    ! [m]
H2          =      5.0e-3,    ! [m]
H3          =      7.0e-3,    ! [m]
H4          =      2.0e-3,    ! [m]
H5          =      8.0e-3,    ! [m]
H_HOLE      =      2.5e-3,    ! [m]
```

```
RH          =      1.5e-3,     ! [m]
P_LENGTH    =      1.2e-2,     ! Tile poloidal length [m]
T_GAP       =      1.0e-3,     ! Toroidal gap between hypervapotron cahnnels [m]
ALPHA       =         0.5,     ! Reflector plate tilting [degrees]
BETA        =         1.0,     ! Poloidal chamfer [degrees]
THETA       =         4.2,     ! Toroidal strike angle of magnetic flux lines [degrees]
REVERSE     =           0,     ! Integer, if >0 reverse the heat flux profile from top to bottom
/

&MESH
NN          =          20,     ! \# of mesh points on H4
/



&COOLANT
T_COOLANT   =     128.595,     ! Coolant inlet temperature [C]
M_FLOW      =       0.734,     ! Coolant mass flow rate [kg/s]
CP_COOLANT  =      4.22e3,     ! Coolant specific heat capacity [J/kg/K]
/



&OUTPUT
NPOINTS     =           2,     ! Number of points for output surface temperature
LPOINTS     =  0.,14.0e-3,     ! Location of output surface temperature
NPLOTS      =          10,     ! Plot the 2D profile each NPLOTS cuts
/
```

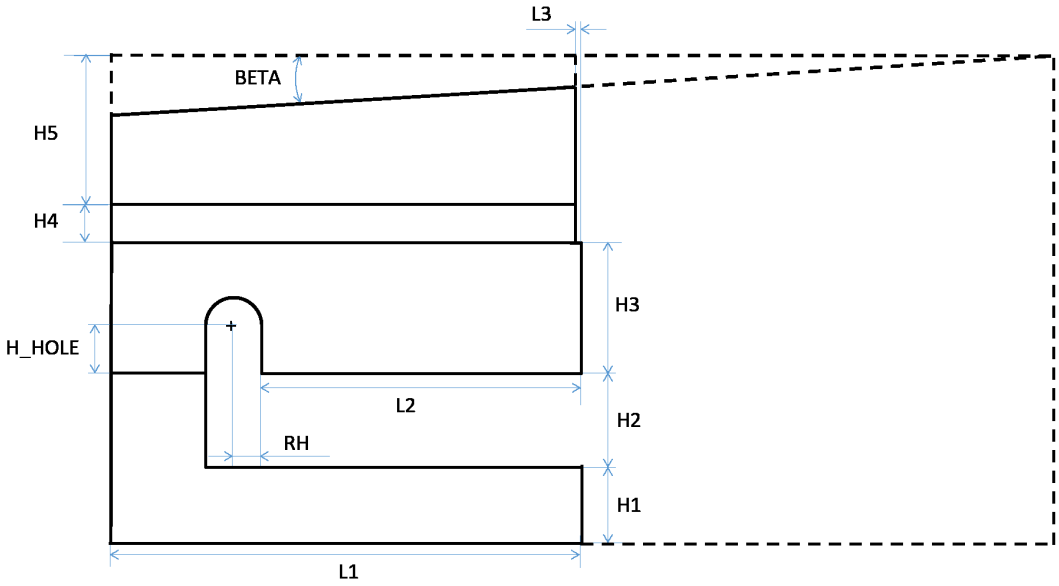


Figure 3.8: Reflector plate tile cross-section

3.20 b2ydm, post processing of dome wall loading

This routine first reads the formatted file `b2ydm.dat`. Then, it reads the file `wlld_dome.dat` produced by **b2plot** with the `"1000 wlld"` command and writes the unformatted file `dome_out.dat` in which is stored the coolant temperature at the outlet of the dome hypervapotron cooling system. This temperature can be used as inlet temperature for the outer reflector plate heat transfer model. To use the routine:

1. Define in the `DG` model the plot zone variable for the dome
2. Create the file `b2ydm.dat` in the run directory (see below for an example)
3. `echo "1000 wlld" | b2plot` will produce the `wlld_dome.dat` file
4. Copy the `wlld_dome.dat` file in the run directory
5. `b2plot b2ydm`

Example of the `b2ydm.dat` file:

```
&GEOMETRY
HEIGHT      =     28.0e-3,    ! Tile height [m]
T_LENGTH    =     50.0e-3,    ! Tile toroidal length [m]
P_LENGTH    =      1.2e-2,    ! Tile poloidal length [m]
T_GAP       =      1.0e-3     ! Toroidal gap between hypervapotron channels [m]
ALPHA       =         0.5,    ! Reflector plate tilting [degrees]
BETA        =         1.0,    ! Poloidal chamfer [degrees]
THETA       =         4.2,    ! Toroidal strike angle of magnetic flux lines [degrees]
/

&COOLANT
T_COOLANT   =        70.0,    ! Coolant inlet temperature [degrees Celsius]
M_FLOW      =       0.734,    ! Coolant mass flow rate [kg/s]
CP_COOLANT  =      4.22e3,    ! Coolant specific heat capacity [J/kg/K]
/
```

3.21 b2_ual_write, writing last state to IDS

The program can be run with one of the following

```
gmake -f $SOLPSTOP/runs/Makefile b2_ual_write.prt
```

or

```
b2run b2_ual_write
```

This program saves the final output state of a run into an IDS (Interface Data Structure). Currently, it will populate the `edge_profiles`, `edge_transport`, `edge_sources`, `radiation`, `divertors`, `dataset_description`, and `summary` data structures. If the current run being saved is a time continuation of the run already present in that IDS, then a new time slice is appended, otherwise the IDS is overwritten. If the existing IDS was written with an earlier Data Dictionary version, it is recreated with the current one.

The program reads the `b2mndr_id*` switch group, defined in `b2mn.dat`:

- `b2mndr_user` defines the user name and it is set by default to be the same as the `USER` environment variable.
- `b2mndr_database` defines the IMAS IDS database name and it is set by default to be the same as the `DEVICE` environment variable. If `DEVICE` has been left undefined, the default database name is `'solps-iter'`. If `DEVICE` has been set to `'iter'`, the code internally modifies it to `'ITER'`, following the IMAS convention.
- `b2mndr_device`, superseded by `b2mndr_database`.
- `b2mndr_ids_path` defines the path to the IDS being saved. May or may not contain the environment variable `IMASDIR` as prefix. If not provided, it is read from the last line of the `shotnumber.history` file. Only available when using IMAS Access Layer 5. If it starts with a `/` character and does not contain `IMASDIR`, it is understood to be an absolute path.
- `b2mndr_pulse_number` defines the pulse (in Access Layer 5) or shot (in Access Layer 4) number. If not provided, it is read from the last line of the `shotnumber.history` file. Superseded by `b2mndr_ids_path`.
- `b2mndr_shot_number`, superseded by `b2mndr_pulse_number`.
- `b2mndr_run_number` defines the run number. If not provided, it is deduced from the last line of the `shotnumber.history` file. Superseded by `b2mndr_ids_path`.

These switches are used as identification parameters for the IDS. Afterwards, a `b2mn` run is initialized but *not* started, so that the final output state data is read. Then the data is written to an IDS, with the `b2mndr_id*` switchgroup used as identification descriptors.

It is possible to replace (or supplement) the IDS switch settings coming from the `b2mn.dat` file by using the command line `-o` option, with the following syntax:

```
b2run -o "-s 123456 -r 1" b2_ual_write
```

which will recognize, within the quotes following the `-o` additional options opener, the switches:

- `-p`, `--path` The path to the directory where to save the IMAS data entry (only available with Access Layer 5).
- `-s`, `--shot` `--pulse` The pulse (in Access Layer 5) or shot (in Access Layer 4) number (superseded by `--path`). If used, must be positive.
- `-r`, `--run` The run number (superseded by `--path`). If used, must be positive or zero.
- `-u`, `--user` The user name (default is `USER`).
- `-d`, `--database`, `--device` The IMAS database name (default is `DEVICE`, if set, or `'solps-iter'` otherwise).
- `-v`, `--version` The IMAS version to be used (default is `3`).

Concerning the assignment of pulse (in Access Layer 5) or shot (in Access Layer 4) and run numbers, the following rules are in effect. If not using IMAS Access Layer 5, the shot number must a six digit number between 1 and 214748. ITER pulse or shot numbers shall start with a 1, the second digit represents the majority fuelling species (0:H, 1:He, 2:D, 3:D-T, 4:T, 5:others, the sequence reflecting the order in which fuel gases are expected to be used in the ITER Research Plan), and the last four digits identify the simulation. Because multiple codes are all writing to the same database and we want to avoid clashes, different "slices" of numbers have been assigned for various codes. We will use numbers `1x2000` to `1x2999` for transferring existing simulations from the **SOLPS 4.3** database to IMAS, and `1x3000` to `1x3499` for new **SOLPS-ITER** cases. Run numbers should start at 1, and be incremented every time a new version of the same simulation (say after a re-run because a bug in the code was fixed) is being stored in the archive. When changing important run parameters such as gas puffing rates, turning on drifts, etc..., a new pulse or shot number should be used. If intending to publish the IMAS data entry in the IO database, please send an email to the IO code Responsible Officer (`Xavier.Bonnin@iter.org`), when you wish to assign a new shot number, as the IO will keep track of those already used. IMPORTANT: *When publishing SOLPS-ITER results with IO co-authors involved, archiving into the public IMAS database of the simulations on which the published work is based is mandatory.*

With Access Layer 5, the IMAS data entries will be written in the directory with absolute location given by `path`, with the prefix `$IMASDIR/` if it was not provided. `IMASDIR` is expected to be of the form `<path_to_user_public_directory>/imasdb/${database}/${version}`. It should be set as part of the local environment variables in a `setup.csh.local` file. Use of the `user`, `database`, or `version` input flags allows to overwrite their values in the `IMASDIR` path. The `path` may also contain `$HOME`, `$IMASDIR`, or `$SOLPSTOP`, which will be parsed by the code.

After the IDS files are created, if using IMAS Access Layer 3 or 4, one should also run the `create_db_entry` script, with the syntax:

```
create_db_entry [-u USER] [-d DATABASE] -s SHOT -r RUN
```

which will create `ids_*.yaml` and `ids_*.watcher` files for that shot and run number, to be used in the IMAS scenario database. These files should be edited by hand according to the instructions within them and the IMAS documentation.

3.21.1 b2_ual_rewrite, rewriting an existing IDS

The program can be run with one of the following

```
gmake -f $SOLPSTOP/runs/Makefile b2_ual_rewrite.prt
```

or

```
b2run b2_ual_rewrite
```

This program rewrites an existing IDS in case a correction needs to be made (because of a bug fix, for instance). It will rewrite the `edge_profiles`, `edge_transport`, `edge_sources`, `radiation`, `divertors`, `dataset_description`, and `summary` data structures. It uses the same inputs as `b2_ual_write`. In addition, one can have the IDS rewritten with a new run number, by means of the optional `-n` *new_run_number* argument. Only the last time slice present in the IDS can be rewritten in this way, and data from older time slices will be lost. If a new run number is specified, the `.yaml` file of the old run number will be modified to indicate that it is now obsolete and replaced by the new run number, and the `.yaml` file for the new IDS will also indicate which old run it is replacing. The new run number must be larger than the old one.

3.22 b2xx, the others

Chapter 4

Running the standalone version of Eirene

The complete Eirene manual is provided as part of the **SOLPS-ITER** distribution as the $SOLPSTOP/modules/Eirene/Manual/eirene.pdf file, automatically built with the code compilation. If not present, it can also be found at http://www.eirene.de/. The user is strongly encouraged to check out that very complete manual. Reference [3] also provides a useful description.

The Eirene standalone executable is built by doing gmake eirene (if you want the Eirene graphics) or gmake eirene_nox (without graphics) from the top **SOLPS-ITER** directory. MPI and/or OpenMP parallelization and debugging options can be requested by adding _mpi, _openmp and _debug (last) to the compilation target name, or by using, respectively, the set_mpi, set_openmp, and set_debug scripts before launching the compilation. The executables are named eirobj and eirobjx respectively, and can be invoked directly.

Eirene can also be compiled from its submodule directory with:

```
sei
gmake depend
gmake listobj
gmake tags
gmake depend
gmake
```

See also the next chapter and section K.1 for setting the array sizes and useful file links for Eirene before compilation.

4.1 Specific SOLPS-ITER blocks in the Eirene input file

In addition to the very complete description of the Eirene input file in the Eirene online manual, the last part of the Eirene input file (blocks 14 and 15) is specific to **SOLPS-ITER** coupling. We therefore document its content here.

4.1.1 Block 14. Data for interfacing routine INFUSR

The data in this block is meant to allow communication between the B2.5 and Eirene halves of **SOLPS-ITER** .

The format of this input block for coupling to **SOLPS-ITER** is as follows:

****14. Data for interfacing routine "infusr"*

```
LSYMET  LBALAN  LCOARSE NLTRIMESH  LSPRCL  LOLD31
NFLA    NCUTB   NCUTL   IMF   NTRFRM   NFULL   IBRAD   IBPOL   IBTOR
```

read NPLSI species lines

```
DO 20, IPL=1,NPLSI
   I   IFLB   FCTE   BMASS   LKINDP
20 CONTINUE
```

Recycling target definitions

```
NDXA   NDYA
NTARGI
NTGPRT(IT), IT=1,NTARGI
DO 30, IT=1,NTARGI
   DO 33, IPRT=1,NTGPRT(IT)
      I NDT    NINCT  NIXY   NTIN   NTEN   NIFLG   NPTC   NPTCM
        NSPZI  NSPZE  NEMOD
   33 CONTINUE
30 CONTINUE
```

Short cycle parameters

```
CHGP   CHGEE   CHGEI   CHGMOM
NAINB  NCOPIB  NCOPEB
DO 40, IAIN=1,NAINB
   I   NAINS   NAINT
   TXTPLS(IAIN,12)
   TXTPSP(IAIN,12)  TXTPUN(IAIN,12)
40 CONTINUE
NAOTB
DO 50, IAOT=1,NAOTB
   I   NAOTS   NAOTT
50 CONTINUE
```

Meaning of the input variables for interfacing routine INFUSR

We only include here the variables or aspects thereof not already described in the `Eirene` manual, section 2.14.

LCOARSE Coarse mesh switch ???

NLTRIMESH When `.TRUE.`, `Eirene` is only run to obtain the contours necessary to build the triangular grid, which are then written in `fort.78`. This is used by the `triang` script toolchain. No particle trajectories are followed.

LSPRCL Turns on species-specific source rescaling when `.TRUE.`

LOLD31 Switch for backward compatibility with SOLPS5.3 `fort.31` file format (should always be set to `.FALSE.` in **SOLPS-ITER** runs)

NCUTB For **SOLPS-ITER** couplings, should be 0

NCUTL For **SOLPS-ITER** couplings, should be 1

IMF `fort.30` format switch, should be 0

NTRFRM `fort.33` format switch, should be 0

LKINDP Index of the `Eirene` atom this species is linked to

NPTC In addition to the description in the `Eirene` manual, note that only the value from the first segment of the stratum (`IPRT=1`) is used.

NPTCM This corresponds to the NMINPTS flag in input block 7, and is needed only if INDSRC=6, i.e., if the source is specified by data from block 14 alone. As for NPTC, only the value from the first stratum segment is used.

NEMOD This corresponds to the NEMODS flag in input block 7, and is needed only if INDSRC=6, i.e., if the source is specified by data from block 14 alone. As for NPTC, only the value from the first stratum segment is used. It is overwritten internally by the code to be 7 for E/W surfaces (i.e. include sheath acceleration) and 6 for N/S surfaces (do not include sheath acceleration).

NCOPIB Not in use

NCOPEB Maximum number of additional tallies per plasma background species

4.1.2 Block 15. Data for interfacing routine GEOUSR

When using the JSON formatted input, the data from this block will be found in the user_data.input file.

This input block is used to adjust the edges of the additional walls in the Eirene description of the vacuum vessel walls to match exactly the location of the grid corners from the plasma solver in B2.5. This block may begin with an optional text line containing either GENERAL or BIASED_GARCHING, which changes the format below. The **SOLPS-ITER** default is to not include this text line. With that default, the specified additional surfaces will be adjusted to meet up exactly with the grid corners at the edges of the divertor targets, and there should be one such additional surface specification per grid corner. The corners will be listed in the following order, in reference to the rectangular computational domain: lower left, upper left, lower right, upper right. For double-null configurations, each separate half of the computational domain will be listed in turn, first the four corners of the left half, then the four corners of the right half. An additional surface can only accommodate one such end vertex adjustment, not more. The number of such corner adjustments is provided by NASMOD.

It is also possible to move the vertices of other additional surfaces with the NADMOD block, and to flip cell face normals with the NORMOD block, although this is not recommended in standard **SOLPS-ITER** runs.

****15. Data for interfacing routine "geousr"*

If there is no preceding text line (recommended default), the reading format is:

```
NADMOD   NASMOD
DO I=1,NADMOD
   NRS   IPUNKT   XCOOR   YCOOR   ZCOOR
ENDDO
DO I=1,NASMOD
   LIMPOS   ONETWO
   IF (ONETWO.LT.0) XPOLPOS   YPOLPOS
ENDDO
```

If the text line is GENERAL, the reading format is:

```
NADMOD   NASMOD   NORMOD
DO I=1,NADMOD
   NRS   IPUNKT   XCOOR   YCOOR   ZCOOR
ENDDO
DO I=1,NASMOD
   NAS   IPUNKT   NSSIR   NSSIP
ENDDO
DO I=1,NORMOD
   IDIR  IR   IP
ENDDO
```

If the text line is BIASED_GARCHING, the reading format is:

```
DO I=1,max(NPPLG/3,1)*4
   ONETWO   LIMPOS
   IF (ONETWO.LT.0) XPOLPOS   YPOLPOS
ENDDO
```

Meaning of the input variables for interfacing routine GEOUSR

NADMOD Number of additional surfaces to be fixed to another position specified by `(XCOOR,YCOOR,ZCOOR)`. Should be 0 for standard **SOLPS-ITER** couplings.

NASMOD Number of additional surfaces for which one end vertex is to be matched to a B2.5 grid corner. Should be 2 per divertor target, and 2 for a limiter geometry case.

NORMOD Number of cell faces for which the normal is to be inverted. Should be 0 for standard **SOLPS-ITER** couplings.

NRS Index of the additional surface being modified

IPUNKT Index of the vertex being modified in that additional surface. Values from 1 to 6 are allowed in `NADMOD` block, and 1 or 2 only in `NASMOD` block.

XCOOR New X-coordinate of the vertex being modified (in cm).

YCOOR New Y-coordinate of the vertex being modified (in cm).

ZCOOR New Z-coordinate of the vertex being modified (in cm).

LIMPOS Index of the additional surface to be fixed to a grid corner. The particular grid corner to which this additional surface is fixed is given by the ordering specified in the introductory paragraph of this subsection.

ONETWO `ABS(ONETWO)` is the index of the vertex being modified in that additional surface, and can only be 1 or 2

XPOLPOS Integer `X`-index (poloidal) of the `Eirene` grid vertex to which the additional surface vertex must be matched. To be used only if the default setting does not give the desired result.

YPOLPOS Integer `Y`-index (radial) of the `Eirene` grid vertex to which the additional surface vertex must be matched. To be used only if the default setting does not give the desired result.

NAS Index of the additional surface to be modified

NSSIR Radial index of the `Eirene` grid vertex to which the additional surface vertex is to be matched

NSSIP Poloidal index of the `Eirene` grid vertex to which the additional surface vertex is to be matched

IDIR Direction of the face which normal is being inverted (1 = radial, 2 = poloidal)

IR Radial index of the `Eirene` grid cell which face normal is being inverted

IP Poloidal index of the `Eirene` grid cell which face normal is being inverted

4.1.3 Information for MPI

The standard way to choose the MPI parallelization strategy in `Eirene` is *via* the `NPRLL` variable in block 1 of the input file. In order to obtain the default **SOLPS-ITER** behaviour as quoted below, one should set `NPRLL = -1`. However, one can also use an optional block at the very end of the `Eirene` input file which will overwrite the `NPRLL` setting. Its format is:

**** INFORMATION_FOR_MPI*

`AUTOMATIC_DISTRIBUTION`

There are currently six possible choices supported. They are:

BALANCED A balanced strategy in which the workload is attempted to be divided evenly among processors. If the initial setup for the **BALANCED** strategy fails, the code then attempts the **APCAS** strategy. Also obtainable by setting `NPRLL = 3`.

IMPROVED An improvement over the **BALANCED** strategy to minimize communication overhead and increase stability and parallelization efficiency. If the initial setup for the **IMPROVED** strategy fails, the code then attempts the **APCAS** strategy. Also obtainable by setting `NPRLL = 4`.

APCAS meaning *All Processors Compute All Strata*, where the work of each stratum is divided among all processors. Usually involves more communication between processors. Also obtainable by setting `NPRLL = 2`.

ORIGINAL The original strategy implemented in versions of `Eirene` prior to **SOLPS-ITER** version 3.0.5. Also called "proportional allocation" in the `Eirene` manual. In that strategy, each processor is handed either part of a large stratum or one or several full smaller strata. This strategy works well if the number of processors is larger than the number of strata, but is not recommended if the number of strata is larger than the number of processors. Also obtainable by setting `NPRLL = 1`.

EMBARRASS meaning the "embarassingly parallel" scheme by which each processor is assigned the same number of particle trajectories, as given in the `Eirene` input file. This is the default in non-**SOLPS-ITER** versions of `EIRENE`. Also obtainable if setting `NPRLL = 0` or `NPRLL > 4`.

AUTOMATIC tells the code to use the current default, whatever it may be.

The format to read the strategy name is free-form. The input file is scanned for the words describing the chosen strategy after the title line `*** INFORMATION_FOR_MPI` is found. The *first* match to an (uppercase) strategy name will be the strategy followed by the code. If the title line is missing or no match is found to any of the strategy names, the default strategy is used. See the `Eirene_balanced_MPI` document on the **SOLPS-ITER** web site `Manuals and Documentation` folder for more details on the performance and algorithm corresponding to each strategy.

The default strategy depends on the version of the MPI library with which `Eirene` was compiled. If MPI version 3 or greater was used, then the default strategy is the **BALANCED** one. Otherwise, the **ORIGINAL** strategy is chosen. If `Eirene` is used outside the **SOLPS-ITER** environment, the default strategy will be the **EMBARASSINGLY PARALLEL** one.

4.2 Pressure Feedback Loop

This type of boundary condition modifies the recycling probability `RECYCT` by means of a proportional-integral loop so that the pressure in a given cell matches a given reference value. It is activated by setting `ILREF = 4` in the desired reflection model in block 6a of the Eirene input file. Two new values, related with the reference cell and pressure, have to be provided. The first one in the line:

```
ILREF ILSPT ISRS ISRC ILCHSPNWL REFCELL
```

in which `REFCELL` is an integer referencing the cell in the Eirene mesh in which the pressure will be computed. This value can be easily obtained with `triang` by means of the `(I)nquire` function. The second value is provided in the line:

```
RECYCS RECYCC SPTPRM ESPUTS ESPUTC REFPRESS
```

in which `REFPRESS` is the reference pressure in Pa.

This loop will not provide values of `RECYCT` larger than 1 or below 0. The coefficients of the proportional-integral loop are hard-coded.

4.3 Modelling for Beryllium migration

A model for the self-consistent modelling of Beryllium (Be) erosion and migration is provided through the SPTUSR routines within Eirene. This model tracks the erosion and re-deposition of Be for selected wall elements. It computes the fractional surface coverage of deposited Be originating from wall erosion (from any location) on the designated wall elements as a function of time and allows for the further sputtering and transport of that deposited Be.

For this model to be activated, the file input_for_refusr.dat should be present in the run directory. This file has the format

```
BEGIN     END   DELAY     SBE        TSTEP      AVCOF   TSTEP_FLAG  NOACUM_FLAG
  182     253      10   3.240E-20  1.000E-02    0.990            0            1
```

and contains the variables:

BEGIN Starting Eirene wall element number over which the Be deposition will be computed (should be in the range [1:NLIMI+NSTSI]). If given a negative value, its absolute value will be understood as the index of an Eirene non-standard surface and mapped to its proper location in the array.

END Ending Eirene wall element number over which the Be deposition will be computed (should be in the range [1:NLIMI+NSTSI] and larger than or equal to BEGIN once mapped). If given a negative value, its absolute value will be understood as the index of an Eirene non-standard surface and mapped to its proper location in the array.

DELAY Frequency of Eirene calls at which the conbe.dat output file is written (should be at least 1)

SBE Characteristic area occupied by one Be atom on the surface (in m^2)

TSTEP Time-step used in the dynamic re-sputtering model (in s)

TSTEP_FLAG Flag to indicate that all surfaces use the same time-step (default), or if a separate array of different time-steps for each wall element is to be provided (if TSTEP_FLAG = 1, see below)

AVCOF Averaging coefficient in the Robbins-Monro method for smoothing the time-evolution of the beryllium surface concentration (should be between 0 and 1)

NOACUM_FLAG If set to 1, limits the maximum Be concentration to one complete monolayer (recommended)

An example of an input file where different time-steps are provided for each wall element looks like:

```
BEGIN     END     DELAY       SBE         TSTEP       AVCOF     TSTEP_FLAG   NOACUM_FLAG
  182     253        10     3.240E-20   1.000E-02     0.990              1             1
TSTEP(MSURF):
 7.9880147E-02 6.8972709E-02 5.7468701E-02 5.0992880E-02 4.7954093E-02 4.3538353E-02 3.4336616E-02 2.6990613E-
```

where the array of timesteps is provided after the initial set of variables.

For wall elements between BEGIN and END, to activate the Be dynamic sputtering model, one must in addition select the user-supplied sputtering model as part of their surface properties, namely ILSPT = 9. The surface block first two lines will then look like:

```
SURFMOD_3_TUNGSTEN_BESPUT_1160K
      1      9      4      0      1
```

where 4 refers to the species index of Be in the list of atomic species followed by Eirene in block 4a of its input file. With these settings, only Be will be sputtered from that surface, not its original material.

For wall elements within the [BEGIN:END] range that do not have ILSPT = 9, the Be surface concentration will remain constant. To speed up the convergence of the model, it is recommended to use a TSTEP value that is 4-5 orders of magnitude larger than the B2.5 timestep, since the deposition and sputtering processes occur on much longer timescales than the plasma transport itself.

If this model is active, the conbe.dat file will be written at least on the first and last iterations of the run, and every DELAY iterations in between. If the file is already present at the beginning of the run, its contents will be read and

used for the starting values of the Be data evolution. If the file is not found, it will be created and assume zero Be concentration on the wall elements and `1%` Be concentration on the `Eirene` non-standard surfaces at the start. The file will contain blocks of six lines (one per writing call) which correspond to:

1. `SPUMPADD` Instantaneous values of the Be deposition fluxes (in $A.m^{-2}$)
2. `SPTTOTADD` Instantaneous values of the Be sputtering fluxes (in $A.m^{-2}$)
3. `SPUMPSAVE` Time-averaged values of the Be deposition fluxes (in $A.m^{-2}$)
4. `SPTTOTSAVE` Time-averaged values of the Be sputtering fluxes (in $A.m^{-2}$)
5. `CONBE` Proportion of wall element surface coated with Be
6. `DENSBE` Surface concentration of Be on the wall element (in m^{-2})

Only the last block of 6 lines of the `conbe.dat` file needs to be kept from one run to the next. However, the number of wall elements to which the model is applied should remain the same, or the `conbe.dat` file should be modified by hand to reflect the new number of wall elements, or removed, in which case the model restarts from scratch.

Chapter 5

Running the coupled version of B2.5-Eirene

In the run directory, you should execute the command

```
setup_baserun_eirene_links
```

if using the formatted version of the `Eirene` input file (`input.dat`), or

```
setup_baserun_eirene_json_links
```

if using the JSON formatted `eirene.input.json` file, to establish the links `Eirene` needs to the relevant input and atomic data files, as well as some links needed for diagnostic scripts. Using the `-r` option with these scripts creates relative links. Default behaviour is to create absolute links.

If only the `input.dat` file is present (or is used), `Eirene` will translate it to the JSON format as part of the run. The input file that `Eirene` actually uses is the one to which the symbolic link `fort.1` file points.

5.1 Input files

Below are given the input files used for the `D+C+He` benchmark case that can be found on the SVN server. These files can be found in the *Examples* folder in the **SOLPS-ITER** SharePoint web site (https://sharepoint.iter.org/departments/POP/CM/IMAS/SOLPS-ITER), or directly among the examples provided in `${SOLPSTOP}/runs/examples`, as part of the `AUG_16151_D+C+He` case.

5.1.1 b2ag.dat

```
*dimens          (nx, ny; free format)
  96  36   96  36
*param           (param(0:99); free format)
  -1.0, 0.0, 0.75, 1.0, -0.010, 0.0, 0.040,
  5.0, 1.0, 0.02, 0.05, 0.05, 0.05,
  0.06, 1.0,
  100*0.0
'b2agfs_geometry'      '16151.4000.AUGD.EQI.00._96x36.geo'
'b2agfs_min_pitch'     '0.01'
```

5.1.2 b2ai.dat

```
*dimens          (ns; free format)
   12
*label           (lblps, character*60; free format)
```

```
  'Flat initial state for D + C + He, all charge states'
*specs      za       zn       am      za2
   '0'     0.0      1.0      2.0      0.0
   '1'     1.0      1.0      2.0      1.0
   '2'     0.0      6.0     12.011    0.0
   '3'     1.0      6.0     12.011    1.0
   '4'     2.0      6.0     12.011    4.0
   '5'     3.0      6.0     12.011    9.0
   '6'     4.0      6.0     12.011   16.0
   '7'     5.0      6.0     12.011   25.0
   '8'     6.0      6.0     12.011   36.0
   '9'     0.0      2.0      4.003    0.0
  '10'     1.0      2.0      4.003    1.0
  '11'     2.0      2.0      4.003    4.0
*naini          (na0(0:ns-1); free format)
  1.0e12  1.0e20  7*1.0e16  3*6.0e16  100*1.0e0
*ttini          (tt0; free format)
   1.0e2
```

5.1.3 b2ar.dat

```
*tlohi          (tlo, thi; real, free format)
  1.0e-1  1.0e4
*nlohi          (nlo, nhi; real, free format)
  1.0e18  1.0e22
*numnuc         (nnuc; integer, free format)
   3
*nucspec        (nz, izlo, izhi; integers, one triplet per line)
   1  0  1
   6  0  6
   2  0  2
*flag           (either 'adpak' or 'strahl', free format)
  'strahl'
*tailep         (real, free format)
  0.08
```

5.1.4 b2ah.dat

```
*dimens         (ns; free format)
  12
*label          (lblps, character*60; free format)
  'Species D + C +He'
*b2cmpa         basic parameters
*specs      za       zn       am      za2
   '0'     0.0      1.0      2.0      0.0
   '1'     1.0      1.0      2.0      1.0
   '2'     0.0      6.0     12.011    0.0
   '3'     1.0      6.0     12.011    1.0
   '4'     2.0      6.0     12.011    4.0
   '5'     3.0      6.0     12.011    9.0
   '6'     4.0      6.0     12.011   16.0
   '7'     5.0      6.0     12.011   25.0
   '8'     6.0      6.0     12.011   36.0
   '9'     0.0      2.0      4.003    0.0
  '10'     1.0      2.0      4.003    1.0
  '11'     2.0      2.0      4.003    4.0
*b2cmpb         boundary conditions
*cbregs
 'south'    3       0.25     0.75     2.00
 'north'    1       2.0
 'west'     1       2.0
 'east'     1       2.0
*region
```

```
 0  'south, private flux'
*cbsna     (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0    -1.0e-02       0         0         0         0         0
   '1'       0         0    -1.0e-03       0         0         0         0         0
   '2'       0         0           0       0         0         0         0         0
   '3'       0         0    -1.0e-03       0         0         0         0         0
   '4'       0         0    -1.0e-03       0         0         0         0         0
   '5'       0         0    -1.0e-03       0         0         0         0         0
   '6'       0         0    -1.0e-03       0         0         0         0         0
   '7'       0         0    -1.0e-03       0         0         0         0         0
   '8'       0         0    -1.0e-03       0         0         0         0         0
   '9'       0         0    -1.0e-02       0         0         0         0         0
  '10'       0         0    -1.0e-03       0         0         0         0         0
  '11'       0         0    -1.0e-03       0         0         0         0         0
*cbsmo     (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0    -1.0e+10       0         0      -1.0         0
   '1'       0         0         0           0       0         0      -1.0         0
   '2'       0         0         0    -1.0e+10       0         0      -1.0         0
   '3'       0         0         0           0       0         0      -1.0         0
   '4'       0         0         0           0       0         0      -1.0         0
   '5'       0         0         0           0       0         0      -1.0         0
   '6'       0         0         0           0       0         0      -1.0         0
   '7'       0         0         0           0       0         0      -1.0         0
   '8'       0         0         0           0       0         0      -1.0         0
   '9'       0         0         0    -1.0e+10       0         0      -1.0         0
  '10'       0         0         0           0       0         0      -1.0         0
  '11'       0         0         0           0       0         0      -1.0         0
*cbshi     (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'  1.00e-18  -100.0  -1.0e-01       0         0         0         0      -2.5
   '1'  1.00e-18  -100.0  -1.0e-02       0         0         0         0      -2.5
   '2'  1.00e-18  -100.0  -1.0e-01       0         0         0         0      -2.5
   '3'  1.00e-18  -100.0  -1.0e-02       0         0         0         0      -2.5
   '4'  1.00e-18  -100.0  -1.0e-02       0         0         0         0      -2.5
   '5'  1.00e-18  -100.0  -1.0e-02       0         0         0         0      -2.5
   '6'  1.00e-18  -100.0  -1.0e-02       0         0         0         0      -2.5
   '7'  1.00e-18  -100.0  -1.0e-02       0         0         0         0      -2.5
   '8'  1.00e-18  -100.0  -1.0e-02       0         0         0         0      -2.5
   '9'  1.00e-18  -100.0  -1.0e-01       0         0         0         0      -2.5
  '10'  1.00e-18  -100.0  -1.0e-02       0         0         0         0      -2.5
  '11'  1.00e-18  -100.0  -1.0e-02       0         0         0         0      -2.5
*cbshe     (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'  1.00e-18  -100.0  -1.0e-04       0         0         0         0      -2.5
*cbsch     (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'       0         0         0         0         0         0         0         0
*cbrec     (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0         0         0         0         0         0
   '1'     1.0       0.3         0         0         0         0         0         0
   '2'       0         0         0         0         0         0         0         0
   '3'       0         0         0         0         0         0         0         0
   '4'       0         0         0         0         0         0         0         0
   '5'       0         0         0         0         0         0         0         0
   '6'       0         0         0         0         0         0         0         0
   '7'       0         0         0         0         0         0         0         0
   '8'       0         0         0         0         0         0         0         0
   '9'       0         0         0         0         0         0         0         0
  '10'     1.0       0.3         0         0         0         0         0         0
  '11'     1.0       0.3         0         0         0         0         0         0
*cbmsa     (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'       0         0         0         0         0         0         0         0
*cbmsb     (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0         0         0         0         0         0
   '1'       0         0         0         0         0         0         0         0
   '2'       0         0         0         0         0         0         0         0
   '3'       0         0         0         0         0         0         0         0
   '4'       0         0         0         0         0         0         0         0
   '5'       0         0         0         0         0         0         0         0
   '6'       0         0         0         0         0         0         0         0
```

```
   '7'       0         0         0         0         0         0         0         0
   '8'       0         0         0         0         0         0         0         0
   '9'       0         0         0         0         0         0         0         0
  '10'       0         0         0         0         0         0         0         0
  '11'       0         0         0         0         0         0         0         0
*region
 1  'south, core plasma'
*cbsna      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0    -1.0e+10       0         0         0         0         0         0
   '1'  4.00e+29 -1.0e+10       0         0         0         0         0         0
   '2'       0         0         0         0         0         0         0         0
   '3'       0         0         0         0         0         0         0         0
   '4'       0         0         0         0         0         0         0         0
   '5'       0         0         0         0         0         0         0         0
   '6'       0         0         0         0         0         0         0         0
   '7'       0         0         0         0         0         0         0         0
   '8'       0         0         0         0         0         0         0         0
   '9'       0    -1.0e+10       0         0         0         0         0         0
  '10'       0    -1.0e+10       0         0         0         0         0         0
  '11'  1.00e+26 -1.0e+10       0         0         0         0         0         0
*cbsmo      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0    -1.0e+10       0         0         0         0
   '1'       0         0         0    -1.0e+10       0         0         0         0
   '2'       0         0         0    -1.0e+10       0         0         0         0
   '3'       0         0         0    -1.0e+10       0         0         0         0
   '4'       0         0         0    -1.0e+10       0         0         0         0
   '5'       0         0         0    -1.0e+10       0         0         0         0
   '6'       0         0         0    -1.0e+10       0         0         0         0
   '7'       0         0         0    -1.0e+10       0         0         0         0
   '8'       0         0         0    -1.0e+10       0         0         0         0
   '9'       0         0         0    -1.0e+10       0         0         0         0
  '10'       0         0         0    -1.0e+10       0         0         0         0
  '11'       0         0         0    -1.0e+10       0         0         0         0
*cbshi      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'  5.15e+03      0         0         0         0         0         0        -2.5
   '1'  5.15e+03      0         0         0         0         0         0        -2.5
   '2'  5.15e+03      0         0         0         0         0         0        -2.5
   '3'  5.15e+03      0         0         0         0         0         0        -2.5
   '4'  5.15e+03      0         0         0         0         0         0        -2.5
   '5'  5.15e+03      0         0         0         0         0         0        -2.5
   '6'  5.15e+03      0         0         0         0         0         0        -2.5
   '7'  5.15e+03      0         0         0         0         0         0        -2.5
   '8'  5.15e+03      0         0         0         0         0         0        -2.5
   '9'  5.15e+03      0         0         0         0         0         0        -2.5
  '10'  5.15e+03      0         0         0         0         0         0        -2.5
  '11'  5.15e+03      0         0         0         0         0         0        -2.5
*cbshe      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'  5.15e+03      0         0         0         0         0         0        -2.5
*cbsch      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'       0         0         0         0         0         0         0         0
*cbrec      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0         0         0         0         0         0
   '1'       0         0         0         0         0         0         0         0
   '2'       0         0         0         0         0         0         0         0
   '3'       0         0         0         0         0         0         0         0
   '4'       0         0         0         0         0         0         0         0
   '5'       0         0         0         0         0         0         0         0
   '6'       0         0         0         0         0         0         0         0
   '7'       0         0         0         0         0         0         0         0
   '8'       0         0         0         0         0         0         0         0
   '9'       0         0         0         0         0         0         0         0
  '10'       0         0         0         0         0         0         0         0
  '11'       0         0         0         0         0         0         0         0
*cbmsa      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'       0         0         0         0         0         0         0         0
*cbmsb      (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'       0         0         0         0         0         0         0         0
```

```
   '1'      0        0        0        0        0        0        0        0
   '2'      0        0        0        0        0        0        0        0
   '3'      0        0        0        0        0        0        0        0
   '4'      0        0        0        0        0        0        0        0
   '5'      0        0        0        0        0        0        0        0
   '6'      0        0        0        0        0        0        0        0
   '7'      0        0        0        0        0        0        0        0
   '8'      0        0        0        0        0        0        0        0
   '9'      0        0        0        0        0        0        0        0
  '10'      0        0        0        0        0        0        0        0
  '11'      0        0        0        0        0        0        0        0
*region
 2  'south, private flux'
*cbsna     (0)      (1)      (2)      (3)      (4)      (5)      (6)      (7)
   '0'      0        0    -1.0e-02     0        0        0        0        0
   '1'      0        0    -1.0e-03     0        0        0        0        0
   '2'      0        0        0        0        0        0        0        0
   '3'      0        0    -1.0e-03     0        0        0        0        0
   '4'      0        0    -1.0e-03     0        0        0        0        0
   '5'      0        0    -1.0e-03     0        0        0        0        0
   '6'      0        0    -1.0e-03     0        0        0        0        0
   '7'      0        0    -1.0e-03     0        0        0        0        0
   '8'      0        0    -1.0e-03     0        0        0        0        0
   '9'      0        0    -1.0e-02     0        0        0        0        0
  '10'      0        0    -1.0e-03     0        0        0        0        0
  '11'      0        0    -1.0e-03     0        0        0        0        0
*cbsmo     (0)      (1)      (2)      (3)      (4)      (5)      (6)      (7)
   '0'      0        0        0    -1.0e+10     0        0      -1.0       0
   '1'      0        0        0        0        0        0      -1.0       0
   '2'      0        0        0    -1.0e+10     0        0      -1.0       0
   '3'      0        0        0        0        0        0      -1.0       0
   '4'      0        0        0        0        0        0      -1.0       0
   '5'      0        0        0        0        0        0      -1.0       0
   '6'      0        0        0        0        0        0      -1.0       0
   '7'      0        0        0        0        0        0      -1.0       0
   '8'      0        0        0        0        0        0      -1.0       0
   '9'      0        0        0    -1.0e+10     0        0      -1.0       0
  '10'      0        0        0        0        0        0      -1.0       0
  '11'      0        0        0        0        0        0      -1.0       0
*cbshi     (0)      (1)      (2)      (3)      (4)      (5)      (6)      (7)
   '0'  1.00e-18  -100.0  -1.0e-01     0        0        0        0      -2.5
   '1'  1.00e-18  -100.0  -1.0e-02     0        0        0        0      -2.5
   '2'  1.00e-18  -100.0  -1.0e-01     0        0        0        0      -2.5
   '3'  1.00e-18  -100.0  -1.0e-02     0        0        0        0      -2.5
   '4'  1.00e-18  -100.0  -1.0e-02     0        0        0        0      -2.5
   '5'  1.00e-18  -100.0  -1.0e-02     0        0        0        0      -2.5
   '6'  1.00e-18  -100.0  -1.0e-02     0        0        0        0      -2.5
   '7'  1.00e-18  -100.0  -1.0e-02     0        0        0        0      -2.5
   '8'  1.00e-18  -100.0  -1.0e-02     0        0        0        0      -2.5
   '9'  1.00e-18  -100.0  -1.0e-01     0        0        0        0      -2.5
  '10'  1.00e-18  -100.0  -1.0e-02     0        0        0        0      -2.5
  '11'  1.00e-18  -100.0  -1.0e-02     0        0        0        0      -2.5
*cbshe     (0)      (1)      (2)      (3)      (4)      (5)      (6)      (7)
  '-1'  1.00e-18  -100.0  -1.0e-04     0        0        0        0      -2.5
*cbsch     (0)      (1)      (2)      (3)      (4)      (5)      (6)      (7)
  '-1'      0        0        0        0        0        0        0        0
*cbrec     (0)      (1)      (2)      (3)      (4)      (5)      (6)      (7)
   '0'      0        0        0        0        0        0        0        0
   '1'     1.0      0.3       0        0        0        0        0        0
   '2'      0        0        0        0        0        0        0        0
   '3'      0        0        0        0        0        0        0        0
   '4'      0        0        0        0        0        0        0        0
   '5'      0        0        0        0        0        0        0        0
   '6'      0        0        0        0        0        0        0        0
   '7'      0        0        0        0        0        0        0        0
   '8'      0        0        0        0        0        0        0        0
   '9'      0        0        0        0        0        0        0        0
```

```
  '10'     1.0       0.3       0         0         0         0         0         0
  '11'     1.0       0.3       0         0         0         0         0         0
*cbmsa     (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'     0         0         0         0         0         0         0         0
*cbmsb     (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'     0         0         0         0         0         0         0         0
   '1'     0         0         0         0         0         0         0         0
   '2'     0         0         0         0         0         0         0         0
   '3'     0         0         0         0         0         0         0         0
   '4'     0         0         0         0         0         0         0         0
   '5'     0         0         0         0         0         0         0         0
   '6'     0         0         0         0         0         0         0         0
   '7'     0         0         0         0         0         0         0         0
   '8'     0         0         0         0         0         0         0         0
   '9'     0         0         0         0         0         0         0         0
  '10'     0         0         0         0         0         0         0         0
  '11'     0         0         0         0         0         0         0         0
*region
 3  'north, SOL/DIV outer wall'
*cbsna     (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'     0         0     -1.0e-02      0         0         0         0         0
   '1'     0         0     -1.0e-03      0         0         0         0         0
   '2'     0         0         0         0         0         0         0         0
   '3'     0         0     -1.0e-03      0         0         0         0         0
   '4'     0         0     -1.0e-03      0         0         0         0         0
   '5'     0         0     -1.0e-03      0         0         0         0         0
   '6'     0         0     -1.0e-03      0         0         0         0         0
   '7'     0         0     -1.0e-03      0         0         0         0         0
   '8'     0         0     -1.0e-03      0         0         0         0         0
   '9'     0         0     -1.0e-02      0         0         0         0         0
  '10'     0         0     -1.0e-03      0         0         0         0         0
  '11'     0         0     -1.0e-03      0         0         0         0         0
*cbsmo     (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'     0         0         0     -1.0e+10      0         0       -1.0        0
   '1'     0         0         0         0         0         0       -1.0        0
   '2'     0         0         0     -1.0e+10      0         0       -1.0        0
   '3'     0         0         0         0         0         0       -1.0        0
   '4'     0         0         0         0         0         0       -1.0        0
   '5'     0         0         0         0         0         0       -1.0        0
   '6'     0         0         0         0         0         0       -1.0        0
   '7'     0         0         0         0         0         0       -1.0        0
   '8'     0         0         0         0         0         0       -1.0        0
   '9'     0         0         0     -1.0e+10      0         0       -1.0        0
  '10'     0         0         0         0         0         0       -1.0        0
  '11'     0         0         0         0         0         0       -1.0        0
*cbshi     (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'  1.00e-18  -100.0   -1.0e-01      0         0         0         0       -2.5
   '1'  1.00e-18  -100.0   -1.0e-02      0         0         0         0       -2.5
   '2'  1.00e-18  -100.0   -1.0e-01      0         0         0         0       -2.5
   '3'  1.00e-18  -100.0   -1.0e-02      0         0         0         0       -2.5
   '4'  1.00e-18  -100.0   -1.0e-02      0         0         0         0       -2.5
   '5'  1.00e-18  -100.0   -1.0e-02      0         0         0         0       -2.5
   '6'  1.00e-18  -100.0   -1.0e-02      0         0         0         0       -2.5
   '7'  1.00e-18  -100.0   -1.0e-02      0         0         0         0       -2.5
   '8'  1.00e-18  -100.0   -1.0e-02      0         0         0         0       -2.5
   '9'  1.00e-18  -100.0   -1.0e-01      0         0         0         0       -2.5
  '10'  1.00e-18  -100.0   -1.0e-02      0         0         0         0       -2.5
  '11'  1.00e-18  -100.0   -1.0e-02      0         0         0         0       -2.5
*cbshe     (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'  1.00e-18  -100.0   -1.0e-04      0         0         0         0       -2.5
*cbsch     (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
  '-1'     0         0         0         0         0         0         0         0
*cbrec     (0)       (1)       (2)       (3)       (4)       (5)       (6)       (7)
   '0'     0         0         0         0         0         0         0         0
   '1'     1.0       0.3       0         0         0         0         0         0
   '2'     0         0         0         0         0         0         0         0
   '3'     0         0         0         0         0         0         0         0
```

```
   '4'      0        0        0        0        0        0        0        0
   '5'      0        0        0        0        0        0        0        0
   '6'      0        0        0        0        0        0        0        0
   '7'      0        0        0        0        0        0        0        0
   '8'      0        0        0        0        0        0        0        0
   '9'      0        0        0        0        0        0        0        0
  '10'    1.0      0.3        0        0        0        0        0        0
  '11'    1.0      0.3        0        0        0        0        0        0
*cbmsa     (0)      (1)      (2)      (3)      (4)      (5)      (6)      (7)
  '-1'     0        0        0        0        0        0        0        0
*cbmsb     (0)      (1)      (2)      (3)      (4)      (5)      (6)      (7)
   '0'      0        0        0        0        0        0        0        0
   '1'      0        0        0        0        0        0        0        0
   '2'      0        0        0        0        0        0        0        0
   '3'      0        0        0        0        0        0        0        0
   '4'      0        0        0        0        0        0        0        0
   '5'      0        0        0        0        0        0        0        0
   '6'      0        0        0        0        0        0        0        0
   '7'      0        0        0        0        0        0        0        0
   '8'      0        0        0        0        0        0        0        0
   '9'      0        0        0        0        0        0        0        0
  '10'      0        0        0        0        0        0        0        0
  '11'      0        0        0        0        0        0        0        0
*region
 4  'west, target plate'
*cbsna     (0)      (1)      (2)      (3)      (4)      (5)      (6)      (7)
   '0'      0        0        0        0        0        0        0     -1.0
   '1'      0        0        0        0        0        0        0     -1.0
   '2'      0        0        0        0        0        0        0     -1.0
   '3'      0        0        0        0        0        0        0     -1.0
   '4'      0        0        0        0        0        0        0     -1.0
   '5'      0        0        0        0        0        0        0     -1.0
   '6'      0        0        0        0        0        0        0     -1.0
   '7'      0        0        0        0        0        0        0     -1.0
   '8'      0        0        0        0        0        0        0     -1.0
   '9'      0        0        0        0        0        0        0     -1.0
  '10'      0        0        0        0        0        0        0     -1.0
  '11'      0        0        0        0        0        0        0     -1.0
*cbsmo     (0)      (1)      (2)      (3)      (4)      (5)      (6)      (7)
   '0'      0        0        0        0   -1.0e+10      0        0        0
   '1'      0        0   1.0e+10       0   -1.0e+10      0        0        0
   '2'      0        0        0        0   -1.0e+10      0        0        0
   '3'      0        0   1.0e+10       0   -1.0e+10      0        0        0
   '4'      0        0   1.0e+10       0   -1.0e+10      0        0        0
   '5'      0        0   1.0e+10       0   -1.0e+10      0        0        0
   '6'      0        0   1.0e+10       0   -1.0e+10      0        0        0
   '7'      0        0   1.0e+10       0   -1.0e+10      0        0        0
   '8'      0        0   1.0e+10       0   -1.0e+10      0        0        0
   '9'      0        0        0        0   -1.0e+10      0        0        0
  '10'      0        0   1.0e+10       0   -1.0e+10      0        0        0
  '11'      0        0   1.0e+10       0   -1.0e+10      0        0        0
*cbshi     (0)      (1)      (2)      (3)      (4)      (5)      (6)      (7)
   '0'  1.00e-18  -100.0      0        0        0        0        0     -1.5
   '1'  1.00e-18  -100.0      0     -0.1        0        0        0     -1.5
   '2'  1.00e-18  -100.0      0        0        0        0        0     -1.5
   '3'  1.00e-18  -100.0      0     -0.1        0        0        0     -1.5
   '4'  1.00e-18  -100.0      0     -0.1        0        0        0     -1.5
   '5'  1.00e-18  -100.0      0     -0.1        0        0        0     -1.5
   '6'  1.00e-18  -100.0      0     -0.1        0        0        0     -1.5
   '7'  1.00e-18  -100.0      0     -0.1        0        0        0     -1.5
   '8'  1.00e-18  -100.0      0     -0.1        0        0        0     -1.5
   '9'  1.00e-18  -100.0      0        0        0        0        0     -1.5
  '10'  1.00e-18  -100.0      0     -0.1        0        0        0     -1.5
  '11'  1.00e-18  -100.0      0     -0.1        0        0        0     -1.5
*cbshe     (0)      (1)      (2)      (3)      (4)      (5)      (6)      (7)
  '-1'  1.00e-18  -100.0      0     -0.1        0        0        0     -1.5
*cbsch     (0)      (1)      (2)      (3)      (4)      (5)      (6)      (7)
```

'-1'	0	0	0	0	0	0	1.0	0.5
*cbrec	(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)
'0'	0	0	0	0	0	0	0	0
'1'	1.0	0.3	0	0	0	0	0	0
'2'	0	0	0	0	0	0	0	0
'3'	0	0	0	0	0	0	0	0
'4'	0	0	0	0	0	0	0	0
'5'	0	0	0	0	0	0	0	0
'6'	0	0	0	0	0	0	0	0
'7'	0	0	0	0	0	0	0	0
'8'	0	0	0	0	0	0	0	0
'9'	0	0	0	0	0	0	0	0
'10'	1.0	0.3	0	0	0	0	0	0
'11'	1.0	0.3	0	0	0	0	0	0
*cbmsa	(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)
'-1'	0	0	0	0	0	0	0	0
*cbmsb	(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)
'0'	0	0	0	0	0	0	0	0
'1'	0	0	0	0	0	0	0	0
'2'	0	0	0	0	0	0	0	0
'3'	0	0	0	0	0	0	0	0
'4'	0	0	0	0	0	0	0	0
'5'	0	0	0	0	0	0	0	0
'6'	0	0	0	0	0	0	0	0
'7'	0	0	0	0	0	0	0	0
'8'	0	0	0	0	0	0	0	0
'9'	0	0	0	0	0	0	0	0
'10'	0	0	0	0	0	0	0	0
'11'	0	0	0	0	0	0	0	0
*region								
5 'east, target plate'								
*cbsna	(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)
'0'	0	0	0	0	0	0	0	-1.0
'1'	0	0	0	0	0	0	0	-1.0
'2'	0	0	0	0	0	0	0	-1.0
'3'	0	0	0	0	0	0	0	-1.0
'4'	0	0	0	0	0	0	0	-1.0
'5'	0	0	0	0	0	0	0	-1.0
'6'	0	0	0	0	0	0	0	-1.0
'7'	0	0	0	0	0	0	0	-1.0
'8'	0	0	0	0	0	0	0	-1.0
'9'	0	0	0	0	0	0	0	-1.0
'10'	0	0	0	0	0	0	0	-1.0
'11'	0	0	0	0	0	0	0	-1.0
*cbsmo	(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)
'0'	0	0	0	0	-1.0e+10	0	0	0
'1'	0	0	1.0e+10	0	-1.0e+10	0	0	0
'2'	0	0	0	0	-1.0e+10	0	0	0
'3'	0	0	1.0e+10	0	-1.0e+10	0	0	0
'4'	0	0	1.0e+10	0	-1.0e+10	0	0	0
'5'	0	0	1.0e+10	0	-1.0e+10	0	0	0
'6'	0	0	1.0e+10	0	-1.0e+10	0	0	0
'7'	0	0	1.0e+10	0	-1.0e+10	0	0	0
'8'	0	0	1.0e+10	0	-1.0e+10	0	0	0
'9'	0	0	0	0	-1.0e+10	0	0	0
'10'	0	0	1.0e+10	0	-1.0e+10	0	0	0
'11'	0	0	1.0e+10	0	-1.0e+10	0	0	0
*cbshi	(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)
'0'	1.00e-18	-100.0	0	0	0	0	0	-1.5
'1'	1.00e-18	-100.0	0	-0.1	0	0	0	-1.5
'2'	1.00e-18	-100.0	0	0	0	0	0	-1.5
'3'	1.00e-18	-100.0	0	-0.1	0	0	0	-1.5
'4'	1.00e-18	-100.0	0	-0.1	0	0	0	-1.5
'5'	1.00e-18	-100.0	0	-0.1	0	0	0	-1.5
'6'	1.00e-18	-100.0	0	-0.1	0	0	0	-1.5
'7'	1.00e-18	-100.0	0	-0.1	0	0	0	-1.5
'8'	1.00e-18	-100.0	0	-0.1	0	0	0	-1.5

```
   '9'  1.00e-18  -100.0      0        0        0        0        0      -1.5
  '10'  1.00e-18  -100.0      0      -0.1       0        0        0      -1.5
  '11'  1.00e-18  -100.0      0      -0.1       0        0        0      -1.5
*cbshe      (0)       (1)      (2)      (3)      (4)      (5)      (6)      (7)
  '-1'  1.00e-18  -100.0      0      -0.1       0        0        0      -1.5
*cbsch      (0)       (1)      (2)      (3)      (4)      (5)      (6)      (7)
  '-1'        0         0        0        0        0        0      1.0      0.5
*cbrec      (0)       (1)      (2)      (3)      (4)      (5)      (6)      (7)
   '0'        0         0        0        0        0        0        0        0
   '1'      1.0       0.3        0        0        0        0        0        0
   '2'        0         0        0        0        0        0        0        0
   '3'        0         0        0        0        0        0        0        0
   '4'        0         0        0        0        0        0        0        0
   '5'        0         0        0        0        0        0        0        0
   '6'        0         0        0        0        0        0        0        0
   '7'        0         0        0        0        0        0        0        0
   '8'        0         0        0        0        0        0        0        0
   '9'        0         0        0        0        0        0        0        0
  '10'      1.0       0.3        0        0        0        0        0        0
  '11'      1.0       0.3        0        0        0        0        0        0
*cbmsa      (0)       (1)      (2)      (3)      (4)      (5)      (6)      (7)
  '-1'        0         0        0        0        0        0        0        0
*cbmsb      (0)       (1)      (2)      (3)      (4)      (5)      (6)      (7)
   '0'        0         0        0        0        0        0        0        0
   '1'        0         0        0        0        0        0        0        0
   '2'        0         0        0        0        0        0        0        0
   '3'        0         0        0        0        0        0        0        0
   '4'        0         0        0        0        0        0        0        0
   '5'        0         0        0        0        0        0        0        0
   '6'        0         0        0        0        0        0        0        0
   '7'        0         0        0        0        0        0        0        0
   '8'        0         0        0        0        0        0        0        0
   '9'        0         0        0        0        0        0        0        0
  '10'        0         0        0        0        0        0        0        0
  '11'        0         0        0        0        0        0        0        0
*b2cmpt  transport coefficients
*cfdf0      (0)       (1)      (2)      (3)      (4)      (5)      (6)      (7)
   '0'   1.0e+02        0      1.0   5.0e-20  1.5e-18        0        0        0
   '1'      0.5         0        0        0        0        0        0        0
   '2'   1.0e+02        0      1.0   2.0e-20  5.0e-20        0        0        0
   '3'      0.5         0        0        0        0        0        0        0
   '4'      0.5         0        0        0        0        0        0        0
   '5'      0.5         0        0        0        0        0        0        0
   '6'      0.5         0        0        0        0        0        0        0
   '7'      0.5         0        0        0        0        0        0        0
   '8'      0.5         0        0        0        0        0        0        0
   '9'   1.0e+02        0      1.0   5.0e-20  1.5e-18        0        0        0
  '10'      0.5         0        0        0        0        0        0        0
  '11'      0.5         0        0        0        0        0        0        0
*cfdna      (0)       (1)      (2)      (3)      (4)      (5)      (6)      (7)
   '0'        0         0        0        0        0        0        0        0
   '1'        0         0      1.0        0        0        0        0        0
   '2'        0         0        0        0        0        0        0        0
   '3'        0         0      1.0        0        0        0        0        0
   '4'        0         0      1.0        0        0        0        0        0
   '5'        0         0      1.0        0        0        0        0        0
   '6'        0         0      1.0        0        0        0        0        0
   '7'        0         0      1.0        0        0        0        0        0
   '8'        0         0      1.0        0        0        0        0        0
   '9'        0         0        0        0        0        0        0        0
  '10'        0         0      1.0        0        0        0        0        0
  '11'        0         0      1.0        0        0        0        0        0
*cfdpa      (0)       (1)      (2)      (3)      (4)      (5)      (6)      (7)
   '0'        0         0      1.0        0        0        0        0        0
   '1'        0         0        0        0        0        0        0        0
   '2'        0         0      1.0        0        0        0        0        0
   '3'        0         0        0        0        0        0        0        0
```

```
   '4'      0        0        0        0        0        0        0        0
   '5'      0        0        0        0        0        0        0        0
   '6'      0        0        0        0        0        0        0        0
   '7'      0        0        0        0        0        0        0        0
   '8'      0        0        0        0        0        0        0        0
   '9'      0        0      1.0        0        0        0        0        0
  '10'      0        0        0        0        0        0        0        0
  '11'      0        0        0        0        0        0        0        0
*cfvla     (0)      (1)      (2)      (3)      (4)      (5)      (6)      (7)
   '0'      0        0        0        0        0        0        0        0
   '1'      0        0        0        0        0        0        0        0
   '2'      0        0        0        0        0        0        0        0
   '3'      0        0        0        0        0        0        0        0
   '4'      0        0        0        0        0        0        0        0
   '5'      0        0        0        0        0        0        0        0
   '6'      0        0        0        0        0        0        0        0
   '7'      0        0        0        0        0        0        0        0
   '8'      0        0        0        0        0        0        0        0
   '9'      0        0        0        0        0        0        0        0
  '10'      0        0        0        0        0        0        0        0
  '11'      0        0        0        0        0        0        0        0
*cfvsa     (0)      (1)      (2)      (3)      (4)      (5)      (6)      (7)
   '0'      0        0      1.0        0        0        0        0        0
   '1'      0        0      1.0        0        0        0        0        0
   '2'      0        0      1.0        0        0        0        0        0
   '3'      0        0      1.0        0        0        0        0        0
   '4'      0        0      1.0        0        0        0        0        0
   '5'      0        0      1.0        0        0        0        0        0
   '6'      0        0      1.0        0        0        0        0        0
   '7'      0        0      1.0        0        0        0        0        0
   '8'      0        0      1.0        0        0        0        0        0
   '9'      0        0      1.0        0        0        0        0        0
  '10'      0        0      1.0        0        0        0        0        0
  '11'      0        0      1.0        0        0        0        0        0
*cfhci     (0)      (1)      (2)      (3)      (4)      (5)      (6)      (7)
   '0'      0        0      1.4        0        0        0        0        0
   '1'      0        0      1.4        0        0        0        0        0
   '2'      0        0      1.4        0        0        0        0        0
   '3'      0        0      1.4        0        0        0        0        0
   '4'      0        0      1.4        0        0        0        0        0
   '5'      0        0      1.4        0        0        0        0        0
   '6'      0        0      1.4        0        0        0        0        0
   '7'      0        0      1.4        0        0        0        0        0
   '8'      0        0      1.4        0        0        0        0        0
   '9'      0        0      1.4        0        0        0        0        0
  '10'      0        0      1.4        0        0        0        0        0
  '11'      0        0      1.4        0        0        0        0        0
*cfhce     (0)      (1)      (2)      (3)      (4)      (5)      (6)      (7)
  '-1'     0.7        0        0        0        0        0        0        0
*cfsig     (0)      (1)      (2)      (3)      (4)      (5)      (6)      (7)
  '-1'   1.0e-07      0        0        0        0        0        0        0
*cfalf     (0)      (1)      (2)      (3)      (4)      (5)      (6)      (7)
  '-1'       0        0        0        0        0        0        0        0
*cflim    cflme    cflmi    cflmv    cflal    cflab     ...
  '-1'     0.15     0.15      0.5      0.5      0.5        0        0        0
*endphy
```

5.1.5 b2mn.dat

```
*label          (lblmn: character*60)
  'D+C+He 15190 3.5s L-mode SNU'
*b2cmpa  basic parameters
*b2cmpb  boundary conditions
*b2cmpt  transport coefficients
*cflim     cflme    cflmi    cflmv    cflal    cflab     ...
```

```
  '-1'      0.3     10.0      0.5      0.5      0.5       0        0        0
*endphy
'b2mndr_ntim'     '100'
'b2mndr_dtim'     '1.0e-4'
'b2mndt_nstg0'    '1'
'b2mndt_nstg1'    '1'
'b2mndt_nstg2'    '10'
'b2mndt_rxf'      '0.5'
'b2mndr_mvinc'    '1'
'b2mndr_mvnum'    '0'
'b2mndr_b2time'   '1'
'b2mndr_cpu'      '82800.0'
'b2mndr_savecpu'  '0.0'
'tallies_netcdf'  '1'
'b2ux5p_style'    '2'
'b2ux5p_nltrsol'  '2'
'b2stbc_sna0ep'   '1.0e-10'
'b2trcl_lluciani' '1'
'b2trcl_lvis21' '0'
'b2trcl_lthf21' '0'
'b2news_facdrift_target' '0'
'b2news_facdrift_start'  '0'
'b2news_facdrift_inc'    '1.00'
'b2news_facdrift_dec'    '0.5'
'b2news_potit'    '100'
'b2news_potok'    '1e-2'
'b2sihs_phm3'    '1.0'
'b2stbr_sput_frc'        '0.0100'
'b2stbr_sput_phys'       '1.00'
'b2stbr_sput_chem_model'   '1'
'b2stbr_sput_dst'          '2'
'b2stbc_boundary_namelist' '1'
'b2srdt_numerics_namelist' '1'
'b2tqna_transport_namelist' '1'
'b2tqna_new_df0'  '0'
'b2mndr_eirene'                    '1'
'eirene_repeat_first_call'         '1'
'b2mndr_rescale_neutrals'          '1e-0'
'b2mndr_rescale_neutrals_sources' '1e-10'
'b2mwti_jxi'  39
'b2mwti_jxa'  54
'b2news_recalculate_contributions' '0'
'eirene_savef31' '0'
'eirene_dpc_fix'  '1'
```

To activate a coupled run, the master switch is `'b2mndr_eirene'` which should be set to `'1'`. In addition, the densities of the fluid neutral species need to be scaled down, along with the sources that affect them. This is achieved, respectively, by use of the switches `'b2mndr_rescale_neutrals'` and `'b2mndr_rescale_neutral_sources'`. Typically, it is recommended to use a value of the order of 10^{-10} for scaling down the fluid neutral sources. For the fluid neutral densities, one can scale them down only on the first run, by the same amount as the sources, and then turn off the rescaling, as they are no longer being replenished.

In order to attempt to improve convergence of coupled **SOLPS-ITER** runs, currently four averaging schemes are available to reduce the impact of the Monte Carlo noise, which is intrinsic to the `Eirene` sources, to the plasma solution (especially concerning the particle balance of impurities, which may be worsened because of bias errors derived from high levels of noise in their `Eirene` source terms). The user may want to activate one of these schemes, although there is no firm recommendation as to which will perform best. The switches for those are:

eirene_neutr_avg The neutral sources are averaged over the last up to $neutr_avg$ `Eirene` calls. After $neutr_avg$ calls, the history is cleared and the average arrays are rebuilt progressively again (ratcheting average scheme).

eirene_underrelax The neutral sources are multiplied by an under-relaxation factor of $1/eirene_underrelax$ and added to the sources from the previous timestep (under-relaxation scheme).

b2stbr_eir_src_nhist The neutral sources are averaged over the last $nhist$ `Eirene` calls (moving average scheme).

eirene_averaging_multiple_calls `Eirene` is called $multiple_calls$ times within a single time step, before a new plasma solution is computed by `B2.5`, and the `Eirene` sources are averaged over all performed calls before being used to compute the new plasma solution.

The first three schemes should be used only when converged steady-state solution is of interest, since sources computed in previous time steps also contribute to the neutral background in use for any current time step. The last scheme, despite coming at the cost of a significant increase in computation time, may also be used for time-dependent simulations.

In addition to those, please see the recommended averaging scheme described in Section 5.6.

5.1.6 b2.boundary.parameters

```
&boundary
NBC=6,
BCCHAR=     'S',    'S',    'S',    'N',    'W',    'E',
BCPOS=       -1,     -1,     -1,     36,     -1,     96,
BCSTART=     24,      0,     72,      0,     -1,     -1,
BCEND=       71,     23,     95,     95,     36,     36,
BCENE=        8,      9,      9,      9,      3,      3,
BCENI=        8,      9,      9,      9,      3,      3,
BCPOT=        0,      0,      0,      0,      3,      3,
ENEPAR=  0.80e6,   0.01,   0.01,   0.01,   0.90,   0.90,    ENEPAR(2,2)=1e2, 1e2, 1e2,
ENIPAR=  0.80e6,   0.01,   0.01,   0.01,   2.50,   2.50,    ENIPAR(2,2)=1e2, 1e2, 1e2,
POTPAR=    0.00,   0.00,   0.00,   0.00,   0.00,   0.00,    POTPAR(1,2)=6*0,
BCCON(0,1)    =      8,     1,     8,     8,     8,     8,     8,     8,     8,     8,     8,     1,
  CONPAR(0,1,1) = 0e19,  2e19,  0e19,  0e19,  0e19,  0e19,  0e19,  0e19,  0e19,  0e19,  0e19,  1e16,
BCCON(0,2)    =     10,    10,    10,    10,    10,    10,    10,    10,    10,    10,    10,    10,
  CONPAR(0,2,1) =-1e-2, -1e-3, -1e-2, -1e-3, -1e-3, -1e-3, -1e-3, -1e-3, -1e-3, -1e-2, -1e-3, -1e-3,
BCCON(0,3)    =     10,    10,    10,    10,    10,    10,    10,    10,    10,    10,    10,    10,
  CONPAR(0,3,1) =-1e-2, -1e-3, -1e-2, -1e-3, -1e-3, -1e-3, -1e-3, -1e-3, -1e-3, -1e-2, -1e-3, -1e-3,
BCCON(0,4)    =     10,    10,    10,    10,    10,    10,    10,    10,    10,    10,    10,    10,
  CONPAR(0,4,1) =-1e-2, -1e-3, -1e-2, -1e-3, -1e-3, -1e-3, -1e-3, -1e-3, -1e-3, -1e-2, -1e-3, -1e-3,
BCCON(0,5)    =      5,     3,     5,     3,     3,     3,     3,     3,     3,     5,     3,     3,
  CONPAR(0,5,1) = 0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,
BCCON(0,6)    =      5,     3,     5,     3,     3,     3,     3,     3,     3,     5,     3,     3,
  CONPAR(0,6,1) = 0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,
BCMOM(0,1)    =      4,     4,     4,     4,     4,     4,     4,     4,     4,     4,     4,     4,
  MOMPAR(0,1,1) = 0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,
  mompar(0,1,2)=12*1e5,
BCMOM(0,2)    =      4,     4,     4,     4,     4,     4,     4,     4,     4,     4,     4,     4,
  MOMPAR(0,2,1) = 0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,
  mompar(0,2,2)=12*1e5,
BCMOM(0,3)    =      4,     4,     4,     4,     4,     4,     4,     4,     4,     4,     4,     4,
  MOMPAR(0,3,1) = 0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,
  mompar(0,3,2)=12*1e5,
BCMOM(0,4)    =      4,     4,     4,     4,     4,     4,     4,     4,     4,     4,     4,     4,
  MOMPAR(0,4,1) = 0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,  0.00,
  mompar(0,4,2)=12*1e5,
BCMOM(0,5)    =      1,     3,     1,     3,     3,     3,     3,     3,     3,     1,     3,     3,
  MOMPAR(0,5,1) = 0.00,  1.00,  0.00,  1.00,  1.00,  1.00,  1.00,  1.00,  1.00,  0.00,  1.00,  1.00,
  mompar(0,5,2)=12*0,
BCMOM(0,6)    =      1,     3,     1,     3,     3,     3,     3,     3,     3,     1,     3,     3,
  MOMPAR(0,6,1) = 0.00,  1.00,  0.00,  1.00,  1.00,  1.00,  1.00,  1.00,  1.00,  0.00,  1.00,  1.00,
  mompar(0,6,2)=12*0,
GAMMAI=1.666666667,GAMMAE=0.5,
LBNDUSR=f,
/
```

5.1.7 b2.numerics.parameters

```
&numerics
dtco(0,1)= 12*1.0, dtco(0,2)=12*1.0, dtco(0,3)=12*1.0, dtco(0,4)=12*1.0,
dtmo(0,1)= 12*1.0, dtmo(0,2)=12*1.0, dtmo(0,3)=12*1.0, dtmo(0,4)=12*1.0,
dtee(1)= 1.0, 1.0, 1.0, 1.0,
dtei(1)= 1.0, 1.0, 1.0, 1.0,
/
```

5.1.8 b2.transport.parameters

```
&transport
flag_dna=1, parm_dna=12*0.4,
flag_dpa=0, parm_dpa=12*0,
flag_vla=0, parm_vla=12*0,
flag_vsa=0, parm_vsa=12*0,
flag_hci=1, parm_hci=12*1.6,
flag_hce=1, parm_hce=1.6,
flag_sig=0, parm_sig=0,
flag_alf=0, parm_alf=0,
/
```

5.1.9 b2.neutrals.parameters

```
&NEUTRALS
NSTRAI=8,
CRCSTRA= 'W','E','S','S','N','V','V','V','T',
RCPOS=    -1, 96, -1, -1, 36,  0,  0,  0,  0,
RCSTART=   0,  0,  0, 72,  0,  0,  0,  0,  0,
RCEND=    35, 35, 23, 95, 95,  0,  0,  0,  0,
SPECIES_START= 0, 0, 0, 0, 0, 0, 2, 9, 0,
SPECIES_END=  11,11,11,11,11, 1, 8,11,11,
recyc(0,1)=2*1.00,7*0.00,3*1.00,
recyc(0,2)=2*1.00,7*0.00,3*1.00,
recyc(0,3)=2*1.00,7*0.00,3*1.00,
recyc(0,4)=2*1.00,7*0.00,3*1.00,
recyc(0,5)=2*1.00,7*0.00,3*1.00,
recyc(0,6)=2*1.00,7*0.00,3*1.00,
recyc(0,7)=2*1.00,7*0.00,3*1.00,
recyc(0,8)=2*1.00,7*0.00,3*1.00,
recyc(0,9)=2*1.00,7*0.00,3*1.00,
userfluxparm(1,1)=9*0,
userfluxparm(1,2)=9*0,
volrecinc=1.1,
chemical_sputter_yield(1)=200*0.01,
chemical_sputter_yield(201)=6*0.0,
chemical_sputter_yield(202)=0.01,
chemical_sputter_yield(203)=0.01,
chemical_sputter_yield(204)=0.00,
chemical_sputter_yield(205)=0.00,
chemical_sputter_yield(206)=0.00,
fchar_chemical=1,
igass_chemical=2,
itsput_chemical=1,
issput_chemical=1206,
EIRENE_MOD=1,
eirene_step_dt=1e-0,
eirene_step_cpu=120.0,
l_neutrad=1,
lmolscl(1)=1,
mlcmp(1,1)=2,
targsp(1:5,1)=5*2,
phys_sput(0:11,1:5)=60*1.00,
```

```
chem_sput(0:1,1:5)=10*0.010,
track_index(0:11)=2*0,7*1,3*0,
track_chem_sput(1)=.TRUE.,.FALSE.,
/
```

In this file the user defines many parameters that are used for communication between the `B2.5` and `Eirene` parts of the code. Please see Appendix A.5 for full details. In particular, it is essential that the neutrals boundary conditions declared here match, in number and in order, the strata declared in the `Eirene` input file.

Some differences between the **SOLPS-ITER** and **SOLPS5.0** behaviours should be noted here.

First, the number of strata `NSTRAI` need no longer include the time-dependent `'T'` stratum.

Second, the array sizes of `chemical_sputter_yield`, if used, now need to match the actual number of wall and non-standard surfaces `NLIM` and `NSTS` as they are found in the `Eirene` input file. Non-zero values should only be assigned to actual walls (so the 200 in the above example corresponds to the real number of walls declared in block 3B of the `Eirene input.dat` file and/or in the `eirene_add_surfaces.json.input` file used for the same case) and to the non-standard surfaces (block 3A) corresponding to the targets. Note that the surface numbering considers block 3B first, and then block 3A, so non-standard surfaces will have indices varying between `NLIM+1` and `NLIM+NSTS`. Moreover, `Eirene` adds a last non-standard (absorbing) surface that is used as a "time boundary" (or "horizon"). If the `Eirene` input file is being produced by use of the `Uinp` utility and/or the `triang` script, then a `b2.neutrals.parameters.2` file meeting these criteria will be created, provided a `b2.neutrals.parameters` file containing any other data not deducible from the `DG` model is present.

Third, in **SOLPS-ITER**, the North and South boundaries of the `B2.5` grid are no longer recycling/sputtering surfaces but rather only switching surfaces to the triangular grid regions.

5.2 Output files

When running the coupled version of **SOLPS-ITER**, you will obtain additional output files, named `fort.44` and `fort.46`, which contain data computed from `Eirene`. The contents of the `fort.44` file is a series of data blocks, each starting with a title line, after a header providing the grid size and species list, as well as a format identifier. The relevant dimensions are (please also refer to the `Eirene` manual for further details and to the code for the actual way the arrays are written out):

NDX Number of poloidal cells in the `Eirene` computational grid

NDY Number of radial cells in the `Eirene` computational grid

NATM Number of atom species declared in the `Eirene` input file

NMOL Number of molecule species declared in the `Eirene` input file

NION Number of test ion species declared in the `Eirene` input file

NFLA Number of plasma species followed by the plasma solver, as declared in the `Eirene` input file

NPLS Number of plasma background species declared in the `Eirene` input file (including BGK species)

NSTRA Number of source strata declared in the `Eirene` input file (Stratum 0 is the sum over strata)

NLIM Number of "additional surfaces" (physical walls) declared in block 3B of the `Eirene` input file

NSTS Number of "non-standard surfaces" (plasma boundaries) declared in block 3A of the `Eirene` input file

NTRII Number of triangles forming the `Eirene` computational grid

NCL Number of segments for resolved tallies on non-standard surfaces

The data blocks are:

dab2 Atom density (m^{-3}), dimension (`NDX`,`NDY`,`NATM`)

tab2 Atom temperature (eV), dimension (NDX,NDY,NATM)

dmb2 Molecule density (m^{-3}), dimension (NDX,NDY,NMOL)

tmb2 Molecule temperature (eV), dimension (NDX,NDY,NMOL)

dib2 Test ion density (m^{-3}), dimension (NDX,NDY,NION)

tib2 Test ion temperature (eV), dimension (NDX,NDY,NION)

rfluxa Radial flux density of atoms ($m^{-2}.s^{-1}$), dimension (NDX,NDY,NATM)

rfluxm Radial flux density of molecules ($m^{-2}.s^{-1}$), dimension (NDX,NDY,NMOL)

pfluxa Poloidal flux density of atoms ($m^{-2}.s^{-1}$), dimension (NDX,NDY,NATM)

pfluxm Poloidal flux density of molecules ($m^{-2}.s^{-1}$), dimension (NDX,NDY,NMOL)

refluxa Radial energy flux density carried by atoms ($W.m^{-2}$), dimension (NDX,NDY,NATM)

refluxm Radial energy flux density carried by molecules ($W.m^{-2}$), dimension (NDX,NDY,NMOL)

pefluxa Poloidal energy flux density carried by atoms ($W.m^{-2}$), dimension (NDX,NDY,NATM)

pefluxm Poloidal energy flux density carried by molecules ($W.m^{-2}$), dimension (NDX,NDY,NMOL)

emiss H_α emissivity due to atoms ($photons.m^{-3}.s^{-1}$), dimension (NDX,NDY)

emissmol H_α emissivity due to molecules and molecular ions ($photons.m^{-3}.s^{-1}$), dimension (NDX,NDY)

srcml Molecule particle source (if the header is older than 20240627, then $A.cm^{-3}$, otherwise $molecules.s^{-1}$), dimension (NDX,NDY,NMOL)

edissml Energy spent for dissociating hydrogenic molecules (if the header is older than 20240627, then $W.cm^{-3}$, otherwise W), dimension (NDX,NDY,NMOL)

wldnek Net kinetic energy deposited by neutrals (W), total over strata, dimension (NLIM+NSTS)

wldnep Energy released by recombination of neutrals and ions into molecules (W), total over strata, dimension (NLIM+NSTS)

wldna Flux of atoms impinging on surface ($particles.s^{-1}$), total over strata, dimension (NLIM+NSTS,NATM)

ewlda Average energy of impinging atoms on surface (eV), total over strata, dimension (NLIM+NSTS,NATM)

wldnm Flux of molecules impinging on surface ($particles.s^{-1}$), total over strata, dimension (NLIM+NSTS,NMOL)

ewldm Average energy of impinging molecules on surface (eV), total over strata, dimension (NLIM+NSTS,NMOL)

p1,p2 Endpoints of surface (X and Y coordinates, in m), total over strata, dimension (NLIM)

wldra Flux of reflected atoms from surface ($particles.s^{-1}$), total over strata, dimension (NLIM+NSTS,NATM)

wldrm Flux of reflected molecules from surface ($particles.s^{-1}$), total over strata, dimension (NLIM+NSTS,NMOL)

For each stratum $i = 1, nstra$ The above data, resolved per stratum:

- **wldnek(i)** Net kinetic energy deposited by neutrals (W), dimension (NLIM+NSTS)
- **wldnep(i)** Energy released by recombination of neutrals and ions into molecules (W), dimension (NLIM+NSTS)
- **wldna(i)** Flux of atoms impinging on surface ($particles.s^{-1}$), dimension (NLIM+NSTS,NATM)
- **ewlda(i)** Average energy of impinging atoms on surface (eV), dimension (NLIM+NSTS,NATM)
- **wldnm(i)** Flux of molecules impinging on surface ($particles.s^{-1}$), dimension (NLIM+NSTS,NMOL)
- **ewldm(i)** Average energy of impinging molecules on surface (eV), dimension (NLIM+NSTS,NMOL)

wldra(i) Flux of reflected atoms from surface ($particles.s^{-1}$), dimension (NLIM+NSTS,NATM)

wldrm(i) Flux of reflected molecules from surface ($particles.s^{-1}$), dimension (NLIM+NSTS,NMOL)

wldpp Flux of plasma ions impinging on surface ($particles.s^{-1}$), total over strata, dimension (NLIM+NSTS,NPLS)

wldpa Net flux of atoms emitted from surface ($particles.s^{-1}$), total over strata, dimension (NLIM+NSTS,NATM)

wldpm Net flux of molecules emitted from surface ($particles.s^{-1}$), total over strata, dimension (NLIM+NSTS,NMOL)

wldpeb Power carried by particles emitted from surface (W), total over strata, dimension (NLIM+NSTS)

wldspt Flux of sputtered wall material ($particles.s^{-1}$), total over strata, dimension (NLIM+NSTS)

wldspta Flux of sputtered wall material per atom ($particles.s^{-1}$), total over strata, dimension (NLIM+NSTS,NATM)

wldsptm Flux of sputtered wall material per molecule ($particles.s^{-1}$), total over strata, dimension (NLIM+NSTS,NMOL)

For each stratum $i = 1, nstra$ The above data, resolved per stratum:

wldpp(i) Flux of plasma ions impinging on surface ($particles.s^{-1}$), dimension (NLIM+NSTS,NPLS)

wldpa(i) Net flux of atoms emitted from surface ($particles.s^{-1}$), dimension (NLIM+NSTS,NATM)

wldpm(i) Net flux of molecules emitted from surface ($particles.s^{-1}$), dimension (NLIM+NSTS,NMOL)

wldpeb(i) Power carried by particles emitted from surface (W), dimension (NLIM+NSTS)

wldspt(i) Flux of sputtered wall material ($particles.s^{-1}$), dimension (NLIM+NSTS)

wldspta(m) Flux of sputtered wall material per atom ($particles.s^{-1}$), dimension (NLIM+NSTS,NATM)

wldsptm(m) Flux of sputtered wall material per molecule ($particles.s^{-1}$), dimension (NLIM+NSTS,NMOL)

isrftype ILIIN surface type variable in Eirene, dimension (NLIM+NSTS)

wlarea Surface area (m^2), dimension (NLIM+NSTS)

wlabsrp(A) Absorption rate for atoms, dimension (NATM, NLIM+NSTS)

wlabsrp(M) Absorption rate for molecules, dimension (NMOL, NLIM+NSTS)

wlabsrp(I) Absorption rate for test ions, dimension (NION, NLIM+NSTS)

wlabsrp(P) Absorption rate for plasma ions, dimension (NPLS, NLIM+NSTS)

wlpump(A) Pumped flux per atom ($particles.s^{-1}$), dimension (NATM, NLIM+NSTS)

wlpump(M) Pumped flux per molecule ($particles.s^{-1}$), dimension (NMOL, NLIM+NSTS)

wlpump(I) Pumped flux per test ion ($particles.s^{-1}$), dimension (NION, NLIM+NSTS)

wlpump(P) Pumped flux per plasma ion ($particles.s^{-1}$), dimension (NPLS, NLIM+NSTS)

eneutrad Radiation rate due to atoms (W), dimension (NDX,NDY,NATM)

emolrad Radiation rate due to molecules (W), dimension (NDX,NDY,NMOL)

eionrad Radiation rate due to test ions (W), dimension (NDX,NDY,NION)

eirdiag_nds Indices for segments on resolved non-standard surfaces, dimension ($5\times$ NSTS $+1$)

sarea_res Surface area of surface segment (m^2), dimension (NCL)

wldna_res Flux of atoms impinging on surface segment ($particles.m^{-2}.s^{-1}$), dimension (NATM, NCL)

wldnm_res Flux of molecules impinging on surface segment ($particles.m^{-2}.s^{-1}$), dimension (NMOL, NCL)

ewlda_res Energy flux carried by impinging atoms on surface segment ($W.m^{-2}$), dimension (NATM, NCL)

ewldm_res Energy flux carried by impinging molecules on surface segment ($W.m^{-2}$), dimension (NMOL, NCL)

ewldea_res Energy flux carried by emitted atoms from surface segment ($W.m^{-2}$), dimension (NATM, NCL)

ewldem_res Energy flux carried by emitted molecules from surface segment ($W.m^{-2}$), dimension (NMOL, NCL)

ewldrp_res Kinetic energy flux carried by recycling atoms and molecules ($W.m^{-2}$), dimension (NCL)

ewldmr_res Energy flux due to recombination of atoms and atomic ions into molecules ($W.m^{-2}$), dimension (NMOL, NCL)

wldspt_res Flux of sputtered wall material ($A.m^{-2}$), dimension (NCL)

wldspta_res Flux of sputtered wall material per atom ($A.m^{-2}$), dimension (NCL, NATM)

wldsptm_res Flux of sputtered wall material per molecule ($A.m^{-2}$), dimension (NCL, NMOL)

wlpump_res(A) Pumped flux per atom ($A.m^{-2}$), dimension (NCL, NATM)

wlpump_res(M) Pumped flux per molecule ($A.m^{-2}$), dimension (NCL, NMOL)

wlpump_res(I) Pumped flux per test ion ($A.m^{-2}$), dimension (NCL, NION)

wlpump_res(P) Pumped flux per plasma ion ($A.m^{-2}$), dimension (NCL, NPLS)

ewldt_res Total wall power loading from Eirene particles ($W.m^{-2}$), dimension (NCL)

pdena_int Integral number of atoms over the entire Eirene computational grid, dimension (NATM, 0:NSTRA)

pdenm_int Integral number of molecules over the entire Eirene computational grid, dimension (NMOL, 0:NSTRA)

pdeni_int Integral number of test ions over the entire Eirene computational grid, dimension (NION, 0:NSTRA)

pdena_int_b2 Integral number of atoms over the B2.5 computational grid, dimension (NATM, 0:NSTRA)

pdenm_int_b2 Integral number of molecules over the B2.5 computational grid, dimension (NMOL, 0:NSTRA)

pdeni_int_b2 Integral number of test ions over the B2.5 computational grid, dimension (NION, 0:NSTRA)

edena_int Integral energy carried by atoms over the entire Eirene computational grid (J), dimension (NATM, 0:NSTRA)

edenm_int Integral energy carried by molecules over the entire Eirene computational grid (J), dimension (NMOL, 0:NSTRA)

edeni_int Integral energy carried by test ions over the entire Eirene computational grid (J), dimension (NION, 0:NSTRA)

edena_int_b2 Integral energy carried by atoms over the B2.5 computational grid (J), dimension (NATM, 0:NSTRA)

edenm_int_b2 Integral energy carried by molecules over the B2.5 computational grid (J), dimension (NMOL, 0:NSTRA)

edeni_int_b2 Integral energy carried by test ions over the B2.5 computational grid (J), dimension (NION, 0:NSTRA)

Similarly, the fort.46 file contains the following data blocks, this time all defined on the Eirene triangle grid, which are:

PDENA Atom particle density (in cm^{-3}), dimension (NTRII,NATM)

PDENM Molecule particle density (in cm^{-3}), dimension (NTRII,NMOL)

PDENI Test ion particle density (in cm^{-3}), dimension (NTRII,NION)

EDENA Energy density carried by atoms (in $eV.cm^{-3}$), dimension (NTRII,NATM)

EDENM Energy density carried by molecules (in $eV.cm^{-3}$), dimension (NTRII,NMOL)

EDENI Energy density carried by test ions (in $eV.cm^{-3}$), dimension (NTRII,NION)

VXDENA X-directed momentum density carried by atoms (in $g.cm^{-2}.s^{-1}$), dimension (NTRII,NATM)

VXDENM X-directed momentum density carried by molecules (in $g.cm^{-2}.s^{-1}$), dimension (NTRII,NMOL)

VXDENI X-directed momentum density carried by test ions (in $g.cm^{-2}.s^{-1}$), dimension (NTRII,NION)

VYDENA Y-directed momentum density carried by atoms (in $g.cm^{-2}.s^{-1}$), dimension (NTRII,NATM)

VYDENM Y-directed momentum density carried by molecules (in $g.cm^{-2}.s^{-1}$), dimension (NTRII,NMOL)

VYDENI Y-directed momentum density carried by test ions (in $g.cm^{-2}.s^{-1}$), dimension (NTRII,NION)

VZDENA Z-directed momentum density carried by atoms (in $g.cm^{-2}.s^{-1}$), dimension (NTRII,NATM)

VZDENM Z-directed momentum density carried by molecules (in $g.cm^{-2}.s^{-1}$), dimension (NTRII,NMOL)

VZDENI Z-directed momentum density carried by test ions (in $g.cm^{-2}.s^{-1}$), dimension (NTRII,NION)

VOL Triangle volumes (in cm^3), dimension (NTRII)

PUX X-component of the poloidal unit vector at the triangle center, dimension (NTRII)

PUY Y-component of the poloidal unit vector at the triangle center, dimension (NTRII)

PVX X-component of the radial unit vector at the triangle center, dimension (NTRII)

PVY Y-component of the radial unit vector at the triangle center, dimension (NTRII)

Please note that, for the momentum densities, the coordinate system used by Eirene is cartesian, i.e. independent of the local magnetic field, and that this data is given in CGS units. For cases without a certain species type, for example a pure Helium case with no molecules, the relevant data fields have a species dimension of 1 as they are mapped internally in Eirene to a dummy pointer array.

It is also possible to save long-term averages of some of the fort.44 and fort.46 quantities, using the Robbins-Monro averaging method. These will be found in the fort.44_aver and fort.46_aver files respectively. To activate the calculation and saving of these averages, create the two files, in the current run directory, with the first line empty, and the second line containing the real number which will be used to weigh the averages. If that number is zero, the corresponding averages will not be computed. If that number is negative or larger than 1, then the default value of 0.99 will be used and any existing further data in the file will be ignored and the averages restarted from scratch. The averaging scheme applied is of the form:

$$X^t_{aver} = (1 - F)X^t + FX^{t-1}_{aver} \tag{5.1}$$

where F is the number read from the fort.4[4-6]_aver files.

The list of averaged quantities stored in fort.44_aver is:

wldnek_aver Time-averaged net kinetic energy deposited by neutrals (W), total over strata, dimension (NLIM+NSTS)

wldnep_aver Time-averaged energy released by recombination of neutrals and ions into molecules (W), total over strata, dimension (NLIM+NSTS)

wldna_aver Time-averaged flux of atoms impinging on surface ($particles.s^{-1}$), total over strata, dimension (NLIM+NSTS,NATM)

ewlda_aver Time-averaged average energy of impinging atoms on surface (eV), total over strata, dimension (NLIM+NSTS,NATM)

wldnm_aver Time-averaged flux of molecules impinging on surface ($particles.s^{-1}$), total over strata, dimension (NLIM+NSTS,NMOL)

ewldm_aver Time-averaged average energy of impinging molecules on surface (eV), total over strata, dimension (NLIM+NSTS,NMOL)

wldra_aver Time-averaged flux of reflected atoms from surface ($particles.s^{-1}$), total over strata, dimension (NLIM+NSTS,NATM)

wldrm_aver Time-averaged flux of reflected molecules from surface ($particles.s^{-1}$), total over strata, dimension (NLIM+NSTS,NMOL)

wldpp_aver Time-averaged flux of plasma ions impinging on surface ($particles.s^{-1}$), total over strata, dimension (NLIM+NSTS,NFLA)

wldpa_aver Time-averaged net flux of atoms emitted from surface ($particles.s^{-1}$), total over strata, dimension (NLIM+NSTS,NATM)

wldpm_aver Time-averaged net flux of molecules emitted from surface ($particles.s^{-1}$), total over strata, dimension (NLIM+NSTS,NMOL)

wldpeb_aver Time-averaged power carried by particles emitted from surface (W), total over strata, dimension (NLIM+NSTS)

wldspt_aver Time-averaged flux of sputtered wall material ($particles.s^{-1}$), total over strata, dimension (NLIM+NSTS)

wldspta_aver Time-averaged flux of sputtered wall material per atom ($particles.s^{-1}$), total over strata, dimension (NLIM+NSTS,NATM)

wldsptm_aver Time-averaged flux of sputtered wall material per molecule ($particles.s^{-1}$), total over strata, dimension (NLIM+NSTS,NMOL)

wlpump_aver(A) Time-averaged pumped flux per atom ($particles.s^{-1}$), dimension (NATM, NLIM+NSTS)

wlpump_aver(M) Time-averaged pumped flux per molecule ($particles.s^{-1}$), dimension (NMOL, NLIM+NSTS)

wlpump_aver(I) Time-averaged pumped flux per test ion ($particles.s^{-1}$), dimension (NION, NLIM+NSTS)

wlpump_aver(P) Time-averaged pumped flux per plasma ion ($particles.s^{-1}$), dimension (NFLA, NLIM+NSTS)

The list of averaged quantities stored in `fort.46_aver` is:

PDENA_AVER Time-averaged atom particle density (in cm^{-3}), dimension (NTRII,NATM)

PDENM_AVER Time-averaged molecule particle density (in cm^{-3}), dimension (NTRII,NMOL)

EDENA_AVER Time-averaged energy density carried by atoms (in $eV.cm^{-3}$), dimension (NTRII,NATM)

EDENM_AVER Time-averaged energy density carried by molecules (in $eV.cm^{-3}$), dimension (NTRII,NMOL)

VXDENA_AVER Time-averaged X-directed momentum density carried by atoms (in $g.cm^{-2}.s^{-1}$), dimension (NTRII,NATM)

VXDENM_AVER Time-averaged X-directed momentum density carried by molecules (in $g.cm^{-2}.s^{-1}$), dimension (NTRII,NMOL)

VYDENA_AVER Time-averaged Y-directed momentum density carried by atoms (in $g.cm^{-2}.s^{-1}$), dimension (NTRII,NATM)

VYDENM_AVER Time-averaged Y-directed momentum density carried by molecules (in $g.cm^{-2}.s^{-1}$), dimension (NTRII,NMOL)

VZDENA_AVER Time-averaged Z-directed momentum density carried by atoms (in $g.cm^{-2}.s^{-1}$), dimension (NTRII,NATM)

VZDENM_AVER Time-averaged Z-directed momentum density carried by molecules (in $g.cm^{-2}.s^{-1}$), dimension (NTRII,NMOL)

5.3 Changing dt

With **SOLPS 4.x** one was often limited to a timestep of $1 \times 10^{-6}s$ – $1 \times 10^{-5}s$. With **SOLPS 5.x** this restriction seems to have been lifted, at least for some parameter regimes. In figure 5.1 we show the time evolution of the upstream separatrix temperature for time-steps of $1 \times 10^{-6}s$, $1 \times 10^{-5}s$, $1 \times 10^{-4}s$, $1 \times 10^{-3}s$ and $1 \times 10^{-2}s$. All of the runs seem to be behaving similarly, though at the highest time-step there is a time shift.

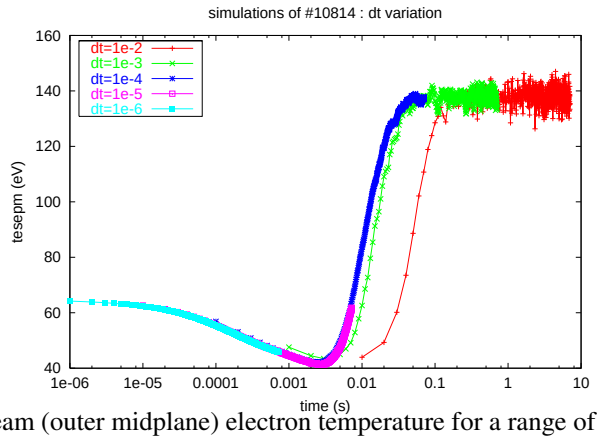


Figure 5.1: Upstream (outer midplane) electron temperature for a range of time-steps.

In figure 5.2 we see the same quantity plotted for a range of time-steps but with an additional core speed-up factor coming from `b2.numerics.parameters`.

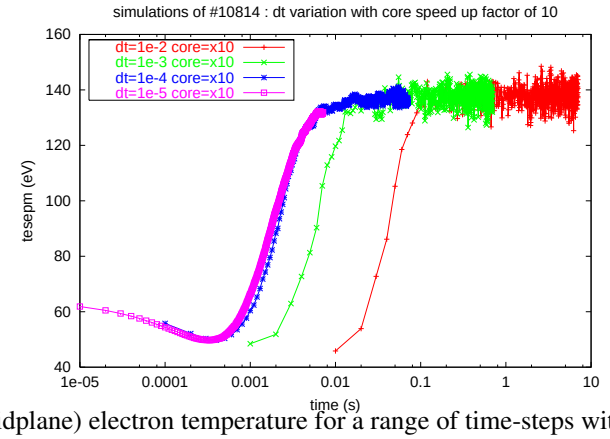


Figure 5.2: Upstream (outer midplane) electron temperature for a range of time-steps with a core speed-up factor.

Figure 5.3 combines the data form figures 5.1 and 5.2.

Figures 5.4, 5.5 and 5.6 show the outer target separatrix temperatures corresponding to figures 5.1, 5.2 and 5.3.

Figures 5.7, 5.8 and 5.9 show continuation runs.

A similar study was undertaken using **SOLPS-ITER** with fluid neutrals, where a subset of the PFR region radial particle boundary conditions were set as total particle flux to simulate a gas puff. In this case it was found that a time step on the order of 10^{-7} was required to obtain converged behavior. In particular, large time steps resulted in significant deviation in the total number of particles injected after an elapsed time, meaning the effective puff rate varied with the value of Δ_t. Figures 5.10 and 5.11 show the evolution of the midplane temperature and the total number of electron after the puff B.C. was applied for several values of Δ_t.

5.4 Dynamic (”Time-dependent”) mode

This feature is experimental and should be used with caution!

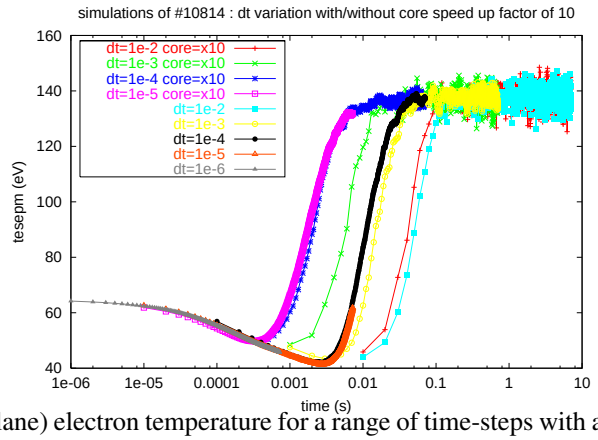


Figure 5.3: Upstream (outer midplane) electron temperature for a range of time-steps with and without a core speed-up factor.

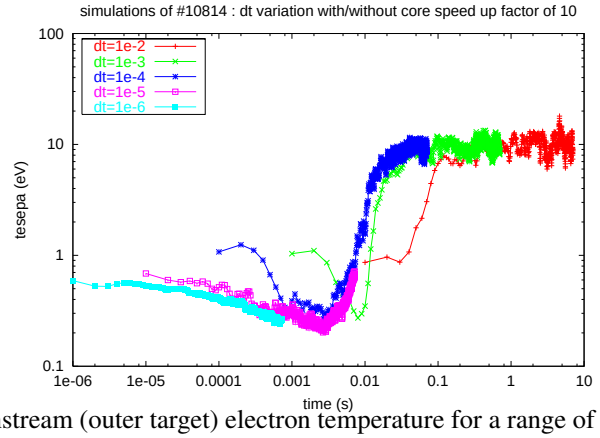


Figure 5.4: Downstream (outer target) electron temperature for a range of time-steps.

There is an alternative solution method available, which does not rely on under-relaxation, and is called the dynamic mode, by abuse of language, the "time-dependent" mode. This solution mode allows for a more precise estimate of the time derivative terms, particularly for the momentum and energy equations, and can be helpful to damp out numerical oscillations. This is obtained by setting the switch `b2mndt_style` to `2`. Please note that the timestep may need to be decreased compared to a similar traditional case, using under-relaxation, to achieve convergence.

The dynamic mode triggers the use of the **SOLPS 5.2** physics model. It changes the order of solution of the equations to solve first for the density equations, then the parallel momentum equations, followed by the electrical potential, electron and ion temperature equations. The total momentum and energy equations are not solved.

When using the traditional solution method, relying on under-relaxation of the equations, the evolution in time of the solution is not a true reflection of the physical evolution of the system, but rather only a numerical approximation, and only the final steady-state solution can be fully trusted. If one is interested in the details of the time evolution of the plasma state, then this new solution method should be considered.

More specifically, the way the time derivative terms are discretized is changed between the two code operation modes. For the traditional way of running the code, using the under-relaxation scheme, the time derivative term for the parallel momentum equations, (C.55) or (C.56), is discretized as:

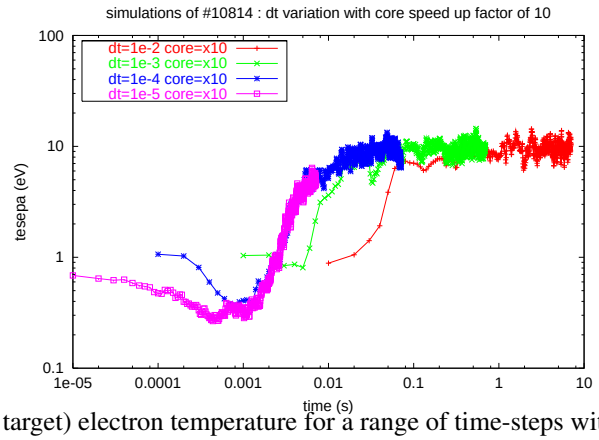


Figure 5.5: Downstream (outer target) electron temperature for a range of time-steps with a core speed-up factor.

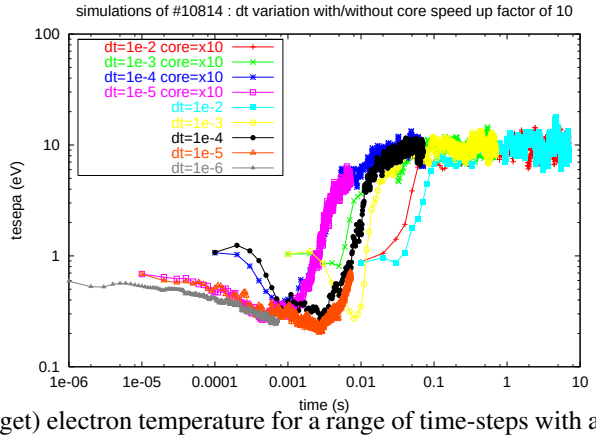


Figure 5.6: Downstream (outer target) electron temperature for a range of time-steps with and without a core speed-up factor.

$$m_a h_z \sqrt{g} \frac{n_a^{(k)} V_{\|a}^{(k+1)} - n_a^{(k)} V_{\|a}^{(k)}}{\tau \delta_{ma} \tau_f} \tag{5.2}$$

and the time derivative terms for the energy equations, (C.105) and (C.154), are discretized as:

$$\sqrt{g} \frac{3}{2} \frac{n_\alpha^{(k+1)} T_\alpha^{(k+1)} - n_\alpha^{(k)} T_\alpha^{(k)}}{\tau \delta_{e\alpha} \tau_f} \tag{5.3}$$

where α stands for e or i, τ is the time step size, δ_x is the time step multiplier for the x equation (i.e. the value of the `DTMO`, `DTEE`, or `DTEI` arrays for that region and species), and τ_f is an additional optional time step multiplier, governed by the `CORE_DT_FACTOR` switch from the `NUMERICS` namelist (see Section 3.5.4 and Appendix A.5 for details).

According to the under-relaxation scheme, the updated values of the parallel velocity and temperatures at time step $k+1$ are given by:

$$V_{\|a}^{(k+1)} = V_{\|a}^{(k)} + r_{xf_{V_{\|a}}} \left(\widetilde{V}_{\|a}^{(k+1)} - V_{\|a}^{(k)} \right)$$

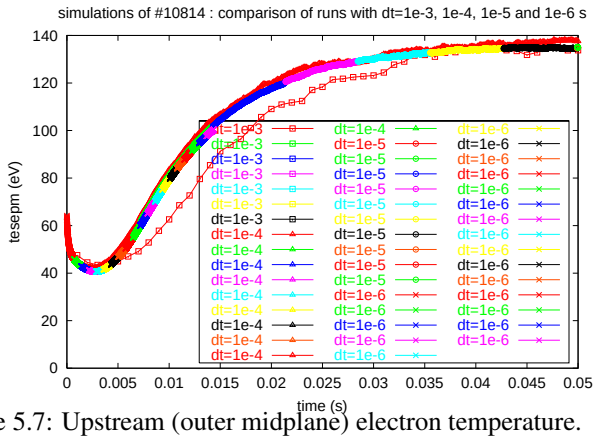


Figure 5.7: Upstream (outer midplane) electron temperature.

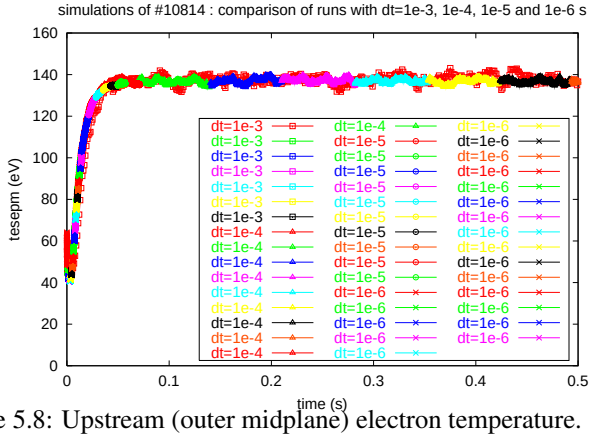


Figure 5.8: Upstream (outer midplane) electron temperature.

$$T_\alpha^{(k+1)} = T_\alpha^{(k)} + r_{xf_{T_\alpha}} \left(\widetilde{T}_\alpha^{(k+1)} - T_\alpha^{(k)} \right)$$

where $r_{xf_{V_{\|a}}}$ and $r_{xf_{T_\alpha}}$ are the under-relaxation factors, which are equal to `rxf*xfm1` and `rxf*xfm3` respectively, and the tilde quantities are the actual solutions to the equations. The factors `rxf`, `xfm1`, and `xfm3`, are given by the values of the `b2mndr_rxf`, `b2news_xfm1`, and `b2news_xfm3` switches, respectively.

When applying the dynamic mode, the discretization of the time derivative terms becomes, for the parallel momentum equations:

$$m_a h_z \sqrt{g} \frac{\widetilde{V}_{\|a}^{(k+1)} - V_{\|a}^{(k)}}{\tau \delta_{ma} \tau_f} n_a^{(k+1)} + m_a h_z \sqrt{g} \frac{n_a^{(k+1)} - n_a^{(k)}}{\tau} V_{\|a}^{(k)} \tag{5.4}$$

and, for the energy equations:

$$\sqrt{g} \frac{3}{2} \frac{\widetilde{T}_\alpha^{(k+1)} - T_\alpha^{(k)}}{\tau \delta_{e\alpha} \tau_f} n_\alpha^{(k+1)} + \sqrt{g} \frac{3}{2} \frac{n_\alpha^{(k+1)} - n_\alpha^{(k)}}{\tau} T_\alpha^{(k)} \tag{5.5}$$

Note that, since the order of solution of the equations has been changed in the dynamic mode, the updated density $n_a^{(k+1)}$ is now available for use in Eq. (5.4), providing a more accurate evaluation of the parallel momentum time derivative term.

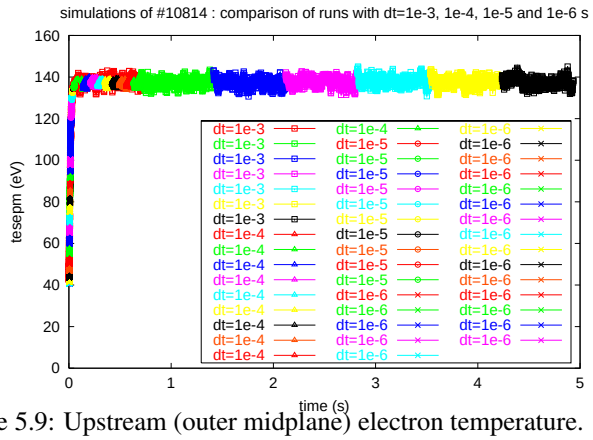


Figure 5.9: Upstream (outer midplane) electron temperature.

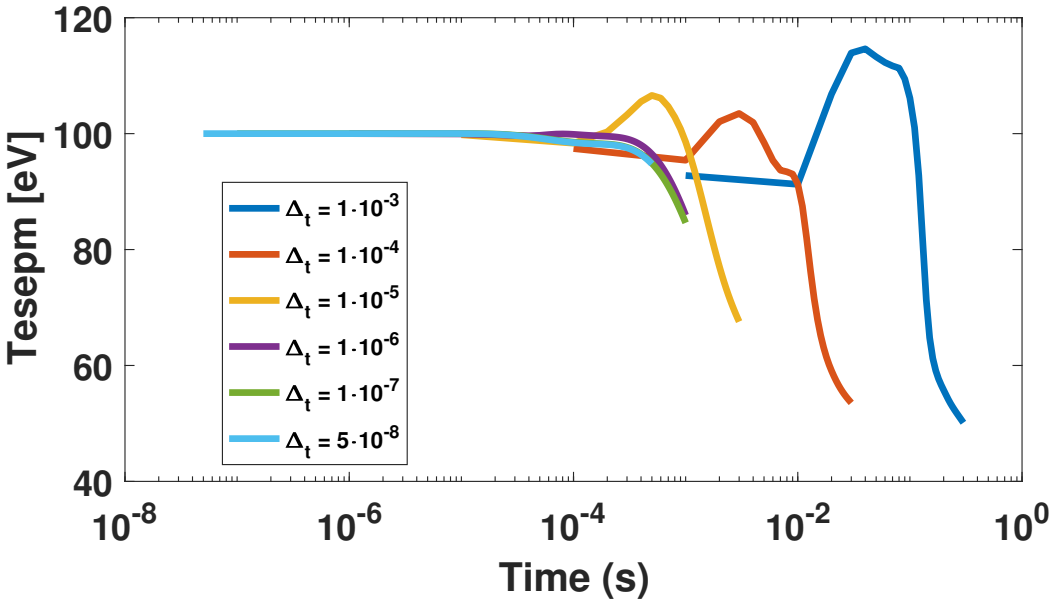


Figure 5.10: Upstream electron temperature for several timesteps after puff b.c. applied.

The conditions

$$\delta_{m_a} \tau_f r_{x f_{V_{\|a}}} = 1 \tag{5.6}$$

$$\delta_{e\alpha} \tau_f r_{x f_{T_\alpha}} = 1 \tag{5.7}$$

$$\delta_{n_a} \tau_f r_{x f_{f_{n_a}}} = 1 \tag{5.8}$$

must be fulfilled for the numerical solution to converge to the exact solution as $\tau \to 0$. Setting all parameters equal to one could be the best option, if it were not for the process of sequential separate solution of the equations, which can lead to a limitation of the time step τ. To expand the region of convergence of this process, one can try to find parameters in Eqs. (5.6)-(5.8) other than unity.

When computing the residuals in dynamic mode, the contribution from the time derivative terms is not included. If using internal iterations (`b2mndt_nstg0`, `b2mndt_nstg1`, or `b2mndt_nstg2` larger than 1), the time derivative is computed from one internal iteration to the next.

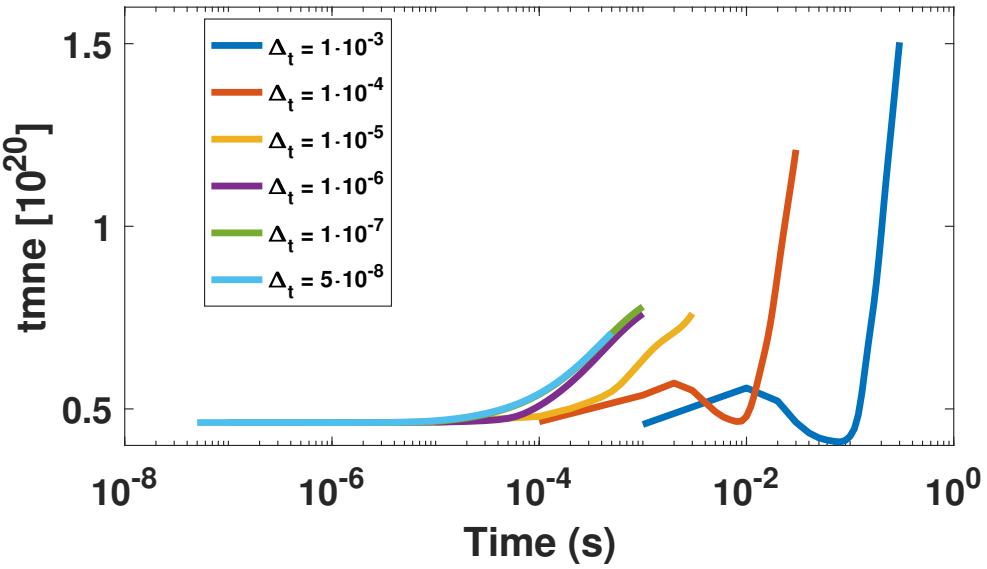


Figure 5.11: Total electron content for several timesteps after puff b.c. applied.

5.5 Time-dependent puff functions

Several functional forms for setting time-dependent gas puff inputs can be set using `b2.neutrals.parameters`. To use these options, the switches `TIME_DEP_PUFF` and `TIME_DEP_PUFF_FUNC` should be set to `true` for the intended stratum. Six functional forms can be selected, corresponding to the value of `TIME_DEP_PUFF_CASE`, with parameters controlled by `TIME_DEP_PUFF_PARAM`. These are described in Table 5.1 with examples shown in Fig. 5.12.

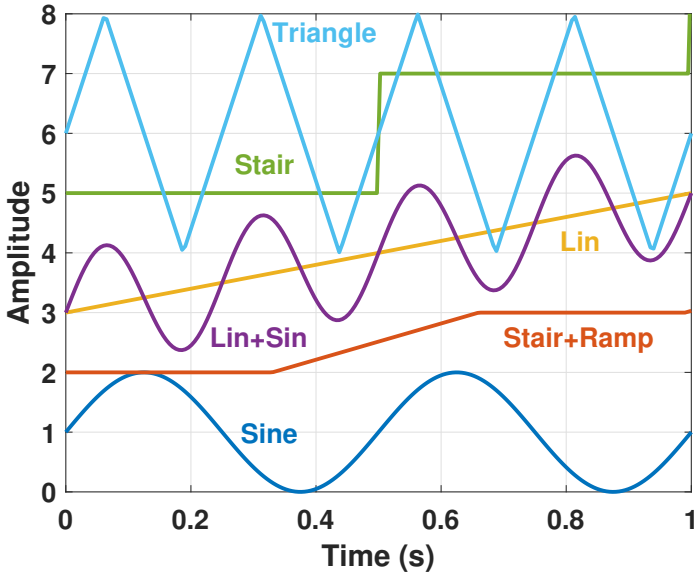


Figure 5.12: Examples of the time-dependent puff functions.

Function	Case	Param	Meaning	Fig. Example
Sine	0	1	offset ($\#/s$)	1
		2	amplitude ($\#/s$)	1
		3	frequency (Hz)	2
		4	phase (rad)	0
Stair+Ramp	1	1	offset ($\#/s$)	2
		2	step size ($\#/s$)	1
		3	step duration (s)	0.33
Linear	2	1	offset ($\#/s$)	3
		2	slope ($\#/s^2$)	2
Lin+Sine	3	1	offset ($\#/s$)	3
		2	amplitude ($\#/s$)	1
		3	frequency (Hz)	4
		4	phase (rad)	0
		5	slope ($\#/s^2$)	2
Stair	4	1	offset ($\#/s$)	5
		2	step size ($\#/s$)	2
		3	step duration (s)	0.5
Triangle	5	1	offset ($\#/s$)	6
		2	slope ($/s^2$)	2
		3	period (s)	0.25
		4	phase (rad)	0

Table 5.1: Input switch settings for time-dependent puff functions.

5.6 Averaging scheme

When running a coupled **SOLPS-ITER** case to obtain a steady-state solution, it is recommended to use the averaging scheme detailed in `${SOLPSTOP}/doc/SOLPS-ITER_Eirene_averaging.pdf` and references therein. This averaging scheme is meant to yield significant run speed-up and lower the statistical error introduced by the `Eirene` Monte Carlo noise, as well as provide means to estimate the errors on the solution, following the formalism detailed in [12, 13]. Error estimates are obtained by means of the `b2ye` program (See Section 5.7 below). It should be noted that this averaging scheme does NOT affect the physical run behaviour as such and therefore does not have any stabilizing effect on the run.

The averaging scheme works in three stages:

1. A transient phase during which the code is run normally, until all the short transients die out and the solution approaches its steady state.
2. An averaging phase, turned on by means of the `'b2mndt_av'` switch in the `b2mn.dat` input file, usually with a smaller number of `Eirene` trajectories, during which batch averaging will be done to obtain an averaged steady-state solution. This stage can be prepared by means of the `run_av` script and uses the `input_avg.eir` `Eirene` input file prepared by `Uinp` if available. When continuing a run in this phase, the user needs to provide both the instantaneous and averaged solutions from the previous run, i.e. do `mv b2favere b2faveri` in addition to `mv b2fstate b2fstati`.
3. A finalization phase where the averaged steady-state solution is used for a small number of timesteps, and a larger number of `Eirene` trajectories, to reduce statistical noise. This is done by running the `res_av` script and uses the `input_fin.eir Eirene` input file prepared by `Uinp` if available.

It is possible to write the averaged state as part of an IMAS data entry by means of the `b2mndr_ids_av` switch. If non-zero, the value of the switch is the frequency, in number of batch averages, at which time-slices are appended to the IMAS data entry. The averaged solution is saved as the second occurrence of the `edge_profiles` and

edge_sources IDSs.

5.7 b2ye, statistical error assessment

The program b2ye is meant for computing statistical error assessment according to [14] of a coupled B2.5-Eirene run where the averaging scheme described in Section 5.6 above has been used. It is called by means of the command:

```
b2run b2ye
```

or

```
gmake -f $SOLPSTOP/runs/Makefile b2ye.prt
```

It uses the data in the b2time.nc and b2favere files and calculates statistical errors based on averaged quantities collected during the averaging phase of a B2.5-Eirene calculation. For each averaged quantity stored in b2time.nc it evaluates:

- The sample mean.
- The correlation time according to the *batching* method.
- The correlation time according to the *autocorrelation* method.
- The statistical error σ_s according to the maximum of the two correlation times.

Chapter 6

Using MDSPLUS

A routine has been added, b2md, which allows one to save the output from **SOLPS 5.x** and **SOLPS-ITER** to an MDSplus server in Garching. (For more information about MDSplus see the Web site at www.mdsplus.org.)

The reverse operation, namely recovering an archived run from the MDSplus server, is performed by b2rd.

In order to use this facility, you need to ask David Coster at IPP-Garching (david.coster@ipp.mpg.de) to add you to the list of authorized MDSplus writers, supplying the username which will be used, as well as the machine/-domain.

In order to compile the code with MDSplus, the MDSPLUS_DIR environment needs to be set, as well as the compiler linking instruction LD_MDSPLUS — consult your systems people about this, or the MDSplus documentation. The MDSPLUS_DIR should in principle be provided by the ${SOLPSTOP}/SETUP/setup.csh.${HOST_NAME}.${COMPILER} file for your system, but can be overwritten locally in ${SOLPSTOP}/SETUP/setup.csh.${HOST_NAME}.${COMPILER}.local file. Similarly, the LD_MDSPLUS linking instructions should be found in the ${SOLPSTOP}/SETUP/config.${HOST_NAME}.${COMPILER}(.local) file. Then, a full recompilation with

```
sbb ; gmake clean ; gmake depend ; gmake
```

is safest. Before doing this, the MDSPLUS_DIR environment needs to be set — consult your systems people about this, or the MDSplus documentation.

On many systems, it is sufficient to do

```
module load MDSplus/7.153.3-GCCcore-13.2.0
```

or whatever the latest version of MDSplus is.

b2md requires as input b2md.dat and this is searched for in the present working directory, then the baserun directory, in order. See Appendix A.5 for a content description.

An example b2md.dat is:

```
&b2md_namelist
exp='JET',
shot=45462,
time=59.381,
comment='Parameter scan based (mostly) on g045462.59381.txt.x4.96x24.mod',
timedep=.true.,
snapshot=.true.,
tallies=.true.,
movies=.false.,
&end
```

To save a new run to the MDSplus server, execute

```
save_mds
```

which should create `shotnumber.history` which contains the ”shotnumber” for the run on the server (not to be confused with the experimental shot number). The output from the operation will be saved in `b2md.log` and this should be checked for errors.

A run can be re-saved to the same shotnumber with

```
resave_mds
```

which uses the ”shotnumber” in `shotnumber.history`.

The `shotnumber.history` file can later be read by `b2rd` which will then fetch from the `MDSplus` tree the archived shot corresponding to the last number stored within.

6.1 SOLPS tree structure

The `solps` tree consists of the main tree, `SOLPS`, and six sub-trees: `IDENT`, `SNAPSHOT`, `TALLIES`, `TIMEDEP`, `MOVIES` and `DIAG`.

The signals in, for example, `TIMEDEP` can be accessed as
`SOLPS::TOP.TIMEDEP.DP:IMPSEP` or alternatively
`TIMEDEP::TOP.DP:IMPSEP`.

6.1.1 SOLPS

```
\SOLPS::TOP.DIAG

\SOLPS::TOP.IDENT

\SOLPS::TOP.MOVIES

\SOLPS::TOP.SNAPSHOT

\SOLPS::TOP.TALLIES

\SOLPS::TOP.TIMEDEP
```

6.1.2 IDENT

```
\IDENT::TOP "Top level for information identifying a run"

\IDENT::TOP.IMPSEP
\IDENT::TOP.IMPSEP:CR "Inner midplane separatrix R"
\IDENT::TOP.IMPSEP:CZ "Inner midplane separatrix Z"
\IDENT::TOP.IMPSEP:D "Inner midplane separatrix diffusion coefficient"
\IDENT::TOP.IMPSEP:KYE "Inner midplane separatrix electron diffusivity"
\IDENT::TOP.IMPSEP:KYI "Inner midplane separatrix ion diffusivity for ns=1"
\IDENT::TOP.IMPSEP:KYI0 "Inner midplane separatrix ion/atom diffusivity"
\IDENT::TOP.IMPSEP:NE "Inner midplane separatrix ne"
\IDENT::TOP.IMPSEP:TE "Inner midplane separatrix te"
\IDENT::TOP.IMPSEP:TI "Inner midplane separatrix ti"

\IDENT::TOP.OMPSEP
\IDENT::TOP.OMPSEP:CR "Outer midplane separatrix R"
\IDENT::TOP.OMPSEP:CZ "Outer midplane separatrix Z"
\IDENT::TOP.OMPSEP:D "Outer midplane separatrix diffusion coefficient"
```

```
\IDENT::TOP.OMPSEP:KYE "Outer midplane separatrix electron diffusivity"
\IDENT::TOP.OMPSEP:KYI "Outer midplane separatrix ion diffusivity for ns=1"
\IDENT::TOP.OMPSEP:KYI0 "Outer midplane separatrix ion/atom diffusivity"
\IDENT::TOP.OMPSEP:NE "Outer midplane separatrix ne"
\IDENT::TOP.OMPSEP:TE "Outer midplane separatrix te"
\IDENT::TOP.OMPSEP:TI "Outer midplane separatrix ti"

\IDENT::TOP.SEP
\IDENT::TOP.SEP:FHE "core-sol electron energy flux"
\IDENT::TOP.SEP:FHI "core-sol ion energy flux"
\IDENT::TOP.SEP:FNE "core-sol electron flux"
\IDENT::TOP.SEP:FNI "core-sol ion flux"
\IDENT::TOP:B2FGMTRYID "md5sum hash of b2fgmtry"
\IDENT::TOP:B2FPLASMAID "md5sum hash of b2fplasma"
\IDENT::TOP:B2FRATESID "md5sum hash of b2frates"
\IDENT::TOP:B2FSTATEID "md5sum hash of b2fstate"
\IDENT::TOP:B2FSTATIID "md5sum hash of b2fstati"
\IDENT::TOP:COMMENTS "NOT_SET"
\IDENT::TOP:DATE "Date when the run was saved"
\IDENT::TOP:DIRECTORY "Directory in which the run was stored"
\IDENT::TOP:EXP "Experiment name, derived from the b2md_namelist in b2md.dat"
\IDENT::TOP:GEOMETRY "Label in b2fgmtry (derived from b2ag.dat)"
\IDENT::TOP:HOST "Hostname from which the run was saved"
\IDENT::TOP:MAIN "Label in b2fstate derived from b2mn.dat"
\IDENT::TOP:NS "Number of species"
\IDENT::TOP:NX "Grid size in poloidal direction"
\IDENT::TOP:NY "Grid size in radial direction"
\IDENT::TOP:OBJECTCODE "SOLPS $OBJECTCODE"
\IDENT::TOP:PHYSICS "Label in b2fparam derived from b2ah.dat"
\IDENT::TOP:SHOT "Experiment shot number (int) derived from the b2md_namelist in b2md.dat"
\IDENT::TOP:SOLPSTOP "SOLPS $OBJECTCODE"
\IDENT::TOP:SOLPSVERSION "Version number from SOLPS"
\IDENT::TOP:SPECIES "List of plasma species"
\IDENT::TOP:TIME "Experiment time point (double) derived from the b2md_namelist in b2md.dat"
\IDENT::TOP:USER "Username"
\IDENT::TOP:VERSION "Version number of the SOLPS MDS+ data output routine"
\IDENT::TOP:VERSION:B2FMOVIE "Version of b2fmovie"
\IDENT::TOP:VERSION:B2FPLASMA "Version of b2fplasma"
\IDENT::TOP:VERSION:B2FRATES "Version of b2frates"
\IDENT::TOP:VERSION:B2FSTATE "Version of b2fstate"
\IDENT::TOP:VERSION:B2FSTATI "Version of b2ftati"
\IDENT::TOP:VERSION:B2FTRACK "Version of b2ftrack"
\IDENT::TOP:VERSION:GIT "SOLPS GIT Repository Revision"
\IDENT::TOP:VERSION:PLASMASTATE "Version of plasmastate"
\IDENT::TOP:VERSION:SVN "SOLPS SVN Repository Revision"
\IDENT::TOP:WALL_SPECIES "List of wall species"
```

6.1.3 SNAPSHOT

```
\SNAPSHOT::TOP "Detailed results from the end of the run"


\SNAPSHOT::TOP.DIMENSIONS "Dimensions of the grid, number of species, position of separatrix etc."
\SNAPSHOT::TOP.DIMENSIONS:IMP "Index of the inner midplane"
\SNAPSHOT::TOP.DIMENSIONS:NATM "Number of atomic species"
\SNAPSHOT::TOP.DIMENSIONS:NION "Number of test ion species"
\SNAPSHOT::TOP.DIMENSIONS:NMOL "Number of molecular species"
\SNAPSHOT::TOP.DIMENSIONS:NS "Number of fluid species"
\SNAPSHOT::TOP.DIMENSIONS:NSW "Number of wall species"
\SNAPSHOT::TOP.DIMENSIONS:NWALL "Number of wall elements"
\SNAPSHOT::TOP.DIMENSIONS:NX "Number of cells in the parallel or poloidal direction"
\SNAPSHOT::TOP.DIMENSIONS:NY "Number of cells in the radial direction"
\SNAPSHOT::TOP.DIMENSIONS:OMP "Position of the outer midplane"
\SNAPSHOT::TOP.DIMENSIONS:SEP "Position of the separatrix"
```

```
\SNAPSHOT::TOP.DIMENSIONS:TARG1 "Position of target 1 (usually inner)"
\SNAPSHOT::TOP.DIMENSIONS:TARG2 "Position of target 2"
\SNAPSHOT::TOP.DIMENSIONS:TARG3 "Position of target 3"
\SNAPSHOT::TOP.DIMENSIONS:TARG4 "Position of target 4 (usually outer)"


\SNAPSHOT::TOP.GRID "Details of the grid"
\SNAPSHOT::TOP.GRID:AM "Atom/ion mass"
\SNAPSHOT::TOP.GRID:BOTTOMIX "bottom ix neighbourhood array"
\SNAPSHOT::TOP.GRID:BOTTOMIY "bottom iy neighbourhood array"
\SNAPSHOT::TOP.GRID:CFLAGS "cell flags array"
\SNAPSHOT::TOP.GRID:CR "R-coordinate of the cell centre"
\SNAPSHOT::TOP.GRID:CR_X "R-coordinate of the cell x-face"
\SNAPSHOT::TOP.GRID:CR_Y "R-coordinate of the cell y-face"
\SNAPSHOT::TOP.GRID:CZ "Z-coordinate of the cell centre"
\SNAPSHOT::TOP.GRID:CZ_X "Z-coordinate of the cell x-face"
\SNAPSHOT::TOP.GRID:CZ_Y "Z-coordinate of the cell y-face"
\SNAPSHOT::TOP.GRID:DSPAR "Parallel distance"
\SNAPSHOT::TOP.GRID:DSPOL "Poloidal distance"
\SNAPSHOT::TOP.GRID:DSRAD "Radial distance wrt separatrix"
\SNAPSHOT::TOP.GRID:IMAPX "IX-position of wall elements"
\SNAPSHOT::TOP.GRID:IMAPY "IY-position of wall elements"
\SNAPSHOT::TOP.GRID:LEFTIX "left ix neighbourhood array"
\SNAPSHOT::TOP.GRID:LEFTIY "left iy neighbourhood array"
\SNAPSHOT::TOP.GRID:POT "Potential Energy"
\SNAPSHOT::TOP.GRID:POTI "Ionization Potential"
\SNAPSHOT::TOP.GRID:R "R-coordinates of cell vertices"
\SNAPSHOT::TOP.GRID:REGFLX "x-directed flux region indices"
\SNAPSHOT::TOP.GRID:REGFLY "y-directed flux region indices"
\SNAPSHOT::TOP.GRID:REGVOL "volume region indices"
\SNAPSHOT::TOP.GRID:RIGHTIX "right ix neighbourhood array"
\SNAPSHOT::TOP.GRID:RIGHTIY "right iy neighbourhood array"
\SNAPSHOT::TOP.GRID:TOPIX "top ix neighbourhood array"
\SNAPSHOT::TOP.GRID:TOPIY "top iy neighbourhood array"
\SNAPSHOT::TOP.GRID:VESSEL "Vessel structure"
\SNAPSHOT::TOP.GRID:XYMAP "Wall index array"
\SNAPSHOT::TOP.GRID:Z "Z-coordinates of cell vertices"
\SNAPSHOT::TOP.GRID:ZA "Atom charge"
\SNAPSHOT::TOP.GRID:ZAMAX "Atom charge (max over bundle)"
\SNAPSHOT::TOP.GRID:ZAMIN "Atom charge (min over bundle)"
\SNAPSHOT::TOP.GRID:ZN "Nuclear charge"
\SNAPSHOT::TOP:ALF "Thermo-electric coefficient"
\SNAPSHOT::TOP:B "B field"
\SNAPSHOT::TOP:BRHE "Electron heat source from recycling"
\SNAPSHOT::TOP:BRHI "Ion heat source from recycling"
\SNAPSHOT::TOP:BRMO "Momentum source from recycling"
\SNAPSHOT::TOP:BRNA "Particle source from recycling"
\SNAPSHOT::TOP:CEQP "Electron-ion thermal equipartition factor"
\SNAPSHOT::TOP:D "D"
\SNAPSHOT::TOP:DAB2 "Atomic density"
\SNAPSHOT::TOP:DIB2 "Test ion density"
\SNAPSHOT::TOP:DMB2 "Molecular density"
\SNAPSHOT::TOP:DP "Dpa"
\SNAPSHOT::TOP:EDISSML "Power to atoms due to molecule dissociation (W)"
\SNAPSHOT::TOP:EIONRAD "Power loss due to (test) ion radiation (W)"
\SNAPSHOT::TOP:EMISS "Atomic H-alpha emissivity (photons/m3/s)"
\SNAPSHOT::TOP:EMISSMOL "Molecular H-alpha emissivity (photons/m3/s)"
\SNAPSHOT::TOP:EMOLRAD "Power loss due to molecule radiation (W)"
\SNAPSHOT::TOP:ENEUTRAD "Power loss due to atom radiation (W)"
\SNAPSHOT::TOP:FCHDIAX "Poloidal diamagnetic current"
\SNAPSHOT::TOP:FCHDIAY "Radial diamagnetic current"
\SNAPSHOT::TOP:FCHINERTX "Poloidal ion inertial current"
\SNAPSHOT::TOP:FCHINERTY "Radial ion inertial current"
\SNAPSHOT::TOP:FCHINX "Poloidal ion-neutral current"
\SNAPSHOT::TOP:FCHINY "Radial ion-neutral current"
\SNAPSHOT::TOP:FCHVISPARX "Poloidal parallel viscosity current"
\SNAPSHOT::TOP:FCHVISPARY "Radial parallel viscosity current"
```

```
\SNAPSHOT::TOP:FCHVISPERX "Poloidal perpendicular viscosity current"
\SNAPSHOT::TOP:FCHVISPERY "Radial perpendicular viscosity current"
\SNAPSHOT::TOP:FCHVISQX "Poloidal heat viscosity current"
\SNAPSHOT::TOP:FCHVISQY "Radial heat viscosity current"
\SNAPSHOT::TOP:FCHX "Poloidal current"
\SNAPSHOT::TOP:FCHX_32 "Convective poloidal current"
\SNAPSHOT::TOP:FCHX_52 "Conductive poloidal current"
\SNAPSHOT::TOP:FCHY "Radial current"
\SNAPSHOT::TOP:FCHY_32 "Convective radial current"
\SNAPSHOT::TOP:FCHY_52 "Conductive radial current"
\SNAPSHOT::TOP:FCH_P "Parallel current"
\SNAPSHOT::TOP:FHEPSCHX "Poloidal electron Pfirsch-Schlueter energy flux"
\SNAPSHOT::TOP:FHEPSCHY "Radial electron Pfirsch-Schlueter energy flux"
\SNAPSHOT::TOP:FHEX "Poloidal electron energy flux"
\SNAPSHOT::TOP:FHEX_COND "Poloidal electron energy flux, conductive part"
\SNAPSHOT::TOP:FHEX_CONV "Polodial electron energy flux, convective part"
\SNAPSHOT::TOP:FHEX_EIR "Poloidal electron energy flux (passed to Eirene)"
\SNAPSHOT::TOP:FHEX_MDF "Poloidal electron energy flux (with modified divergent diamagnetic flow)"
\SNAPSHOT::TOP:FHEY "Radial electron energy flux"
\SNAPSHOT::TOP:FHEY_COND "Radial electron energy flux, conductive part"
\SNAPSHOT::TOP:FHEY_CONV "Radial electron energy flux, convective part"
\SNAPSHOT::TOP:FHEY_EIR "Radial electron energy flux (passed to Eirene)"
\SNAPSHOT::TOP:FHEY_MDF "Radial electron energy flux (with modified divergent diamagnetic flow)"
\SNAPSHOT::TOP:FHIPSCHX "Poloidal ion Pfirsch-Schlueter energy flux"
\SNAPSHOT::TOP:FHIPSCHY "Radial ion Pfirsch-Schlueter energy flux"
\SNAPSHOT::TOP:FHIX "Poloidal ion energy flux"
\SNAPSHOT::TOP:FHIX_COND "Poloidal ion energy flux, conductive part"
\SNAPSHOT::TOP:FHIX_CONV "Poloidal ion energy flux, convective part"
\SNAPSHOT::TOP:FHIX_EIR "Poloidal ion energy flux (passed to Eirene)"
\SNAPSHOT::TOP:FHIX_MDF "Poloidal ion energy flux (with modified divergent diamagnetic flow)"
\SNAPSHOT::TOP:FHIY "Radial ion energy flux"
\SNAPSHOT::TOP:FHIY_COND "Radial ion energy flux, conductive part"
\SNAPSHOT::TOP:FHIY_CONV "Radial ion energy flux, convective part"
\SNAPSHOT::TOP:FHIY_EIR "Radial ion energy flux (passed to Eirene)"
\SNAPSHOT::TOP:FHIY_MDF "Radial ion energy flux (with modified divergent diamagnetic flow)"
\SNAPSHOT::TOP:FHJX "Poloidal electrostatic energy flux"
\SNAPSHOT::TOP:FHJY "Radial electrostatic energy flux"
\SNAPSHOT::TOP:FHMX "Poloidal kinetic energy flux"
\SNAPSHOT::TOP:FHMY "Radial kinetic energy flux"
\SNAPSHOT::TOP:FHPX "Poloidal potential energy flux"
\SNAPSHOT::TOP:FHPY "Radial potential energy flux"
\SNAPSHOT::TOP:FHTX "Poloidal total energy flux"
\SNAPSHOT::TOP:FHTY "Radial total energy flux"
\SNAPSHOT::TOP:FLLIME "Electron thermal flux limiter factor"
\SNAPSHOT::TOP:FLLIMI "Ion thermal flux limiter factor"
\SNAPSHOT::TOP:FLOEX_NOC "Poloidal electron energy flow"
\SNAPSHOT::TOP:FLOEY_NOC "Radial electron energy flow"
\SNAPSHOT::TOP:FLOIX_NOC "Poloidal ion energy flow"
\SNAPSHOT::TOP:FLOIY_NOC "Radial ion energy flow"
\SNAPSHOT::TOP:FMOX "Poloidal momentum flux"
\SNAPSHOT::TOP:FMOY "Radial momentum flux"
\SNAPSHOT::TOP:FNAPSCHX "Poloidal Pfirsch-Schlueter particle flux"
\SNAPSHOT::TOP:FNAPSCHY "Radial Pfirsch-Schlueter particle flux"
\SNAPSHOT::TOP:FNAX "Poloidal particle flux"
\SNAPSHOT::TOP:FNAX_32 "Poloidal particle flux (3/2 piece)"
\SNAPSHOT::TOP:FNAX_52 "Poloidal particle flux (5/2 piece)"
\SNAPSHOT::TOP:FNAX_EIR "Poloidal particle flux (passed to Eirene)"
\SNAPSHOT::TOP:FNAX_FCOR "Poloidal particle flux (including HZ toroidal factors)"
\SNAPSHOT::TOP:FNAX_HE "Poloidal particle flux (for electron heat equation)"
\SNAPSHOT::TOP:FNAX_MDF "Poloidal particle flux (with modified divergent diamagnetic flow)"
\SNAPSHOT::TOP:FNAX_NODRIFT "Poloidal particle flux (not including drift contributions)"
\SNAPSHOT::TOP:FNAY "Radial particle flux"
\SNAPSHOT::TOP:FNAY_32 "Radial particle flux (3/2 piece)"
\SNAPSHOT::TOP:FNAY_52 "Radial particle flux (5/2 piece)"
\SNAPSHOT::TOP:FNAY_EIR "Radial particle flux (passed to Eirene)"
\SNAPSHOT::TOP:FNAY_FCOR "Radial particle flux (including HZ toroidal factors)"
\SNAPSHOT::TOP:FNAY_HE "Radial particle flux (for electron heat equation)"
```

```
\SNAPSHOT::TOP:FNAY_MDF "Radial particle flux (with modified divergent diamagnetic flow)"
\SNAPSHOT::TOP:FNAY_NODRIFT "Radial particle flux (not including drift contributions)"
\SNAPSHOT::TOP:FNEX_32 "Poloidal electron flux (3/2 piece)"
\SNAPSHOT::TOP:FNEX_52 "Poloidal electron flux (5/2 piece)"
\SNAPSHOT::TOP:FNEX_EIR "Poloidal electron flux (passed to Eirene)"
\SNAPSHOT::TOP:FNEY_32 "Radial electron flux (3/2 piece)"
\SNAPSHOT::TOP:FNEY_52 "Radial electron flux (5/2 piece)"
\SNAPSHOT::TOP:FNEY_EIR "Radial electron flux (passed to Eirene)"
\SNAPSHOT::TOP:FNIX_32 "Poloidal total ion flux (3/2 piece)"
\SNAPSHOT::TOP:FNIX_52 "Poloidal total ion flux (5/2 piece)"
\SNAPSHOT::TOP:FNIY_32 "Radial total ion flux (3/2 piece)"
\SNAPSHOT::TOP:FNIY_52 "Radial total ion flux (5/2 piece)"
\SNAPSHOT::TOP:HX "Length of cell"
\SNAPSHOT::TOP:HY "Width of cell"
\SNAPSHOT::TOP:HY1 "Corrected width of cell"
\SNAPSHOT::TOP:KINRGY "Total kinetic energy"
\SNAPSHOT::TOP:KYE "kye"
\SNAPSHOT::TOP:KYI "kyi"
\SNAPSHOT::TOP:KYI0 "kyi0"
\SNAPSHOT::TOP:NA "Ion/Neutral density"
\SNAPSHOT::TOP:NA0 "Ion/Neutral density (previous time-step)"
\SNAPSHOT::TOP:NE "Electron density"
\SNAPSHOT::TOP:NE0 "Electron density (previous time-step)"
\SNAPSHOT::TOP:PEFA "Poloidal atomic energy flux"
\SNAPSHOT::TOP:PEFM "Poloidal molecular energy flux"
\SNAPSHOT::TOP:PFLA "Poloidal atomic flux"
\SNAPSHOT::TOP:PFLM "Poloidal molecular flux"
\SNAPSHOT::TOP:PO "Electric potential"
\SNAPSHOT::TOP:PO0 "Electric potential (previous time-step)"
\SNAPSHOT::TOP:PR "Pressure"
\SNAPSHOT::TOP:QZ "qz"
\SNAPSHOT::TOP:RCXHI "CX ion energy neutral losses"
\SNAPSHOT::TOP:RCXMO "CX momentum neutral losses"
\SNAPSHOT::TOP:RCXNA "CX particle neutral losses"
\SNAPSHOT::TOP:REFA "Radial atomic energy flux"
\SNAPSHOT::TOP:REFM "Radial molecular energy flux"
\SNAPSHOT::TOP:RESCO "Continuity equation residual"
\SNAPSHOT::TOP:RESHE "Electron heat equation residual"
\SNAPSHOT::TOP:RESHI "Ion heat equation residual"
\SNAPSHOT::TOP:RESMO "Momentum equation residual"
\SNAPSHOT::TOP:RESMT "Total momentum equation residual"
\SNAPSHOT::TOP:RESPO "Potential equation residual"
\SNAPSHOT::TOP:RFLA "Radial atomic flux"
\SNAPSHOT::TOP:RFLM "Radial molecular flux"
\SNAPSHOT::TOP:RPI "Effective Ionisation Potential (function of ix,iy,is)"
\SNAPSHOT::TOP:RPT "Cumulative Ionisation Potential (function of ix,iy,is)"
\SNAPSHOT::TOP:RQAHE "Electron cooling rate"
\SNAPSHOT::TOP:RQBRM "Bremsstrahlung radiation rate"
\SNAPSHOT::TOP:RQRAD "Line radiation rate"
\SNAPSHOT::TOP:RRAHI "Recombination ion energy losses"
\SNAPSHOT::TOP:RRAMO "Recombination momentum losses"
\SNAPSHOT::TOP:RRANA "Recombination particle losses"
\SNAPSHOT::TOP:RSAHI "Ionization ion energy losses"
\SNAPSHOT::TOP:RSAMO "Ionization momentum losses"
\SNAPSHOT::TOP:RSANA "Ionization particle losses"
\SNAPSHOT::TOP:RZ2 "Ion charge squared (function of ix,iy,is)"
\SNAPSHOT::TOP:RZA "Ion charge (function of ix,iy,is)"
\SNAPSHOT::TOP:SIG "Anomalous conductivity"
\SNAPSHOT::TOP:SMAF "smaf (extended friction term)"
\SNAPSHOT::TOP:SMAG "smag (pressure gradient term)"
\SNAPSHOT::TOP:SMAV "smav (additional viscosity term)"
\SNAPSHOT::TOP:SMFR "Friction force"
\SNAPSHOT::TOP:SMO "Parallel momentum source"
\SNAPSHOT::TOP:SMPR "Thermal force (proportional to exhp)"
\SNAPSHOT::TOP:SMPT "Thermal force (proportional to Ti and Te gradients)"
\SNAPSHOT::TOP:SMQ "smq (momentum sources from atomic physics)"
\SNAPSHOT::TOP:SMTH "Thermal force"
```

```
\SNAPSHOT::TOP:SRCML "Molecule dissociation rate (#/s)"
\SNAPSHOT::TOP:SX "sx (poloidal contact area)"
\SNAPSHOT::TOP:SXPERP "sxperp (poloidal area perpendicular to flux tube)"
\SNAPSHOT::TOP:SY "sy"
\SNAPSHOT::TOP:TAB2 "Neutral atom temperature"
\SNAPSHOT::TOP:TE "Electron temperature"
\SNAPSHOT::TOP:TE0 "Electron temperature (previous time-step)"
\SNAPSHOT::TOP:TEXTAN "Atom species"
\SNAPSHOT::TOP:TEXTCOMP "Components"
\SNAPSHOT::TOP:TEXTDRIFT "Directions of drift components"
\SNAPSHOT::TOP:TEXTIN "Test ion species"
\SNAPSHOT::TOP:TEXTLIN "Linearization components"
\SNAPSHOT::TOP:TEXTMN "Molecule species"
\SNAPSHOT::TOP:TEXTPL "Plasma species"
\SNAPSHOT::TOP:TEXTWL "Wall species"
\SNAPSHOT::TOP:TI "Ion temperature"
\SNAPSHOT::TOP:TI0 "Ion temperature (previous time-step)"
\SNAPSHOT::TOP:TIB2 "Test ion temperature"
\SNAPSHOT::TOP:TMB2 "Neutral molecule temperature"
\SNAPSHOT::TOP:UA "Ion/neutral parallel velocity"
\SNAPSHOT::TOP:UA0 "Ion/neutral parallel velocity (previous time-step)"
\SNAPSHOT::TOP:UADIA "Drift velocities in diamagnetic and radial directions"
\SNAPSHOT::TOP:VADIA "Divergent diamagnetic drift velocities in diamagnetic and radial directions"
\SNAPSHOT::TOP:VAECRB "ExB drift velocities in diamagnetic and radial directions"
\SNAPSHOT::TOP:VEDIA "Divergent diamagnetic drift electron velocities in diamagnetic and radial direc
\SNAPSHOT::TOP:VEECRB "Electron ExB drift velocities in diamagnetic and radial directions"
\SNAPSHOT::TOP:VISC "Viscosity"
\SNAPSHOT::TOP:VLAX "Poloidal pinch velocity"
\SNAPSHOT::TOP:VLAY "Radial pinch velocity"
\SNAPSHOT::TOP:VOL "Volume"
\SNAPSHOT::TOP:WADIA "Diamagnetic drift velocities in diamagnetic and radial directions"
\SNAPSHOT::TOP:WALL_TEMP "Wall temperature"
```

Note that the `SNAPSHOT::TOP.GRID:DSPAR` and `SNAPSHOT::TOP.GRID:DSPOL` quantities are respectively the parallel and poloidal distances along the magnetic field to the Western edge of the computational domain, and not, strictly speaking, connection lengths.

6.1.4 TALLIES

```
\TALLIES::TOP "Various tallies (volume or surface integrals) as a function of time"
\TALLIES::TOP:B2BREMREG "Bremsstrahlung energy loss per region"
\TALLIES::TOP:B2DIVUA "B2DIVUA"
\TALLIES::TOP:B2DIVUE "B2DIVUE"
\TALLIES::TOP:B2EXBA "B2EXBA"
\TALLIES::TOP:B2EXBE "B2EXBE"
\TALLIES::TOP:B2FRAA "B2FRAA"
\TALLIES::TOP:B2JOULE "B2JOULE"
\TALLIES::TOP:B2QIE "B2QIE"
\TALLIES::TOP:B2RADREG "Line radiation energy loss per region"
\TALLIES::TOP:B2SEXTSCHREG "External charge source"
\TALLIES::TOP:B2SEXTSHEREG "External electron energy source"
\TALLIES::TOP:B2SEXTSHIREG "External ion energy source"
\TALLIES::TOP:B2SEXTSMOREG "External parallel momentum source"
\TALLIES::TOP:B2SEXTSNAREG "External particle source"
\TALLIES::TOP:B2SEXTSNEREG "External electron particle source"
\TALLIES::TOP:B2SHE "B2SHE"
\TALLIES::TOP:B2SHE0 "B2SHE0"
\TALLIES::TOP:B2SHI "B2SHI"
\TALLIES::TOP:B2SHI0 "B2SHI0"
\TALLIES::TOP:B2STBCSCHREG "B2STBCSCHREG"
\TALLIES::TOP:B2STBCSHEREG "B2STBCSHEREG"
\TALLIES::TOP:B2STBCSHIREG "B2STBCSHIREG"
\TALLIES::TOP:B2STBCSMOREG "B2STBCSMOREG"
\TALLIES::TOP:B2STBCSNAREG "B2STBCSNAREG"
```

```
\TALLIES::TOP:B2STBCSNEREG "Boundary electron particle source"
\TALLIES::TOP:B2STBMSCHREG "Additional charge source"
\TALLIES::TOP:B2STBMSHEREG "Additional electron energy source"
\TALLIES::TOP:B2STBMSHIREG "Additional ion energy source"
\TALLIES::TOP:B2STBMSMOREG "Additional parallel momentum source"
\TALLIES::TOP:B2STBMSNAREG "Additional particle source"
\TALLIES::TOP:B2STBMSNEREG "Additional electron particle source"
\TALLIES::TOP:B2STBRSCHREG "B2STBRSCHREG"
\TALLIES::TOP:B2STBRSHEREG "B2STBRSHEREG"
\TALLIES::TOP:B2STBRSHIREG "B2STBRSHIREG"
\TALLIES::TOP:B2STBRSMOREG "B2STBRSMOREG"
\TALLIES::TOP:B2STBRSNAREG "B2STBRSNAREG"
\TALLIES::TOP:B2STBRSNEREG "Recycling electron particle source"
\TALLIES::TOP:B2STR "B2STR"
\TALLIES::TOP:B2VISA "B2VISA"
\TALLIES::TOP:B2WRONG1 "B2WRONG1"
\TALLIES::TOP:B2WRONG2 "B2WRONG2"
\TALLIES::TOP:B2WRONG3 "B2WRONG3"
\TALLIES::TOP:FCHXREG "FCHXREG"
\TALLIES::TOP:FCHYREG "FCHYREG"
\TALLIES::TOP:FHEXREG "FHEXREG"
\TALLIES::TOP:FHEYREG "FHEYREG"
\TALLIES::TOP:FHIXREG "FHIXREG"
\TALLIES::TOP:FHIYREG "FHIYREG"
\TALLIES::TOP:FHJXREG "FHJXREG"
\TALLIES::TOP:FHJYREG "FHJYREG"
\TALLIES::TOP:FHMXREG "FHMXREG"
\TALLIES::TOP:FHMYREG "FHMYREG"
\TALLIES::TOP:FHPXREG "FHPXREG"
\TALLIES::TOP:FHPYREG "FHPYREG"
\TALLIES::TOP:FHTXREG "fhtxreg"
\TALLIES::TOP:FHTYREG "fhtyreg"
\TALLIES::TOP:FNAXREG "FNAXREG"
\TALLIES::TOP:FNAYREG "FNAYREG"
\TALLIES::TOP:NAREG "NAREG"
\TALLIES::TOP:NE2REG "NE2REG"
\TALLIES::TOP:NEREG "NEREG"
\TALLIES::TOP:NIREG "NIREG"
\TALLIES::TOP:NS "NS"
\TALLIES::TOP:POREG "POREG"
\TALLIES::TOP:QCONVEXREG "qconvexreg"
\TALLIES::TOP:QCONVEYREG "qconveyreg"
\TALLIES::TOP:QCONVIXREG "qconvixreg"
\TALLIES::TOP:QCONVIYREG "qconviyreg"
\TALLIES::TOP:RCXHIREG "RCXHIREG"
\TALLIES::TOP:RCXMOREG "RCXMOREG"
\TALLIES::TOP:RCXNAREG "RCXNAREG"
\TALLIES::TOP:RESCOREG "RESCOREG"
\TALLIES::TOP:RESCO_REG "RESCO_REG"
\TALLIES::TOP:RESHEREG "RESHEREG"
\TALLIES::TOP:RESHE_REG "RESHE_REG"
\TALLIES::TOP:RESHIREG "RESHIREG"
\TALLIES::TOP:RESHI_REG "RESHI_REG"
\TALLIES::TOP:RESMOREG "RESMOREG"
\TALLIES::TOP:RESMO_REG "RESMO_REG"
\TALLIES::TOP:RESMTREG "RESMTREG"
\TALLIES::TOP:RESMT_REG "RESMT_REG"
\TALLIES::TOP:RESPOREG "RESPOREG"
\TALLIES::TOP:RESPO_REG "RESPO_REG"
\TALLIES::TOP:RQAHEREG "RQAHEREG"
\TALLIES::TOP:RQBRMREG "Bremsstrahlung radiation rate per region"
\TALLIES::TOP:RQRADREG "Line radiation rate per region"
\TALLIES::TOP:RRAHIREG "RRAHIREG"
\TALLIES::TOP:RRAMOREG "RRAMOREG"
\TALLIES::TOP:RRANAREG "RRANAREG"
\TALLIES::TOP:RSAHIREG "RSAHIREG"
\TALLIES::TOP:RSAMOREG "RSAMOREG"
```

```
\TALLIES::TOP:RSANAREG "RSANAREG"
\TALLIES::TOP:SPECIES "Species"
\TALLIES::TOP:TEREG "TEREG"
\TALLIES::TOP:TIME "TIME"
\TALLIES::TOP:TIREG "TIREG"
\TALLIES::TOP:VOLREG "VOLREG"
\TALLIES::TOP:VREG "VREG"
\TALLIES::TOP:XREG "XREG"
\TALLIES::TOP:YREG "YREG"
```

6.1.5 TIMEDEP

```
\TIMEDEP::TOP "Various time dependent quantities"


\TIMEDEP::TOP.DN "diffusion coefficient"
\TIMEDEP::TOP.DN:IMPSEP
\TIMEDEP::TOP.DN:OMPSEP "diffusion coefficient, outer midplane"


\TIMEDEP::TOP.DP "pressure diffusion coefficient"
\TIMEDEP::TOP.DP:IMPSEP
\TIMEDEP::TOP.DP:OMPSEP "pressure diffusion coefficient, outer midplane"

\TIMEDEP::TOP.FCH
\TIMEDEP::TOP.FCH:DIVINT "integrated radial current, divertor region"
\TIMEDEP::TOP.FCH:MCWINT "integrated radial current, main chamber"
\TIMEDEP::TOP.FCH:TARG1INT "integrated poloidal current, inboard divertor"
\TIMEDEP::TOP.FCH:TARG2INT
\TIMEDEP::TOP.FCH:TARG3INT
\TIMEDEP::TOP.FCH:TARG4INT "integrated poloidal current, outboard divertor"

\TIMEDEP::TOP.FEE
\TIMEDEP::TOP.FEE:DIVINT "integrated radial electron energy flux, divertor region"
\TIMEDEP::TOP.FEE:MCWINT "integrated radial electron energy flux, main chamber"
\TIMEDEP::TOP.FEE:TARG1INT "integrated poloidal electron energy flux, inboard divertor"
\TIMEDEP::TOP.FEE:TARG2INT
\TIMEDEP::TOP.FEE:TARG3INT
\TIMEDEP::TOP.FEE:TARG4INT "integrated poloidal electron energy flux, outboard divertor"

\TIMEDEP::TOP.FEI
\TIMEDEP::TOP.FEI:DIVINT "integrated radial ion energy flux, divertor region"
\TIMEDEP::TOP.FEI:MCWINT "integrated radial ion energy flux, main chamber"
\TIMEDEP::TOP.FEI:TARG1INT "integrated poloidal ion energy flux, inboard divertor"
\TIMEDEP::TOP.FEI:TARG2INT
\TIMEDEP::TOP.FEI:TARG3INT
\TIMEDEP::TOP.FEI:TARG4INT "integrated poloidal ion energy flux, outboard divertor"

\TIMEDEP::TOP.FET
\TIMEDEP::TOP.FET:DIVINT
\TIMEDEP::TOP.FET:MCWINT
\TIMEDEP::TOP.FET:TARG1INT
\TIMEDEP::TOP.FET:TARG2INT
\TIMEDEP::TOP.FET:TARG3INT
\TIMEDEP::TOP.FET:TARG4INT

\TIMEDEP::TOP.FNI
\TIMEDEP::TOP.FNI:DIVINT "integrated radial particle flux, divertor region"
\TIMEDEP::TOP.FNI:MCWINT "integrated radial particle flux, main chamber"
\TIMEDEP::TOP.FNI:TARG1INT "integrated poloidal particle flux, inboard divertor"
\TIMEDEP::TOP.FNI:TARG2INT
\TIMEDEP::TOP.FNI:TARG3INT
\TIMEDEP::TOP.FNI:TARG4INT "integrated poloidal particle flux, outboard divertor"
```

```
\TIMEDEP::TOP.IMP
\TIMEDEP::TOP.IMP:AN "atom density, inboard midplane"
\TIMEDEP::TOP.IMP:DN "diffusion coefficient, inboard midplane"
\TIMEDEP::TOP.IMP:DP "pressure diffusion coefficient, inboard midplane"
\TIMEDEP::TOP.IMP:DS "S-S_sep (inner midplane)"
\TIMEDEP::TOP.IMP:KE
\TIMEDEP::TOP.IMP:KI "ion thermal diffusivity, inboard midplane"
\TIMEDEP::TOP.IMP:LH "Conductive flux limit factor for neutrals, inner midplane"
\TIMEDEP::TOP.IMP:LN "Convective flux limit factor for neutrals, inner midplane"
\TIMEDEP::TOP.IMP:MN "molecule density, inboard midplane"
\TIMEDEP::TOP.IMP:NE "electron density, inboard midplane"
\TIMEDEP::TOP.IMP:PO "potential, inboard midplane"
\TIMEDEP::TOP.IMP:TE "electron temperature, inboard midplane"
\TIMEDEP::TOP.IMP:TI "ion temperature, inboard midplane"
\TIMEDEP::TOP.IMP:VS "viscosity coefficient, inboard midplane'"
\TIMEDEP::TOP.IMP:VX "poloidal pinch velocity, inboard midplane'"
\TIMEDEP::TOP.IMP:VY "radial pinch velocity, inboard midplane"

\TIMEDEP::TOP.INPUT
\TIMEDEP::TOP.INPUT:DS "S-S_sep (input profile location)"


\TIMEDEP::TOP.KE "electron thermal diffusivity"
\TIMEDEP::TOP.KE:IMPSEP
\TIMEDEP::TOP.KE:OMPSEP "electron thermal diffusivity, outer midplane"


\TIMEDEP::TOP.KI "ion thermal diffusivity"
\TIMEDEP::TOP.KI:IMPSEP
\TIMEDEP::TOP.KI:OMPSEP "ion thermal diffusivity, outer midplane"

\TIMEDEP::TOP.NE
\TIMEDEP::TOP.NE:IMPSEP
\TIMEDEP::TOP.NE:OMPSEP "separatrix electron density, outer midplane"
\TIMEDEP::TOP.NE:PWDENMAX
\TIMEDEP::TOP.NE:TARG1MAX "maximum electron density, inboard divertor"
\TIMEDEP::TOP.NE:TARG1SEP "separatrix electron density, inboard divertor"
\TIMEDEP::TOP.NE:TARG2MAX
\TIMEDEP::TOP.NE:TARG2SEP
\TIMEDEP::TOP.NE:TARG3MAX
\TIMEDEP::TOP.NE:TARG3SEP
\TIMEDEP::TOP.NE:TARG4MAX "maximum electron density, outboard divertor"
\TIMEDEP::TOP.NE:TARG4SEP "separatrix electron density, outboard divertor"

\TIMEDEP::TOP.OMP
\TIMEDEP::TOP.OMP:AN "atom density, outboard midplane"
\TIMEDEP::TOP.OMP:DN "diffusion coefficient, outboard midplane"
\TIMEDEP::TOP.OMP:DP "pressure diffusion coefficient, outboard midplane"
\TIMEDEP::TOP.OMP:DS "S-S_sep (outer midplane)"
\TIMEDEP::TOP.OMP:KE
\TIMEDEP::TOP.OMP:KI "ion thermal diffusivity, outboard midplane"
\TIMEDEP::TOP.OMP:LH "Conductive flux limit factor for neutrals, outer midplane"
\TIMEDEP::TOP.OMP:LN "Convective flux limit factor for neutrals, outer midplane"
\TIMEDEP::TOP.OMP:MN "molecule density, outboard midplane"
\TIMEDEP::TOP.OMP:NE "electron density, outboard midplane"
\TIMEDEP::TOP.OMP:PO "potential, outboard midplane"
\TIMEDEP::TOP.OMP:TE "electron temperature, outboard midplane"
\TIMEDEP::TOP.OMP:TI "ion temperature, outboard midplane"
\TIMEDEP::TOP.OMP:VS "viscosity coefficient, outboard midplane'"
\TIMEDEP::TOP.OMP:VX "poloidal pinch velocity, outboard midplane'"
\TIMEDEP::TOP.OMP:VY "radial pinch velocity, outboard midplane"

\TIMEDEP::TOP.PO
\TIMEDEP::TOP.PO:IMPSEP
\TIMEDEP::TOP.PO:OMPSEP "separatrix potential, outer midplane"
\TIMEDEP::TOP.PO:PWDENMAX
\TIMEDEP::TOP.PO:TARG1MAX "maximum potential, inboard divertor"
```

```
\TIMEDEP::TOP.PO:TARG1SEP "separatrix potential, inboard divertor"
\TIMEDEP::TOP.PO:TARG2MAX
\TIMEDEP::TOP.PO:TARG2SEP
\TIMEDEP::TOP.PO:TARG3MAX
\TIMEDEP::TOP.PO:TARG3SEP
\TIMEDEP::TOP.PO:TARG4MAX "maximum potential, outboard divertor "
\TIMEDEP::TOP.PO:TARG4SEP "separatrix potential, outboard divertor max"

\TIMEDEP::TOP.TARGET1
\TIMEDEP::TOP.TARGET1:AN "atom density, inboard divertor"
\TIMEDEP::TOP.TARGET1:DA "Area element, target1 (usually inner target)"
\TIMEDEP::TOP.TARGET1:DS "S-S_sep, target1 (usually inner target)"
\TIMEDEP::TOP.TARGET1:FC "poloidal current, inboard divertor "
\TIMEDEP::TOP.TARGET1:FE "poloidal electron energy flux, inboard divertor"
\TIMEDEP::TOP.TARGET1:FI "poloidal ion energy flux, inboard divertor"
\TIMEDEP::TOP.TARGET1:FL "poloidal electron flux, inboard divertor"
\TIMEDEP::TOP.TARGET1:FN "poloidal spec one flux, inboard divertor"
\TIMEDEP::TOP.TARGET1:FO "poloidal ion flux, inboard divertor"
\TIMEDEP::TOP.TARGET1:FT "poloidal total energy flux, inboard divertor"
\TIMEDEP::TOP.TARGET1:MN "molecule density, inboard divertor"
\TIMEDEP::TOP.TARGET1:NE "electron density, inboard divertor"
\TIMEDEP::TOP.TARGET1:PO "potential, inboard diverto"
\TIMEDEP::TOP.TARGET1:TE "electron temperature, inboard divertor"
\TIMEDEP::TOP.TARGET1:TI "ion temperature, inboard divertor"
\TIMEDEP::TOP.TARGET1:TP "Target surface temperature, inboard divertor"

\TIMEDEP::TOP.TARGET2
\TIMEDEP::TOP.TARGET2:AN
\TIMEDEP::TOP.TARGET2:DA
\TIMEDEP::TOP.TARGET2:DS
\TIMEDEP::TOP.TARGET2:FC
\TIMEDEP::TOP.TARGET2:FE
\TIMEDEP::TOP.TARGET2:FI
\TIMEDEP::TOP.TARGET2:FL
\TIMEDEP::TOP.TARGET2:FN
\TIMEDEP::TOP.TARGET2:FO
\TIMEDEP::TOP.TARGET2:FT
\TIMEDEP::TOP.TARGET2:MN
\TIMEDEP::TOP.TARGET2:NE
\TIMEDEP::TOP.TARGET2:PO
\TIMEDEP::TOP.TARGET2:TE
\TIMEDEP::TOP.TARGET2:TI
\TIMEDEP::TOP.TARGET2:TP "Target surface temperature"

\TIMEDEP::TOP.TARGET3
\TIMEDEP::TOP.TARGET3:AN
\TIMEDEP::TOP.TARGET3:DA
\TIMEDEP::TOP.TARGET3:DS
\TIMEDEP::TOP.TARGET3:FC
\TIMEDEP::TOP.TARGET3:FE
\TIMEDEP::TOP.TARGET3:FI
\TIMEDEP::TOP.TARGET3:FL
\TIMEDEP::TOP.TARGET3:FN
\TIMEDEP::TOP.TARGET3:FO
\TIMEDEP::TOP.TARGET3:FT
\TIMEDEP::TOP.TARGET3:MN
\TIMEDEP::TOP.TARGET3:NE
\TIMEDEP::TOP.TARGET3:PO
\TIMEDEP::TOP.TARGET3:TE
\TIMEDEP::TOP.TARGET3:TI
\TIMEDEP::TOP.TARGET3:TP "Target surface temperature"

\TIMEDEP::TOP.TARGET4
\TIMEDEP::TOP.TARGET4:AN "atom density, outboard divertor"
\TIMEDEP::TOP.TARGET4:DA "Area element, target4 (usually outer target)"
\TIMEDEP::TOP.TARGET4:DS "S-S_sep, target4 (usually outer target)"
\TIMEDEP::TOP.TARGET4:FC "poloidal current, outboard divertor "
```

```
\TIMEDEP::TOP.TARGET4:FE "poloidal electron energy flux, outboard divertor"
\TIMEDEP::TOP.TARGET4:FI "poloidal ion energy flux, outboard divertor"
\TIMEDEP::TOP.TARGET4:FL "poloidal electron flux, outboard divertor"
\TIMEDEP::TOP.TARGET4:FN "poloidal spec one flux, outboard divertor"
\TIMEDEP::TOP.TARGET4:FO "poloidal ion flux, outboard divertor"
\TIMEDEP::TOP.TARGET4:FT "poloidal total energy flux, outboard divertor"
\TIMEDEP::TOP.TARGET4:MN "molecule density, outboard divertor"
\TIMEDEP::TOP.TARGET4:NE "electron density, outboard divertor"
\TIMEDEP::TOP.TARGET4:PO "potential, outboard divertor"
\TIMEDEP::TOP.TARGET4:TE "electron temperature, outboard divertor"
\TIMEDEP::TOP.TARGET4:TI "ion temperature, outboard divertor"
\TIMEDEP::TOP.TARGET4:TP "Target surface temperature, outboard divertor"

\TIMEDEP::TOP.TE
\TIMEDEP::TOP.TE:IMPSEP
\TIMEDEP::TOP.TE:OMPSEP "separatrix electron temperature, outer midplane"
\TIMEDEP::TOP.TE:PWDENMAX
\TIMEDEP::TOP.TE:TARG1MAX "maximum electron temperature, inboard divertor"
\TIMEDEP::TOP.TE:TARG1SEP "separatrix electron temperature, inboard divertor"
\TIMEDEP::TOP.TE:TARG2MAX
\TIMEDEP::TOP.TE:TARG2SEP
\TIMEDEP::TOP.TE:TARG3MAX
\TIMEDEP::TOP.TE:TARG3SEP
\TIMEDEP::TOP.TE:TARG4MAX "maximum electron temperature, outboard divertor"
\TIMEDEP::TOP.TE:TARG4SEP "separatrix electron temperature, outboard divertor"

\TIMEDEP::TOP.TI
\TIMEDEP::TOP.TI:IMPSEP
\TIMEDEP::TOP.TI:OMPSEP "separatrix ion temperature, outer midplane"
\TIMEDEP::TOP.TI:PWDENMAX
\TIMEDEP::TOP.TI:TARG1MAX "maximum ion temperature, inboard divertor"
\TIMEDEP::TOP.TI:TARG1SEP "separatrix ion temperature, inboard divertor"
\TIMEDEP::TOP.TI:TARG2MAX
\TIMEDEP::TOP.TI:TARG2SEP
\TIMEDEP::TOP.TI:TARG3MAX
\TIMEDEP::TOP.TI:TARG3SEP
\TIMEDEP::TOP.TI:TARG4MAX "maximum ion temperature, outboard divertor "
\TIMEDEP::TOP.TI:TARG4SEP "separatrix ion temperature, outboard divertor"

\TIMEDEP::TOP.TP
\TIMEDEP::TOP.TP:TARG1MAX "Maximum surface temperature along target, inboard divertor"
\TIMEDEP::TOP.TP:TARG1SEP "Target surface temperature at strike point, inboard divertor"
\TIMEDEP::TOP.TP:TARG2MAX "Maximum surface temperature along target"
\TIMEDEP::TOP.TP:TARG2SEP "Target surface temperature at strike point"
\TIMEDEP::TOP.TP:TARG3MAX "Maximum surface temperature along target"
\TIMEDEP::TOP.TP:TARG3SEP "Target surface temperature at strike point"
\TIMEDEP::TOP.TP:TARG4MAX "Maximum surface temperature along target, outboard divertor"
\TIMEDEP::TOP.TP:TARG4SEP "Target surface temperature at strike point, outboard divertor"


\TIMEDEP::TOP.VS "viscosity coefficient"
\TIMEDEP::TOP.VS:IMPSEP
\TIMEDEP::TOP.VS:OMPSEP "viscosity coefficient, outer midplane"


\TIMEDEP::TOP.VX "poloidal pinch velocity"
\TIMEDEP::TOP.VX:IMPSEP
\TIMEDEP::TOP.VX:OMPSEP "poloidal pinch velocity, outer midplane"


\TIMEDEP::TOP.VY "radial pinch velocity"
\TIMEDEP::TOP.VY:IMPSEP
\TIMEDEP::TOP.VY:OMPSEP "radial pinch velocity, outer midplane"
\TIMEDEP::TOP:HALCORE
\TIMEDEP::TOP:HALDIV
\TIMEDEP::TOP:HALSOL
\TIMEDEP::TOP:TIME "time"
```

6.1.6 MOVIES

```
\MOVIES::TOP "Saved slices of quantities for movies"
\MOVIES::TOP:DAB2 "Atom density"
\MOVIES::TOP:DMB2 "Molecule density"
\MOVIES::TOP:NA "Ion density"
\MOVIES::TOP:NE "Electron density"
\MOVIES::TOP:RQAHESUM "Electron cooling rate (summed over all species)"
\MOVIES::TOP:RQRADSUM "Radiation rate"
\MOVIES::TOP:TE "Electron temperature"
\MOVIES::TOP:TI "Ion temperature"
\MOVIES::TOP:TIME "Time base"
```

6.1.7 DIAG

```
\DIAG::TOP "Top level for simulated experimental diagnostic signals"


\DIAG::TOP.AUG "AUG simulated experimental diagnostic signals'"

\DIAG::TOP.AUG.BLS_II_DPC
\DIAG::TOP.AUG.BLS_II_DPC:AHAL_MHAL
\DIAG::TOP.AUG.BLS_II_DPC:CIII_4650
\DIAG::TOP.AUG.BLS_II_DPC:CII_6581
\DIAG::TOP.AUG.BLS_II_DPC:H0_4340
\DIAG::TOP.AUG.BLS_II_DPC:H0_4861
\DIAG::TOP.AUG.BLS_II_DPC:H0_6561
\DIAG::TOP.AUG.BLS_II_DPC:HALPHA
\DIAG::TOP.AUG.BLS_II_DPC:HBETA
\DIAG::TOP.AUG.BLS_II_DPC:HGAMMA

\DIAG::TOP.AUG.LAST10
\DIAG::TOP.AUG.LAST10:AN3DA
\DIAG::TOP.AUG.LAST10:AN3DI
\DIAG::TOP.AUG.LAST10:AN3DL
\DIAG::TOP.AUG.LAST10:AN3DR
\DIAG::TOP.AUG.LAST10:DN3DA
\DIAG::TOP.AUG.LAST10:DN3DI
\DIAG::TOP.AUG.LAST10:DP3DA
\DIAG::TOP.AUG.LAST10:DP3DI
\DIAG::TOP.AUG.LAST10:FC3DL
\DIAG::TOP.AUG.LAST10:FC3DR
\DIAG::TOP.AUG.LAST10:FE3DL
\DIAG::TOP.AUG.LAST10:FE3DR
\DIAG::TOP.AUG.LAST10:FI3DL
\DIAG::TOP.AUG.LAST10:FI3DR
\DIAG::TOP.AUG.LAST10:FL3DL
\DIAG::TOP.AUG.LAST10:FL3DR
\DIAG::TOP.AUG.LAST10:FN3DL
\DIAG::TOP.AUG.LAST10:FN3DR
\DIAG::TOP.AUG.LAST10:FO3DL
\DIAG::TOP.AUG.LAST10:FO3DR
\DIAG::TOP.AUG.LAST10:FT3DL
\DIAG::TOP.AUG.LAST10:FT3DR
\DIAG::TOP.AUG.LAST10:KE3DA
\DIAG::TOP.AUG.LAST10:KE3DI
\DIAG::TOP.AUG.LAST10:KI3DA
\DIAG::TOP.AUG.LAST10:KI3DI
\DIAG::TOP.AUG.LAST10:MN3DA
\DIAG::TOP.AUG.LAST10:MN3DI
\DIAG::TOP.AUG.LAST10:MN3DL
\DIAG::TOP.AUG.LAST10:MN3DR
\DIAG::TOP.AUG.LAST10:NE3DA
```

```
\DIAG::TOP.AUG.LAST10:NE3DI
\DIAG::TOP.AUG.LAST10:NE3DL
\DIAG::TOP.AUG.LAST10:NE3DR
\DIAG::TOP.AUG.LAST10:PO3DA
\DIAG::TOP.AUG.LAST10:PO3DI
\DIAG::TOP.AUG.LAST10:PO3DL
\DIAG::TOP.AUG.LAST10:PO3DR
\DIAG::TOP.AUG.LAST10:TE3DA
\DIAG::TOP.AUG.LAST10:TE3DI
\DIAG::TOP.AUG.LAST10:TE3DL
\DIAG::TOP.AUG.LAST10:TE3DR
\DIAG::TOP.AUG.LAST10:TI3DA
\DIAG::TOP.AUG.LAST10:TI3DI
\DIAG::TOP.AUG.LAST10:TI3DL
\DIAG::TOP.AUG.LAST10:TI3DR
\DIAG::TOP.AUG.LAST10:TP3DL
\DIAG::TOP.AUG.LAST10:TP3DR
\DIAG::TOP.AUG.LAST10:VS3DA
\DIAG::TOP.AUG.LAST10:VS3DI
\DIAG::TOP.AUG.LAST10:VX3DA
\DIAG::TOP.AUG.LAST10:VX3DI
\DIAG::TOP.AUG.LAST10:VY3DA
\DIAG::TOP.AUG.LAST10:VY3DI

\DIAG::TOP.AUG.Z_0_2
\DIAG::TOP.AUG.Z_0_2:AHAL_MHAL
\DIAG::TOP.AUG.Z_0_2:CIII_4650
\DIAG::TOP.AUG.Z_0_2:CII_6581
\DIAG::TOP.AUG.Z_0_2:H0_4340
\DIAG::TOP.AUG.Z_0_2:H0_4861
\DIAG::TOP.AUG.Z_0_2:H0_6561
\DIAG::TOP.AUG.Z_0_2:HALPHA
\DIAG::TOP.AUG.Z_0_2:HBETA
\DIAG::TOP.AUG.Z_0_2:HGAMMA
\DIAG::TOP.AUG:DIA
\DIAG::TOP.AUG:DIA:AHAL_MHAL
\DIAG::TOP.AUG:DIA:CIII_4650
\DIAG::TOP.AUG:DIA:CII_6581
\DIAG::TOP.AUG:DIA:H0_4340
\DIAG::TOP.AUG:DIA:H0_4861
\DIAG::TOP.AUG:DIA:H0_6561
\DIAG::TOP.AUG:DIA:HALPHA
\DIAG::TOP.AUG:DIA:HBETA
\DIAG::TOP.AUG:DIA:HGAMMA
```

6.2 Accessing data from the MDSPLUS server

6.2.1 Using standard experimental data viewing programs

The `MDSplus` server name is `solps-mdsplus.ipp.mpg.de:8001`.

The tree name is `solps`.

An example shotnumber is `4293`.

Example signal names are
`top.timedep.ne:ompsep`,
`top.timedep.te:ompsep` (Outer midplane electron density and temperature), and
`top.timedep.ti:ompsep` (Outer midplane ion temperature).

cview

- start `cview`
- `Shot = 4293`
- In the table:
  - `shot = 0`
  - `shot = +solps`
  - `diag = solps`
  - `name = top.timedep.ne:ompsep`

Adding `top.timedep.ne:ompsep` and `top.timedep.ne:ompsep`, the graph in figure 6.1 is produced.

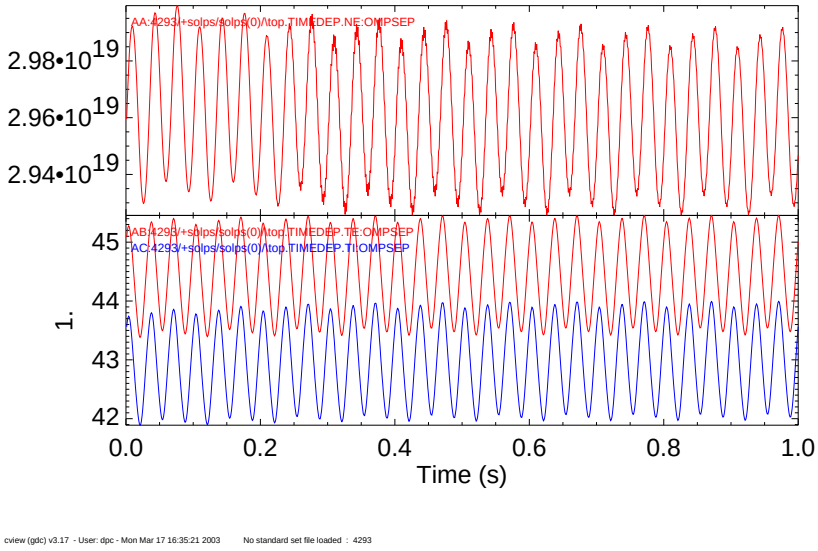


Figure 6.1: Using CVIEW to access SOLPS MDSplus data.

jetdsp

- start `JETDSP`
- `File|Read Signal`
- `MDS+` option
- `MDS+ server: = solps-mdsplus.ipp.mpg.de:8001`
- `Shot number: = 4293`
- `Tree: = solps`
- `Signal: = top.timedep.ne:ompsep`

- `Read` button

You can also use

- start `JETDSP`
- `File|Read Signal`
- `MDS+` option
- `MDS+ server: = solps-mdsplus.ipp.mpg.de:8001`
- `Shot number: = 4293`
- `Tree: = timedep`
- `Signal: = top.ne:ompsep`
- `Read` button

Adding `te:ompsep` and `ti:ompsep`, the graph in figure 6.2 is produced.

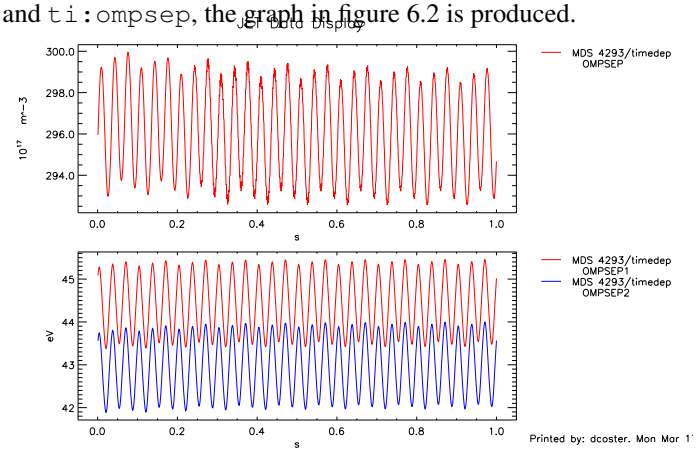


Figure 6.2: Using JETDSP to access SOLPS MDSplus data.

reviewplus

6.2.2 Using IDL

```
mdsconnect,'solps-mdsplus.ipp.mpg.de:8001'
mdsopen,'solps',251
te_coarse=mdsvalue('\top.snapshot:te')
cr_coarse=mdsvalue('\top.snapshot.grid:cr')
mdsopen,'solps',288
te_fine=mdsvalue('\top.snapshot:te')
cr_fine=mdsvalue('\top.snapshot.grid:cr')
mdsclose
mdsdisconnect
plot,cr_coarse[*,8],te_coarse[*,8],linestyle=1,psym=-1,yrange=[0,50]
oplot,cr_fine[*,10],te_fine[*,10],linestyle=2,psym=-2
```

6.2.3 Using MATLAB

```
mdsopen('solps-mdsplus.ipp.mpg.de:8001::solps',nshot);
onx= mdsvalue('\SOLPS::TOP.SNAPSHOT.DIMENSIONS:NX');
ny = mdsvalue('\SOLPS::TOP.SNAPSHOT.DIMENSIONS:NY'); %number of radial cells
ne = mdsvalue('\SOLPS::TOP.SNAPSHOT:NE');
te = mdsvalue('\SOLPS::TOP.SNAPSHOT:TE');
ti = mdsvalue('\SOLPS::TOP.SNAPSHOT:TI');
sep= mdsvalue('\SOLPS::TOP.TIMEDEP.OMP:DS');
qe = 1.602e-19; %Elementary charge in C
amu= 1.660542e-27; %Atomic mass unit in kg

%Calculate the ion sound speed at outer target
cs2o = qe .* (ti(onx-1,:) + te(onx-1,:)) ./ (2*amu);

%Outer target
neo  = ne(onx-1,:);
osja = neo .* qe .* sqrt(cs2o) / 1.0e4;

cr = mdsvalue('\SOLPS::TOP.SNAPSHOT.GRID:CR'); %R-coordinate of cell centre
ocr= cr(onx,:); %R coordinates of outer target cells
cz = mdsvalue('\SOLPS::TOP.SNAPSHOT.GRID:CZ'); %Z-coordinate of cell centre
ocz= cz(onx,:); %Z coordinate of outer target cells
mdsclose('MDSIP');

[map2mid,oms]=jhmap2mid(26730,ocr,ocz,0.7);
rrsep= map2mid.Rmid-map2mid.Rlcfs;

plot(rrsep,osja,'k');
```

6.2.4 Using Python

You will need the MDSplus interface for python.

```
import MDSplus

conn = MDSplus.Connection('solps-mdsplus.ipp.mpg.de:8001')
conn.openTree('solps', 12380)
r=conn.get('\\top.snapshot.grid:r')
z=conn.get('\\top.snapshot.grid:z')
conn.closeTree('solps', 12380)
conn.disconnect()

print(r)
print(z)
```

For additional information about using python, see
https://wiki.eu-euforia.eu/index.php/Fusion_dataset_with_Python_and_VisIt
(this activity occurred within the EUFORIA project, http://www.euforia-project.eu/).

Typing

```
python
import compare_shots
```

with the following as `compare_shots.py`:

```
#! /usr/bin/env python3
'''
Date : 05/12/2008
Authors : Matthieu Haefele based on IDL scripts provided by David Coster

This script demonstrates the capabilities of Python/NumPy/matplotlib tools to show fusion data.
Basically it reads datasets from solps server through MDS+ and either display or write a
    resulting
picture on disk thanks to matplotlib.

It provides the function compare_shots which reads 1D datasets from a list of shots on a server
through MDS+ and either displays or writes a resulting picture on disk thanks to matplotlib.
The picture consists in a 9X9 plot on which data from all the specified  shots are displayed. If
filename is omitted, a X window pop up, if filename is specified, no window pop up, but a picture
is generated on disk with name filename.
'''

import MDSplus
import numpy

# matplotlib.pyplot is a collection of command style functions that make matplotlib work like
    matlab
import matplotlib.pyplot as plt

# matplotlib.ticker contains classes to support completely configurable tick locating and
    formatting.
import matplotlib.ticker as ticker

def compare_shots(shot, filename=None, server='solps-mdsplus.ipp.mpg.de:8001'):
        '''
This function reads 1D datasets from shot <shot> on solps server <server> through MDS+ and either
displays or writes a resulting picture on disk thanks to matplotlib.
* shot : python list of integer which are the shots. Mandatory argument
* filename : python string which contains the name of the genereated file. Optional argument.
* server : MDS+ server to connect. Optional argument, default value=solps-mdsplus.aug.ipp.mpg.de
    :8001
        '''

        params = {'axes.labelsize': 10,
                                                'axes.titlesize': 10,
                                                'xtick.labelsize': 4,
                                                'ytick.labelsize': 4,
                                                'lines.markersize' : 3,
                                                'text.usetex': True
                                                }


        if filename == None:
                # No filename supplied, displaying on screen
                params['backend'] =  'GTK3Agg'              #'QtAgg', 'ps', 'svg','png'
        else:
                # Filename provided, then it"s a png
                params['backend'] =  'png'              #'QtAgg', 'ps', 'svg','png'


        plt.rcParams.update(params)


        # compare_shots is called with an array of shots. Let's get is length
        ncase = len(shot)

  # Let's create a few identical arrays whose dimensions and type are in arrayspec
        arrayspec=(ncase,numpy.int32)

        # Creating arrays expanding array specification
        nx  = numpy.zeros(*arrayspec)
```

```
ny  = numpy.zeros(*arrayspec)
ns  = numpy.zeros(*arrayspec)
imp = numpy.zeros(*arrayspec)
omp = numpy.zeros(*arrayspec)
sep = numpy.zeros(*arrayspec)



conn = MDSplus.Connection(server)
for i in range(ncase):
        conn.openTree('solps',shot[i])
        nx[i]=conn.get('\\top.snapshot.dimensions:nx')
        ny[i]=conn.get('\\top.snapshot.dimensions:ny')
        print('opening shot '+ str(shot[i]) + ' => nx, ny='+str(nx[i]) + ' ' + str(ny[i])
            )
        # Someday maybe, all this will work in Python 3. One shall then write
        # print(''opening shot '+ str(shot[i]) + ' => nx, ny='+str(nx[i]) + ' ' + str(ny[
            i]))

        ns[i]=conn.get('\\top.snapshot.dimensions:ns')
        imp[i]=conn.get('\\top.snapshot.dimensions:imp')
        omp[i]=conn.get('\\top.snapshot.dimensions:omp')
        sep[i]=conn.get('\\top.snapshot.dimensions:sep')
        conn.closeTree('solps',shot[i])

# Another array specification
arrayspec=((ncase,ny[0],nx[0]),numpy.float64)

# Let's create these float arrays
dsrad = numpy.zeros( *arrayspec)
dspol = numpy.zeros( *arrayspec)
dspar = numpy.zeros (*arrayspec)
te    = numpy.zeros (*arrayspec)
ti    = numpy.zeros (*arrayspec)
nel   = numpy.zeros (*arrayspec)



for i in range(ncase):
        conn.openTree('solps',shot[i])
        dsrad[i,:,:]=conn.get('\\top.snapshot.grid:dsrad')
        dspol[i,:,:]=conn.get('\\top.snapshot.grid:dspol')
        dspar[i,:,:]=conn.get('\\top.snapshot.grid:dspar')
        te[i,:,:]=conn.get('\\top.snapshot:te')
        ti[i,:,:]=conn.get('\\top.snapshot:ti')
        nel[i,:,:]=conn.get('\\top.snapshot:ne')
        conn.closeTree('solps',shot[i])

conn.disconnect()


#build the figure (Let's go Picasso)
# Layout is a list composed of
# [number of sub-plot, legend of the left axis, data, location in the tokamak (Inner
    target, Outer midplane or Outer target), subplot upper title,subplot lower title ]
layout=[[1,'Te',te, 0, 'Inner target', ''],
  [2,'',te,omp[0], 'Outer midplane', ''],
  [3,'',te,nx[0]-1, 'Outer target', ''],
  [4,'Ti',ti, 0, '', ''],
  [5,'',ti,omp[0], '', ''],
  [6,'',ti,nx[0]-1, '', ''],
  [7,'ne',nel, 0, '', 'distance wrt sep'],
  [8,'',nel,omp[0], '', 'distance wrt sep'],
  [9,'',nel,nx[0]-1, '', 'distance wrt sep']
 ]

# Matplotlib plot definitions : point shape, linked or not, color
```

```
        line_style = ['o-b', '^-g', 's-r', '+-c', 'x-m', 'D-y', '1-k', 'h-']

        # sub-plot number j[0] in a 3x3 subplot. last number ranges from 1 to 3x3, line-major
        print('generating figures')
        for j in layout:
                plotgridshape=(3,3,j[0])
                cur_fig = plt.subplot(*plotgridshape)

                # Finds with the ticker module nice ticks locations, up to 5 locators
                cur_fig.xaxis.set_major_locator(ticker.MaxNLocator(5))

                cur_fig.set_xlabel(j[5])
                cur_fig.set_ylabel(j[1])
                cur_fig.set_title(j[4])

                # In each subplot, plot all the cases, each with its line style
                for i in range(ncase):
                        cur_fig.plot(dsrad[i,:,j[3]],j[2][i,:,j[3]], line_style[i])


        # Some more space between sub-plots
        plt.subplots_adjust(wspace=0.15, hspace=0.2)

        if filename == None:
                plt.show()
        else:
                plt.savefig(filename)

compare_shots([11437,11438,11447,11448,11449,11450])
```

produced Figure 6.3.

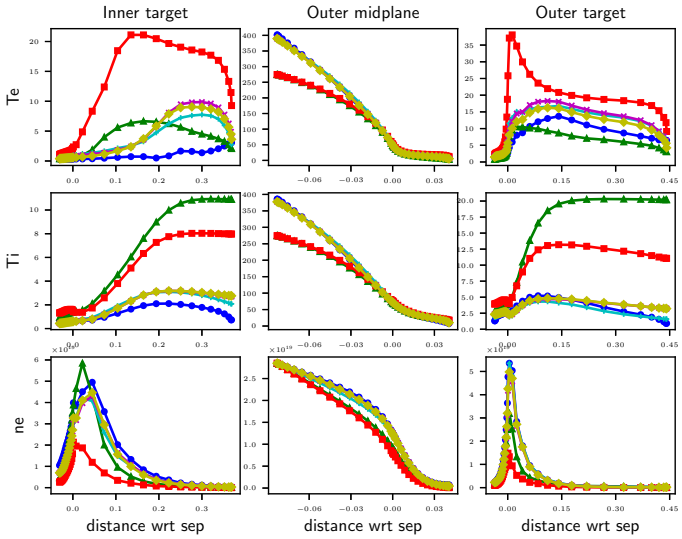


Figure 6.3: Comparison of profiles for a range of shots.

In another, more complicated, example, python was used to create the frames of a movie on the ITM Gateway machine.

```
set c=0
foreach i ( `cat shots` )
  set tbase=`echo $c | awk '{print $1*1000*1e-5}'`
```

```
  mkdir $i
  echo Submitting $i with offset $tbase
  ( cd $i ; sed -e "s:#1#:${i}:g" -e "s:#2#:${tbase}:g" < ../QSUB.shot | bsub )
  @ c++
  sleep 5
end
```

with `QSUB.shot` containing

```
#! /bin/tcsh

#BSUB -e PYTHON.e%J.%I
#BSUB -o PYTHON.o%J.%I
#BSUB -J PYTHON
#BSUB -W 36:00
#BSUB -q priority

setenv PATH /afs/efda-itm.eu/imp3/user/coster/public/python/bin:$PATH
setenv PYTHONPATH /afs/efda-itm.eu/imp3/user/coster/python:/afs/efda-itm.eu/imp3/user/coster/
    public/python/lib/python2.5/site-packages:/afs/efda-itm.eu/imp3/user/coster/public/python:
    $PYTHONPATH

echo 1200 | awk '{srand();print int($1*rand())}' | xargs sleep
echo "import movie_patch ; movie_patch.movie_patch(shot=#1#,data='rqahesum',fmin=0.0,fmax
    =812649219.41426456,tbase=#2#,rmin=1.155,rmax=1.725,zmin=-1.25,zmax=-0.65)" | python
```

and `shots` containing

```
18559
18577
18593
18615
18637
18662
18685
18711
18731
18744
18757
18772
18785
18795
18808
18822
18835
18847
18879
18897
18917
18951
18978
19005
19034
19059
19089
19113
19139
19157
19190
19211
19232
19258
19280
19300
19331
19352
19379
19405
```

19432
19456
19485
19523
19540
19575
19605
19628
19658
19690
19710
19732
19770
19793
19827
19856
19896
19928
19950
19971
19994
20025
20051
20076
20268
20269
20301
20323
20349
20379
20405
20445
20473
20516
20537
20559
20588
20613
20638
20663
20687
20710
20732
20763
20780
20806
20824
20845
20869
20890
20911
21116
21117
21118
21119
21120
21121
21160
21161
21182
21203
21224
21245
21266
21287
21308
21329

21350
21371
21392
21414
21435
21459
21478
21499
21520
21541
21562
21583
21604
21625
21646
21669
21840
21841
21842
21843
21845
21847
21848
21850
21868
21886
21904
21929
21958
21995
22025
22052
22079
22106
22125
22160
22188
22215
22240
22268
22308
22336
22408
22452
22486
22539
22568
22611
22649
22685
22710
22745
22767
22803
22831
22872
22902
22935
22971
23024
23058
23103
23141
23179
23209
23239
23269

```
23299
23333
23363
23395
23436
23465
23498
23533
23563
23586
23624
23664
23738
23793
23839
23887
23946
23982
24049
24050
24078
24106
24134
24169
24204
```

This launches one job per SOLPS shot-number. Then the frames were renumbered by

```
mkdir frames
cd frames/
set c=0
foreach i ( `cat ../shots` )
  set p=`echo $c | awk '{printf "%03i\n",$1}'`
  echo $p
  @ c++
  foreach j ( `cd ../$i ; echo tmp_????.png`)
    ln -s ../$i/$j `echo $j | sed "s:tmp_0:tmp_${p}:"`
  end
end
```

The individual frames were then combined into a movie with

```
mencoder 'mf://frames/tmp_??????.png' -mf type=png:fps=30 -ovc lavc -lavcopts \
 vcodec=wmv2 -oac copy -o fb_nesepm=5.0e19.ELM_2cm_X10.rqahesum.mpg
```

The resultant movie can be found at
http://solps-mdsplus.ipp.mpg.de/SOLPS/fb_nesepm=5.0e19.ELM_2cm_X10.rqahesum.mpg

6.3 Querying the database

All cases are also entered into a `MySQL` (http://www.mysql.com/) database.

The database can also be queried via `MySQL` commands passed via `MDSplus`. An `IDL` example of querying the database to find the number of shots supplied by each user follows:

```
mdsconnect,'solps-mdsplus.ipp.mpg.de:8001'
ans=mdsvalue("sql('select user,count(*) from id group by user','solps')")
mdsdisconnect
help,ans
print,transpose(ans)
```

Another, more complicated, `IDL` example is:

```
mdsconnect,'solps-mdsplus.ipp.mpg.de:8001'
ans=mdsvalue("sql('select solpsindex,shot,time,ompsep_te,ompsep_ti,user,exp
   from id where ompsep_te>110.0 and ompsep_te<120.0','solps')")
mdsdisconnect
help,ans
print,format='(i6,1x,i6,1x,f6.3,1x,f6.2,1x,1x,f6.2,1x,a10,1x,a)',transpose(ans)
plot,ans(*,3),ans(*,4),psym=4,xtitle='Te (eV)',ytitle='Ti (eV)'
```

And a similar example in python:

`python sql_solps.py`

with `sql_solps.py` containing

```
#! /usr/bin/env python3
import MDSplus
import numpy as np

conn = MDSplus.Connection('solps-mdsplus.ipp.mpg.de:8001')
shape = conn.get("shape(_s=sql(\'select solpsindex,shot,time,ompsep_ne,ompsep_te,user,exp from id
     where ompsep_te>10.0 and ompsep_te<1000.0\',\'solps\'))")
tmp = conn.get("_s")
ans = np.array(tmp).reshape(shape[::-1])

solpsindex = np.array(ans[0,:].tolist()).astype('i8')
shot       = np.array(ans[1,:].tolist()).astype('i8')
time       = np.array(ans[2,:].tolist()).astype('f8')
ompsep_ne  = np.array(ans[3,:].tolist()).astype('f8')
ompsep_te  = np.array(ans[4,:].tolist()).astype('f8')
user       = np.array([e.decode('utf8').strip() for e in ans[5]])
exp        = np.array([e.decode('utf8').strip() for e in ans[6]])

import matplotlib.cm as cm
import matplotlib.pyplot as plt

plt.clf()
plt.loglog(ompsep_ne, ompsep_te, 'o', alpha=0.1)
plt.xlabel('ompsep_ne')
plt.ylabel('ompsep_te')
plt.savefig('sql_scatter.png', dpi=200)
plt.savefig('sql_scatter.ps', dpi=200, bbox_inches='tight')
plt.savefig('sql_scatter.eps', dpi=200)

# filter a subset of the data
filter = (ompsep_ne > 1e19) & (ompsep_ne < 1e20) & (ompsep_te > 10.0) & (ompsep_te <  500.0)

plt.clf()
plt.hexbin(ompsep_ne[filter],ompsep_te[filter],xscale='log', yscale='log', cmap=cm.OrRd)
plt.title("SOLPS")
plt.xlabel('ompsep_ne')
plt.ylabel('ompsep_te')
cb = plt.colorbar()
cb.set_label('N')
plt.savefig('sql_hexbin_lin.png', dpi=200)
plt.savefig('sql_hexbin_lin.ps', dpi=200, bbox_inches='tight')
plt.savefig('sql_hexbin_lin.eps', dpi=200)

plt.clf()
plt.hexbin(ompsep_ne[filter], ompsep_te[filter], bins='log', xscale='log', yscale='log', cmap=cm.
    OrRd)
plt.title("SOLPS")
plt.xlabel('ompsep_ne')
plt.ylabel('ompsep_te')
cb = plt.colorbar()
cb.set_label('N')
plt.savefig('sql_hexbin_log.png', dpi=200, bbox_inches='tight')
```

```
plt.savefig('sql_hexbin_log.ps', dpi=200, bbox_inches='tight')
plt.savefig('sql_hexbin_log.eps', dpi=200,)
```

The resultant plots are given in figures 6.4, 6.5 and 6.6.

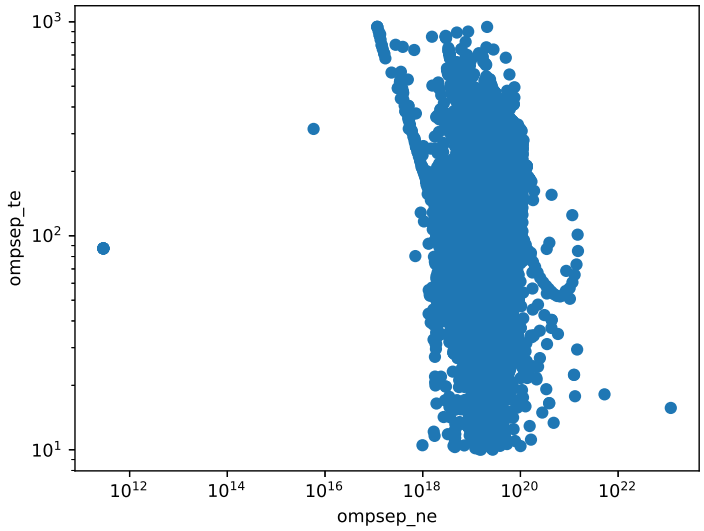


Figure 6.4: Scatter plot for separatrix temperature versus density.

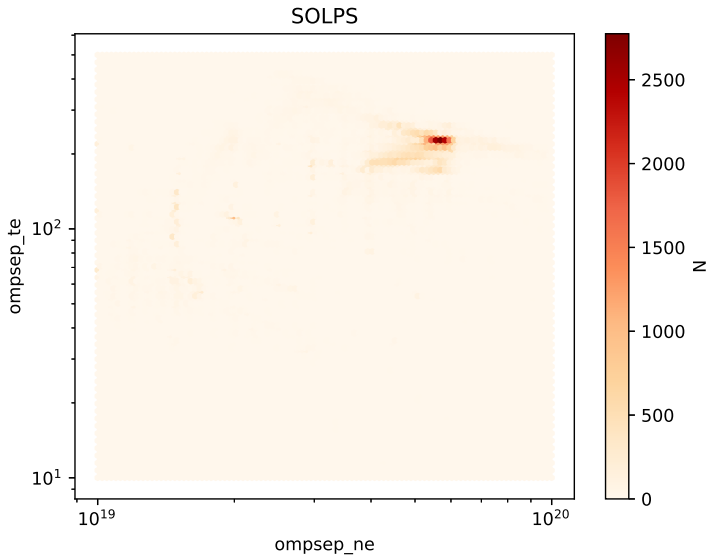


Figure 6.5: Histogram plot for separatrix temperature versus density.

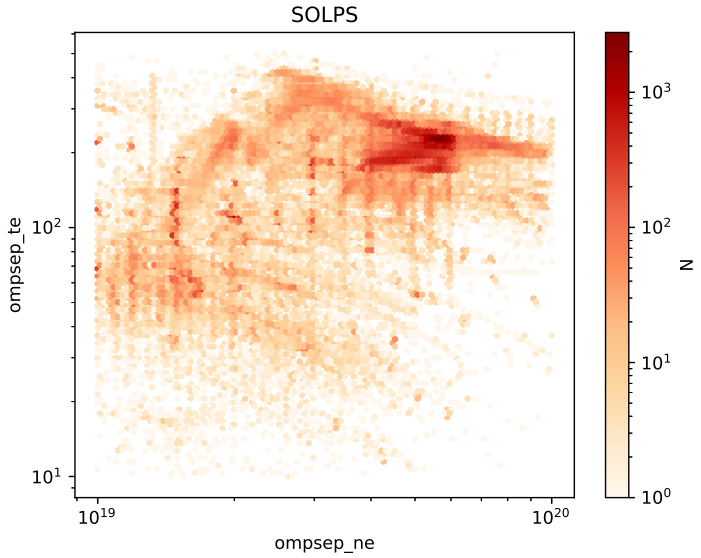


Figure 6.6: Log histogram plot for separatrix temperature versus density.

6.4 Coupling to other codes via MDSplus

Coupling of `SOLPS` to other codes has in the past been done mostly by exchanging files. The preferred route for the future is to use the `SOLPS MDSplus` tree.

As an example of how to do this, a short example `FORTRAN` code which queries the `SOLPS MDSplus` server and then writes geometry information in the standard `Sonnet` format is presented below. The code uses Wolfgang Suttrop's `libitdb` library (http://www.ipp.mpg.de/ Wolfgang.Suttrop/mdsplus/libitdb/).

```
      program write_files
      implicit none
      integer nxdmax, nydmax
      parameter (nxdmax=1024, nydmax=1024)
      integer ishot, status, iargc, lng,
     1 sizes(4), ndim, ndim1, ndim2, ndim3,
     2 icell, ix, iy, nx, ny
      character arg*256, string*256
      real*8 cr(-1:nxdmax, -1:nydmax),  cz(-1:nxdmax, -1:nydmax),
     1 crx(-1:nxdmax, -1:nydmax, 0:3), cry(-1:nxdmax, -1:nydmax, 0:3),
     2 bb(-1:nxdmax, -1:nydmax, 0:3),
     3 dum1(1), dum2(1), dum3(1), b0r0

      if(iargc().ne.1) then
        write(*,*) 'One argument expected, not ',iargc()
        stop
      endif
      call getarg(1,arg)
      read(arg,*) ishot

      call itdb_open(status,"solps-mdsplus.ipp.mpg.de:8001",
     1 "solps", ishot)
```

```
 call itdb_err_msg('itdb_open',status)

 lng=len(string)
 call itdb_gettext(status,"\TOP.IDENT:USER",string,lng)
 call itdb_err_msg('itdb_gettext',status)
 write(*,*) string(1:lng)

 ndim=2
 ndim1=0                 ! nxdmax+2
 sizes(1)=nxdmax+2
 ndim2=0                 ! nydmax+2
 sizes(2)=nydmax+2
 ndim3=0                 ! 1+3
 sizes(3)=1+3

 ndim=2
 ndim1=0                 ! nxdmax+2
 sizes(1)=nxdmax+2
 ndim2=0                 ! nydmax+2
 sizes(2)=nydmax+2
 ndim3=0                 ! 1+3
 CALL itdb_getsig(status,'\TOP.SNAPSHOT.GRID:CR',
*     ndim,sizes,cr,ndim1,dum1,ndim2,dum2,ndim3,dum3)
 call itdb_err_msg('itdb_getsig',status)
 write(*,*) 'CR: ',ndim,sizes(1:ndim),ndim1,ndim2,ndim3
 ndim=2
 ndim1=0                 ! nxdmax+2
 sizes(1)=nxdmax+2
 ndim2=0                 ! nydmax+2
 sizes(2)=nydmax+2
 ndim3=0                 ! 1+3
 CALL itdb_getsig(status,'\TOP.SNAPSHOT.GRID:CZ',
*     ndim,sizes,cz,ndim1,dum1,ndim2,dum2,ndim3,dum3)
 call itdb_err_msg('itdb_getsig',status)
 write(*,*) 'CZ: ',ndim,sizes(1:ndim),ndim1,ndim2,ndim3
 ndim=3
 ndim1=0                 ! nxdmax+2
 sizes(1)=nxdmax+2
 ndim2=0                 ! nydmax+2
 sizes(2)=nydmax+2
 ndim3=0                 ! 1+3
 sizes(3)=1+3
 CALL itdb_getsig(status,'\TOP.SNAPSHOT.GRID:R',
*     ndim,sizes,crx,ndim1,dum1,ndim2,dum2,ndim3,dum3)
 call itdb_err_msg('itdb_getsig',status)
 write(*,*) 'R: ',ndim,sizes(1:ndim),ndim1,ndim2,ndim3
 ndim=3
 ndim1=0                 ! nxdmax+2
 sizes(1)=nxdmax+2
 ndim2=0                 ! nydmax+2
 sizes(2)=nydmax+2
 ndim3=0                 ! 1+3
 sizes(3)=1+3
```

```
      CALL itdb_getsig(status,'\TOP.SNAPSHOT.GRID:Z',
     *     ndim,sizes,cry,ndim1,dum1,ndim2,dum2,ndim3,dum3)
      call itdb_err_msg('itdb_getsig',status)
      write(*,*) 'Z: ',ndim,sizes(1:ndim),ndim1,ndim2,ndim3
      ndim=3
      ndim1=0                 ! nxdmax+2
      sizes(1)=nxdmax+2
      ndim2=0                 ! nydmax+2
      sizes(2)=nydmax+2
      ndim3=0                 ! 1+3
      sizes(3)=1+3
      CALL itdb_getsig(status,'\TOP.SNAPSHOT:B',
     *     ndim,sizes,bb,ndim1,dum1,ndim2,dum2,ndim3,dum3)
      call itdb_err_msg('itdb_getsig',status)
      write(*,*) 'B: ',ndim,sizes(1:ndim),ndim1,ndim2,ndim3

      call itdb_close(status)
      call itdb_err_msg('itdb_close',status)

      nx=sizes(1)-2
      ny=sizes(2)-2

      open(99,file=trim(arg)//'.sonnet')
      icell=0
      write(99,*)
      write(99,'(3x,a/)') 'Element output:'
      b0r0=cr(-1,-1)*bb(-1,-1,3)

      write(99,*) '   R*Btor =',b0r0
      write(99,'(/3x,2a)')
     1 '======================================',
     2 '================================================'
      do iy=-1,ny
        do ix=-1,nx
          write(99,10001) icell,ix+1,iy+1,
     1     crx(ix,iy,2),cry(ix,iy,2),crx(ix,iy,3),cry(ix,iy,3)
10001     format(3x,'Element',i5,' = (',i3,',',i3,'): (',
     .     1pe17.10,',',1pe17.10,')',
     .     6x,'(',1pe17.10,',',1pe17.10,')')
          write(99,10002)
     1     sqrt(bb(ix,iy,0)**2+bb(ix,iy,1)**2)/bb(ix,iy,3),
     2     cr(ix,iy),cz(ix,iy)
10002     format(3x,'Field ratio  = ',1pe17.10,13x,
     .     '(',1pe17.10,',',1pe17.10,')')
          write(99,10003)
     1     crx(ix,iy,0),cry(ix,iy,0),crx(ix,iy,1),cry(ix,iy,1)
10003     format(
     .     t30,'(',1pe17.10,',',1pe17.10,')',
     .     6x,'(',1pe17.10,',',1pe17.10,')')
          write(99,10005)
10005     format(3x,
     1     '-------------------------------------------------',
     2     '-----------------------------------')
```

```
    icell=icell+1
  enddo
enddo
close(99)

end

subroutine itdb_err_msg(name,status)
implicit none
character*(*) name
integer status

if (status.ne.0) then
  write(*,*) 'Routine ',trim(name),' returned error code ',status
endif
return
end
```

6.5 Some information about the server

The IPP MDSplus server is a dual processor 2.8 GHz AMD Opteron machine with 4 GB RAM running SUSE LINUX 10.1 (X86-64). The data is stored in MR-AFS.

As of Wed Jun 1 21:30:57 CEST 2005 the SOLPS MDSplus server "solps-mdsplus.ipp.mpg.de:8001" contained 12497 cases consisting of 387.158 GB in 261099 files.

As of Wed Jul 26 19:12:53 CEST 2006 the SOLPS MDSplus server "solps-mdsplus.ipp.mpg.de:8001" contained 15999 cases consisting of 995.904 GB in 334534 files.

As of Sat Mar 8 00:46:36 CET 2008 the SOLPS MDSplus server "solps-mdsplus.ipp.mpg.de:8001" contained 22190 cases consisting of 2824.17 GB in 464547 files.

As of Tue Oct 7 19:45:57 CEST 2008 the SOLPS MDSplus server "solps-mdsplus.ipp.mpg.de:8001" contained 25623 cases consisting of 3646.37 GB in 536644 files.

As of Tue Jan 13 10:05:51 MET 2009 the SOLPS MDSplus server "solps-mdsplus.ipp.mpg.de:8001" contained 26448 cases consisting of 3802.68 GB in 553970 files.

As of Thu Jun 8 11:22:51 CEST 2017 the SOLPS MDSplus server "solps-mdsplus.ipp.mpg.de:8001" contained 95977 cases consisting of 19859.1 GB in 1929218 files.

As of 2025-07-31T09:02:17 CEST the SOLPS MDSplus server "solps-mdsplus.ipp.mpg.de:8001" contained 186288 cases consisting of 42 TB in 3909532 files.

Chapter 7

Running DivGeo

You will need to choose your device, *e.g.*:

```
setenv DEVICE iter
```

Available choices are:

aditya ADITYA (Gandhinagar, India)

asdex ASDEX (Garching, Germany), now HL–2A (Chengdu, China)

cfns Compact Fusion Neutron Source (China)

cmod Alcator C–Mod (Boston, USA)

demo DEMO (many incarnations)

d3d DIII–D (San Diego, USA)

dream (Japan)

east EAST (Hefei, China)

globus Globus–2M (St. Petersburg, Russia)

hl2a HL-2A (Chengdu, China)

hl2m HL-2M (Chengdu, China)

iter ITER (Cadarache, France) (default)

iter.dg link to `iter`

iterm ITER (modified)

jet JET (Culham, UK)

jt60sa JT60–SA (Naka, Japan)

jt60u JT60–U (Naka, Japan)

kstar KSTAR (Daejon, Korea)

mast MAST (Culham, UK)

nhtx NHTX (Hefei, China)

sst SST (Gandhinagar, India)

t15m T15–M (Moscow, Russia)

tcv TCV (Lausanne, Switzerland)

ts Tore Supra (Cadarache, France)

upgrade ASDEX–Upgrade (Garching, Germany)

upgrade.dg link to `upgrade`

The default setting for `DEVICE` can be changed for each local implementation *via* use of the `setup.csh.*` files listed in section 2.1.

If `DivGeo` is not yet compiled, you will need to do

```
cd $DG
gmake tags
gmake listobj
gmake depend
gmake
cd equtrn
gmake depend
gmake
cd ../convert
gmake
```

The environment variable `DG` contains the path `${SOLPSTOP}/modules/DivGeo`. To start `DivGeo`, the command is:

```
./dg &
```

or

```
./dg case.dg &
```

where `case.dg` is an existing case.

The reader is referred to the `DG` manual in Appendix L for further information of running `DG` itself.

After saving (`File|Save`) and outputting (`File|Output`) you will then need to

```
./lns case
```

where `case` is the name of the just saved case (without the `.dg` extension). This links the `DG` output files to files named `dg.*` in the `Carre` run directory.

The `lns` script establishes a series of symbolic links from the `DivGeo` output files to some files with standardized names on which other parts of the workflow rely. It also copies the `case.trg` target and `case.str` structure description files to the `${DGSONNETMASK}` directory (with default value `${DG}/device/${DEVICE}` or `${DG}/device/${DEVICE}/${USER}` for central installation).

Then you will probably want to generate the grid — this is described in the next chapter. After the grid has been created, you can import it back into `DG` (`File|Import|Mesh`) to check it.

Chapter 8

Running Carre

If `Carre` is not yet compiled, you will need to do

```
ssc
gmake tags
gmake listobj
gmake depend
gmake
```

With the installation instructions from Section 1.1, users get a local clone of the `Carre` GIT submodule, and **this is the only supported option!** The files in `src` will be version-controlled and updated whenever the user will pull in new code version. If, however, one wishes to make local changes to a file and protect those from GIT updates, the procedure for that purpose is to copy the file you want to change to `src.local` and then edit that file. Once you have copied and edited the file(s) you need to change in `src.local`, you should recompile with:

```
stop
gmake listobj
gmake tags
gmake depend
gmake (carre)
```

so that the local file is used. The last `gmake` command may of course be replaced by `gmake carre(_debug)`, `gmake solps(_debug)`, `gmake solps_mpi(_debug)`, `gmake all(_debug)` or `gmake all_mpi(_debug)` as needed.

If this local change should no longer be needed, then it should be removed from `src.local` and the following performed:

```
stop
gmake clean(_carre)
gmake listobj
gmake tags
gmake depend
gmake (carre)
```

If you want to avoid a long recompilation, you need to remove from the `builds/*/` directories , the object and module files corresponding to your locally changed file(s), and then ask for a normal recompilation with a single `gmake` command.

To run `Carre`, assuming that you are working on a case from `DG`, you would then need to do

```
carre -
```

and then either type `return` to accept the default, or a `y` if asked to accept `Carre`'s choices.

The presence in the run directory of a file called `ncar.cfg`, with the first character of the first line non-blank, will cause the program to use 'publication-ready' settings for the graphs, namely thicker and visible lines and easier-to-read font.

The program will propose a table (see next chapter for an example) containing, for each region of the grid, the relevant meshing parameters. The first table (titled `SEPARATRIX SEGMENTS`) includes the variables for the poloidal meshing along the separatrix. For a single-null case, segment 1 goes from the inner divertor to the X-point, segment 2 from the outer divertor to the X-point, and segment 3 around the core in the clockwise direction. For more complicated geometries, see the `Carre` original paper [15]. The variables in the table mean:

nptseg Number of poloidal mesh points along the segment

lg Length of the segment (in m)

deltp1 Distance from the 1^{st} to the 2^{nd} mesh point (The first mesh point is defined to be the X-point) (in m)

deltpn Distance from the penultimate to the last mesh point (The last mesh point is defined to be the strike point) (in m)

dpmin Minimum distance achieved between two neighbouring points (in m)

dpmax Maximum distance achieved between two neighbouring points (in m)

Of the above, the user is only allowed to change `nptseg`, `deltp1` and `deltpn`. If the code, when using the default parameters (which are imported from the `DG` run), initially fails, re-run trying a different set of parameters. The code in some cases gives some indication of which region and which surface was problematic. For the most common error scenarios, some sanity checks are performed at this stage and the code will not allow the user to continue unless all of these are satisfied. In particular, make sure that for each individual segment `lg`, `deltp1`, `deltpn`, `dpmin` and `dpmax` are all of the same sign. It is recommended to have the sizes of the mesh steps on all sides of the X-points to be comparable, and to increase resolution gradually towards the strike points. The values of `dpmin` and `dpmax` can be used to check the monotonicity of the mesh step size. For reasons related to numerical convergence of the plasma solvers, the grid spacing between neighbouring cells should not change too quickly and must be prevented from collapsing to zero. To this effect, `Carre` imposes a minimum grid spacing given by the `pasmin` value. However, the spacing of the points as initially provided by `DG` is unaware of that limit, and thus it is often the case that the user must adapt either the grid spacings or `pasmin` (or both) before being allowed to continue.

The second table (titled `RADIAL DISTRIBUTIONS`) includes the variables for the radial meshing away from the separatrix. For a single-null case, region 1 is the private flux region (PFR), and region 2 the scrape-off layer (SOL). For more complicated geometries, see the `Carre` original paper [15].

repart Radial repartition of points is done according to distance if `repart=1` (recommended) or according to the magnetic flux function `psi` if `repart=2`. Units for the quantities below are then either m or Wb.

npr Number of radial mesh points in the region

width Width of the region

deltr1 Distance from the 1^{st} to the 2^{nd} mesh point (The first mesh point is defined to be on the separatrix)

deltrn Distance from the penultimate to the last mesh point

drmin Minimum distance achieved between two neighbouring points

drmax Maximum distance achieved between two neighbouring points

The user is only allowed to change `repart`, `npr`, `deltr1` and `deltrn`. Again, make sure that for each individual segment `width`, `deltr1`, `deltrn`, `drmin` and `drmax` are all of the same sign. Regions on opposite sides of the separatrix will usually have opposite signs for their respective `width` values. It is recommended to have the sizes of the mesh steps on both sides of the separatrix be comparable, and to decrease resolution gradually towards the

boundaries. For disconnected double-null cases, the region between the two separatrices must be meshed with special care. The values of drmin and drmax can be used to check the monotonicity of the mesh step size.

The third table (titled CENTRAL REGION) includes the variables for the radial meshing in the core. For a single-null case, this is region 3. The repart switch from above applies here as well. For more complicated geometries, see the Carre original paper [15].

pntrat Radial penetration from the separatrix into the core to the innermost meshed flux surface. The O-point is located at the value pntrat max.

npr Number of radial mesh points in the core

width Width of the region

deltr1 Distance from the 1^{st} to the 2^{nd} mesh point (The first mesh point is defined to be on the separatrix)

deltrn Distance from the penultimate to the last mesh point

drmin Minimum distance achieved between two neighbouring points

drmax Maximum distance achieved between two neighbouring points

The user is only allowed to change pntrat, npr, deltr1 and deltrn. Again, the code forces the user to ensure that width, deltr1, deltrn, drmin and drmax are all of the same sign. It is recommended to keep the sizes of the mesh steps on both sides of the separatrix to be comparable (compare with the RADIAL DISTRIBUTIONS table), and to decrease resolution gradually towards the inner core boundary and outer regions. The values of drmin and drmax can be used to check the monotonicity of the mesh step size. For a B2 or B2.5 run, it is necessary that npr in the PFR and core regions be the same. This check will be performed by the b2ag grid preprocessor and an error message returned if the condition is not satisfied. Similar more complicated checks will be done for double-null geometries.

The fourth table (titled GUARD LENGTH FOR EACH DIVERTOR PLATE) gives the poloidal length (in m) tgarde over which the grid is to be adjusted from strictly orthogonal to coincident with the target geometry. This is a highly device-dependent quantity, but a value of the order of the length of the relevant separatrix segment is recommended. In single-null geometries, tgarde(1) refers to the inner target, and tgarde(2) to the outer target.

The last table (titled RELAXATION PARAMETERS USED TO CONSTRUCT THE MESH) gives numerical parameters used to smooth out the mesh. Carre uses a relaxation scheme to optimize the mesh so as to make it as orthogonal as possible within the given constraints of the field geometry and the shape of the target plates. The parameters governing this scheme are:

nrelax The maximum number of iterations allowed

relax The relaxation factor used

pasmin The minimal spacing required between neighbouring surfaces in the poloidal direction

rlcept The maximum desired deviation from orthogonality (averaged over a flux surface)

If the code fails to position the points poloidally in a way that satisfies the optimization criteria, an error message will be printed showing the deviation from orthogonality obtained and options likely to improve it. The user may ignore the message and proceed, but the mesh provided may not then be of sufficient quality.

Lastly, there is a "hidden" language variable sellan (default value 'english') that can be changed to its original 'francais'.

Once all the settings have been chosen, it is often convenient to save them, so as to not have to re-enter them on the next Carre meshing attempt. This is achieved by means of the (S)ave option within the carre script, provided that the user does not start the next meshing attempt with the (P)repare option, but rather jumps directly to (G)rid.

Upon succesful termination, one needs to (C)onvert the grid files to the .geo and .sno formats understandable by B2.5 and DivGeo. It is also possible to s(T)ore the files used to create the grid and the grid output for archival. The input files will be copied to the

${CARRE_STOREDIR} directory (default value is ${SOLPSTOP}/modules/Carre/meshes/${DEVICE}, or ${SOLPSTOP}/modules/Carre/meshes/${DEVICE}/${USER} for a central installation). The Carre grid output and .geo file will be stored in the same location, but that can be changed by means of the (O)utput command in the script itsef, or by specifying the ${CARRE_GRIDDIR} environment variable. If ${CARRE_GRIDDIR} is not specified, the *.sno converted output file will be copied instead to ${SOLPSTOP}/modules/DivGeo/device/${DEVICE} (or ${SOLPSTOP}/modules/DivGeo/device/${DEVICE}/${USER} for a central installation).

After running Carre, it is recommended to have a look at the mesh and decide if one wants to modify it before proceeding much further (see previous section). The default settings may not be appropriate for all cases. It is particularly important to check the mesh near the separatrix, the X-point(s) and the target plates.

In some cases, the carre – script will fail. In that case, one can still run Carre in "standalone" mode by invoking:

```
ssc
builds/${HOST_NAME}.${COMPILER}(.debug)/carre
```

Chapter 9

Running agg

`agg` is an interface to `Carre` for generating the grid for a specified ASDEX Upgrade shot at a particular time point. There is also a similar program, `aggj` for JET. `agg` was written by Heimo Bürbaumer.

Running the code at IPP produces:

```
15:36> ./aggs
AGG: Automatic Grid Generator
For fully automatic use type: aggs -s shotnumber -t time
and optionally: -nx pol.grid -ny rad.grid -depth distance
with shotnumber as integer, time in s (2.7), poloidal grid (96,...)
rad.grid (32, ... ) and distance betw. X-Point and inner boundary in cm
For half-automatic use type: aggs -i
Requesting shot 10095(1) of DPC:EQI 10095(1)
Requesting shot 8650 of AUGD:YGC 8650
 kkEQPFM: ierr=0
 kkEQpfx: ierr=0
 kkrzBrzt: ierr0
 1.74294448,  1.52229261,  2.15042973,  1.68617392,  1.11173415
 0.130604938,  -0.858187616,  0.449555516,  1.26027465,  -0.581599176
 ngr,ngz=65,  129
 10095,  3.20000005,  1 dpc  EQI 96x24
 Starting
 newpag OK
 cpsets ok
 cprect ok
 cpcldr ok
 Pre-selected points are identified.
 o-point:     1.7426E+00    1.3054E-01
 x-point:     1.5218E+00   -8.5777E-01


 Do you accept the selection (y/n)?

 Entering sptris: npx=  1

 ipx,ix,jx,ptx,pty =  1  26  28    1.5217840062009  -0.85776841963238
 psi(ix,jx),CCW =   -7.9810600000000D-04    8.9943400000000D-05
     4.3189500000000D-04    -1.7131000000000D-03
 fctpx =   -1.5440357642937D-04

 idir,nbdfav,isep =  1  0  0
 i1,i2,j1,j2,iref,jref =  28  28  27  28  28  27
 psi(i1,j1),psi(i2,j2),fctpx(ipx)=   -2.4685900000000D-03   -5.0395700000000D-03
    -1.5440357642937D-04
 i1,i2,j1,j2,iref,jref =  28  28  28  29  28  28
 psi(i1,j1),psi(i2,j2),fctpx(ipx)=   -5.0395700000000D-03   -3.9870700000000D-03
    -1.5440357642937D-04
```

```
i1,i2,j1,j2,iref,jref =  28  28  29  30  28  29
psi(i1,j1),psi(i2,j2),fctpx(ipx)=   -3.9870700000000D-03    8.0841800000000D-04
   -1.5440357642937D-04
Cell crossing: k,isep,ipx,separx,separy =  2  1  1    1.5600000000000
  -0.82721824912211
insert: indstr,nbdef,ipx  1  0  1

idir,nbdfav,isep =  2  1  1
i1,i2,j1,j2,iref,jref =  28  27  30  30  27  30
psi(i1,j1),psi(i2,j2),fctpx(ipx)=    8.0841800000000D-04    4.4249300000000D-03
   -1.5440357642937D-04
i1,i2,j1,j2,iref,jref =  27  26  30  30  26  30
psi(i1,j1),psi(i2,j2),fctpx(ipx)=    4.4249300000000D-03    9.2840200000000D-04
   -1.5440357642937D-04
i1,i2,j1,j2,iref,jref =  26  25  30  30  25  30
psi(i1,j1),psi(i2,j2),fctpx(ipx)=    9.2840200000000D-04   -8.7230800000000D-03
   -1.5440357642937D-04
Cell crossing: k,isep,ipx,separx,separy =  2  2  1    1.4966342819380
  -0.82250000000000
insert: indstr,nbdef,ipx  2  1  1

idir,nbdfav,isep =  3  2  2
i1,i2,j1,j2,iref,jref =  25  25  30  29  24  29
psi(i1,j1),psi(i2,j2),fctpx(ipx)=   -8.7230800000000D-03   -9.5526000000000D-03
   -1.5440357642937D-04
i1,i2,j1,j2,iref,jref =  25  25  29  28  24  28
psi(i1,j1),psi(i2,j2),fctpx(ipx)=   -9.5526000000000D-03   -7.2130600000000D-03
   -1.5440357642937D-04
i1,i2,j1,j2,iref,jref =  25  25  28  27  24  27
psi(i1,j1),psi(i2,j2),fctpx(ipx)=   -7.2130600000000D-03   -1.7010600000000D-03
   -1.5440357642937D-04

idir,nbdfav,isep =  4  2  2
i1,i2,j1,j2,iref,jref =  25  26  27  27  25  26
psi(i1,j1),psi(i2,j2),fctpx(ipx)=   -1.7010600000000D-03    3.5539300000000D-03
   -1.5440357642937D-04
Cell crossing: k,isep,ipx,separx,separy =  2  3  1    1.4788296443394
  -0.89300000000000
insert: indstr,nbdef,ipx  1  2  1
i1,i2,j1,j2,iref,jref =  26  27  27  27  26  26
psi(i1,j1),psi(i2,j2),fctpx(ipx)=    3.5539300000000D-03    3.4159400000000D-03
   -1.5440357642937D-04
i1,i2,j1,j2,iref,jref =  27  28  27  27  27  26
psi(i1,j1),psi(i2,j2),fctpx(ipx)=    3.4159400000000D-03   -2.4685900000000D-03
   -1.5440357642937D-04
Cell crossing: k,isep,ipx,separx,separy =  2  4  1    1.5482020156738
  -0.89300000000000
insert: indstr,nbdef,ipx  2  2  1

After idir loop: isep,ipx,nbdef,nbdfav =   4  1  2  2

after x-point loop: isep,ipx,nbdef,nbdfav =   4  2  2  2
ixp_hlp :  1  1
xsttmp :    1.2536294134836    1.6897170950977
ysttmp :   -1.0691543089581   -1.0565671252574
nptot
  226 220  18  14
    0   0   0   0
indplq
    1   2   1   2
    0   0   0   0
Entering trgarng: nbdef=  2
inddef:       1         2
indxpt:       1         1
xst:     1.254E+00 1.690E+00
yst:    -1.069E+00-1.057E+00
Single intersections :   2
```

```
 jj :   1  2
 ll :   1  1
 Complete x-points:   1
 kk :   1
 Before re-arrangement. k =   2
 jj :   1  2
 After re-arrangement:   2
 inddef:       1         2
 xst:     1.254E+00 1.690E+00
 yst:    -1.069E+00-1.057E+00
 Leaving trgarng:   2
 Entering orddef: npx,nbdef=   1  2
 inddef:    1   2
 xst:   1.25E+00  1.69E+00
 yst:  -1.07E+00 -1.06E+00
 ptx:   1.52E+00
 pty:  -8.58E-01
 After ordering in y
 inddef:    2   1
   i ipx    xy      pty     aire  :                   wx                :                   wy
   1  1 -8.58E-01 1.31E-01-4.44E-02 1.52E+00 1.69E+00 1.25E+00 1.52E+00-8.58E-01-1.06E+00-1.07E+00-8.58E-01
 After ordering in x
 inddef:    2   1
 Leaving sptris...

 === carre *..10.0 - before frtier
 ptx:  1.5218E+00  1.7426E+00  0.0000E+00  0.0000E+00
 pty: -8.5777E-01  1.3054E-01  0.0000E+00  0.0000E+00
 nptot(4,nxpoints)
   204    1   17   13    0    0    0    0    0    0    0    0    0    0    0    0
 Strike points (presumably)
 x:  1.5218E+00  1.5218E+00  1.2536E+00  1.6897E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00

 y: -8.5777E-01 -8.5777E-01 -1.0692E+00 -1.0566E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00



         SEPARATRICES

 #   nptseg(i)   lg(i)      deltp1(i)   deltpn(i)     dpmin       dpmax
=============================================================================
 1      25      2.6059E-01  5.2083E-03  5.2083E-03  5.2083E-03  1.4051E-02
 2      25      3.4168E-01  5.2083E-03  5.2083E-03  5.2083E-03  1.9340E-02
 3      49      4.1421E+00  5.2083E-03  5.2083E-03  5.2083E-03  1.2942E-01


         RADIAL DISTRIBUTIONS
             Distribution in psi: repart=    2

region  npr(i)    width     deltr1(i)   deltrn(i)     drmin       drmax
=============================================================================
 1      13     -1.3291E-01 -1.0000E-03 -2.0833E-03 -1.7192E-02 -1.0000E-03
 2      13      5.0702E-02  1.0000E-03  2.0833E-03  1.0000E-03  5.9821E-03


         CENTRAL REGION: region i=  3     pntrat max.=1.01267691

pntrat  npr(i)    width     deltr1(i)   deltrn(i)     drmin       drmax
=============================================================================
0.3000   13     2.3836E-01  1.0000E-03  2.0833E-03  1.0000E-03  3.1572E-02


         GUARD LENGTH FOR EACH DIVERTOR PLATE

tgarde(1)      tgarde(2)
=============================================================================
0.00000         0.00000
```

```
         RELAXATION PARAMETERS USED TO CONSTRUCT THE MESH

================================================================================
nrelax        relax          pasmin         rlcept
  200         0.200           5.000E-03      1.000E-06


Do you wish to accept these values (y/n)?
 ir=  2
 ir=  3
 ir=  4
 ir=  5
 ir=  6
 ir=  7
 ir=  8
 ir=  9
 ir=  10
 ir=  11
 ir=  12
 ir=  13
 ir=  2
 ir=  3
 ir=  4
 ir=  5
 ir=  6
 ir=  7
 ir=  8
 ir=  9
 ir=  10
 ir=  11
 ir=  12
 ir=  13
 ir=  2
 ir=  3
 ir=  4
 ir=  5
 ir=  6
 ir=  7
 ir=  8
 ir=  9
 ir=  10
 ir=  11
 ir=  12
 ir=  13
Warning: Distance Separatix-outer grid line is small: 2.40 cm !
But mesh-generation is valid !
 Distance Separatix-outer grid line in private flux region is: 6.37 cm !
 Name of the file containing the carre grid
 Select the output format
 1: standard mailtri format
 2: format B2.5
 3: format SONNET-DIVIMP
 4: format DG-SONNET-B2-EIRENE
 ===> Entering b2agfz. nreg =   3
 Input: b0r0 =    -4.1984658200000
 topology: sn-down
15:37>
```

Appendix A

Internal B2.5 documentation

A.1 b2cdca.F

```
*-----------------------------------------------------------------------
*.specification

      subroutine b2cdca ()
      implicit none

*-----------------------------------------------------------------------
*.documentation
*
*  1. purpose
*
*     B2CDCA provides some general documentation of the B2 code.
*     No computation is performed in this routine.
*
*-----------------------------------------------------------------------
*.text
*
*  0. Introduction
*
*     The B2 code /1,2/ solves a system of fluid equations that models
*     a multi-species plasma in a two-dimensional geometry. Its main
*     application is simulation of the edge plasma in tokamaks.
*
*     B2 is composed of several independent programs that communicate
*     through external files. The main sequence involves the programs
*     b2ag (prepares a geometry and magnetic field), b2ah (prepares
*     a table of default physics parameters), b2ai (prepares an
*     initial plasma state), b2mn (principal calculation), and b2yp
*     (postprocessing).
*     Additional programs may prepare tables of rate coefficients for
*     atomic processes, tables of plasma-wall interaction processes,
*     external source terms, and other physics, or may perform further
*     postprocessing.
*
*     ...
*
*  1. Mathematical model
*
*     ...
*
*  2. Geometry
*
*     B2 employs a two-dimensional, curvilinear, topologically
```

```
*     rectangular mesh. The coordinates of the mesh are labelled 'x'
*     and 'y'. There are nx+2 cells along the x-coordinate and ny+2
*     cells along the y-coordinate. The cells of the mesh are indexed
*     by (ix,iy) in (-1:nx,-1:ny); the interior cells have index in
*     (0:nx-1,0:ny-1). The mesh may contain cuts, which typically
*     originate from an x-point in the real*8 geometry.
*
*     The primary data describing the mesh are the coordinates of the
*     four vertices of all the cells of the mesh. These data are stored
*     in the arrays crx and cry. The metric parameters (vol, hx, hy, qz,
*     qc and gs; described elsewhere) are derived from crx and cry.
*
*     The primary data describing the magnetic field are the flux
*     functions fpsi and ffbz at the cell vertices. The magnetic field
*     components at cell centers are derived from fpsi and ffbz.
*
*     Singular regions in the geometry are specified by the data in the
*     commmon block /b2cmgs/.
*
*     ...
*
*  3. Numerical method
*
*     ...
*
*  4. References
*
*     (Refs. /1/ and /2/ are out of date; !! to be replaced.)
*     /1/ B. J. Braams, Computational Studies in Tokamak Equilibrium
*     and Transport, thesis, State University at Utrecht, June 1986.
*     /2/ B. J. Braams, A Multi-Fluid Code for Simulation of the Edge
*     Plasma in Tokamaks, NET Report Nr 68, January 1987.
*     /3/ R. A. Hulse, Numerical Studies of Impurities in Fusion
*     Plasmas, Nuclear Technol./Fusion, vol.3 (1983) pp.259-272.
*     /4/ R. A. Hulse, !! (detailed description?).
*     /5/ K. Behringer, Description of the Impurity Transport Code
*     STRAHL, JET Report R(87)08, 1987.
*     /6/ K. Behringer, !! (journal article?).
*     /7/ Jon C. Weisheit, Recombination in Dense Plasmas, J. Phys. B
*     (Atom. Molec. Phys.) 8 (1975) 2556.
*     /8/ R. K. Janev, D. E. Post, W. D. Langer, K. Evans, D. B.
*     Heifetz and J. C. Weisheit, Survey of Atomic Processes in Edge
*     Plasmas, J. Nuclear Mater. 121 (1984) 10-16.
*
*-A1. Notes
*
*   ..Adherence to ANSI Fortran-77.
*     We respect the standard with the following exceptions.
*     - 'implicit none' is used to force type declarations for all
*       constants, variables and functions. We also declare, as external
*       or intrinsic, all procedures.
*     - 'do, enddo' syntax is used; a pre-processor based on Emacs
*       (courtesy of Charles Karney) may convert to labelled do loops.
*     - CPP 'include' statements are used for common blocks and for some
*       parameter statements.
*     - symbolic names may have up to eight characters.
*     These extensions to ANSI Fortran-77 are accepted in the identical
*     syntax on all popular workstations as well as on the Cray.
*     The code can be distributed using do-enddo syntax or using
*     labelled do loops, and with or without use of include statements.
*
*   ..Scientific libraries.
*     The BLAS (basic linear algebra subroutines) and LAPACK (linear
*     algebra package) are employed; these are included in the standard
*     scientific libraries on many computer systems, and are also in
*     NAG and SLATEC; moreover, they are available by electronic mail
*     or ftp from netlib@ornl.gov.  Some simple additional routines
```

```
*     from the NAG library may be employed, for constants of the machine
*     arithmetic and for time and date.
*     An implementation of these scientific libraries is required in
*     which real*8 variables occupy eight bytes.
*
*   ..Graphics libraries.
*     As distributed, B2 relies on NCAR and GKS for graphics output.
*     The graphics postprocessing codes for B2 are completely separate
*     from the main calculation.
*
*   ..Concerning 'un*formatted' files.
*     The programs that make up B2 communicate through external files,
*     which could be unformatted if all programs run on the same
*     computer system, and otherwise would be formatted. In the latter
*     case the format is very simple, and not designed for convenience
*     of a human reader. We refer to these files as 'un*formatted'.
*     Reading from and writing to an un*formatted file is done through
*     the routines cfruch and cfwuch for character variables, cfruin
*     and cfwuin for integer variables, and cfrure and cfwure for real
*     variables.
*     The un*formatted files could easily be replaced by files written
*     using 'NetCDF' or 'HDF' routines if either one of those packages
*     is desired as a standard interface.
*
*   ..Concerning internal documentation.
*     The style of the internal documentation follows that of the NAG
*     library routine documents.
*     It has been the intent to specify the meaning of any non-trivial
*     variable in the outermost routine in which it is declared, and
*     also in the routine in which it is first computed.
*
*   ..Special patterns in comments.
*     The pattern !! is used to draw attention to segments of code
*     that might have to be changed. The pattern *$$$ in columns 1-4 is
*     used for code that is commented out with the text retained as a
*     reminder.
*
*   ..Contacting the author.
*     Use e-mail to braams@cims.nyu.edu, phone to 212-998-3276,
*     FAX to 212-995-4121, or regular mail to
*       B. J. Braams
*       Courant Institute of Mathematical Sciences
*       New York University
*       251 Mercer Street
*       New York, NY 10012
*
*-----------------------------------------------------------------------
*.end b2cdca

      end subroutine b2cdca
```

A.2 b2cdcr.F

```
*-----------------------------------------------------------------------
*.specification

      subroutine b2cdcr ()
      implicit none

*-----------------------------------------------------------------------
*.documentation
*
*  1. purpose
*
*     B2CDCR contains a brief statement of the purpose of many of the
*     subroutines of the B2 code. No computation is performed in this
*     routine.
*
*-----------------------------------------------------------------------
*.text
*
*  0. Overview and naming conventions.
*
*     B2A*, B2B* - Auxiliary programs and their subroutines.
*       B2AG* - Preparation of a geometry.
*       B2AH* - Preparation of default physics parameters.
*       B2AI* - Preparation of an initial state.
*       B2AR* - Preparation of atomic rate coefficients.
*       B2AW* - Reserved for preparation of wall data.
*     B2C* - Documentation and 'include' segments.
*       B2CD* - Documentation.
*       B2CM* - 'include' segments: parameters and common blocks.
*     B2D* - Reserved for DEGAS interface.
*     B2E* - Reserved for EIRENE interface.
*     B2F*-B2L* - Not used.
*     B2M* - Core routines for main program: driver and timestepping.
*       B2MN_INIT - Main program initialisation.
*       B2MN_STEP - Main program call.
*       B2MN_FIN - Main program finalization.
*       B2MNDR_0 - Main driver routine initialisation.
*       B2MNDR_1 - Main driver routine.
*       B2MNDR_2 - Main driver routine finalisation.
*       B2MNDS - Initialisation.
*       B2MNDT - Time stepping.
*       B2MW* - Monitoring of the calculation.
*       B2MX* - Auxiliaries for b2m*.
*     B2N* - Driver routines for inner iteration.
*       B2NEWS - Controls inner iteration.
*       B2NP* - Iteration for individual equations.
*       B2NX* - Auxiliary routines for b2n*.
*     B2O*-B2Q* - Reserved for alternative numerical core.
*     B2R* - Read from an external file.
*       B2RF* - Read formatted.
*       B2RU* - Read un*formatted.
*     B2S* - Evaluate source term coefficients.
*       B2SI* - Evaluate internal source terms.
*       B2SP* - Evaluate log-log linearised rate coefficients.
*       B2SQ* - Evaluate rate coefficients.
*       B2SR* - Evaluate or modify total source term coefficients.
*       B2ST* - Evaluate partial source term coefficients.
*     B2T* - Evaluate transport coefficients and fluxes.
*       B2TF* - Evaluate fluxes.
*       B2TL* - Apply flux limits to transport coefficients.
*       B2TQ* - Evaluate differential transport coefficients.
*       B2TR* - Evaluate cell-integrated transport coefficients.
*       B2TX* - Auxiliary routines for b2t*.
*     B2U* - Inner routines for the inner iteration.
```

```
*       B2UP* - Update some quantity.
*       B2UR* - Evaluate a residual.
*       B2US* - Solve a correction equation.
*       B2UX* - Auxiliaries related to b2u*.
*     B2V* - Not used.
*     B2W* - Write to an external file.
*       B2WF* - Write formatted.
*       B2WG* - Write graphical.
*       B2WS* - Write special purpose.
*       B2WU* - Write un*formatted.
*     B2X* - Basic auxiliary routines.
*       B2XG* - Auxiliaries for geometry.
*       B2XP* - Auxiliaries for plasma state.
*       B2XV* - Test validity of some quantities.
*       B2XX* - Miscellaneous auxiliaries.
*     B2Y* - Postprocessing.
*       B2YG* - Display the geometry and magnetic field.
*       B2YH* - Display the table of physics parameters.
*       B2YI* - Display the plasma state.
*       B2YM* - Display the evolution of the plasma state.
*       B2YN* - Display the progress of the inner iterations.
*       B2YP* - Comprehensive postprocessing of a calculation.
*       B2YQ* - Quick display of the evolution of a calculation.
*       B2YR* - Display the table of atomic rate coefficients.
*       B2YS* - Display the externally imposed source terms.
*     B2Z* - Reserved for application-dependent postprocessing.
*
*
*  1. Purpose of individual B2 routines.
*
*     B2AG (main program) prepares a geometry and magnetic field for
*     B2. This program calls the driver routine b2agdr for the actual
*     computation.
*
*     B2AGDR computes and writes to an external file the specification
*     of a B2 geometry and magnetic field.
*
*     B2AGBB computes the magnetic field from the flux functions.
*
*     B2AGBS computes a correction to the pbsx array to ensure the
*     constancy of magnetic flux along flux tubes.
*
*     B2AGCF computes the locations of the cell vertices, and computes
*     the flux functions at those locations.
*
*     B2AGFS obtains a description of the flux surface structure.
*
*     B2AGGS marks a singular region in the geometry.
*
*     B2AGMT computes the metric coefficients given the locations of
*     cell vertices.
*
*     B2AGMX projects the metric coefficients from a fine mesh to a
*     coarsening of that mesh.
*
*     B2AGSP evaluates the coordinates and the flux functions at one
*     single point on the mesh. This routine is application dependent,
*     and defines the specific geometry and magnetic field for a B2
*     calculation.
*
*     B2AGX0 reads the mesh dimensions from a formatted file.
*
*     B2AH (main program) prepares default physics parameters for B2.
*     This program calls the driver routine b2ahdr for the actual
*     computation.
*
*     B2AHDR prepares and writes to an external file a table of default
```

```
*     physics parameters for B2.
*
*     B2AHX0 reads the number of species from a formatted file.
*
*     B2AI (main program) prepares an initial plasma state for B2.
*     This program calls the driver routine b2aidr for the actual
*     computation.
*
*     B2AIDR prepares and writes to an external file an initial plasma
*     state for B2.
*
*     B2AIX0 reads the mesh dimensions and the number of species from a
*     formatted file.
*
*     B2AR (main program) prepares a table of atomic rate coefficients
*     for B2. This program calls the driver routine b2ardr for the
*     actual computation.
*
*     B2ARDR computes and writes to an external file a table of atomic
*     rate coefficients for B2. The coefficients are obtained using
*     either the ADPAK or the STRAHL set of routines.
*
*     B2ARIN reads from a formatted file the list of nuclear species
*     and atomic states for which the rate coefficients are to be
*     computed.
*
*     B2CDCA provides some general documentation of the B2 code.
*     No computation is performed in this routine.
*
*     B2CDCR contains a brief statement of the purpose of many of the
*     subroutines of the B2 code. No computation is performed in this
*     routine.
*
*     B2CDCV contains a description of some of the variables used in
*     the B2 code. No computation is performed in this routine.
*
*     B2MN is the main program for the B2 code. The code solves a
*     system of fluid equations that models a multi-species plasma
*     in a two-dimensional geometry. This program calls the driver
*     routines b2mndr_[0-2] for the actual computation.
*
*     B2MNDR is the main driver routine for the B2 code.
*
*     B2MNDS performs initialisation functions for a B2 calculation.
*
*     B2MNDT performs one implicit timestep for the B2 system of
*     equations.
*
*     B2MWMV writes to an un*formatted file a selection of data
*     describing the state of the plasma.
*
*     B2MWIT writes to an un*formatted file detailed monitoring output
*     for the inner iterations of the B2 solution process.
*
*     B2MWQ0 writes a header for the output from b2mwqt.
*
*     B2MWQT writes brief data concerning the numerical evolution.
*
*     B2MXAC computes the (maximum-)norms of the corrections.
*
*     B2MXAR computes scaled (L1-)norms of the residuals.
*
*     B2MXNP extrapolates a positive quantity to the next time.
*
*     B2MXNU extrapolates an indefinite quantity to the next time.
*
*     B2MXZR sets the residuals to zero along the boundaries of the
```

```
*     mesh and on the special regions.
*
*     B2NEWS performs one inner iteration for the B2 equations.
*     This routine defines the core structure of the solution
*     algorithm.
*
*     B2NPCO performs one iteration of a relaxation process for the
*     solution of the continuity equation for one atomic species.
*
*     B2NPHT performs one iteration of a relaxation process for the
*     solution of the equations for internal electron and atom energy.
*
*     B2NPMO performs one iteration of a relaxation process for the
*     solution of the B2 equations for parallel velocity.
*
*     B2NPPO performs one iteration of a relaxation process for the
*     solution of the charge conservation equation, which governs the
*     electric potential.
*
*     B2NXCM computes the adjustment to the coefficients pcca that
*     comes from the combined parallel momentum equation solver.
*
*     B2NXDP computes the correction to the electric potential following
*     an update to the electron density.
*
*     B2NXDU computes the artificial velocity shift that is
*     required to make the total numerical friction force cancel.
*
*     B2NXDV computes the correction to the parallel momentum source
*     that is required to make the total friction force cancel.
*
*     B2NXFC computes the transport terms in the parallel momentum
*     correction equation.
*
*     B2NXFM computes the flow terms in the parallel momentum equation
*     (not used anymore).
*
*     B2NXFV_ZH calculates the convective term due to new form of ion
*     viscosity term in the momentum correction equation if Zhdanov
*     closure is applied.
*
*     B2NXFX computes an expression that occurs in the electron-atom
*     thermal force.
*
*     B2NXST accumulates the parallel momentum source linearisation
*     over the distinct atomic species.
*
*     B2PRNTI prints out the residuals in integer format.
*
*     B2RFCN reads from a formatted file some parameters in the
*     numerics common block.
*
*     B2RFCP reads from a formatted file some parameters in the
*     physics common blocks.
*
*     B2RFLB reads from a formatted file a label for one run of one
*     of the programs that make up B2.
*
*     B2RUCP reads from an un*formatted file the parameters in the
*     physics common blocks.
*
*     B2RUGM reads from an un*formatted file the geometry and magnetic
*     field.
*
*     B2RUPS reads the state of the plasma from an un*formatted file.
*
*     B2RURC reads a table of atomic rate data from an un*formatted
```

```
*     file.
*
*     B2RUSR reads a table of source term data from an un*formatted
*     file.
*
*     B2SCDT modifies the source coefficients to account for the
*     implicit time discretisation for the isb-th density equation.
*
*     B2SHDT modifies the source coefficients to account for the
*     implicit time discretisation for Te and Ti equations.
*
*     B2SIAN computes contribution to the parallel momentum source due
*     to the anomalous current.
*
*     B2SIAV computes a linearised representation of the additional
*     viscosity.
*
*     B2SIAV_ZH computes a linearised representation of the additional
*     viscosity if the Zhdanov closure is applied.
*
*     B2SIFR computes a linearised representation of the friction
*     force between one species and all others.
*
*     B2SIFRTF computes a linearised representation of the thermal
*     and friction forces.
*
*     B2SIGP computes the pressure gradient term on the right hand
*     of the parallel momentum balance equation for one species.
*
*     B2SIHS computes the source coefficients for a set of terms in the
*     heat equations. These terms specify the contributions of the
*     divergence of the flow field, of viscous heating, and of the
*     electron-atom and atom-atom friction.
*
*     B2SMDT modifies the source coefficients to account for the
*     implicit time discretisation for parallel momentum equations.
*
*     B2SPCX computes on all cells of the mesh a log-log representation
*     of the rate coefficients for charge exchange.
*
*     B2SPEL computes on all cells of the mesh a log-log representation
*     of the rate coefficients for ionisation, recombination, and
*     electron heat loss.
*
*     B2SQCX computes on all cells of the mesh the rate coefficients
*     for charge exchange with one neutral hydrogen species.
*
*     B2SQEL computes on all cells of the mesh the rate coefficients
*     for ionisation, recombination, and electron heat loss.
*
*     B2SRAL provides linearised expressions for the sources of
*     density, parallel momentum, electron heat, atom heat, and charge.
*
*     B2SRDT modifies the source coefficients to account for the
*     implicit time discretisation.
*
*     B2SRSM smoothes expressions the sources strengths of
*     density, parallel momentum, electron heat, atom heat, and charge.
*
*     B2SRST modifies the source coefficients to provide numerical
*     stabilisation.
*
*     B2STBC provides linearised expressions for the sources of
*     density, parallel momentum, electron heat, atom heat, and charge,
*     due to the boundary conditions.
*
*     B2STBM provides linearised expressions for the sources of
```

```
*     density, parallel momentum, and heat, due to miscellaneous
*     application specific processes.
*
*     B2STBR provides linearised expressions for the sources of
*     density, parallel momentum, and heat, due to special boundary
*     conditions and the neutral gas recycling model.
*
*     B2STCX provides linearised expressions for the sources of atom
*     density and atom parallel momentum, due to charge exchange.
*
*     B2STEL provides linearised expressions for the sources of
*     density, parallel momentum, electron heat and atom heat, due
*     to the electron-atom collision processes of ionisation,
*     recombination, and radiation.
*
*     B2TANML computes the X and Y components of the anomalous current
*     for each species and calculates the X and Y components of
*     the total anomalous current.
*
*     B2TBFL computes the total flux through the boundaries.
*
*     B2TDIA computes the X and Y components of the divergent part of
*     the diamagnetic current.
*
*     B2TEPSCH computes the X and Y components of the Pfirsch-Schlueter
*     heat flux.
*
*     B2TFCH computes the electric current between neighboring cells,
*     and the coefficients that specify its linearisation.
*
*     B2TFED computes the electron drift velocities.
*
*     B2TFHE computes the electron heat flux between neighboring cells,
*     and the coefficients that specify its linearisation.
*
*     B2TFHI computes the all atom heat flux between neighboring cells,
*     and the coefficients that specify its linearisation.
*
*     B2TFNB computes the particle flux of one atomic species between
*     neighboring cells, and the coefficients that specify its
*     linearisation.
*
*     B2TFRN computes the particle and energy fluxes between
*     neighbouring cells which are to be provided to Eirene.
*
*     B2TFVH computes the ion velocity-dependent heat flux.
*
*     B2TINER computes the X and Y components of the inertial current
*     for each species and calculates the X and Y components of
*     the total inertial current.
*
*     B2TINNT computes the ion-neutral current.
*
*     B2TIPSCH computes the Pfirsch-Schlueter all atom heat flux between
*     neighboring cells, and the coefficients that specify its linearization.
*
*     B2TLHE applies a flux limit to the electron heat and charge
*     transport coefficients.
*
*     B2TLHI applies a flux limit to the all atom heat transport
*     coefficients.
*
*     B2TLH0 computes the conductive neutral heat transport flux limit.
*
*     B2TLMV applies a flux limit to the viscosity.
*
*     B2TNPSCH computes the Pfirsch-Schlueter particle fluxes of one atomic
```

```
*     species between neighbouring cells
*
*     B2TQCA computes the classical contributions to the atom
*     transport coefficients.
*
*     B2TQCE computes the classical contributions to the electron
*     transport coefficients.
*
*     B2TQNA computes the anomalous contributions to the transport
*     coefficients.
*
*     B2TRAL computes the transport coefficients, including geometric
*     factors such as vol/hx**2 and vol/hy**2.
*
*     B2TRCA computes the inversed matrix for heat flux/R-moments.
*
*     B2TRCL computes the classical transport coefficients, including
*     geometric factors such as vol/hx**2 and vol/hy**2.
*
*     B2TRCN computes the N-coefficients and all mass-related variables.
*
*     B2TRCV computes the inverted matrix for the viscous stress tensor.
*
*     B2TRCW calculates charge state corrections for the viscous
*     stress tensor.
*
*     B2TRCZ calculates charge state corrections to the thermal and
*     friction force terms.
*
*     B2TREQ computes the cell-integrated energy equipartition
*     multiplier.
*
*     B2TRFL computes the flux-limiting coefficients.
*
*     B2TRFL_AN computes the flux-limiting coefficients using
*     analytically calculated ion heat conductivity.
*
*     B2TRFL_VIS computes the flux-limiting coefficient for the stress
*     viscosity.
*
*     B2TRFR computes the friction force transport coefficient.
*
*     B2TRHRTF_AN computes the heat flux, the R-moment and the thermal
*     force transport coefficient analytically.
*
*     B2TRHT computes the ion heat conductivity coefficient.
*
*     B2TRHT_AN computes the ion heat conductivity coefficient using
*     the analytic method.
*
*     B2TRHW computes the heat flux and R-moment velocity-dependent part
*     of the transport coefficient.
*
*     B2TRNO computes the anomalous transport coefficients, including
*     geometric factors such as vol/hx**2 and vol/hy**2.
*
*     B2TRNU computes the number of isonuclear sequences (ions with
*     different nuclei) and the maximal atom number Z in the plasma
*     composition.
*
*     B2TRSC computes the X and Y components of electrical conductivity
*     associated with stochastization of the magnetic field.
*
*     B2TRSH computes the X and Y components of electron thermal conductivity
*     associated with stochastization of the magnetic field.
*
*     B2TRTF computes the thermal force transport coefficient.
```

```
*
*     B2TRVIS_AN computes the viscous stress tensor transport coefficient
*     analytically.
*
*     B2TRZH computes the transport coefficients using the 21N-moment
*     approach, as detailed in [V. M. Zhdanov. Transport Processes in
*     Multicomponent Plasma, vol. 44. 10 2002].
*
*     B2TSTCH computes the Y component of the electron current
*     associated with stochastization of the magnetic field.
*
*     B2TVSPA computes the X and Y components of the parallel viscosity
*     current for each species and calculates the X and Y components of
*     the total parallel viscosity current.
*
*     B2TVSPR computes the X and Y components of the perpendicular viscosity
*     current for each species and calculates the X and Y components of
*     the total perpendicular viscosity current.
*
*     B2TVSQ computes the X and Y components of the heat viscosity
*     current for each species and calculates the X and Y components of
*     the total heat viscosity current.
*
*     B2TXCV computes viscous conductances between neighbouring
*     staggered cells, given the viscosities (not used anymore).
*
*     B2TXCX computes conductances between neighbouring cells in the
*     x-direction, given the conductivities.
*
*     B2TXCY computes conductances between neighbouring cells in the
*     y-direction, given the conductivities.
*
*     B2TXFX computes conductances between neighbouring cells in the
*     x-direction, given the flux-limited conductivities.
*
*     B2TXFY computes conductances between neighbouring cells in the
*     y-direction, given the flux-limited conductivities.
*
*     B2TXSX computes the integrals over cell x-faces of a function
*     that is provided on the cells.
*
*     B2TXSY computes the integrals over cell y-faces of a function
*     that is provided on the cells.
*
*     B2UNAVSPR7A evaluates the coefficients of matrix of the contribution
*     from divergence of the viscosity perpendicular current by analytical
*     manipulation with respect to main ion density using a 7-point stencil.
*
*     B2UPCO corrects the state of the plasma towards the solution of
*     continuity equation for one atomic species.
*
*     B2UPHT adjusts the state of the plasma towards the solution
*     of the heat equations.
*
*     B2UPPO corrects the electric potential po.
*
*     B2URMO computes the residual of the parallel momentum balance
*     equation for one atomic species.
*
*     B2URSC computes the residual of a scalar conservation equation.
*
*     B2URSD computes the residual of a scalar conservation equation.
*
*     B2USCO computes a correction towards the solution of the
*     continuity equation for one atomic species.
*
*     B2USHT computes a correction to the fields te and ti.
```

```
*
*     B2USMO Computes a correction towards the solution of the
*     parallel momentum balance equation for one atomic species.
*
*     B2USPO computes a correction towards the solution of the
*     potential equation.
*
*     B2UX5P solves a five-point system of equations. This routine is
*     an interface to an iterative or a direct solver; the choice
*     depends upon an internal parameter.
*
*     B2WAOS prints out the results of a B2 calculation in old-fashioned
*     style
*
*     B2WDAT is an output routine that writes out B2 arrays into ASCII
*     data files.
*
*     B2WFCP writes to a formatted file the parameters in the physics
*     common blocks.
*
*     B2WFGM writes to a formatted file a summary of the B2 geometry
*     and magnetic field.
*
*     B2WFPI writes a summary of the plasma state to a formatted file.
*     This routine is an interface to b2wfpj.
*
*     B2WFPJ writes a summary of the plasma state to a formatted file.
*     This routine does not make use of any geometric information.
*
*     B2WFPK writes supplementary information about the plasma state to
*     a formatted file.
*
*     B2WFPS writes a summary of the plasma state to a formatted file.
*
*     B2WFR0 writes to a formatted file a text containing credits for
*     the ADPAK atomic physics rate coefficients.
*
*     B2WFR1 writes to a formatted file a text containing credits for
*     the STRAHL atomic physics rate coefficients.
*
*     B2WFRC writes to a formatted file a table of atomic rate
*     coefficients that has been prepared for B2.
*
*     B2WGCR provides graphical output of the residuals and the
*     corrections.
*
*     B2WGDT provides graphical output of the change in the state.
*
*     B2WGGM produces graphical displays of the B2 geometry and
*     magnetic field.
*
*     B2WGPI provides graphical output of the plasma state on a uniform
*     rectangular mesh. This routine is an interface to b2wgpj.
*
*     B2WGPJ provides graphical output of the plasma state on a uniform
*     cartesian mesh.
*
*     B2WGPS provides graphical output of the plasma state.
*     This routine is an interface to b2wgst.
*
*     B2WGST provides graphical output of the plasma state.
*
*     B2WGRC displays a table of atomic rate coefficients that has been
*     prepared for B2.
*
*     B2WGX0 displays the B2 mesh in a cartesian representation.
*
```

```
*     B2WGX1 displays the B2 mesh on the physical plane.
*
*     B2WGX2 produces contour plots on a sequence of cartesian domains.
*
*     B2WGX3 produces contour plots on the physical domain.
*
*     B2WGX4 produces log-contour plots on a sequence of cartesian
*     domains.
*
*     B2WGX5 produces log-contour plots on the physical domain.
*
*     B2WGXB plots an outline of the B2 mesh on the physical domain.
*
*     B2WGXP produces contour plots on a cartesian domain.
*
*     B2WGXQ produces log-contour plots on a cartesian domain.
*
*     B2WGYP performs the core NCAR operations to produce a contour
*     plot.
*
*     B2WGYQ performs the core NCAR operations to produce a log-contour
*     plot.
*
*     B2WSAV writes the state of the plasma to an unformatted file,
*     in the style required for input into the 'old B2' code.
*     The routine operates as an interface to b2wsax.
*
*     B2WSAX writes the state of the plasma to an unformatted file,
*     in the style required for input into the 'old B2' code.
*
*     B2WSCP provides special purpose printed output of data in the
*     physics common blocks.
*
*     B2WSPS provides special purpose printed output concerning the
*     plasma state. This routine is an interface to b2wsst.
*
*     B2WSST provides special purpose printed output concerning the
*     plasma state.
*
*     B2WUCP writes to an un*formatted file the parameters in the
*     physics common blocks.
*
*     B2WUGM writes to an un*formatted file the geometry and magnetic
*     field.
*
*     B2WUPS writes the state of the plasma to an un*formatted file.
*
*     B2WURC writes a table of atomic rate coefficients to an
*     un*formatted file.
*
*     B2XBZB calculates Bz/B**2 - Bz/<B**2>.
*
*     B2XGBS computes the product (magnetic field pitch)*(surface area)
*     at cell faces.
*
*     B2XGXY computes cartesian coordinates suitable for graphical and
*     printed output.
*
*     B2XPFE computes the electron particle flux, fne:
*       fne(,,) = (sum is :: rza(,,is)*fna(,,,is))-(1/qe)*fch(,,) .
*
*     B2XPFI computes the total atom particle flux, fni:
*       fni(,,) = (sum is :: fna(,,,is)) .
*
*     B2XPIC computes the poloidal projection of the ion parallel
*     current on cell centers.
*
```

```
*     B2XPIF computes the ion parallel current on cell faces.
*
*     B2XPMC computes the mach number of the parallel flow.
*
*     B2XPNE computes the electron density, ne:
*       ne(,) = (sum is :: rza(,,is)*na(,,is)) .
*
*     B2XPNI computes the total atom density, ni0:
*       ni(,,0) = (sum is :: na(,,is)) .
*     and the total ion density, ni1:
*       ni(,,1) = (sum is :: na(,,is) if is_neutral(is).eqv..false.) .
*
*     B2XPNM computes the atom density weighted by rz2/am, ne2m:
*       ne2m(,) = (sum is :: (rz2(,,is)/am(is))*na(,,is)) .
*
*     B2XPNR computes the atom density weighted by 1/sqrt(am), nirm:
*       nirm(,) = (sum is :: na(,,is)/sqrt(am(is))) .
*
*     B2XPPB computes the partial pressure, pb = nb*(rzb*te+ti).
*
*     B2XPPE computes the electron pressure, pe = ne*te.
*
*     B2XPPR computes the total pressure, pr:
*       pr(,) = ne(,)*te(,) + (sum is :: na(,,is)*ti(,)) .
*
*     B2XPPZ computes the plasma pressure, pz:
*       pz(,) = ne(,)*te(,) + (sum is :: na(,,is)*ti(,), not including neutrals).
*
*     B2XPRO computes the mass density, ro:
*       ro(,) = (sum is :: am(is)*mp*na(,,is)) .
*
*     B2XPRZ computes the ion mass density, rz:
*       rz(,) = (sum is :: am(is)*mp*na(,,is), including only charged species).
*
*     B2XPVE computes the parallel electron velocity on cell centers.
*
*     B2XPXC interpolates a function from x-faces to cell centers.
*
*     B2XVCP tests the validity of parameters in the physics common
*     blocks.
*
*     B2XVFF verifies the zero entries in the edge locations of some
*     quantity defined on faces of the B2 mesh. This routine is an
*     interface to b2xvfx and b2xvfy.
*
*     B2XVFV verifies the zero entries in the edge locations of some
*     quantity that is defined in the manner of a viscosity on the
*     cells and corners of the B2 mesh (not used anymore).
*
*     B2XVFX verifies the zero entries in the edge locations of some
*     quantity defined on x-faces of the B2 mesh.
*
*     B2XVFY verifies the zero entries in the edge locations of some
*     quantity defined on y-faces of the B2 mesh.
*
*     B2XVGM tests the validity of the geometric parameters.
*
*     B2XVPS tests the validity of the state of the plasma.
*
*     B2XVSG verifies the sign of some vector quantity.
*
*     B2XXGS sets a mesh function to a constant value on the singular
*     regions of the B2 geometry.
*
*     B2XXID returns an identification of the current version of B2.
*
*     B2XXMM computes the locations of the minimum and maximum entries
```

```
*     of a quantity defined on the B2 mesh.
*
*     B2XZ11 computes the z11 quantity needed for the Grad-Zhdanov
*     closure scheme:
*       z11(,) = (sum over j /= i ::
*                 (m(j)/m(i)*kappa(i,j))*sqrt(mu(i,j)/m(i))*
*                 Z_2_n_i(j)/Z_2_n_i(i)*(1+16/13*m(i)/m(j)+
*                                        30/13*(m(i)/m(j))^2))
*
*     B2XZ12 computes the z12 quantity needed for the Grad-Zhdanov
*     closure scheme:
*       z12(,) = (sum over j /= i ::
*                  mu(i,j)/m(i)*(m(j)/m(i)*kappa(i,j))*sqrt(mu(i,j)/m(i))*
*                  Z_2_n_i(j)/Z_2_n_i(i)*(1+32/23*m(i)/m(j)+
*                                         84/23*(m(i)/m(j))^2))
*
*     B2XZ22 computes the z22 quantity needed for the Grad-Zhdanov
*     closure scheme:
*       z22(,) = (sum over j /= i ::
*                 (m(j)/m(i)*kappa(i,j))^2*sqrt(mu(i,j)/m(i))*
*                  Z_2_n_i(j)/Z_2_n_i(i)*(1+1088/433*m(i)/m(j)+
*                  3672/433*(m(i)/m(j))^2+1792/433*(m(i)/m(j))^3+
*                  1400/433*(m(i)/m(j))^4))
*
*     B2XZEF computes zeff:
*       zeff(,) = (sum is :: rz2(,,is)*na(,,is))/ne .
*
*     B2XZF1 computes the zf1 quantity needed for the Grad-Zhdanov
*     closure scheme:
*       zf1(,) = (sum over j /= i :: mu(i,j)/m(i)*sqrt(mu(i,j)/m(i))*
*                                     Z_2_n_i(j)/Z_2_n_i(i))
*
*     B2XZF2 computes the zf2 quantity needed for the Grad-Zhdanov
*     closure scheme:
*       zf2(,) = (sum over j /= i :: (mu(i,j)/m(i))^2*sqrt(mu(i,j)/m(i))*
*                                     Z_2_n_i(j)/Z_2_n_i(i))
*
*     B2XZMC computes the zmc quantity needed for the Grad-Zhdanov
*     closure scheme:
*       zmc(,) = (sum over j /= i :: sqrt(mu(i,j)/m(i))*
*                                     Z_2_n_i(j)/Z_2_n_i(i))
*
*     B2XZMF computes zmff:
*       zmff(,) = (sum over is /= ismain :: rz2(,,is)*na(,,is))/
*                  na(,,ismain)*rz2(,,ismain)
*
*     B2YE (main program) is a postprocessor for B2. It calculates
*     statistical errors based on averaged quantities collected during
*     the averaging phase of a B2 calculation.
*
*     B2YG (main program) is a postprocessor for B2. It reads and
*     displays the geometry and magnetic field. This program calls the
*     driver routine b2ygdr for the actual computation.
*
*     B2YGDR is a postprocessor for B2. It reads and displays the
*     geometry and magnetic field.
*
*     B2YH (main program) is a postprocessor for B2. It reads and
*     displays the table of physics parameters. This program calls the
*     driver routine b2yhdr for the actual computation.
*
*     B2YHDR is a postprocessor for B2. It reads and displays the table
*     of physics parameters.
*
*     B2YI (main program) is a postprocessor for B2. It reads and
*     displays the plasma state. This program calls the driver routine
*     b2yidr for the actual computation.
```

```
*
*     B2YIDR is a postprocessor for B2. It reads and displays the
*     plasma state. This routine does not make use of any metric
*     information.
*
*     B2YM (main program) is a postprocessor for B2. It is meant to
*     produce movie output. This program calls the driver routine b2ymdr
*     for the actual computation.
*
*     B2YMDR is a postprocessor for B2. It is meant to produce movie
*     output.
*
*     B2YN (main program) is a postprocessor for B2. It displays the
*     progress of the inner iterations. This program calls the driver
*     routine b2yndr for the actual computation.
*
*     B2YNDR is a postprocessor for B2. It displays the progress of
*     the inner iterations.
*
*     B2YP (main program) is a postprocessor for B2. It displays the
*     plasma state. This program calls the driver routine b2ypdr for
*     the actual computation.
*
*     B2YPDR is a postprocessor for B2. It displays the plasma state,
*     making use also of the geometric information.
*
*     B2YQ (main program) is a postprocessor for B2. It provides a
*     quick display of the evolution data. This program calls the driver
*     routine b2yqdr for the actual computation.
*
*     B2YQDR is a postprocessor for B2. It provides a quick display of
*     the evolution data.
*
*     B2YR (main program) is a postprocessor for B2. It reads and
*     displays a table of atomic rate coefficients. This program calls
*     the driver routine b2yrdr for the actual computation.
*
*     B2YRDR is a postprocessor for B2. It reads and displays a table
*     of atomic rate coefficients.
*
*
*  2. Purpose of some additional routines employed with B2.
*
*   ..2a program management and error handling
*
*     PRGINI performs any tasks required at the start of program
*     execution.
*
*     PRGEND performs any tasks required at the end of program
*     execution.
*
*     SUBSYS is a group of routines (entry points in SUBSYS) that
*     perform certain functions related to workspace management and
*     error handling. The entry points are subini, subend, and subxnd.
*     SUBINI performs any tasks required at the start of subprogram
*     execution.
*     SUBEND performs any tasks required at the end of subprogram
*     execution.
*     SUBXND is employed to build down the stack when the program
*     is terminated because of an error condition.
*
*     IPMAIN is a group of routines that
*     operate on a collection of names and values representing
*     "internal parameters" in a code. The entry points are ipsetc,
*     ipgetc, and ipfini.
*     IPSETC saves the value of an internal parameter.
*     IPGETC returns the value of an internal parameter.
```

```
*     IPFINI writes an error message for any parameter that has been
*     set but not read.
*
*     IPGETI is an interface to ipgetc; it retrieves an integer value
*     ival from the table of internal parameters, or returns ival
*     unchanged if the name is not found or if the corresponding entry
*     is a blank string. The internal parameter may have been set by an
*     earlier call to ipseti or to ipsetc.
*
*     IPGETR is an interface to ipgetc; it retrieves a real*8 value
*     rval from the table of internal parameters, or returns rval
*     unchanged if the name is not found or if the corresponding entry
*     is a blank string. The internal parameter may have been set by an
*     earlier call to ipsetr or to ipsetc.
*
*     IPSETI is an interface to ipsetc; it saves an integer value in
*     the table of internal parameters.
*
*     IPSETR is an interface to ipsetc; it saves a real*8 value in the
*     table of internal parameters.
*
*     LWIMAI is a group of routines (entry points in lwimai) that
*     maintain a base address for free space in the workspace area
*     b2mod_stack%wi. The entry points are lwiset, lwiget, and lwimax.
*     LWISET frees workspace by resetting the base address.
*     LWIGET reserves workspace by incrementing the base address.
*     LWIMAX returns the maximum workspace use.
*
*     LWMAIN is a group of routines (entry points in lwmain) that
*     maintain a base address for free space in the workspace area
*     b2mod_stack%w. The entry points are lwset, lwget, and lwmax.
*     LWSET frees workspace by resetting the base address.
*     LWGET reserves workspace by incrementing the base address.
*     LWMAX returns the maximum workspace use.
*
*     XERGET gets the unit number for error messages.
*
*     XERSET sets the unit number for error messages.
*
*     XERRAB writes an error message and the call chain, and terminates
*     execution of the program.
*
*     XERTST tests a condition; if violated, calls xerrab to terminate
*     execution.
*
*    ..2b File management and un*formatted input/output
*
*     CFENDF tests whether an un*formatted file is positioned at
*     end-of-file.
*
*     CFOPEN opens an external file. The file may be formatted or
*     unformatted.
*!!   This routine may contain system dependencies.
*
*     CFRUCH reads a character array from an un*formatted file.
*
*     CFRUIN reads an integer array from an un*formatted file.
*
*     CFRURE reads a real*8 array from an un*formatted file.
*
*     CFWUCH writes a character array to an un*formatted file.
*
*     CFWUIN writes an integer array to an un*formatted file.
*
*     CFWURE writes a real*8 array to an un*formatted file.
*
*    ..2c Low-level auxiliary routines
```

```
*
*     BLAS and BLAS-like routines (basic linear algebra subprograms):
*       IFILL (BLAS-like) fills an integer vector with a constant value.
*       SMIN (BLAS-like) computes the minimum value in a vector.
*       SMAX (BLAS-like) computes the maximum value in a vector.
*       SAMAX (BLAS-like) computes the maximum absolute value in
*       a vector.
*       B2SSUM (BLAS-like) sums the elements of a vector.
*       B2SASUM (BLAS) sums the absolute values of elements of a vector.
*       SFILL (BLAS-like) fills a vector with a constant value.
*       SBIAS (BLAS-like) adds a constant value to a vector.
*       B2SSCAL (BLAS) scales a vector by a constant.
*       B2SCOPY (BLAS) copies a vector to another.
*       B2SAXPY (BLAS) adds a constant-times-vector to a vector.
*
*     ESSEQ determines whether two real*8 numbers differ by no more than
*     a small multiple of their machine precision.
*
*     EVAL returns its real*8 argument through the function name.  It may
*     be used to force a compiler to store an intermediate operand.
*
*     RATIO evaluates very carefully the real*8 ratio of two integers.
*
*     SCAL10 is meant to compute a suitable scale for graphical output.
*
*     CHCASE converts a character variable to either all uppercase or
*     all lowercase.
*
*     STRCOPY copies the first l0 characters of string str1 into str0.
*
*     STREQL checks for equality between two strings, ignoring case
*     shift and trailing blanks.
*
*     STRMAS checks for a certain matching between two strings.
*
*   ..2d Auxiliary routines for discretised PDEs
*
*     ILU5F computes the ILU factorisation of a 2d 5-point matrix.
*
*     ILU5G computes the ILU factorisation of a 2d 5-point matrix and
*     applies the factors for one iteration of an update equation.
*
*     ILU5S solves an equation of the form L*U*x=r, where L and U are
*     the factors in an ILU factorisation of a 2d 5-point matrix.
*
*     ILUTER performs a number of iterations of the Stone strongly
*     implicit procedure for a system of equations of five-point form.
*
*     SIP5F computes the SIP-ILU factorisation of a 2d 5-point matrix.
*
*     SIP5G computes the SIP-ILU factorisation of a 2d 5-point matrix,
*     and applies the factors for one iteration of an update equation.
*
*     SLV5PT solves a five-point system of equations by a direct
*     method.
*
*     STABEQ adjusts the diagonal of a five-point matrix to satisfy
*     a property related to diagonal dominance.
*
*   ..2e System-dependent routines
*
*     DAYTIM returns the date and time in a character*17 format.
*!!   This routine may contain system dependencies.
*
*     JOBNAM returns an identification of the present job.
*!!   This routine may contain system dependencies.
*
```

```
*     MACHEPS returns the machine precision.
*!!   This routine may contain system dependencies.
*
*     MACHSFR returns the safe numerical range (a tiny positive machine
*     value e0 for which -e0, 1/e0, -1/e0 are all computable).
*!!   This routine may contain system dependencies.
*
*     PRVRT sets all elements of a real*8 array to a value that is
*     expected to cause an error when used in an arithmetic operation.
*     This is used to protect against certain programming errors.
*!!   This routine may contain system dependencies.
*
*     PRVRTI sets all elements of an integer array to a value that is
*     expected to cause an error when used in an arithmetic operation.
*     This is used to protect against certain programming errors.
*!!   This routine may contain system dependencies.
*
*     SYSINI performs any system dependent calls that may be required
*     at the start of program execution.
*!!   This routine may contain system dependencies.
*
*     SYSEND performs any system dependent calls that may be required
*     at the finish of program execution.
*!!   This routine may contain system dependencies.
*
*     USRNAM returns the name or other appropriate identification of
*     the submitter of the present job.
*!!   This routine may contain system dependencies.
*
*    ..2f Basic graphics routines
*
*     GNOPEN opens a graphics file for use with GKS.
*
*     GWSREP defines the workstation representation for GKS. It should
*     normally be called just once for each workstation, immediately
*     after opening the workstation.
*
*     PAGDEF sets GKS and NCAR properties to default values. It should
*     normally be called at the start of each new plot.
*
*     PAGINI starts a new plot.
*
*     PAGEND terminates a plot.
*
*     CPMPXY is a substitute for a routine of the same name in the
*     CONPACK sub-package of NCARG graphics. It defines a mapping from
*     rectangular array index coordinates in (1:*,1:*) to the plotting
*     window.
*
*     GNMAP2 (together with the alternate entry GNDFM2) is a service
*     routine for use with the CONPACK routines in the NCARG graphics
*     package. It defines a rectangular map for CONPACK. The map is
*     specified by a call to the entry gndfm2.
*
*     GNMAP3 (together with the alternate entry GNDFM3) is a service
*     routine for use with the CONPACK routines in the NCARG graphics
*     package. It defines a curvilinear, logically rectangular map for
*     CONPACK. The map is specified by a call to the entry gndfm3.
*
*     GNCOLR is for use by the routine ARSCAM in the AREAS utility of
*     NCAR graphics. It defines the filling of a polygon.
*
*     LBFILL is a substitute for a routine of the same name in the
*     LABELBAR utility of NCAR graphics. It fills a specified area.
*     LBFILX and LBFILQ are alternate entries in LBFILL. LBFILX is
*     called to define the filling style and LBFILQ is called to query
*     the filling style.
```

```
*
*   ..2g Atomic physics and surface physics
*
*     RATADP uses the ADPAK code to prepare tables of rate coefficients
*     for ionisation, recombination, electron energy loss, and charge
*     exchange; as function of temperature and density.
*
*     RATSTR uses the STRAHL code to prepare tables of rate coefficients
*     for ionisation, recombination, electron energy loss, and charge
*     exchange; as function of temperature and density.
*
*     RATPRT prints an excerpt from a table of atomic rate
*     coefficients.
*
*     RATTAI computes a smoothed set of rate coefficients to represent
*     the effect of a power-law tail on the distribution function.
*
*     RATWEI uses WEISHEIT data to improve tables of rate coefficients
*     for ionisation, recombination, electron energy loss, and charge
*     exchange; as function of temperature and density.
*     The WEISHEIT dataset is incomplete.  The call to ratwei must
*     be preceded by a call to one of the comprehensive routines.
*
*-----------------------------------------------------------------------
*.end b2cdcr

      end subroutine b2cdcr
```

A.3 b2cdcv.F

```
*----------------------------------------------------------------------
*.specification

      subroutine b2cdcv ()
      implicit none

*----------------------------------------------------------------------
*.documentation
*
*  1. purpose
*
*     B2CDCV contains a description of some of the variables used in
*     the B2 code. No computation is performed in this routine.
*
*----------------------------------------------------------------------
*.text
*
*  0. Introduction
*
*     ...
*
*
*  1. Principal code variables
*
*   ..variables describing the geometry
*
*     nx, ny - integer.
*     nx and ny specify the number of interior cells along the first
*     and the second coordinate, respectively. The total number of
*     cells is (nx+2)*(ny+2); they are indexed by (-1:nx,-1:ny).
*     It will hold that 0.le.nx and 0.le.ny.
*
*     crx, cry - (-1:nx,-1:ny,0:3) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), crx(ix,iy,0:3) specify the
*     horizontal coordinates and cry(ix,iy,0:3) specify the vertical
*     coordinates (in the physical plane) of the four corners of the
*     (ix,iy) cell, mapped as follows:
*     (,,0): lower left corner,  (,,1): lower right corner,
*     (,,2): upper left corner,  (,,3): upper right corner.
*     One expects the following identities to hold away from the
*     boundaries:
*     crx(ix,iy,0)=crx(ix-1,iy,1)=crx(ix,iy-1,2)=crx(ix-1,iy-1,3),
*     cry(ix,iy,0)=cry(ix-1,iy,1)=cry(ix,iy-1,2)=cry(ix-1,iy-1,3),
*     However, there may be cuts in the geometry, in which case these
*     identities need not hold.
*
*     fpsi - (-1:nx,-1:ny,0:3) real*8 array.
*     For (ix,iy,i) in (-1:nx,-1:ny,0:3), fpsi(ix,iy,i) specifies the
*     potential for the magnetic field components (bx,by) at the
*     (ix,iy,i) position.
*     (In the case of straight symmetry, bx=dpsi/dy and by=-dpsi/dx.
*     In the case of toroidal or cylindrical symmetry,
*       bx=(dpsi/dy)/(2*pi*R) and by=-(dpsi/dx)/(2*pi*R),
*     where R is the distance to the axis of symmetry.)
*
*     ffbz - (-1:nx,-1:ny,0:3) real*8 array.
*     For (ix,iy,i) in (-1:nx,-1:ny,0:3), ffbz(ix,iy,i) specifies the
*     field function for the magnetic field component bz at the
*     (ix,iy,i) position.
*     (In the case of straight symmetry, bz=ffbz. In the case of
*     toroidal or cylindrical symmetry, bz=ffbz/(2*pi*R), where R is
*     the distance to the axis of symmetry.)
*
*     bb - (-1:nx,-1:ny,0:3) real*8 array.
```

```
*     For (ix,iy) in (-1:nx,-1:ny), bb(ix,iy,0:3) specifies the
*     magnetic field at the center of the (ix,iy) cell.
*     bb(ix,iy,0:2) are the (x,y,z)-components and bb(ix,iy,3) is
*     the absolute magnetic field strength.
*     (bb(,,3)=sqrt(bb(,,0)**2+bb(,,1)**2+bb(,,2)**2.)
*     It will hold that 0.lt.bb(,,3).
*
*     wbbl - (-1:nx,-1:ny,0:3) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), wbbl(ix,iy,0:3) specifies the
*     magnetic field at the midpoint of the left face of the (ix,iy) cell.
*     Meaning of components is similar to that of the bb array.
*
*     wbbr - (-1:nx,-1:ny,0:3) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), wbbr(ix,iy,0:3) specifies the
*     magnetic field at the midpoint of the right face of the (ix,iy) cell.
*     Meaning of components is similar to that of the bb array.
*
*     wbbv - (-1:nx,-1:ny,0:3) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), wbbv(ix,iy,0:3) specifies the
*     magnetic field at the midpoint of the bottom face of the (ix,iy) cell.
*     Meaning of components is similar to that of the bb array.
*
*     wbbc - (-1:nx+1,-1:ny+1,0:3) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), wbbv(ix,iy,0:3) specifies the
*     magnetic field at the left bottom corner of the (ix,iy) cell.
*     For ix=nx+1 wbbc is related to the right bottom corner of
*     the cell with ix=nx, similarly for iy=ny+1 wbbc is related to
*     the left top corner of the cell iy=ny. wbbc(nx+1,ny+1,:) is related
*     to the right top corner of the cell (ix=nx,iy=ny)
*     Meaning of components are similat to the bb array.
*
*     vol - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), vol(ix,iy) specifies the volume of
*     the (ix,iy) cell. (In the case of toroidal symmetry this volume
*     includes a factor 2*pi*crx, while in the case of cylindrical
*     symmetry it includes a factor 2*pi*cry.)
*     It will hold that 0.lt.vol(,).
*
*     hx, hy, hz - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), hx(ix,iy), hy(ix,iy), and hz(ix,iy)
*     specify the x-diameter, y-diameter, and z-diameter, respectively,
*     of the (ix,iy) cell.
*     If redef_gmtry=1, hz(:,:)*bb(:,:,2)=sum(ffbz(:,:,0:3))/4 is enforced.
*     It will hold that 0.lt.hx(,) and 0.lt.hy(,) and 0.lt.hz(,).
*
*     qz - (-1:nx,-1:ny,0:1) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), qz(ix,iy,0) specifies sin(t) and
*     qz(ix,iy,1) specifies cos(t), where t is the complementary angle
*     between the x-direction and the y-direction on the (ix,iy) cell.
*     t is in the range (-pi/2 to pi/2) and vanishes for an orthogonal
*     geometry. (qz(ix,iy,0) is the inner product of the x and y unit
*     vectors.)
*     It will hold that -1.lt.qz(,,0).lt.1 and 0.lt.qz(,,1).le.1.
*
*     qc - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), qc(ix,iy) specifies cos(t), where
*     t is the complementary angle between the x-direction of the
*     (ix,iy) cell and its left face.
*     t is in the range (-pi/2 to pi/2) and vanishes for an orthogonal
*     geometry.
*     It will hold that 0.lt.qc(,).le.1.
*
*     gs - (-1:nx,-1:ny,0:2) real*8 array.
*     (Let gsx(,)=gs(,,0), gsy(,)=gs(,,1) and gsz(,)=gs(,,2).)
*     For (ix,iy) in (0:nx,-1:ny), gsx(ix,iy) specifies the area of the
*     face between the (ix,iy) cell and its left neighbor. For ix.eq.-1,
*     gsx(ix,-1:ny) will hold 0.
```

```
*     For (ix,iy) in (-1:nx,0:ny), gsy(ix,iy) specifies the area of the
*     face between the (ix,iy) cell and its bottom neighbor. For iy.eq.-1,
*     gsy(-1:nx,iy) will hold 0.
*     For (ix,iy) in (-1:nx,-1:ny), gsz(ix,iy) specifies the area of the
*     (ix,iy) cell normal to the third (ignorable) coordinate.
*
*     hy1 - (-1:nx,-1:ny) real*8 array.
*     Actual radial distance between neighbouring flux surfaces.
*     hy1(:,:) = hy(:,:) * qc(:,:) and
*     vol(:,:) = hx(:,:) * hy1(:,:) * hz(:,:)
*     It will hold that 0.lt.hy1(,).
*
*     hxb, hyl - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), hxb(ix,iy) and hyl(ix,iy) specify the
*     x-diameter and the y-diameter, respectively, of the bottom and
*     left faces of the (ix,iy) cell.
*     It will hold that 0.lt.hxb(,) and 0.lt.hyl(,).
*
*     hzb, hzl - (-1:nx,-1:ny) real*8 array.
*     hzc - (-1:nx+1,-1:ny+1) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), hzb(ix,iy) and hzl(ix,iy) specify the
*     length along the toroidal direction, respectively, of the bottom and
*     left faces of the (ix,iy) cell. It follows that:
*     hzl(:,:) * hyl(:,:) = gs(:,:,0)
*     hzb(:,:) * hxb(:,:) = gs(:,:,1)
*     For (ix,iy) in (-1:nx,-1:ny), hzc(ix,iy) specifies the length along
*     the toroidal direction of the bottom left vertex of cell (ix,iy).
*     For (ix,iy) in (nx+1,-1:ny), hzc(ix,iy) specifies the length along
*     the toroidal direction of the top left vertex of cell (nx,iy).
*     For (ix,iy) in (-1:nx,ny+1), hzc(ix,iy) specifies the length along
*     the toroidal direction of the bottom right vertex of cell (ix,ny).
*     For (ix,iy) in (nx+1,ny+1), hzc(ix,iy) specifies the length along
*     the toroidal direction of the top right vertex of cell (nx,ny).
*     If redef_gmtry=1, the following identities are enforced:
*     hzl(:,:) * wbbl(:,:,2) = (ffbz(:,:,0)+ffbz(:,:,2))/2,
*     hzb(:,:) * wbbv(:,:,2) = (ffbz(:,:,0)+ffbz(:,:,1))/2,
*     hzc(:,:) * wbbc(:,:,2) = ffbz(:,:,:),
*     and conservation of magnetic flux for constant iy (for all ix values):
*     hyl(:,iy) * qc(:,iy) * hzl(:,iy) * wbbl(:,iy,0) = hy(:,iy) * qz(:,iy,1) * hz(:,iy) * bb(:,iy,0)
*     It will hold that 0.lt.hzb(,) and 0.lt.hzl(,) and 0.lt.hzc(,).
*
*     isymm - integer.
*     isymm specifies the type of symmetry of the geometry: isymm.eq.0
*     implies a slab geometry, isymm.eq.1.or.isymm.eq.2 imply rotational
*     symmetry about the crx=0 axis, and isymm.eq.3.or.isymm.eq.4
*     indicate rotational symmetry about the cry=0 axis.
*     Other values are not allowed.
*     isymm.eq.1.or.isymm.eq.3 mean a toroidal geometry case, i.e.
*     with a toroidal (out-of-plane) magnetic field component.
*     isymm.eq.2.or.isymm.eq.4 mean a cylindrical geometry case, i.e.
*     no toroidal magnetic field component.
*
*     periodic_bc - integer.
*     periodic_bc indicates whether the geometry closes upon itself (i.e.
*     whether periodic boundary conditions are necessary). The default value
*     is 0. periodic_bc.eq.1 implies that periodic BC are being used. This
*     case is used for the simulation of diverted stellarator islands or
*     limiters.
*
*     /b2cmfs/ - common block.
*     /b2cmfs/ is reserved for numerical mhd equilibrium data.
*      isymm - integer.
*     See above for description.
*
*     /b2cmgs/ - common block.
*     /b2cmgs/ specifies the singular regions in the geometry.
*      nlrmx - integer constant.
```

```
*     nlrmx is an upper bound for nlreg, and is used in dimension
*     specifications.
*      nlreg - integer.
*     nlreg specifies the number of singular regions. It will hold that
*     0.le.nlreg.le.nlrmx.
*      nlxlo, nlxhi, nlylo, nlyhi - (0:nlrmx-1) integer array.
*     For 0.le.i.lt.nlreg, nlxlo(i), nlxhi(i), nlylo(i) and nlyhi(i) are
*     the bounds of the i-th singular region. Mesh locations (ix,iy)
*     in (nlxlo(i):nlxhi(i),nlylo(i):nlyhi(i)) make up the region, which
*     will be included in (-1:nx,-1:ny).
*     For nlreg.le.i.lt.nlrmx, nlxlo(i), nlxhi(i), nlylo(i) and nlyhi(i)
*     are unspecified.
*      nlloc - (0:nlrmx-1) integer array.
*     For 0.le.i.lt.4, nlloc(i) specifies the class of quantities for
*     which the i-th region is special, as follows:
*       nlloc(i) = 0: cell centers, indices (-1:nx,-1:ny).
*       nlloc(i) = 1: cell x-faces, indices (0:nx,-1:ny).
*       nlloc(i) = 2: cell y-faces, indices (-1:nx,0:ny).
*       nlloc(i) = 3: cell corners, indices (0:nx,0:ny).
*     For nlreg.le.i.lt.nlrmx, nlloc(i) is unspecified.
*
*   ..auxiliary variables describing the geometry
*
*     pbs - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let pbsx(,)=pbs(,,0) and pbsy(,)=pbs(,,1).)
*     For (ix,iy) in (-1:nx,-1:ny), pbsx(ix,iy) specifies the product
*     (bx/bb)*sx and pbsy(ix,iy) specifies (by/bb)*sy, or (magnetic
*     field pitch)*(surface area), namely the parallel contact area,
*     on the (ix,iy) cell left and bottom faces respectively.
*     In the case where the grid covers the full vacuum vessel,
*     pbsy also contains a contribution from the narrowing of the
*     flux tube when impacting the wall structures.
*     See routine b2xgbs for the form of interpolation employed.
*
*     pbshz - (-1:nx,-1:ny,0:1) real*8 array.                       !srv 09.06.08
*     (Let pbshzx(,)=pbshz(,,0) and pbshzy(,)=pbshz(,,1).)
*     For (ix,iy) in (-1:nx,-1:ny), pbshzx(ix,iy) specifies the product
*     hz*(bx/bb)*sx and pbshzy(ix,iy) specifies hz*(by/bb)*sy, or (magnetic
*     field pitch)*(surface area)*hz
*
*     bzb - (-1:nx,-1:ny) real*8 array contains Bz/B**2 - Bz/<B**2> !srv 09.06.08
*     on the cell centre.
*
*     OnedBsq - (-1:nx,-1:ny) real*8 array.                         !srv
*     For (ix,iy) in (-1:nx,-1:ny), OnedBsq(ix,iy) specifies the
*     1/bb(,,3)**2 on the cell centre
*
*   ..basic physical parameters
*
*     /b2cmpa/ - common block
*     /b2cmpa/ specifies basic physics parameters.
*      me, mp, ev, qe, eps0, mu0 - real*8 constant.
*     These parameters are included with /b2cmpa/.
*       me = electron mass (kg)
*       mp = proton mass (kg)
*       ev = 1 electron-Volt (Joule)
*       qe = elementary charge (Coulomb)
*       eps0 = vacuum dielectric permittivity (SI)
*       mu0 = vacuum magnetic permeability (SI)
*      nsdecl - integer constant.
*     nsdecl is an upper bound for the number of species (ns), and is
*     used in dimension specifications.
*      zamin, zamax, zn, am - (0:nsdecl-1) real*8 array.
*     For (is) in (0:ns-1), zn(is) specifies the nuclear charge of
*     species (is), am(is) specifies its atomic mass, and zamin(is) and
*     zamax(is) specify its atomic charge range. All are dimensionless.
*     It will hold that 0.le.zamin(is).le.zamax(is), 0.lt.zn(is) and
```

```
*     0.lt.am(is). It is required that neutral species not be bundled
*     with charged species.
*     For ns.le.is.lt.nsdecl, zamin(is), zamax(is), zn(is), am(is)
*     are undefined.
*
*    ..variables describing the plasma state
*
*     ns - integer.
*     ns specifies the number of atomic species in the calculation.
*     The species are indexed by (0:ns-1).
*     It will hold that 1.le.ns.
*
*     na - (-1:nx,-1:ny,0:ns-1) real*8 array.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), na(ix,iy,is) specifies
*     the density of atomic species (is) on the (ix,iy) cell.
*     It will hold that 0.lt.na(,,).
*
*     ua - (-1:nx,-1:ny,0:ns-1) real*8 array.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), ua(ix,iy,is) specifies
*     the parallel velocity of atomic species (is) on the (ix,iy) cell.
*     The sign of the parallel velocity indicates whether it is flowing
*     in the same direction as the local poloidal magnetic field.
*
*     te - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), te(ix,iy) specifies the electron
*     temperature on the (ix,iy) cell. It will hold that 0.lt.te(,).
*
*     ti - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), ti(ix,iy) specifies the all atom
*     temperature on the (ix,iy) cell. It will hold that 0.lt.ti(,).
*
*     po - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), po(ix,iy) specifies the electric
*     potential on the (ix,iy) cell.
*
*     For all the fluxes below, the sign convention is as follows.
*     Fluxes between poloidal neighbors are positive if directed in the
*     direction of increasing poloidal (ix) index.
*     Fluxes between radial neighbors are positive if directed in the
*     direction of increasing radial (iy) index.
*
*     fna - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     (Let fnax(,,)=fna(,,0,) and fnay(,,)=fna(,,1,).)
*     For (ix,iy,is) in (0:nx,-1:ny,0:ns-1), fnax(ix,iy,is) specifies
*     the flux of atoms of species (is) through the face between the
*     (ix,iy) cell and its left neighbor.
*     For ix.eq.-1, fnax(ix,-1:ny,0:ns-1) will hold 0.
*     For (ix,iy,is) in (-1:nx,0:ny,0:ns-1), fnay(ix,iy,is) specifies
*     the flux of atoms of species (is) through the face between the
*     (ix,iy) cell and its bottom neighbor.
*     For iy.eq.-1, fnay(-1:nx,iy,0:ns-1) will hold 0.
*
*     fna_mdf - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     (Let fnax_mdf(,,)=fna_mdf(,,0,) and fnay_mdf(,,)=fna_mdf(,,1,).)
*     For (ix,iy,is) in (0:nx,-1:ny,0:ns-1), fnax_mdf(ix,iy,is) specifies
*     the modified flux of atoms of species (is) through the face between
*     the (ix,iy) cell and its left neighbor.
*     For ix.eq.-1, fnax_mdf(ix,-1:ny,0:ns-1) will hold 0.
*     For (ix,iy,is) in (-1:nx,0:ny,0:ns-1), fnay_mdf(ix,iy,is) specifies
*     the modified flux of atoms of species (is) through the face between
*     the (ix,iy) cell and its bottom neighbor.
*     For iy.eq.-1, fnay_mdf(-1:nx,iy,0:ns-1) will hold 0.
*     fna_mdf is equal to fna in runs without drifts.
*
*     fna_fcor - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     (Let fna_fcorx(,,)=fna_fcor(,,0,) and fna_fcory(,,)=fna_fcor(,,1,).)
*     For (ix,iy,is) in (0:nx,-1:ny,0:ns-1), fna_fcorx(ix,iy,is) specifies
```

```
*     the flux of atoms of species (is) for calculating momentum transport
*     through the face between the (ix,iy) cell and its left neighbor.
*     For ix.eq.-1, fna_fcorx(ix,-1:ny,0:ns-1) will hold 0.
*     For (ix,iy,is) in (-1:nx,0:ny,0:ns-1), fna_fcory(ix,iy,is) specifies
*     the flux of atoms of species (is) for calculating momentum transport
*     through the face between the (ix,iy) cell and its bottom neighbor.
*     For iy.eq.-1, fna_fcory(-1:nx,iy,0:ns-1) will hold 0.
*     fna_fcor takes into account the hz toroidal length derivative and
*     the Coriolis force.
*
*     fhe - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let fhex(,)=fhe(,,0) and fhey(,)=fhe(,,1).)
*     For (ix,iy) in (0:nx,-1:ny), fhex(ix,iy) specifies the electron
*     heat flux through the face between the (ix,iy) cell and
*     its left neighbor. For ix.eq.-1, fhex(ix,-1:ny) will hold 0.
*     For (ix,iy) in (-1:nx,0:ny), fhey(ix,iy) specifies the electron
*     heat flux through the face between the (ix,iy) cell and
*     its bottom neighbor. For iy.eq.-1, fhey(-1:nx,iy) will hold 0.
*
*     fhe_mdf - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let fhex_mdf(,)=fhe_mdf(,,0) and fhey_mdf(,)=fhe_mdf(,,1).)
*     For (ix,iy) in (0:nx,-1:ny), fhex_mdf(ix,iy) specifies the modified
*     electron heat flux through the face between the (ix,iy) cell
*     and its left neighbor. For ix.eq.-1, fhex_mdf(ix,-1:ny) will hold 0.
*     For (ix,iy) in (-1:nx,0:ny), fhey_mdf(ix,iy) specifies the modified
*     electron heat flux through the face between the (ix,iy) cell and
*     its bottom neighbor. For iy.eq.-1, fhey_mdf(-1:nx,iy) will hold 0.
*     fhe_mdf is equal to fhe in runs without drifts.
*
*     fhi - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let fhix(,)=fhi(,,0) and fhiy(,)=fhi(,,1).)
*     For (ix,iy) in (0:nx,-1:ny), fhix(ix,iy) specifies the all atom
*     heat flux through the face between the (ix,iy) cell
*     and its left neighbor. For ix.eq.-1, fhix(ix,-1:ny) will hold 0.
*     For (ix,iy) in (-1:nx,0:ny), fhiy(ix,iy) specifies the all atom
*     heat flux through the face between the (ix,iy) cell and its
*     bottom neighbor. For iy.eq.-1, fhiy(-1:nx,iy) will hold 0.
*
*     fhi_mdf - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let fhix_mdf(,)=fhi_mdf(,,0) and fhiy_mdf(,)=fhi_mdf(,,1).)
*     For (ix,iy) in (0:nx,-1:ny), fhix_mdf(ix,iy) specifies the
*     modified all atom heat flux through the face between the (ix,iy) cell
*     and its left neighbor. For ix.eq.-1, fhix_mdf(ix,-1:ny) will hold 0.
*     For (ix,iy) in (-1:nx,0:ny), fhiy_mdf(ix,iy) specifies the modified
*     all atom heat flux through the face between the (ix,iy) cell and
*     its bottom neighbor. For iy.eq.-1, fhiy_mdf(-1:nx,iy) will hold 0.
*
*     fch - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let fchx(,)=fch(,,0) and fchy(,)=fch(,,1).)
*     For (ix,iy) in (0:nx,-1:ny), fchx(ix,iy) specifies the electric
*     current through the face between the (ix,iy) cell and its left
*     neighbor. For ix.eq.-1, fchx(ix,-1:ny) will hold 0.
*     For (ix,iy) in (-1:nx,0:ny), fchy(ix,iy) specifies the electric
*     current through the face between the (ix,iy) cell and its bottom
*     neighbor. For iy.eq.-1, fchy(-1:nx,iy) will hold 0.
*
*    ..auxiliary variables describing the plasma state
*
*     nb - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), nb(ix,iy) specifies the density of
*     one selected atomic species on the (ix,iy) cell. If that species
*     has index (is) then nb(,).eq.na(,,is).
*
*     ub - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), ub(ix,iy) specifies the parallel
*     velocity of one selected atomic species on the (ix,iy) cell.
*     The sign of the parallel velocity indicates whether it is flowing
```

```
*     in the same direction as the local poloidal magnetic field.
*
*     ni - (-1:nx,-1:ny,0:1) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), ni(ix,iy,0) specifies the total atom
*     density on the (ix,iy) cell. It will hold that 0.lt.ni(,0).
*     The defining expression for ni is
*       ni(,0) = (sum is : 0.le.is.lt.ns : na(,,is)).
*     For (ix,iy) in (-1:nx,-1:ny), ni(ix,iy,1) specifies the total ion
*     density on the (ix,iy) cell. It will hold that 0.lt.ni(,1).
*     The defining expression for ni is
*       ni(,1) = (sum is : 0.le.is.lt.ns : na(,,is)) when rza(,,is)>0.
*
*     ne - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), ne(ix,iy) specifies the electron
*     density on the (ix,iy) cell. It will hold that 0.lt.ne(,).
*     The defining expression for ne is
*       ne(,) = (sum is : 0.le.is.lt.ns : rza(,,is)*na(,,is)).
*
*     ne2 - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), ne2(ix,iy) specifies the atom
*     density weighted by rz2 on the (ix,iy) cell. It will hold that
*     0.lt.ne2(,).
*     The defining expression for ne2 is
*       ne2(,) = (sum is : 0.le.is.lt.ns : rz2(,,is)*na(,,is)).
*
*     pr - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), pr(ix,iy) specifies the total
*     pressure on the (ix,iy) cell. The defining expression for pr is
*       pr(,) = ne(,)*te(,) + (sum is :: na(,,is)*ti(,)).
*
*     pz - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), pz(ix,iy) specifies the plasma
*     pressure on the (ix,iy) cell. The defining expression for pr is
*       pr(,) = ne(,)*te(,) + (sum is :: na(,,is)*ti(,), not including neutrals).
*
*     pe - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), pe(ix,iy) specifies the electron
*     pressure on the (ix,iy) cell.
*     The defining expression for pe is pe(,) = ne(,)*te(,).
*
*     ro - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), ro(ix,iy) specifies the mass
*     density on the (ix,iy) cell. The defining expression for ro is
*       ro(,) = (sum is : 0.le.is.lt.ns : am(is)*mp**na(,,is)).
*
*     rz - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), rz(ix,iy) specifies the ion mass
*     density on the (ix,iy) cell. The defining expression for rz is
*       rz(,) = (sum is : 0.le.is.lt.ns .and. .not.is_neutral(is) :
*                         am(is)*mp**na(,,is)).
*
*     ne2m - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), ne2m(ix,iy) specifies the atom
*     density weighted by rz2/am on the (ix,iy) cell. It will hold
*     that 0.lt.ne2m(,).
*     The defining expression for ne2m is
*       ne2m(,) = (sum is :: (rz2(,,is)/am(is))*na(,,is)).
*
*     fne - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let fnex(,)=fne(,,0) and fney(,)=fne(,,1).)
*     For (ix,iy) in (0:nx,-1:ny), fnex(ix,iy) specifies the flux of
*     electrons through the face between the (ix,iy) cell and its
*     left neighbor. For ix.eq.-1, fnex(ix,-1:ny) will hold 0.
*     For (ix,iy) in (-1:nx,0:ny), fney(ix,iy) specifies the flux of
*     electrons through the face between the (ix,iy) cell and its
*     bottom neighbor. For iy.eq.-1, fney(-1:nx,iy) will hold 0.
*     The defining expression for fne is
```

```
*       fne(,,) = (sum is :: rza(,,is)*fna(,,,is))-(1/qe)*fch(,,)
*
*     fni - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let fnix(,)=fni(,,0) and fniy(,)=fni(,,1).)
*     For (ix,iy) in (0:nx,-1:ny), fnix(ix,iy) specifies the flux of
*     atoms (all species) through the face between the (ix,iy) cell
*     and its left neighbor. For ix.eq.-1, fnix(ix,-1:ny) will hold 0.
*     For (ix,iy) in (-1:nx,0:ny), fniy(ix,iy) specifies the flux of
*     atoms (all species) through the face between the (ix,iy) cell
*     and its bottom neighbor. For iy.eq.-1, fniy(-1:nx,iy) will hold 0.
*     The defining expression for fni is
*       fni(,,) = (sum is :: fna(,,,is))
*
*     fchvispar - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let fchvisparx(,)=fchvispar(,,0) and fchvispary(,)=fchvispar(,,1).)
*     For (ix,iy) in (0:nx,-1:ny), fchvisparx(ix,iy) specifies the
*     electric current driven by parallel viscosity through the face
*     between the (ix,iy) cell and its left neighbor.
*     For ix.eq.-1, fchvisparx(ix,-1:ny) will hold 0.
*     For (ix,iy) in (-1:nx,0:ny), fchvispary(ix,iy) specifies the
*     electric current driven by parallel viscosity through the face
*     between the (ix,iy) cell and its bottom neighbor.
*     For iy.eq.-1, fchvispary(-1:nx,iy) will hold 0.
*
*     fchvispar_a - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     (Let fchvispar_ax(,,)=fchvispar_a(,,0,) and fchvispar_ay(,,)=fchvispar(,,1,).)
*     For (ix,iy,is) in (0:nx,-1:ny,0:ns-1), fchvispar_ax(ix,iy,is) specifies the
*     electric current driven by parallel viscosity of atoms of species (is) through
*     the face between the (ix,iy) cell and its left neighbor.
*     For ix.eq.-1, fchvispar_ax(ix,-1:ny,0:ns-1) will hold 0.
*     For (ix,iy,is) in (-1:nx,0:ny,0:ns-1), fchvispar_ay(ix,iy,is) specifies the
*     electric current driven by parallel viscosity of atoms of species (is) through
*     the face between the (ix,iy) cell and its left neighbor.
*     For iy.eq.-1, fchvispar_ay(-1:nx,iy,0:ns-1) will hold 0.
*
*     fchvisq - (-1:nx,-1:ny,0:1) real*8 array.
*     fchvisq specifies the electric current produced by the components
*     of the viscosity tensor which is connected with the heat fluxes.
*     (Let fchvisqx(,)=fchvisq(,,0) and fchvisqy(,)=fchvisq(,,1).)
*     For (ix,iy) in (0:nx,-1:ny), fchvisqx(ix,iy) specifies the current
*     through the face between the (ix,iy) cell and its left neighbor.
*     For ix.eq.-1, fchvisqx(ix,-1:ny) will hold 0.
*     For (ix,iy) in (-1:nx,0:ny), fchvisqy(ix,iy) specifies the current
*     through the face between the (ix,iy) cell and its bottom neighbor.
*     For iy.eq.-1, fchvisqy(-1:nx,iy) will hold 0.
*
*     fchvisq_a - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     fchvisq_a specifies the electric current of atoms of species (is)
*     produced by the components of the viscosity tensor which is connected
*     with the heat fluxes.
*     (Let fchvisq_ax(,,)=fchvisq_a(,,0,) and fchvisq_ay(,,)=fchvisq(,,1,).)
*     For (ix,iy,is) in (0:nx,-1:ny,0:ns-1), fchvisq_ax(ix,iy,is) specifies the
*     current of atoms of species (is) through the face between the (ix,iy) cell
*     and its left neighbor.
*     For ix.eq.-1, fchvisq_ax(ix,-1:ny,0:ns-1) will hold 0.
*     For (ix,iy,is) in (-1:nx,0:ny,0:ns-1), fchvisq_ay(ix,iy,is) specifies the
*     current of atoms of species (is) through the face between the (ix,iy) cell
*     and its bottom neighbor.
*     For iy.eq.-1, fchvisq_ay(-1:nx,iy,0:ns-1) will hold 0.
*
*     fchinert - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let fchinertx(,)=fchinert(,,0) and fchinerty(,)=fchinert(,,1).)
*     For (ix,iy) in (0:nx,-1:ny), fchinertx(ix,iy) specifies the
*     contributions from inertia and gyroviscosity current through
*     the face between the (ix,iy) cell and its left neighbor.
*     For ix.eq.-1, fchinertx(ix,-1:ny) will hold 0.
*     For (ix,iy) in (-1:nx,0:ny), fchinerty(ix,iy) specifies the
```

```
*     contributions from inertia and gyroviscosity current through
*     the face between the (ix,iy) cell and its bottom neighbor.
*     For iy.eq.-1, fchinerty(-1:nx,iy) will hold 0.
*
*     fchinert_a - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     (Let fchinert_ax(,,)=fchinert_a(,,0,) and fchinert_ay(,,)=fchinertl(,,1,).)
*     For (ix,iy,is) in (0:nx,-1:ny,0:ns-1), fchinert_ax(ix,iy,is) specifies the
*     contributions from inertia and gyroviscosity current of atoms of species
*     (is) through the face between the (ix,iy) cell and its left neighbor.
*     For ix.eq.-1, fchinert_ax(ix,-1:ny,0:ns-1) will hold 0.
*     For (ix,iy,is) in (-1:nx,0:ny,0:ns-1), fchinert_ay(ix,iy,is) specifies the
*     contributions from inertia and gyroviscosity current of atoms of species (is)
*     through the face between the (ix,iy) cell and its bottom neighbor.
*     For iy.eq.-1, fchinert_ay(-1:nx,iy,0:ns-1) will hold 0.
*
*     fchanml - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let fchanmlx(,)=fchanml(,,0) and fchanmly(,)=fchanml(,,1).)
*     For (ix,iy) in (0:nx,-1:ny), fchanmlx(ix,iy) specifies the
*     anomalous current through the face between the (ix,iy) cell
*     and its left neighbor.
*     For ix.eq.-1, fchanmlx(ix,-1:ny) will hold 0.
*     For (ix,iy) in (-1:nx,0:ny), fchanmly(ix,iy) specifies the
*     amomalous current through the face between the (ix,iy) cell
*     and its bottom neighbor.
*     For iy.eq.-1, fchanmly(-1:nx,iy) will hold 0.
*
*     fchanml_a - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     (Let fchanml_ax(,,)=fchanml_a(,,0,) and fchanml_ay(,,)=fchanml(,,1,).)
*     For (ix,iy,is) in (0:nx,-1:ny,0:ns-1), fchanml_ax(ix,iy,is) specifies the
*     anomalous current of atoms of species (is) through the face between
*     the (ix,iy) cell and its left neighbor.
*     For ix.eq.-1, fchanml_ax(ix,-1:ny,0:ns-1) will hold 0.
*     For (ix,iy,is) in (-1:nx,0:ny,0:ns-1), fchanml_ay(ix,iy,is) specifies the
*     amomalous current of atoms of species (is) through the face between
*     the (ix,iy) cell and its bottom neighbor.
*     For iy.eq.-1, fchanml_ay(-1:nx,iy,0:ns-1) will hold 0.
*
*     fchdia - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let fchdiax(,)=fchdia(,,0) and fchdiay(,)=fchdia(,,1).)
*     For (ix,iy) in (0:nx,-1:ny), fchdiax(ix,iy) specifies the
*     modified diamagnetic current through the face between
*     the (ix,iy) cell and its left neighbor.
*     For ix.eq.-1, fchdiax(ix,-1:ny) will hold 0.
*     For (ix,iy) in (-1:nx,0:ny), fchdiay(ix,iy) specifies the
*     modified diamagnetic current through the face between
*     the (ix,iy) cell and its bottom neighbor.
*     For iy.eq.-1, fchdiay(-1:nx,iy) will hold 0.
*
*     fchin - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let fchinx(,)=fchin(,,0) and fchiny(,)=fchin(,,1).)
*     For (ix,iy) in (0:nx,-1:ny), fchinx(ix,iy) specifies the
*     contribution from ion-neutral friction through the face between
*     the (ix,iy) cell and its left neighbor.
*     For ix.eq.-1, fchinx(ix,-1:ny) will hold 0.
*     For (ix,iy) in (-1:nx,0:ny), fchiny(ix,iy) specifies the
*     contribution from ion-neutral friction through the face between
*     the (ix,iy) cell and its bottom neighbor.
*     For iy.eq.-1, fchiny(-1:nx,iy) will hold 0.
*
*     fch_p - (-1:nx,-1:ny,0:1) real*8 array.
*     For (ix,iy) in (0:nx,-1:ny), fch_p(ix,iy,0) specifies the product of
*     the parallel electric current and the poloidal magnetic field component
*     sign through the face between the (ix,iy) cell and its left neighbor.
*     For ix.eq.-1, fch_p(ix,-1:ny,0) will hold 0.
*     fch_p(-1:nx,-1:ny,1) = 0
*
*     fchvisper - (-1:nx,-1:ny,0:1) real*8 array.
```

```
*     (Let fchvisperx(,)=fchvisper(,,0) and fchvispery(,)=fchvisper(,,1).)
*     For (ix,iy) in (-1:nx,-1:ny), fchvisperx(ix,iy) = 0.
*     For (ix,iy) in (-1:nx, 0:ny), fchvispery(ix,iy) specifies the
*     electric current connected with contribution from
*     the perpendicular viscosity through the face between the (ix,iy) cell
*     and its bottom neighbor.
*     For iy.eq.-1, fchvispery(-1:nx,iy) will hold 0.
*
*     fchvisper_a - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     (Let fchvisper_ax(,,)=fchvisper_a(,,0,) and fchvispery(,,)=fchvisper(,,1,).)
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), fchvisper_ax(ix,iy) = 0.
*     For (ix,iy,is) in (-1:nx, 0:ny,0:ns-1), fchvisper_ay(ix,iy,is) specifies the
*     electric current of atoms of species (is) connected with contribution from
*     the perpendicular viscosity through the face between the (ix,iy) cell
*     and its bottom neighbor.
*     For iy.eq.-1, fchvisper_ay(-1:nx,iy,0:ns-1) will hold 0.
*
*     uadia - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     (Let uadiax(,,)=uadia(,,0,) and uadiay(,)=uadia(,,1,).)
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), uadiax(ix,iy,is) specifies
*     the total effective drift velocity of species (is) in cell (ix,iy)
*     in the diamagnetic direction (i.e. within the flux surface but
*     perpendicular to the magnetic field), but projected onto the
*     poloidal direction.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), uadiay(ix,iy,is) specifies
*     the total effective drift velocity of species (is) in cell (ix,iy)
*     in the radial direction.
*     uadia is computed at cell centers if 'b2tfnb_drift_style' is '0'
*     uadia is computed at cell faces otherwise (default)
*
*     vadia - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     (Let vadiax(,,)=vadia(,,0,) and vadiay(,)=vadia(,,1,).)
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), vadiax(ix,iy,is) specifies
*     the effective diamagnetic drift velocity of species (is) in cell (ix,iy)
*     in the diamagnetic direction (i.e. within the flux surface but
*     perpendicular to the magnetic field), but projected onto the
*     poloidal direction.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), vadiay(ix,iy,is) specifies
*     the effective diamagnetic drift velocity of species (is) in cell (ix,iy)
*     in the radial direction.
*     vadia is computed at cell centers if 'b2tfnb_drift_style' is '0'
*     vadia is computed at cell faces otherwise (default)
*
*     vedia - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let vediax(,)=vedia(,,0) and vediay(,)=vedia(,,1).)
*     For (ix,iy) in (-1:nx,-1:ny), vediax(ix,iy) specifies
*     the effective diamagnetic drift velocity of electrons in cell (ix,iy)
*     in the diamagnetic direction (i.e. within the flux surface but
*     perpendicular to the magnetic field), but projected onto the
*     poloidal direction.
*     For (ix,iy) in (-1:nx,-1:ny), vediay(ix,iy) specifies
*     the effective diamagnetic drift velocity of electrons in cell (ix,iy)
*     in the radial direction.
*     vedia is computed at cell centers if 'b2tfnb_drift_style' is '0'
*     vedia is computed at cell faces otherwise (default)
*
*     wadia - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     (Let wadiax(,,)=wadia(,,0,) and wadiay(,)=wadia(,,1,).)
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), wadiax(ix,iy,is) specifies
*     the total diamagnetic drift velocity of species (is) in cell (ix,iy)
*     in the diamagnetic direction (i.e. within the flux surface but
*     perpendicular to the magnetic field), but projected onto the
*     poloidal direction.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), wadiay(ix,iy,is) specifies
*     the total diamagnetic drift velocity of species (is) in cell (ix,iy)
*     in the radial direction.
*     wadia is computed at cell centers if 'b2tfnb_drift_style' is '0'
```

```
*     wadia is computed at cell faces otherwise (default)
*
*     vaecrb - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     (Let vaecrbx(,,)=vaecrb(,,0,) and vaecrby(,)=vaecrb(,,1,).)
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), vaecrbx(ix,iy,is) specifies
*     the ExB drift velocity of species (is) in cell (ix,iy)
*     in the diamagnetic direction (i.e. within the flux surface but
*     perpendicular to the magnetic field), but projected onto the
*     poloidal direction.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), vaecrby(ix,iy,is) specifies
*     the ExB drift velocity of species (is) in cell (ix,iy)
*     in the radial direction.
*     vaecrb is computed at cell centers if 'b2tfnb_drift_style' is '0'
*     vaecrb is computed at cell faces otherwise (default)
*
*     veecrb - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let veecrbx(,)=veecrb(,,0) and veecrby(,)=veecrb(,,1).)
*     For (ix,iy) in (-1:nx,-1:ny), veecrbx(ix,iy) specifies
*     the ExB drift velocity of electrons in cell (ix,iy)
*     in the diamagnetic direction (i.e. within the flux surface but
*     perpendicular to the magnetic field), but projected onto the
*     poloidal direction.
*     For (ix,iy) in (-1:nx,-1:ny), veecrby(ix,iy) specifies
*     the ExB drift velocity of electrons in cell (ix,iy)
*     in the radial direction.
*     veecrb is computed at cell centers if 'b2tfnb_drift_style' is '0'
*     veecrb is computed at cell faces otherwise (default)
*
*     kinrgy - (-1:nx,-1:ny,0:ns-1) real*8 array.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), kinrgy(ix,iy,is) specifies
*     the kinetic energy of a particle of species (is) in cell (ix,iy).
*
*
*    ..residuals and corrections
*
*     resco - (-1:nx,-1:ny,0:ns-1) real*8 array.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), resco(ix,iy,is) specifies
*     the residual of the continuity equation on the (ix,iy) cell for
*     species (is).
*
*     resmo - (-1:nx,-1:ny,0:ns-1) real*8 array.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), resmo(ix,iy,is) specifies
*     the residual of the parallel momentum equation on the (ix,iy)
*     cell for species (is).
*
*     resmb - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), resmb(ix,iy) specifies the residual
*     of the parallel momentum equation on the (ix,iy) cell for the
*     selected species. If that species has index (is) then
*     resmb(,).eq.resmo(,,is).
*
*     resmt - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), resmt(ix,iy) specifies the residual
*     of the total parallel momentum equation on the (ix,iy) cell.
*
*     reshe - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), reshe(ix,iy) specifies the residual
*     of the electron heat balance equation on the (ix,iy) cell.
*
*     reshi - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), reshe(ix,iy) specifies the residual
*     of the ion heat balance equation on the (ix,iy) cell.
*
*     resht - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), resht(ix,iy) specifies the residual
*     of the total heat balance equation on the (ix,iy) cell.
*
```

```
*     respo - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), respo(ix,iy) specifies the
*     residual of the electric potential equation on the (ix,iy) cell.
*
*     corpa - (-1:nx,-1:ny,0:ns-1) real*8 array.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), corpa(ix,iy,is) specifies
*     the last computed correction to the pressure of species (is) on
*     the (ix,iy) cell. corpa is computed by solving an equation that
*     has resco as its right hand side. It is not necessarily the same
*     as the pressure correction over one complete iteration.
*
*     corua - (-1:nx,-1:ny,0:ns-1) real*8 array.
*     For (ix,iy,is) in (0:nx,-1:ny,0:ns-1), corua(ix,iy,is) specifies
*     the last computed correction to the parallel velocity ua of
*     species (is) on the (ix,iy) cell.
*     corua is computed by solving an equation that has resmo as its
*     right hand side. It is not necessarily the same as the change in
*     the parallel velocity over one complete iteration.
*
*     corut - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (0:nx,-1:ny), corut(ix,iy) specifies the last
*     computed correction to the parallel velocity ua of all species
*     simultaneously on the (ix,iy) cell.
*     corut is computed by solving an equation that has resmt as its
*     right hand side. It is not necessarily the same as the change in
*     the parallel velocity over one complete iteration.
*
*     corte - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), corte(ix,iy) specifies the last
*     computed correction to the electron temperature on the (ix,iy)
*     cell. corte is computed by solving an equation that has reshe as
*     its right hand side. It is not necessarily the same as the change
*     in the electron temperature over one complete iteration.
*
*     corti - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), corti(ix,iy) specifies the last
*     computed correction to the atom temperature on the (ix,iy) cell.
*     corti is computed by solving an equation that has reshi as its
*     right hand side. It is not necessarily the same as the change in
*     the atom temperature over one complete iteration.
*
*     cortt - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), cortt(ix,iy) specifies the last
*     computed correction to the combined temperature on the (ix,iy)
*     cell. cortt is computed by solving an equation that has resht as
*     its right hand side. It is not necessarily the same as the change
*     in the electron or atom temperature over one complete iteration.
*
*     corpo - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), corpo(ix,iy) specifies the last
*     computed correction to the electric potential on the (ix,iy)
*     cell. corpo is computed by solving an equation that has respo as
*     its right hand side. It is not necessarily the same as the change
*     in the electric potential over one complete iteration.
*
*     aresco - (0:ns-1) real*8 array.
*     For (is) in (0:ns-1), aresco(is) specifies the scaled norm of the
*     residual of the continuity equation for species (is).
*
*     aresmo - (0:ns-1) real*8 array.
*     For (is) in (0:ns-1), aresmo(is) specifies the scaled norm of the
*     residual of the parallel momentum equation for species (is).
*
*     aresmt - real.
*     aresmt specifies the scaled norm of the residual of the total
*     parallel momentum equation.
*
```

```
*     areshe - real.
*     areshe specifies the scaled norm of the residual of the electron
*     heat balance equation.
*
*     areshi - real.
*     areshi specifies the scaled norm of the residual of the all atom
*     heat balance equation.
*
*     arespo - real.
*     arespo specifies the scaled norm of the residual of the electric
*     potential equation.
*
*     acorpa - (0:ns-1) real*8 array.
*     For (is) in (0:ns-1), acorpa(is) specifies the norm of the last
*     computed correction for the continuity equation for species (is).
*
*     acorua - (0:ns-1) real*8 array.
*     For (is) in (0:ns-1), acorua(is) specifies the norm of the last
*     computed correction for the momentum balance equation for
*     species (is).
*
*     acorut - real.
*     acorut specifies the norm of the last computed correction for the
*     parallel momentum balance equation for all species simultaneously.
*
*     acorte - real.
*     acorte specifies the norm of the last computed correction for the
*     electron heat balance equation.
*
*     acorti - real.
*     acorti specifies the norm of the last computed correction for the
*     all atom heat balance equation.
*
*     acorpo - real.
*     acorpo specifies the norm of the last computed correction for the
*     electric potential equation.
*
*   ..rate coefficients
*
*     /b2cmrc/ - common block.
*     /b2cmrc/ contains a table of atomic rate coefficients.
*      rtnt - integer constant.
*     rtnt specifies the number of intervals in the discretisation
*     of the temperature range. The table will contain data for
*     (rtnt+1) values of the temperature variable.
*      rtnn - integer constant.
*     rtnn specifies the number of intervals in the discretisation
*     of the density range. The table will contain data for (rtnn+1)
*     values of the density variable.
*      rtns - integer.
*     rtns specifies the total number of atomic species; i.e.
*       (sum inuc : 0.le.inuc.lt.nnuc : izhi(inuc)-izlo(inuc)+1) ,
*     where inuc is a nuclear species index, izlo(inuc) is the lowest
*     and izhi(inuc) is the highest atomic charge considered in B2 for
*     that nuclear species.
*     rtns equals the variable ns used elsewhere in B2.
*      rtdumm - integer.
*     rtdumm is a dummy integer variable, introduced in order to obtain
*     an even-length sequence of integer variables in the common block.
*      rtzmin, rtzmax, rtzn - (0:rtns-1) real*8 array.
*     For 0.le.is.lt.rtns, rtzmin(is) and rtzmax(is) specify the atomic
*     charge range, and rtzn(is) specifies the nuclear charge
*     of atomic species (is).
*     The izhi(inuc)-izlo(inuc)+1 charge states for each nuclear
*     species are taken in natural order, starting at izlo(inuc); the
*     highest state of nuclear species (inuc) is followed by the lowest
*     state of (inuc+1). rtzmin, rtzmax and rtzn are equal to the arrays
```

```
*     zamin, zamax and zn used elsewhere in B2.
*      rtt - (0:rtnt) real*8 array.
*     rtt(0:rtnt) specifies the sequence of values of the temperature
*     variable, expressed in ev, for which the rate coefficients are
*     evaluated. rtt(0:rtnt) will be positive and increasing.
*      rtn - (0:rtnn) real*8 array.
*     rtn(0:rtnn) specifies the sequence of values of the density
*     variable, expressed in m**-3, for which the rate coefficients
*     are evaluated. rtn(0:rtnn) will be positive and increasing.
*      rtlt - (0:rtnt) real*8 array.
*     rtlt(0:rtnt) holds log(rtt(0:rtnt)).
*      rtln - (0:rtnn) real*8 array.
*     rtln(0:rtnn) holds log(rtn(0:rtnn)).
*      rtlsa - (0:rtnt,0:rtnn,0:rtns-1) real*8 array.
*     For (i,j,k) in (0:rtnt,0:rtnn,0:rtns-1), rtlsa(i,j,k) specifies
*     logu(rtrsa), where rtrsa is the ionisation rate coefficient at
*     electron temperature rtt(i), electron density rtn(j), for the
*     process k->k+1. rtrsa has dimension m**3/s. The function logu is
*     the same as (log) except that logu(0.0) is equal to some huge
*     negative number.
*      rtlra - (0:rtnt,0:rtnn,0:rtns-1) real*8 array.
*     For (i,j,k) in (0:rtnt,0:rtnn,0:rtns-1), rtlra(i,j,k) specifies
*     logu(rtrra), where rtrra is the recombination rate coefficient at
*     electron temperature rtt(i) and electron density rtn(j), for the
*     process k->k-1. rtrra has dimension m**3/s.
*      rtlqa - (0:rtnt,0:rtnn,0:rtns-1) real*8 array.
*     For (i,j,k) in (0:rtnt,0:rtnn,0:rtns-1), rtlqa(i,j,k) specifies
*     logu(rtrqa), where rtrqa is the electron heat loss rate
*     coefficient at electron temperature rtt(i) and electron density
*     rtn(j), for processes starting from atomic species k. rtrqa has
*     dimension ev*m**3/s.
*      rtlcx - (0:rtnt,0:rtnn,0:rtns-1) real*8 array.
*     For (i,j,k) in (0:rtnt,0:rtnn,0:rtns-1), rtlcx(i,j,k) specifies
*     logu(rtrcx), where rtrcx is the charge exchange rate coefficient
*     at neutral hydrogen temperature/amu rtt(i) and electron density
*     rtn(j), for the process k->k-1. rtrcx has dimension m**3/s.
*      rtlza - (0:rtnt,0:rtnn,0:rtns-1) real*8 array.
*     For (i,j,k) in (0:rtnt,0:rtnn,0:rtns-1), rtlza(i,j,k) specifies
*     the effective charge state at
*     electron temperature rtt(i), electron density rtn(j), for the
*     superstage k. rtlza has dimension e.
*      rtlz2 - (0:rtnt,0:rtnn,0:rtns-1) real*8 array.
*     For (i,j,k) in (0:rtnt,0:rtnn,0:rtns-1), rtlz2(i,j,k) specifies
*     the effective squared charge at
*     electron temperature rtt(i), electron density rtn(j), for the
*     superstage k. rtlz2 has dimension e**2.
*      rtlpt - (0:rtnt,0:rtnn,0:rtns-1) real*8 array.
*     For (i,j,k) in (0:rtnt,0:rtnn,0:rtns-1), rtlpt(i,j,k) specifies
*     the cumulative ionisation energy at
*     electron temperature rtt(i), electron density rtn(j),
*     to reach superstage k. rtlpt has dimension eV.
*      rtlpi - (0:rtnt,0:rtnn,0:rtns-1) real*8 array.
*     For (i,j,k) in (0:rtnt,0:rtnn,0:rtns-1), rtlpi(i,j,k) specifies
*     the effective ionisation energy at
*     electron temperature rtt(i), electron density rtn(j),
*     to reach superstage k from superstage k-1. rtlpi has dimension eV.
*
*     nscx - integer.
*     nscx is the number of charge exchanging neutral hydrogen species
*     that are present in the calculation.  (So, usually nscx.eq.1).
*     It will hold that 0.le.nscx.lt.10.
*
*     iscx - (0:nscxmax-1) integer array.
*     iscx(0:nscx-1) will hold the species indices for the various kinds
*     of neutral hydrogen that are present in the calculation.
*     For 0.le.k.lt.nscx it will hold that 0.le.iscx(k).lt.ns.
*
```

```
*     rlsa - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), rlsa(ix,iy,0,is) and
*     rlsa(ix,iy,1,is) are the coefficients in the representation
*       rsa = exp(rlsa(,,0,)+rlsa(,,1,)*log(te(ix,iy)/ev))
*     that specifies the rate coefficient for ionisation from atomic
*     species (is) to species (is+1). (For is.eq.ns-1 or in case the
*     atomic species (is) and (is+1) belong to different nuclear
*     species this expression will underflow, and its value is then
*     understood as 0.)
*     rsa has dimension m**3/s.
*
*     rlra - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), rlra(ix,iy,0,is) and
*     rlra(ix,iy,1,is) are the coefficients in the representation
*       rra = exp(rlra(,,0,)+rlra(,,1,)*log(te(ix,iy)/ev))
*     that specifies the rate coefficient for recombination from atomic
*     species (is) to species (is-1). (For is.eq.0 or in case the
*     atomic species (is) and (is-1) belong to different nuclear
*     species this expression will underflow, and its value is then
*     understood as 0.)
*     rra has dimension m**3/s.
*
*     rlqa - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), rlqa(ix,iy,0,is) and
*     rlqa(ix,iy,1,is) are the coefficients in the representation
*       rqa = ev*exp(rlqa(,,0,)+rlqa(,,1,)*log(te(ix,iy)/ev))
*     that specifies the rate coefficient for electron heat loss terms
*     that are proportional to the density of the atomic species (is).
*     rqa has dimension J*m**3/s.
*
*     rlcx - (-1:nx,-1:ny,0:1,0:ns-1,0:nscxmax-1) real*8 array.
*     For (ix,iy,is,k) in (-1:nx,-1:ny,0:ns-1,0:nscx-1),
*     rlcx(ix,iy,0,is,k) and rlcx(ix,iy,1,is,k) are the coefficients in
*     the expression
*       rcx = exp(rlcx(,,0,,)+rlcx(,,1,,)*log(th0/(amh0*ev)))
*     that specifies the rate coefficient for charge exchange
*     recombination from state (is) to state (is-1) by collisions with
*     the neutral hydrogen species iscx(k); here, amh0 is the atomic
*     mass of that neutral species. (For is.eq.0 or in case the atomic
*     species (is) and (is-1) belong to different nuclear species this
*     expression will underflow, and its value is then understood as 0.)
*     rcx has dimension m**3/s.
*
*     rlza - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), rlza(ix,iy,0,is) and
*     rlza(ix,iy,1,is) are the coefficients in the representation
*       rza = rlza(,,0,)+rlza(,,1,)*log(te(ix,iy)/ev)
*     that specifies the effective charge of stage (is).
*     rza has dimension e.
*
*     rlz2 - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), rlz2(ix,iy,0,is) and
*     rlz2(ix,iy,1,is) are the coefficients in the representation
*       rz2 = rlz2(,,0,)+rlz2(,,1,)*log(te(ix,iy)/ev)
*     that specifies the effective square charge of stage (is).
*     rz2 has dimension e**2.
*
*     rlpt - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), rlpt(ix,iy,0,is) and
*     rlpt(ix,iy,1,is) are the coefficients in the representation
*       rpt = rlpt(,,0,)+rlpt(,,1,)*log(te(ix,iy)/ev)
*     that specifies the cumulative ionisation potential to stage (is).
*     rpt has dimension eV.
*
*     rlpi - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), rlpi(ix,iy,0,is) and
*     rlpi(ix,iy,1,is) are the coefficients in the representation
```

```
*       rpi = rlpi(,,0,)+rlpi(,,1,)*log(te(ix,iy)/ev)
*     that specifies the effective ionisation potential to stage (is)
*     from stage (is-1).
*     rpi has dimension eV.
*
*     rsa - (-1:nx,-1:ny,0:ns-1) real*8 array.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), rsa(ix,iy,is) specifies
*     the rate coefficient for ionisation from atomic species (is) to
*     species (is+1). For is.eq.ns-1 or in case the atomic species (is)
*     and (is+1) belong to different nuclear species, this coefficient
*     will be 0.
*     rsa has dimension m**3/s.
*
*     rra - (-1:nx,-1:ny,0:ns-1) real*8 array.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), rra(ix,iy,is) specifies
*     the rate coefficient for recombination from atomic species (is)
*     to species (is-1). For is.eq.0 or in case the atomic species (is)
*     and (is-1) belong to different nuclear species, this coefficient
*     will be 0.
*     rra has dimension m**3/s.
*
*     rqa - (-1:nx,-1:ny,0:ns-1) real*8 array.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), rqa(ix,iy,is) specifies
*     the rate coefficient for electron heat loss terms that are
*     proportional to the density of the atomic species (is).
*     rqa has dimension J*m**3/s.
*
*     rcx - (-1:nx,-1:ny,0:ns-1) real*8 array.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), rcx(ix,iy,is) specifies
*     the rate coefficient for charge exchange recombination from
*     atomic species (is) to species (is-1) by collisions with the
*     selected species of neutral hydrogen. For is.eq.0 or in case the
*     atomic species (is) and (is-1) belong to different nuclear
*     species, this coefficient will be 0.
*     rcx has dimension m**3/s.
*
*   ..boundary conditions, parameters
*
*     /b2cmpb/ - common block.
*     /b2cmpb/ specifies physics parameters for boundary conditions.
*     For this purpose the boundary is divided up into regions, and for
*     each region a set of coefficients defines the local conditions.
*     For detailed description of /b2cmpb/ see the code, especially the
*     routine b2stbc.
*      cbnrmx - integer constant.
*     cbnrmx is an upper bound for the number of boundary regions nreg,
*     (nreg.eq.cbnrso+cbnrno+cbnrwe+cbnrea) and is used in dimension
*     specifications.
*      cbirso, cbirno, cbirwe, cbirea - integer.
*     cbirso(-no,-we,-ea) is the index of the first region on the
*     south (north, west, east) boundary.
*      cbnrso, cbnrno, cbnrwe, cbnrea - integer.
*     cbnrso(-no,-we,-ea) is the number of regions on the south
*     (north, west, east) boundary.
*     It will hold that 1.le.cbnrso, 1.le.cbnrno, 1.le.cbnrwe, and
*     1.le.cbnrea; and 0.eq.cbirso, cbirso+cbnrso.eq.cbirno,
*     cbirno+cbnrno.eq.cbirwe, cbirwe+cbnrwe.eq.cbirea.
*      cbrbrk - (0:cbnrmx-1) real*8 array.
*     cbrbrk defines the boundaries between the boundary regions.
*      cbsna - (0:7,0:nsdecl-1,0:cbnrmx-1) real*8 array.
*     For 0.le.is.lt.ns and 0.le.ireg.lt.nreg, the eight coefficients
*     cbsna(0:7,is,ireg) specify standard form boundary conditions for
*     the continuity equation for species (is) on the (ireg) boundary
*     region.
*      cbsmo - (0:7,0:nsdecl-1,0:cbnrmx-1) real*8 array.
*     For 0.le.is.lt.ns and 0.le.ireg.lt.nreg, the eight coefficients
*     cbsmo(0:7,is,ireg) specify standard form boundary conditions for
```

```
*     the parallel momentum balance equation for species (is) on the
*     (ireg) boundary region.
*      cbshi - (0:7,0:nsdecl-1,0:cbnrmx-1) real*8 array.
*     For 0.le.is.lt.ns and 0.le.ireg.lt.nreg, the eight coefficients
*     cbshi(0:7,is,ireg) specify standard form boundary conditions for
*     the contribution to that atom heat equation by species (is) on
*     the (ireg) boundary region.
*      cbshe - (0:7,0:cbnrmx-1) real*8 array.
*     For 0.le.ireg.lt.nreg, the eight coefficients cbshe(0:7,ireg)
*     specify standard form boundary conditions for the electron heat
*     equation on the (ireg) boundary region.
*      cbsch - (0:7,0:cbnrmx-1) real*8 array.
*     For 0.le.ireg.lt.nreg, the eight coefficients cbsch(0:7,ireg)
*     specify standard form boundary conditions for the electric
*     potential equation on the (ireg) boundary region.
*      cbrec - (0:7,0:nsdecl-1,0:cbnrmx-1) real*8 array.
*     For 0.le.is.lt.ns-1 and 0.le.ireg.lt.nreg, the eight coefficients
*     cbrec(0:7,is,ireg) specify the recycling process for species (is)
*     on the boundary region ireg. The exact interpretation of these
*     coefficients is defined by the routine that computes the source
*     terms due to the recycling process (b2stbr). Routine b2stbr is
*     problem dependent.
*      cbmsa - (0:7,0:cbnrmx-1) real*8 array.
*      cbmsc - (0:7,0:nsdecl-1,0:cbnrmx-1) real*8 array.
*     For 0.le.ireg.lt.nreg, the eight coefficients cbmsa(0:7,ireg)
*     specify miscellaneous processes on the boundary region ireg.
*     For 0.le.is.lt.ns-1 and 0.le.ireg.lt.nreg, the eight coefficients
*     cbmsc(0:7,is,ireg) specify miscellaneous processes for species
*     (is) on the boundary region ireg. The interpretation of these
*     coefficients is defined by the routine that computes the source
*     terms due to miscellaneous processes (b2stbm). Routine b2stbm is
*     application dependent.
*
*    ..source terms from neutral gas model
*
*     /b2cmsr/ - common block.
*     /b2cmsr/ specifies source terms from neutral gas model.
*     (Presently empty--to be used in conjunction with a Monte Carlo
*     neutral gas code.)
*
*    ..source coefficients
*
*     sna - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), sna(ix,iy,0,is) and
*     sna(ix,iy,1,is) are the coefficients in the expression
*       sna(ix,iy,0,is)+sna(ix,iy,1,is)*na(ix,iy,is)
*     that specifies a linear approximation to the assumed particle
*     source of species (is) integrated over the (ix,iy) cell.
*
*     sne - (-1:nx,-1:ny,0:1) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), sne(ix,iy,0) and
*     sne(ix,iy,1) are the coefficients in the expression
*       sne(ix,iy,0)+sne(ix,iy,1)*ne(ix,iy)
*     that specifies a linear approximation to the assumed non-ambipolar
*     electron source of integrated over the (ix,iy) cell.
*
*     smo - (-1:nx,-1:ny,0:3,0:ns-1) real*8 array.
*     For (ix,iy,is) in (0:nx,-1:ny,0:ns-1), smo(ix,iy,0:3,is) are the
*     coefficients in the expression
*       smo(ix,iy,0,is)+
*         smo(ix,iy,1,is)*ua(ix,iy,is)+
*         smo(ix,iy,2,is)*roxa(ix,iy,is)+
*         smo(ix,iy,3,is)*roxa(ix,iy,is)*ua(ix,iy,is)
*     that specifies a linear approximation to the parallel momentum
*     source for species (is) integrated over the (ix,iy) cell; here
*       roxa(ix,iy,is) = am(is)*mp*na(ix,iy,is) .
*
```

```
*     she - (-1:nx,-1:ny,0:3) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), she(ix,iy,0:3) are the coefficients
*     in the expression
*       she(ix,iy,0)+
*         she(ix,iy,1)*te(ix,iy)+
*         she(ix,iy,2)*ne(ix,iy)+
*         she(ix,iy,3)*ne(ix,iy)*te(ix,iy)
*     that specifies a linear approximation to the assumed electron heat
*     source integrated over the (ix,iy) cell.
*
*     shi - (-1:nx,-1:ny,0:3) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), shi(ix,iy,0:3) are the coefficients
*     in the expression
*       shi(ix,iy,0)+
*         shi(ix,iy,1)*ti(ix,iy)+
*         shi(ix,iy,2)*ni(ix,iy)+
*         shi(ix,iy,3)*ni(ix,iy)*ti(ix,iy)
*     that specifies a linear approximation to the assumed all atom heat
*     source integrated over the (ix,iy) cell.
*
*     sch - (-1:nx,-1:ny,0:3) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), sch(ix,iy,0:3) are the coefficients
*     in the expression
*       sch(ix,iy,0)+
*         sch(ix,iy,1)*po(ix,iy)+
*         sch(ix,iy,2)*ne(ix,iy)+
*         sch(ix,iy,3)*ne(ix,iy)*po(ix,iy)
*     that specifies a linear approximation to the assumed electric
*     charge source integrated over the (ix,iy) cell.
*
*   ..partial source coefficients
*
*     sna0 - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), sna0(ix,iy,0,is) and
*     sna0(ix,iy,1,is) are the coefficients in the expression
*       sna0(ix,iy,0,is)+sna0(ix,iy,1,is)*na(ix,iy,is)
*     that specifies a linear approximation to some specific
*     contribution to the particle source of species (is) integrated
*     over the (ix,iy) cell.
*
*     sne0 - (-1:nx,-1:ny,0:1) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), sne0(ix,iy,0) and
*     sne0(ix,iy,1) are the coefficients in the expression
*       sne0(ix,iy,0)+sne0(ix,iy,1)*ne(ix,iy)
*     that specifies a linear approximation to some specific
*     contribution to the particle non-ambipolar electron source
*     integrated over the (ix,iy) cell.
*
*     smo0 - (-1:nx,-1:ny,0:3,0:ns-1) real*8 array.
*     For (ix,iy,is) in (0:nx,-1:ny,0:ns-1), smo0(ix,iy,0:3,is) are the
*     coefficients in the expression
*       smo0(ix,iy,0,is)+
*         smo0(ix,iy,1,is)*ua(ix,iy,is)+
*         smo0(ix,iy,2,is)*roxa(ix,iy,is)+
*         smo0(ix,iy,3,is)*roxa(ix,iy,is)*ua(ix,iy,is)
*     that specifies a linear approximation to some specific
*     contribution to the non-frictional parallel momentum source for
*     species (is) integrated over the (ix,iy) cell. Here,
*       roxa(ix,iy,is) = am(is)*mp*na(ix,iy,is) .
*
*     she0 - (-1:nx,-1:ny,0:3) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), she0(ix,iy,0:3) are the coefficients
*     in the expression
*       she0(ix,iy,0)+
*         she0(ix,iy,1)*te(ix,iy)+
*         she0(ix,iy,2)*ne(ix,iy)+
*         she0(ix,iy,3)*ne(ix,iy)*te(ix,iy)
```

```
*     that specifies a linear approximation to some specific
*     contribution to the electron heat source integrated over the
*     (ix,iy) cell.
*
*     shi0 - (-1:nx,-1:ny,0:3) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), shi0(ix,iy,0:3) are the coefficients
*     in the expression
*       shi0(ix,iy,0)+
*         shi0(ix,iy,1)*ti(ix,iy)+
*         shi0(ix,iy,2)*ni(ix,iy)+
*         shi0(ix,iy,3)*ni(ix,iy)*ti(ix,iy)
*     that specifies a linear approximation to some specific
*     contribution to the all atom heat source integrated over the
*     (ix,iy) cell.
*
*     sch0 - (-1:nx,-1:ny,0:3) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), sch0(ix,iy,0:3) are the coefficients
*     in the expression
*       sch0(ix,iy,0)+
*         sch0(ix,iy,1)*po(ix,iy)+
*         sch0(ix,iy,2)*ne(ix,iy)+
*         sch0(ix,iy,3)*ne(ix,iy)*po(ix,iy)
*     that specifies a linear approximation to some specific
*     contribution to the electric charge source integrated over the
*     (ix,iy) cell.
*
*   ..transport coefficients, parameters
*
*     /b2cmpt/ - common block.
*     (Note: the integer constant nsdecl, which comes with /b2cmpa/,
*     is used instead of (ns) in dimension specifications.)
*      cfdf0 - (0:7,0:ns-1) real*8 array.
*     cfdf0 contains some auxiliary coefficients.  For 0.le.is.lt.ns,
*     the local expression
*       df0(,,is) = cfdf0(0,is)+cfdf0(1,is)/(ni(,,0)/1.0e20)+
*         cfdf0(2,is)*sqrt(ti(,)/(am(is)*mp))/
*          (cfdf0(3,is)*ni(,,0)+cfdf0(4,is)*ne(,))
*     is used in the definition of the anomalous transport coefficients.
*     df0 is meant to represent a diffusivity.  If cfdf0(2,is).eq.0 then
*     the corresponding term in df0 is understood as 0 independent of
*     cfdf0(3:4,is).
*      cfdna - (0:7,0:ns-1) real*8 array.
*     For 0.le.is.lt.ns, cfdna(,is) are the coefficients in the
*     local expression
*       difna(,,is) = cfdna(0,is)+cfdna(1,is)/(ne(,)/1.0e20)+
*         cfdna(2,is)*df0(,,is)+
*         cfdna(3,is)*te(,)/ev/abs(bb(,,3))/16
*     that specifies the anomalous diffusivity with respect to
*     grad.na; the transport equation for species (is) contains a term
*     difna(,,is)*'grad.na'.
*     If cfdna(7,is) is not zero, the dna0 array is flux-scaled with
*     a scaling factor of cfdna(7,is)*(hy1(,)/hy1(ixref,))**2
*     where hy1 is the radial width of the cell and ixref is a reference
*     poloidal position (usually the outer midplane or 'jxa').
*     cfdna(4:6,) are not used in the default version of the code.
*      cfdpa - (0:7,0:ns-1) real*8 array.
*     For 0.le.is.lt.ns, cfdpa(,is) are the coefficients in the
*     local expression
*       difpa(,,is) = cfdpa(0,is)+cfdpa(1,is)/(ne(,)/1.0e20)+
*         cfdpa(2,is)*df0(,,is)+
*         cfdpa(3,is)*te(,)/ev/abs(bb(,,3))/16
*     that specifies the anomalous diffusivity with respect to
*     grad.pa; the transport equation for species (is) contains a term
*     (1/(rza(,,is)*te(,)+ti(,)))*difpa(,,is)*'grad.pa'.
*     If cfdpa(7,is) is not zero, the dpa0 array is flux-scaled with
*     a scaling factor of cfdpa(7,is)*(hy1(,)/hy1(ixref,))**2
*     where hy1 is the radial width of the cell and ixref is a reference
```

```
*     poloidal position (usually the outer midplane or 'jxa').
*     cfdpa(4:6,) are not used in the default version of the code.
*      cfvla - (0:7,0:ns-1) real*8 array.
*     For 0.le.is.lt.ns, cfvla(,is) are the coefficients in the
*     local expression
*       vlay(,,is) = cfvla(0,is)+cfvla(1,is)/(ne(,)/1.0e20)+
*         cfvla(2,is)*df0(,,is)+
*         cfvla(3,is)*te(,)/ev/abs(bb(,,3))/16
*     that specifies the anomalous y-velocity of species (is); the
*     transport equation for species (is) contains a term
*     vlay(,,is)*na(,,is).
*     If cfvla(7,is) is not zero, the vla0 array is flux-scaled with
*     a scaling factor of cfvla(7,is)*(hy1(,)/hy1(ixref,))**2
*     where hy1 is the radial width of the cell and ixref is a reference
*     poloidal position (usually the outer midplane or 'jxa').
*     cfvla(4:6,) are not used in the default version of the code.
*      cfvsa - (0:7,0:ns-1) real*8 array.
*     For 0.le.is.lt.ns, cfvsa(,is) are the coefficients in the
*     local expression
*       dfvsa(,,is) = cfvsa(0,is)+cfvsa(1,is)/(ne(,)/1.0e20)+
*         cfvsa(2,is)*df0(,,is)+
*         cfvsa(3,is)*te(,)/ev/abs(bb(,,3))/16
*     that specifies the diffusivity corresponding to the anomalous
*     transverse viscosity, so that this viscosity is
*       vsa(,,is) = mp*am(is)*na(,,is)*dfvsa
*     If cfvsa(7,is) is not zero, the vsa0 array is flux-scaled with
*     a scaling factor of cfvsa(7,is)*(hy1(,)/hy1(ixref,))**2
*     where hy1 is the radial width of the cell and ixref is a reference
*     poloidal position (usually the outer midplane or 'jxa').
*     cfvsa(4:6,) are not used in the default version of the code.
*      cfhci - (0:7,0:ns-1) real*8 array.
*     For 0.le.is.lt.ns, cfhci(,is) are the coefficients in the
*     local expression
*       dfhca(,,is) = cfhci(0,is)+cfhci(1,is)/(ne(,)/1.0e20)+
*         cfhci(2,is)*df0(,,is)+
*         cfhci(3,is)*te(,)/ev/abs(bb(,,3))/16
*     that specifies the anomalous atom heat conductivity associated
*     with species (is), so that the atom heat conduction coefficient
*     contains a term dfhca(,,is)*na(,,is).
*     If cfhci(7,is) is not zero, the hci arrays are flux-scaled with
*     a scaling factor of cfhci(7,is)*(hy1(,)/hy1(ixref,))**2
*     where hy1 is the radial width of the cell and ixref is a reference
*     poloidal position (usually the outer midplane or 'jxa').
*     cfhci(4:6,) are not used in the default version of the code.
*      cfhce - (0:7) real*8 array.
*     cfhce() are the coefficients in the local expression
*       dfhce(,) = cfhce(0)+cfhce(1)/(ne(,)/1.0e20)+
*         cfhce(3)*te(,)/ev/abs(bb(,,3))/16
*     that specifies the anomalous electron heat conductivity, so that
*     the electron heat conduction coefficient is dfhce(,)*ne(,).
*     If cfhce(7) is not zero, the hce0 array is flux-scaled with
*     a scaling factor of cfhce(7)*(hy1(,)/hy1(ixref,))**2
*     where hy1 is the radial width of the cell and ixref is a reference
*     poloidal position (usually the outer midplane or 'jxa').
*     cfhce(2) and (4:6) are not used in the default version of the code.
*      cfsig - (0:7) real*8 array.
*     cfsig() are the coefficients in the local expression
*       dfsig(,) = cfsig(0)+cfsig(1)/(ne(,)/1.0e20)+
*         cfsig(3)*te(,)/ev/abs(bb(,,3))/16+
*         cfsig(4)*exp(-cfsig(5)*(y/L)**2)
*     that specifies the diffusivity corresponding to the anomalous
*     electrical conductivity, which is
*       sig(,) = qe*ne(,)*dfsig(,).
*     L is the local radial width of the plasma and
*     y is the radial distance from the inner (South) boundary.
*     If cfsig(7) is not zero, the sig0 array is flux-scaled with
*     a scaling factor of cfsig(7)*(hy1(,)/hy1(ixref,))**2
```

```
*     where hy1 is the radial width of the cell and ixref is a reference
*     poloidal position (usually the outer midplane or 'jxa').
*     cfsig(2) and (6) are not used in the default version of the code.
*      cfalf - (0:7) real*8 array.
*     cfalf() are the coefficients in the local expression
*       dfalf(,) = cfalf(0)+cfalf(1)/(ne(,)/1.0e20)+
*         cfalf(3)*te(,)/ev/abs(bb(,,3))/16
*     that specifies the diffusivity corresponding to the anomalous
*     thermo-electric coefficient, which is
*       alf(,) = ne(,)*sqrt(qe/te(,))*dfalf(,).
*     If cfalf(7) is not zero, the alf0 array is flux-scaled with
*     a scaling factor of cfalf(7)*(hy1(,)/hy1(ixref,))**2
*     where hy1 is the radial width of the cell and ixref is a reference
*     poloidal position (usually the outer midplane or 'jxa').
*     cfalf(2) and (4:6) are not used in the default version of the code.
*      cflim - (0:7) real*8 array.
*     cflim contains coefficients for flux limiting. cflim(0)=cflme
*     specifies the flux limit factor for poloidal electron heat
*     transport. cflim(1)=cflmi specifies the flux limit factor for
*     poloidal all atom heat transport. cflim(2)=cflmv specifies the
*     flux limit factor for poloidal atom momentum transport.
*     cflim(3)=cflal specifies the flux limit factor for the
*     thermo-electric coefficient.
*     cflim(4)=cflab specifies the flux limit factor for the
*     friction force.
*     Zero settings turn off these flux limits.
*     cflim(5:7) are not used in the default version of the code.
*
*   ..transport coefficients, differential
*
*     dna - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     (Let dnax(,,)=dna(,,0,) and dnay(,,)=dna(,,1,).)
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), dnax(ix,iy,is) and
*     dnay(ix,iy,is) specify the x-component and the y-component of
*     the diffusivity of species (is) on the (ix,iy) cell. The
*     corresponding term in the particle flux density is
*     -dna*grad.na .
*
*     dpa - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     (Let dpax(,,)=dpa(,,0,) and dpay(,,)=dpa(,,1,).)
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), dpax(ix,iy,is) and
*     dpay(ix,iy,is) specify the x-component and the y-component of
*     the diffusion coefficient (w.r.t. grad.pa) of species (is) on
*     the (ix,iy) cell. The corresponding term in the particle flux
*     density is  -dpa*grad.pa .
*
*     vla - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     (Let vlax(,,)=vla(,,0,) and vlay(,,)=vla(,,1,).)
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), vlax(ix,iy,is) specifies
*     the x-component of the anomalous velocity of species (is) on the
*     (ix,iy) cell.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), vlay(ix,iy,is) specifies
*     the y-component of the anomalous velocity of species (is) on the
*     (ix,iy) cell.
*
*     vsa - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     (Let vsax(,,)=vsa(,,0,) and vsay(,,)=vsa(,,1,).)
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), vsax(ix,iy,is) specifies
*     the x-component and vsay(ix,iy,is) the y-component of the
*     viscosity of species (is) on the (ix,iy) cell.
*
*     hce - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let hcex(,)=hce(,,0) and hcey(,)=hce(,,1).)
*     For (ix,iy) in (-1:nx,-1:ny), hcex(ix,iy) specifies the
*     x-component and hcey(ix,iy) the y-component of the electron
*     thermal conductivity on the (ix,iy) cell.
*
```

```
*     hci - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let hcix(,)=hci(,,0) and hciy(,)=hci(,,1).)
*     For (ix,iy) in (-1:nx,-1:ny), hcix(ix,iy) specifies the
*     x-component and hciy(ix,iy) the y-component of the all atom
*     thermal conductivity on the (ix,iy) cell.
*
*     sig - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let sigx(,)=sig(,,0) and sigy(,)=sig(,,1).)
*     For (ix,iy) in (-1:nx,-1:ny), sigx(ix,iy) specifies the
*     x-component and sigy(ix,iy) the y-component of the electrical
*     conductivity on the (ix,iy) cell.
*
*     alf - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let alfx(,)=alf(,,0) and alfy(,)=alf(,,1).)
*     For (ix,iy) in (-1:nx,-1:ny), alfx(ix,iy) specifies the
*     x-component and alfy(ix,iy) the y-component of the thermo-
*     electric coefficient on the (ix,iy) cell.
*
*     dna0 - (-1:nx,-1:ny,0:ns-1) real*8 array.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), dna0(ix,iy,is) specifies
*     the anomalous diffusivity with respect to grad(na) of species
*     (is) on the (ix,iy) cell.
*
*     dpa0 - (-1:nx,-1:ny,0:ns-1) real*8 array.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), dpa0(ix,iy,is) specifies
*     the anomalous diffusivity with respect to grad(pa) of species
*     (is) on the (ix,iy) cell.
*
*     vla0 - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     (Let vla0x(,,)=vla0(,,0,) and vla0y(,,)=vla0(,,1,).)
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), vla0x(ix,iy,is) specifies
*     the x-component and vla0y(ix,iy,is) the y-component of the
*     anomalous velocity of species (is) on the (ix,iy) cell.
*
*     vsa0 - (-1:nx,-1:ny,0:ns-1) real*8 array.
*     For (ix,iy,is) in (-1:nx,-1:ny,0:ns-1), vsa0(ix,iy,is) specifies
*     the anomalous viscosity of species (is) on the (ix,iy) cell.
*
*     hce0 - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), hce0(ix,iy) specifies the anomalous
*     electron thermal conductivity on the (ix,iy) cell.
*
*     hci0 - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), hci0(ix,iy) specifies the anomalous
*     atom thermal conductivity on the (ix,iy) cell.
*
*     sig0 - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), sig0(ix,iy) specifies the anomalous
*     electrical conductivity on the (ix,iy) cell.
*
*     alf0 - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), alf0(ix,iy) specifies the anomalous
*     thermo-electric coefficient on the (ix,iy) cell.
*
*    ..transport coefficients, discrete
*
*     ceqp - (-1:nx,-1:ny) real*8 array.
*     For (ix,iy) in (-1:nx,-1:ny), ceqp(ix,iy) specifies the energy
*     equipartition multiplier integrated over the (ix,iy) cell.
*     (The cell-integrated energy equipartition coefficient is
*     ceqp*ne*ni/te**1.5.)
*
*     cdna - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     (Let cdnax=cdna(,,0,) and cdnay=cdna(,,1,).)
*     For (ix,iy,is) in (0:nx,-1:ny,0:ns-1), cdnax(ix,iy,is) specifies
*     the particle diffusivity w.r.t. grad(na) of species (is) between
*     the (ix,iy) cell and its left neighbor. This coefficient enters into
```

```
*     the expression for the particle flux in the form
*       -cdnax(ix,iy,is)*(na(ix,iy,is)-na(ix-1,iy,is))
*     For ix.eq.-1, cdnax(ix,-1:ny,0:ns-1) will hold 0.
*     For (ix,iy,is) in (-1:nx,0:ny,0:ns-1), cdnay(ix,iy,is) specifies
*     the particle diffusivity w.r.t. grad(na) of species (is) between
*     the (ix,iy) cell and its bottom neighbor. This coefficient enters into
*     the expression for the particle flux in the form
*       -cdnay(ix,iy,is)*(na(ix,iy,is)-na(bottom(ix,iy),is))
*     For iy.eq.-1, cdnay(-1:nx,iy,0:ns-1) will hold 0.
*
*     cdpa - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     (Let cdpax=cdpa(,,0,) and cdpay=cdpa(,,1,).)
*     For (ix,iy,is) in (0:nx,-1:ny,0:ns-1), cdpax(ix,iy,is) specifies
*     the particle diffusivity w.r.t. grad(pa) of species (is) between
*     the (ix,iy) cell and its left neighbor. This coefficient enters into
*     the expression for the particle flux in the form
*       -cdpax(ix,iy,is)*(pa(ix,iy,is)-pa(ix-1,iy,is))
*     For ix.eq.-1, cdpax(ix,-1:ny,0:ns-1) will hold 0.
*     For (ix,iy,is) in (-1:nx,0:ny,0:ns-1), cdpay(ix,iy,is) specifies
*     the particle diffusivity w.r.t. grad(pa) of species (is) between
*     the (ix,iy) cell and its bottom neighbor. This coefficient enters into
*     the expression for the particle flux in the form
*       -cdpay(ix,iy,is)*(pa(ix,iy,is)-pa(bottom(ix,iy),is))
*     For iy.eq.-1, cdpay(-1:nx,iy,0:ns-1) will hold 0.
*
*     cvla - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     (Let cvlax=cvla(,,0,) and cvlay=cvla(,,1,).)
*     For (ix,iy,is) in (0:nx,-1:ny,0:ns-1), cvlax(ix,iy,is) specifies
*     the surface-integrated anomalous velocity of species (is) between
*     the (ix,iy) cell and its left neighbor. This coefficient enters into
*     the expression for the particle flux in the form
*       cvlax(ix,iy,is)*(na)
*     in which (na) involves some interpolation to cell faces.
*     For ix.eq.-1, cvlax(ix,-1:ny,0:ns-1) will hold 0.
*     For (ix,iy,is) in (-1:nx,0:ny,0:ns-1), cvlay(ix,iy,is) specifies
*     the surface-integrated anomalous velocity of species (is) between
*     the (ix,iy) cell and its bottom neighbor. This coefficient enters into
*     the expression for the particle flux in the form
*       cvlay(ix,iy,is)*(na)
*     in which (na) involves some interpolation to cell faces.
*     For iy.eq.-1, cvlay(-1:nx,iy,0:ns-1) will hold 0.
*
*     cvsa - (-1:nx,-1:ny,0:1,0:ns-1) real*8 array.
*     (Let cvsax=cvsa(,,0,) and cvsay=cvsa(,,1,).)
*     For (ix,iy,is) in (0:nx,-1:ny,0:ns-1), cvsax(ix,iy,is) specifies
*     the viscous conductance of species (is) between the (ix,iy) cell
*     and its left neighbor.
*     For ix.eq.-1, cvsax(ix,-1:ny,0:ns-1) will hold 0.
*     For (ix,iy,is) in (-1:nx,0:ny,0:ns-1), cvsay(ix,iy,is) specifies
*     the viscous conductance of species (is) between the (ix,iy) cell
*     and its bottom neighbor.
*     For iy.eq.-1, cvsay(-1:nx,iy,0:ns-1) will hold 0.
*
*     chce - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let chcex=chce(,,0) and chcey=chce(,,1).)
*     For (ix,iy) in (0:nx,-1:ny), chcex(ix,iy) specifies the electron
*     thermal conductance between the (ix,iy) cell and its left neighbor.
*     This coefficient enters into the expression for the electron
*     heat flux in the form
*       -chcex(ix,iy)*(te(ix,iy)-te(left(ix,iy))) .
*     For ix.eq.-1, chcex(ix,-1:ny) will hold 0.
*     For (ix,iy) in (-1:nx,0:ny), chcey(ix,iy) specifies the electron
*     thermal conductance between the (ix,iy) cell and its bottom neighbor.
*     This coefficient enters into the expression for the electron
*     heat flux in the form
*       -chcey(ix,iy)*(te(ix,iy)-te(bottom(ix,iy))) .
*     For iy.eq.-1, chcey(-1:nx,iy) will hold 0.
```

```
*
*     chve - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let chvex=chve(,,0) and chvey=chve(,,1).)
*     For (ix,iy) in (0:nx,-1:ny), chvex(ix,iy) specifies the electron
*     thermal strange velocity between the (ix,iy) cell and its left
*     neighbor. This coefficient enters into the expression for the
*     electron heat flux in the form  chvex*ne*te , interpolated to
*     cell faces in a suitable manner.
*     For ix.eq.-1, chvex(ix,-1:ny) will hold 0.
*     For (ix,iy) in (-1:nx,0:ny), chvey(ix,iy) specifies the electron
*     thermal strange velocity between the (ix,iy) cell and its bottom
*     neighbor. This coefficient enters into the expression for the
*     electron heat flux in the form  chvey*ne*te , interpolated to
*     cell faces in a suitable manner.
*     For iy.eq.-1, chvey(-1:nx,iy) will hold 0.
*
*     chci - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let chcix=chci(,,0) and chciy=chci(,,1).)
*     For (ix,iy) in (0:nx,-1:ny), chcix(ix,iy) specifies the all atom
*     thermal conductance between the (ix,iy) cell and its left neighbor.
*     This coefficient enters into the expression for the atom heat
*     flux in the form
*       -chcix(ix,iy)*(ti(ix,iy)-ti(left(ix,iy))) .
*     For ix.eq.-1, chcix(ix,-1:ny) will hold 0.
*     For (ix,iy) in (-1:nx,0:ny), chciy(ix,iy) specifies the all atom
*     thermal conductance between the (ix,iy) cell and its bottom neighbor.
*     This coefficient enters into the expression for the atom heat
*     flux in the form
*       -chciy(ix,iy)*(ti(ix,iy)-ti(bottom(ix,iy))) .
*     For iy.eq.-1, chciy(-1:nx,iy) will hold 0.
*
*     chvi - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let chvix=chvi(,,0) and chviy=chvi(,,1).)
*     For (ix,iy) in (0:nx,-1:ny), chvix(ix,iy) specifies the all atom
*     thermal strange velocity between the (ix,iy) cell and its left
*     neighbor. This coefficient enters into the expression for the
*     atom heat flux in the form  chvix*ni*ti , interpolated to cell
*     faces in a suitable manner.
*     For ix.eq.-1, chvix(ix,-1:ny) will hold 0.
*     For (ix,iy) in (-1:nx,0:ny), chviy(ix,iy) specifies the all atom
*     thermal strange velocity between the (ix,iy) cell and its bottom
*     neighbor. This coefficient enters into the expression for the
*     atom heat flux in the form  chviy*ni*ti , interpolated to
*     cell faces in a suitable manner.
*     For iy.eq.-1, chviy(-1:nx,iy) will hold 0.
*
*     csig - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let csigx=csig(,,0) and csigy=csig(,,1).)
*     For (ix,iy) in (0:nx,-1:ny), csigx(ix,iy) specifies the electrical
*     conductance between the (ix,iy) cell and its left neighbor.
*     For ix.eq.-1, csigx(ix,-1:ny) will hold 0.
*     For (ix,iy) in (-1:nx,0:ny), csigy(ix,iy) specifies the electrical
*     conductance between the (ix,iy) cell and its bottom neighbor.
*     For iy.eq.-1, csigx(-1:nx,iy) will hold 0.
*
*     calf - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let calfx=calf(,,0) and calfy=calf(,,1).)
*     For (ix,iy) in (0:nx,-1:ny), calfx(ix,iy) specifies the thermo-
*     electric coefficient between the (ix,iy) cell and its left neighbor.
*     For ix.eq.-1, calfx(ix,-1:ny) will hold 0.
*     For (ix,iy) in (-1:nx,0:ny), calfy(ix,iy) specifies the thermo-
*     electric coefficient between the (ix,iy) cell and its bottom neighbor.
*     For iy.eq.-1, calfy(-1:nx,iy) will hold 0.
*
*     csigin - (-1:nx,-1:ny,0:1,0:ns-1,0:ns-1) real*8 array.                !xpb
*     (Let csiginx=csigin(,,0,,) and csiginy=csigin(,,1,,).)
*     For (ix,iy,is,is0) in (0:nx,-1:ny,0:ns-1,0:ns-1),
```

```
*     csiginx(ix,iy,is,is0) specifies the ion-neutral
*     conductivity between the (ix,iy) cell and its left neighbor, where
*     the ion species is species is and the neutral species is is0.
*     For ix.eq.-1, csiginx(ix,-1:ny) will hold 0.
*     For (ix,iy) in (-1:nx,0:ny), csiginy(ix,iy) specifies the ion-neutral
*     conductivity between the (ix,iy) cell and its bottom neighbor, where
*     the ion species is species is and the neutral species is is0.
*     For iy.eq.-1, csiginx(-1:nx,iy) will hold 0.
*     When is is a neutral species or is0 is an ionised species,
*     csigin(,,,is,is0) will be zero.
*
*     csig_cl - (-1:nx,-1:ny,0:1) real*8 array.                              !xpb
*     (Let csig_clx=csig_cl(,,0) and csig_cly=csig_cl(,,1).)
*     For (ix,iy) in (0:nx,-1:ny), csig_clx(ix,iy) specifies the classical
*     conductivity between the (ix,iy) cell and its left neighbor.
*     For ix.eq.-1, csig_clx(ix,-1:ny) will hold 0.
*     For (ix,iy) in (-1:nx,0:ny), csig_cly(ix,iy) specifies the classical
*     conductivity between the (ix,iy) cell and its bottom neighbor.
*     For iy.eq.-1, csig_clx(-1:nx,iy) will hold 0.
*
*     calf_cl - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let calf_clx=calf_cl(,,0) and calf_cly=calf_cl(,,1).)
*     For (ix,iy) in (0:nx,-1:ny), calf_clx(ix,iy) specifies the
*     classical part of the thermo-electric coefficient
*     between the (ix,iy) cell and its left neighbor.
*     For ix.eq.-1, calf_clx(ix,-1:ny) will hold 0.
*     For (ix,iy) in (-1:nx,0:ny), calf_cly(ix,iy) specifies the
*     classical part of the thermo-electric coefficient
*     between the (ix,iy) cell and its bottom neighbor.
*     For iy.eq.-1, calf_cly(-1:nx,iy) will hold 0.
*
*     csig_an - (-1:nx,-1:ny,0:1) real*8 array.                              !xpb
*     (Let csig_anx=csig_an(,,0) and csig_any=csig_an(,,1).)
*     For (ix,iy) in (0:nx,-1:ny), csig_anx(ix,iy) specifies the anomalous
*     conductivity between the (ix,iy) cell and its left neighbor.
*     For ix.eq.-1, csig_anx(ix,-1:ny) will hold 0.
*     For (ix,iy) in (-1:nx,0:ny), csig_any(ix,iy) specifies the anomalous
*     conductivity between the (ix,iy) cell and its bottom neighbor.
*     For iy.eq.-1, csig_anx(-1:nx,iy) will hold 0.
*
*     calf_an - (-1:nx,-1:ny,0:1) real*8 array.
*     (Let calf_anx=calf_an(,,0) and calf_any=calf_an(,,1).)
*     For (ix,iy) in (0:nx,-1:ny), calf_anx(ix,iy) specifies the
*     anomalous part of the thermo-electric coefficient
*     between the (ix,iy) cell and its left neighbor.
*     For ix.eq.-1, calf_anx(ix,-1:ny) will hold 0.
*     For (ix,iy) in (-1:nx,0:ny), calf_any(ix,iy) specifies the
*     anomalous part of the thermo-electric coefficient
*     between the (ix,iy) cell and its bottom neighbor.
*     For iy.eq.-1, calf_any(-1:nx,iy) will hold 0.
*
*    ..parameters for core numerical iteration
*
*     ntim - integer.
*     ntim specifies the number of timesteps to be performed.
*
*     itim - integer.
*     itim specifies the timestep counter. It is initialised to 0 and
*     incremented by 1 following each successful call to b2mndt.
*
*     dtim - real.
*     dtim specifies the timestep for the present call to b2mndt.
*     It will hold that 0.lt.dtim.
*
*     itcnt - integer.
*     itcnt counts the number of inner iterations. It is initialised to
*     0 and incremented by 1 following each call to b2news.
```

```
*
*     nstg - (0:2) integer array.
*     nstg(0:2) specifies the maximum number of iterations for each of
*     the three main loops in the subroutine b2mndt.
*
*     rxf - real.
*     rxf specifies the main under-relaxation parameter. Computed
*     corrections to the unknowns are scaled by a factor rxf before
*     being applied. It will hold that 0.le.rxf.le.1.
*     Should be set to 1 when running in time-dependent mode.
*
*     rxg - real.
*     rxg specifies a special under-relaxation parameter. A term of the
*     form "abs(residual)/rxg" is added to the diagonal of the matrix
*     of the correction equation, with the effect of limiting the
*     computed correction. rxg is dimensionless and of order unity;
*     smaller values of rxg imply stronger damping.
*     Not used when running in time-dependent mode.
*
*    ..variables for input-output
*
*     nrid - integer.
*     nrid specifies a unit number for formatted input.
*
*     nget - integer.
*     nget specifies a unit number for un*formatted input.
*
*     nprt - integer.
*     nprt specifies a unit number for formatted output.
*
*     nput - integer.
*     nput specifies a unit number for un*formatted output.
*
*     idout - character*(*).
*     idout specifies the desired selection of output segments for
*     printed and graphical output. Associated with each segment is a
*     four-character word, which is identified on the printed output.
*     (The same word may be associated with several segments.)
*     A particular segment will be produced only if its associated word
*     has a match, as defined by the routine strmas, in idout.
*
*     /b2cmwg/ - common block.
*     /b2cmwg/ contains some parameters to control graphical output.
*      nmbpmx - integer constant.
*     nmbpmx is used in dimension declarations instead of nmbpcc or
*     nmbpdm.
*      nmbpcc - integer.
*     nmbpcc specifies the number of cartesian domains on which any
*     quantity may be plotted.
*      nmbpdm - integer.
*     nmbpdm specifies the number of physical domains on which any
*     quantity may be plotted.
*      cwaxr, cwayr - (0:nmbpmx-1) real*8 array.
*     For each iplot in (0:nmbpcc-1), cwaxr(iplot) and cwayr(iplot)
*     specify coordinates for a cartesian domain plot. The coordinates
*     are obtained by integrating the grid metric along two reference
*     lines. The reference line for the x-coordinate is at position
*     nint(cwaxr*nx), and the reference line for the y-coordinate is
*     at nint(cwayr*ny). It will hold that 0.le.cwaxr.le.1 and
*     0.le.cwayr.le.1.
*      cwxmin, cwxmax, cwymin, cwymax - (0:nmbpmx-1) real*8 array.
*     For each iplot in (0:nmbpdm-1), the quadruple of cwxmin(iplot)
*     through cwymax(iplot) specifies a domain in the physical plane
*     for contour plots.
*
*    ..workspace
*
```

```
*     The b2mod_stack module contains two stacks, one for real*8 and
*     one for integer workspace.
*     They are used to provide local dynamic arrays within
*     the rules of Fortran 77.
*      nstckw - integer constant.
*     nstckw specifies the size of the real*8 workspace array w. If
*     nstckw is too small, the program will abort via a call to
*     xertst/xerrab.
*      nstcwi - integer constant.
*     nstcwi specifies the size of the integer workspace array wi. If
*     nstcwi is too small, the program will abort via a call to
*     xertst/xerrab.
*      w - (0:nstckw-1) real*8 array.
*     w is used as a stack for local dynamic real*8 arrays. Base
*     addresses for arrays in this stack are to be maintained by
*     appropriate calls to the subroutines subini, subend, and lwget.
*      wi - (0:nstcwi-1) integer array.
*     wi is used as a stack for local dynamic integer arrays. Base
*     addresses for arrays in this stack are to be maintained by
*     appropriate calls to the subroutines subini, subend, and lwiget.
*
*   ..some other variables
*
*     lblgm - character*120.
*     lblgm identifies the instance of the program employed to prepare
*     the geometry.
*
*     lblcp - character*120.
*     lblcp identifies the instance of the program employed to prepare
*     the table of physics parameters.
*
*     lblps - character*120.
*     lblps identifies the instance of the program employed to prepare
*     the initial state.
*
*     lblrc - character*120.
*     lblrc identifies the instance of the program employed to prepare
*     the table of atomic rate coefficients.
*
*     lblsr - character*120.
*     lblsr identifies the instance of the program employed to prepare
*     the neutral gas source data.
*
*     lblmn - character*120.
*     lblmn identifies the instance of the program employed for the main
*     calculation.
*
*     ncall - integer.
*     ncall counts the number of calls made to the present routine.
*
*
*  2. Notes
*
*     Variables local to a subroutine are not expected to be preserved
*     between calls unless they are declared in a 'save' statement.
*     Variables in common are not expected to be preserved between
*     calls to the outermost routine in which the common block appears.
*     (See ANSI X3.9-1978 Fortran 77, Section 17.3.6d.) The code may
*     be compiled and loaded under an option that initialises all
*     quantities not mentioned in a data statement to an invalid
*     numerical value.
*
*
*-----------------------------------------------------------------------
*.end b2cdcv

      end subroutine b2cdcv
```

A.4 b2cdci.F

```
*-----------------------------------------------------------------------
* Generated by b2cdci.py from b2input.xml. DO NOT EDIT THIS FILE!
*.specification

      subroutine b2cdci ()
      implicit none

*-----------------------------------------------------------------------
*.documentation
*
*  1. purpose
*
*    B2CDCI provides information about internal parameters of the B2.5
*    code. No computation is performed in this routine.
*
*
*-----------------------------------------------------------------------
*.text
*
*  0. Overview and naming conventions.
*
*    Here we list the switches in alphabetical order. By convention, a
*    switch name starts with the name of the routine in which it is used.
*    Unless noted otherwise, the desired switches are to be declared at the
*    end of the b2mn.dat input file.
*    Integer switches are given integer default values, etc...
*    Some switches are used by several routines and may thus also appearin
*    routines other than the one of which they carry the name.
*    The basic philosophy behind the assignment of the default values is
*    that the default values do not change the expected standard behaviour
*    of the solution and solution method. In most cases, the code checks
*    that a switch modified by the user is within some expected range, and
*    stops otherwise. Setting a switch to its default value will not stop
*    the code.
*
*    The switches are divided into 6 broad categories : Geometry, Physics,
*    Run, Output, Numerics and Atomic Physics. The following section will
*    describe each category separately. Within categories, an effort is
*    made to list the most relevant and/or common switches first, and
*    rarely used or esoteric ones last.
*
*    Geometry switches refer to the plasma geometry. These usually will
*    need to be set up only once for a family of runs. The user should be
*    familiar with most of these.
*
*    Run switches refer to parameters fo the code run. These are likely to
*    change for every run and the user should know how to use them. There
*    is some overlap between Physics and Run switches, and such cases will
*    be explained below individually.
*
*    Physics switches refer to the physics terms in the code. These are
*    likely to be different within a family of runs. The user should be
*    familiar with the ones relevant to the case being studied.
*
*    Output switches refer to the level and type of output. A few switches
*    are standard, but most need only be activated for specific debugging
*    purposes.
*
*    Numerics switches refer to the numerical treatment of the equations.
*    Most of these switches should only be modified with the most extreme
*    caution. The user need be concerned with only very few of these.
*
*    Atomic Physics includes switches used in the atomic physics pre-
*    processing module B2AR.
```

```
*
*
*    Switch name:               Default value:         Category:
*  ------------------------------------------------------------------
*    'adas_extrap'              '1'                    Atomic Physics
*    'ank_tracing'              '0'                    Output
*    'b2agdr_nxiso1'            '-2'                   Geometry
*    'b2agdr_nxiso2'            '-2'                   Geometry
*    'b2agdr_nyiso1'            '-2'                   Geometry
*    'b2agdr_nyiso2'            '-2'                   Geometry
*    'b2agdr_redef_gmtry'       '1'                    Geometry
*    'b2agdr_redef_qc'          '0'                    Geometry
*    'b2agfs_Bt_adjust'         '0'                    Geometry
*    'b2agfs_Bt_rescale'        '1.0'                  Geometry
*    'b2agfs_Bt_reversal'       '0'                    Geometry
*    'b2agfs_bottomcut'         See description (integer) Geometry
*    'b2agfs_bottomcut2'        See description (integer) Geometry
*    'b2agfs_geom_match_dist'   '1.0e-6'               Geometry
*    'b2agfs_geometry'          'upgrade.geometry'     Geometry
*    'b2agfs_leftcut'           See description (integer) Geometry
*    'b2agfs_leftcut2'          See description (integer) Geometry
*    'b2agfs_lineardevice'      linear.geometry        Geometry
*    'b2agfs_min_pitch'         '1.0'                  Geometry
*    'b2agfs_nncut'             See description (integer) Geometry
*    'b2agfs_periodic_bc'       '0'                    Geometry
*    'b2agfs_pit_rescale'       '1.0'                  Geometry
*    'b2agfs_rightcut'          See description (integer) Geometry
*    'b2agfs_rightcut2'         See description (integer) Geometry
*    'b2agfs_topcut'            See description (integer) Geometry
*    'b2agfs_topcut2'           See description (integer) Geometry
*    'b2agfs_xoffset'           '0.0'                  Geometry
*    'b2agfs_xrescale'          '1.0'                  Geometry
*    'b2agfs_yoffset'           '0.0'                  Geometry
*    'b2agfs_yrescale'          '1.0'                  Geometry
*    'b2agmt_ld_width'          '1.0'                  Geometry
*    'b2agmt_qc_style'          '1'                    Geometry
*    'b2agmx_pbs_from_basis_mesh' '1'                  Geometry
*    'b2agsi_isymm'             '1'                    Geometry
*    'b2aidr_read_b2fstate'     '0'                    Run
*    'b2ardr_fix_cx'            '1'                    Atomic Physics
*    'b2ardr_fix_recomb'        '0'                    Atomic Physics
*    'b2ardr_no_smoothing'      '0'                    Atomic Physics
*    'b2ardr_no_weisheit'       '0'                    Atomic Physics
*    'b2ardr_rtnn'              '32'                   Atomic Physics
*    'b2ardr_rtnt'              '40'                   Atomic Physics
*    'b2ardr_zmax_recomb'       '1'                    Atomic Physics
*    'b2mndr_astra'             '0'                    Run
*    'b2mndr_atomic_physics_rescale' '0'               Physics
*    'b2mndr_av_read'           '0'                    Output
*    'b2mndr_b2time'            '1'                    Output
*    'b2mndr_cdf_default'       '0'                    Output
*    'b2mndr_cdfmovietim'       '0.0'                  Output
*    'b2mndr_coronal_model'     '0'                    Physics
*    'b2mndr_cpu'               '0.0'                  Run
*    'b2mndr_database'          $(DEVICE)              Run
*    'b2mndr_delta_max'         '0.0'                  Run
*    'b2mndr_delta_min'         '0.0'                  Run
*    'b2mndr_density_rescale'   '1.0'                  Run
*    'b2mndr_device'            $(DEVICE)              Run
*    'b2mndr_dpc_mod_rates_ne_hot_frac' '0.0'          Atomic Physics
*    'b2mndr_dpc_mod_rates_te_hot' '0.0'               Atomic Physics
*    'b2mndr_dt_change_dec'     '1.0'                  Run
*    'b2mndr_dt_change_inc'     '1.0'                  Run
*    'b2mndr_dt_max'            '1.0e+01'              Run
*    'b2mndr_dt_min'            '1.0e-30'              Run
*    'b2mndr_dtim'              '1.0'                  Run
*    'b2mndr_eirene'            '0'                    Run
```

```
*     'b2mndr_elapsed'           '0.0'                  Run
*     'b2mndr_etim'              '0.0'                  Run
*     'b2mndr_hz'                '1.0'                  Physics
*     'b2mndr_idout0'            'pgnl;pgmm;pzmm'       Output
*     'b2mndr_idout1'            'pzmm'                 Output
*     'b2mndr_ids_av'            '0'                    Output
*     'b2mndr_ids_path'          See description        Run
*     'b2mndr_ids_save'          '0'                    Output
*     'b2mndr_ids_time'          '0.0'                  Output
*     'b2mndr_inverse_ua'        '0'                    Output
*     'b2mndr_isfb'              ismain                 Numerics
*     'b2mndr_ismain'            '1'                    Run
*     'b2mndr_ixfb'              See description        Numerics
*     'b2mndr_iyfb'              ny/2                   Numerics
*     'b2mndr_min_aresco'        '0.0'                  Numerics
*     'b2mndr_min_areshe'        '0.0'                  Numerics
*     'b2mndr_min_areshi'        '0.0'                  Numerics
*     'b2mndr_mvinc'             '1'                    Output
*     'b2mndr_mvnum'             '0'                    Output
*     'b2mndr_na_eps'            '1.0e19'               Output
*     'b2mndr_na_min'            '1.0e4'                Numerics
*     'b2mndr_na_new'            '1.0e14'               Numerics
*     'b2mndr_ne_wanted'         '0.0'                  Numerics
*     'b2mndr_ne_wanted_time'    '0.0'                  Numerics
*     'b2mndr_ntim'              '1'                    Run
*     'b2mndr_ntim_save'         '0'                    Output
*     'b2mndr_old_style'         '0'                    Output
*     'b2mndr_plasinc'           '1'                    Output
*     'b2mndr_plasmatim'         '0.0'                  Output
*     'b2mndr_plasnum'           '0'                    Output
*     'b2mndr_po_eps'            '1.0e+1'               Output
*     'b2mndr_pulse_number'      '0'                    Run
*     'b2mndr_rescale_neutrals'  '1.0'                  Run
*     'b2mndr_rescale_neutrals_sources' '1.0'           Run
*     'b2mndr_run_number'        '0'                    Run
*     'b2mndr_savecpu'           '3600.0'               Run
*     'b2mndr_shot_number'       '0'                    Run
*     'b2mndr_stim'              '0.0'                  Run
*     'b2mndr_tally'             '1'                    Output
*     'b2mndr_te_eps'            '1.0e+1'               Output
*     'b2mndr_ti_eps'            '1.0e+1'               Output
*     'b2mndr_trantim'           '0.0'                  Output
*     'b2mndr_ua_eps'            '1.0e+4'               Output
*     'b2mndr_user'              $(USER)                Run
*     'b2mndr_zn_rescale'        '0'                    Run
*     'b2mndt_av'                '0'                    Output
*     'b2mndt_av_batch_all'      '0'                    Output
*     'b2mndt_av_continue'       '1'                    Output
*     'b2mndt_av_ntim_batch'     '500'                  Output
*     'b2mndt_av_ntim_run'       '1000'                 Output
*     'b2mndt_density_control'   '0'                    Run
*     'b2mndt_moitlv'            '-1'                   Output
*     'b2mndt_moqtlv'            '3'                    Output
*     'b2mndt_nstg0'             '1'                    Run
*     'b2mndt_nstg1'             '1'                    Run
*     'b2mndt_nstg2'             '1'                    Run
*     'b2mndt_nstg_aresco'       '0.0'                  Numerics
*     'b2mndt_nstg_areshe'       '0.0'                  Numerics
*     'b2mndt_nstg_areshi'       '0.0'                  Numerics
*     'b2mndt_ntim_step_out'     '1'                    Numerics
*     'b2mndt_rxf'               See description        Numerics
*     'b2mndt_style'             '1'                    Numerics
*     'b2mndt_use_b2srst'        '1'                    Numerics
*     'b2mwqt_style'             '1'                    Output
*     'b2mwti_2dwrite'           '0'                    Output
*     'b2mwti_ismain0'           '0'                    Output
*     'b2mwti_jsep'              See description (integer) Geometry
```

```
*    'b2mwti_jxa'               See description (integer) Geometry
*    'b2mwti_jxi'               See description (integer) Geometry
*    'b2mwti_target_offset'     '1'                       Output
*    'b2news_BoRiS'             '0.0'                     Physics
*    'b2news_ExB'               '0.0'                     Physics
*    'b2news_area_fix'          '3'                       Numerics
*    'b2news_do_2nd_b2npco_call' '0'                      Numerics
*    'b2news_facExB_dec'        '0.0'                     Run
*    'b2news_facExB_inc'        '1.0'                     Run
*    'b2news_facExB_start'      '0.0'                     Run
*    'b2news_facExB_target'     '0.0'                     Run
*    'b2news_fac_ExB_tanh_a'    '0.0'                     Numerics
*    'b2news_fac_ExB_tanh_b'    '0.0'                     Numerics
*    'b2news_fac_ref'           See description (integer) Numerics
*    'b2news_fac_vis_tanh_a'    '0.0'                     Numerics
*    'b2news_fac_vis_tanh_b'    '0.0'                     Numerics
*    'b2news_facdrift_dec'      '0.0'                     Run
*    'b2news_facdrift_inc'      '1.0'                     Run
*    'b2news_facdrift_start'    '0.0'                     Run
*    'b2news_facdrift_tanh_a'   '0.0'                     Numerics
*    'b2news_facdrift_tanh_b'   '0.0'                     Numerics
*    'b2news_facdrift_target'   '0.0'                     Run
*    'b2news_facvis_dec'        '0.0'                     Run
*    'b2news_facvis_inc'        '1.0'                     Run
*    'b2news_facvis_start'      '0.0'                     Run
*    'b2news_facvis_target'     '0.0'                     Run
*    'b2news_guard_flows'       '2'                       Numerics
*    'b2news_iy_nocoreExB'      '-2'                      Numerics
*    'b2news_l_enebound'        '0'                       Numerics
*    'b2news_l_fbound'          '0'                       Numerics
*    'b2news_l_timbound'        '0'                       Numerics
*    'b2news_ncallout'          '-1'                      Output
*    'b2news_no_b2sral_call'    '1'                       Numerics
*    'b2news_no_solve'          '0'                       Run
*    'b2news_nsmax'             ns                        Numerics
*    'b2news_nsmin'             '0'                       Numerics
*    'b2news_poteq'             '1'                       Numerics
*    'b2news_potit'             '50'                      Numerics
*    'b2news_potitmin'          '0'                       Numerics
*    'b2news_potok'             '1.0e-2'                  Numerics
*    'b2news_ramp_slow'         '0'                       Numerics
*    'b2news_re_eval_prtls_fluxes' '0'                    Numerics
*    'b2news_recalculate_contributions' '1'               Numerics
*    'b2news_vis'               '0.0'                     Physics
*    'b2news_xfm0'              '1.0'                     Numerics
*    'b2news_xfm1'              '1.0'                     Numerics
*    'b2news_xfm2'              '1.0'                     Numerics
*    'b2news_xfm3'              '1.0'                     Numerics
*    'b2npco_pcm0'              '1.0'                     Numerics
*    'b2npco_pcm1'              '1.0'                     Numerics
*    'b2npco_rxg'               '1.0'                     Numerics
*    'b2nph9_style'             '1'                       Numerics
*    'b2npht_pcm0'              '1.0'                     Numerics
*    'b2npht_pcm1'              '1.0'                     Numerics
*    'b2npht_rxg'               '1.0'                     Numerics
*    'b2npht_style'             '1'                       Numerics
*    'b2npmo mass multiplicator' '1.0'                    Physics
*    'b2npmo_b2sifr_'           '1'                       Physics
*    'b2npmo_impr_form_fr'      '0'                       Physics
*    'b2npmo_impr_form_tf'      '0'                       Physics
*    'b2npmo_ion_vlct_restrict' '0'                       Numerics
*    'b2npmo_rxg'               '1.0e6'                   Numerics
*    'b2npmo_smbvi_factor'      '1.0'                     Physics
*    'b2npmo_vlct_diagno'       '1'                       Output
*    'b2nxdv_style'             '1'                       Numerics
*    'b2nxfc_style'             '1'                       Numerics
*    'b2nxfx_style'             '1'                       Numerics
```

```
*    'b2sdia_facgt'             '0.0'                Physics
*    'b2sian_phm0'              '1.0'                Physics
*    'b2siav_addvis'            '1.0'                Physics
*    'b2siav_addvis1'           '1.0'                Physics
*    'b2siav_style_qip'         '0'                  Physics
*    'b2sicf_phm0'              '1.0'                Physics
*    'b2sicf_phm1'              '1.0'                Physics
*    'b2sifr_Zhdanov_test'      '0'                  Physics
*    'b2sifr_Zhdanov_tf_fr'     '1'                  Physics
*    'b2sifr_limthee'           '0.3'                Physics
*    'b2sifr_limthii'           '0.3'                Physics
*    'b2sifr_phm0'              '1.0'                Physics
*    'b2sifr_phm1'              '1.0'                Physics
*    'b2sifr_phm2'              '1.0'                Physics
*    'b2sifr_phm3'              '1.0'                Physics
*    'b2sifr_styl0'             '0'                  Numerics
*    'b2sifr_styl1'             '0'                  Numerics
*    'b2sigp_pressure_restriction' '1'               Numerics
*    'b2sigp_style'             '2'                  Numerics
*    'b2sihs_istyle_Joule_heating' '1'               Physics
*    'b2sihs_phm0'              '1.0'                Physics
*    'b2sihs_phm1'              '1.0'                Physics
*    'b2sihs_phm2'              '1.0'                Physics
*    'b2sihs_phm3'              '1.0'                Physics
*    'b2sihs_phm4'              '1.0'                Physics
*    'b2sihs_phm5'              '1.0'                Physics
*    'b2sihs_phm6'              '1.0'                Physics
*    'b2sihs_phm7'              '0.0'                Physics
*    'b2sihs_phm8'              '1.0'                Physics
*    'b2sihs_rf0'               '1.0'                Numerics
*    'b2sihs_rf1'               '1.0'                Numerics
*    'b2sihs_rf2'               '1.0'                Numerics
*    'b2sihs_rf3'               '1.0'                Numerics
*    'b2sihs_rf4'               '1.0'                Numerics
*    'b2sihs_style'             '0'                  Numerics
*    'b2sqcx_phm0'              '1.0'                Physics
*    'b2sqcx_styl0'             '0'                  Physics
*    'b2sqel_artificial_radiation' '0.0'             Physics
*    'b2sqel_phm0'              '1.0'                Physics
*    'b2sqel_phm1'              '1.0'                Physics
*    'b2sqel_phm2'              '1.0'                Physics
*    'b2sral_inputfile'         '0'                  Run
*    'b2sral_style'             '2'                  Physics
*    'b2srdt_numerics_namelist' '0'                  Run
*    'b2srdt_phm0'              '1.0'                Numerics
*    'b2srdt_phm1'              '1.0'                Numerics
*    'b2srdt_phm3'              '1.0'                Numerics
*    'b2srdt_phm4'              '0.0'                Numerics
*    'b2srdt_phm5'              '1.0'                Numerics
*    'b2srsm_diagno'            '0'                  Output
*    'b2srsm_enable'            '0'                  Numerics
*    'b2srst_rf0'               '1.0'                Numerics
*    'b2srst_rf1'               '1.0'                Numerics
*    'b2srst_rf2'               '1.0'                Numerics
*    'b2srst_rf3'               '1.0'                Numerics
*    'b2stbc_bc_ref'            '0.01'               Physics
*    'b2stbc_bc_ref_te'         '0.01'               Physics
*    'b2stbc_bc_ref_ti'         '0.01'               Physics
*    'b2stbc_bcene_15_style'    '1'                  Physics
*    'b2stbc_bcene_16_style'    '0'                  Physics
*    'b2stbc_bceni_16_style'    '0'                  Physics
*    'b2stbc_bcpot_16_step'     '50'                 Numerics
*    'b2stbc_boundary_namelist' See description      Run
*    'b2stbc_cbc'               '1.0'                Numerics
*    'b2stbc_cbsnafac'          '0.01'               Numerics
*    'b2stbc_coreregn2'         '4'                  Geometry
*    'b2stbc_coreregno'         See description      Geometry
```

```
*    'b2stbc_diagno'            '0'                    Output
*    'b2stbc_fchy_dia'          '0.0'                  Numerics
*    'b2stbc_fchy_dia_coreonly' '1'                    Numerics
*    'b2stbc_fchycore'          '-1.0e30'              Physics
*    'b2stbc_fchycore_alpha'    '0.1'                  Numerics
*    'b2stbc_feedback'          '0'                    Run
*    'b2stbc_fheycore'          '0.0'                  Physics
*    'b2stbc_fheycore_alpha'    '0.1'                  Numerics
*    'b2stbc_fhiycore'          '0.0'                  Physics
*    'b2stbc_fhiycore_alpha'    '0.1'                  Numerics
*    'b2stbc_fhiycore_kinetic_energy' '0'              Physics
*    'b2stbc_fix_fch_in_fhe_sheath' '2'                Numerics
*    'b2stbc_fnaycore'          '-1.0e30'              Physics
*    'b2stbc_fnaycore_alpha'    '0.1'                  Numerics
*    'b2stbc_integral_current'  '0.0'                  Numerics
*    'b2stbc_isfeedback'        '0'                    Physics
*    'b2stbc_istyle_cur_contr_on_S_and_N' '2'          Numerics
*    'b2stbc_istyle_fchi'       '0'                    Numerics
*    'b2stbc_iyped'             jsep/2                 Physics
*    'b2stbc_ncallfeedback'     '0'                    Run
*    'b2stbc_ndes'              '0.0'                  Physics
*    'b2stbc_ndes_sol'          '0.0'                  Physics
*    'b2stbc_neoclassical'      '0.0'                  Numerics
*    'b2stbc_nepedm_sol'        '0.0'                  Physics
*    'b2stbc_nesepm'            '0.0'                  Physics
*    'b2stbc_nesepm_alpha'      '1.0e-3'               Numerics
*    'b2stbc_nesepm_beta'       '0.0'                  Numerics
*    'b2stbc_nesepm_gamma'      '0.99'                 Numerics
*    'b2stbc_nesepm_maxpuff'    '0.0'                  Physics
*    'b2stbc_nesepm_minpuff'    '0.0'                  Physics
*    'b2stbc_nesepm_overshoot'  '0.0'                  Physics
*    'b2stbc_nesepm_pfr'        '0.0'                  Physics
*    'b2stbc_nesepm_sol'        '0.0'                  Physics
*    'b2stbc_pfrregno1'         '0'                    Geometry
*    'b2stbc_pfrregno2'         '2'                    Geometry
*    'b2stbc_private_flux_puff' '0.0'                  Physics
*    'b2stbc_secmodel'          '0'                    Physics
*    'b2stbc_she0ep'            '1.0e-36'              Numerics
*    'b2stbc_sheath_drift_fix'  '1'                    Numerics
*    'b2stbc_shi0ep'            '1.0e-36'              Numerics
*    'b2stbc_sna0ep'            '1.0e-36'              Numerics
*    'b2stbc_solregno'          '3'                    Geometry
*    'b2stbc_stab_coeff_sheath_ni' '0.0'               Numerics
*    'b2stbc_stab_coeff_sheath_te' '0.0'               Numerics
*    'b2stbc_stab_coeff_sheath_ti' '0.0'               Numerics
*    'b2stbc_type13_fac'        '1.0'                  Numerics
*    'b2stbc_type13_norm'       '1.0e15'               Numerics
*    'b2stbc_type13_ref'        '1'                    Physics
*    'b2stbc_type16_kinetic_energy' '0'                Physics
*    'b2stbc_type16_ref'        '1'                    Physics
*    'b2stbc_type20_ref'        '1'                    Physics
*    'b2stbc_type21_ref'        '1'                    Physics
*    'b2stbc_volrec'            '0.0'                  Physics
*    'b2stbc_volrec_alpha'      '1.0e-3'               Numerics
*    'b2stbc_volrec_beta'       '1.0'                  Numerics
*    'b2stbc_volrec_overshoot'  '0.0'                  Physics
*    'b2stbm_impgyro_mod'       '0'                    Numerics
*    'b2stbm_linearisation'     '1.0'                  Numerics
*    'b2stbr_alpha'             '0.25'                 Physics
*    'b2stbr_b2wall_netcdf'     '0'                    Output
*    'b2stbr_bas_recycled_neutrals_contr' '1.0'        Physics
*    'b2stbr_c_pumpavr'         '0.02'                 Numerics
*    'b2stbr_core_sources_rescale' '1.0'               Run
*    'b2stbr_eir_src_nhist'     '1'                    Numerics
*    'b2stbr_first_flight'      '0'                    Run
*    'b2stbr_first_flight_dl'   '0.001'                Numerics
*    'b2stbr_first_flight_no_of_flights' '9'           Numerics
```

```
*    'b2stbr_first_flight_no_of_start_points' 2*(nx+2)+2*max(nncut,1)*(ny+2) Geometry
*    'b2stbr_first_flight_table_size' '200000'         Numerics
*    'b2stbr_neutrals_namelist' '0'                    Run
*    'b2stbr_output'            '0'                    Output
*    'b2stbr_plate_model'       '0'                    Physics
*    'b2stbr_plate_option'      '3'                    Physics
*    'b2stbr_plate_temp'        '300.0'                Physics
*    'b2stbr_plate_thick'       '0.00'                 Physics
*    'b2stbr_potential_at_guard_cell' '1'              Numerics
*    'b2stbr_redep_alpha'       '0.00'                 Physics
*    'b2stbr_refl_model'        '1'                    Physics
*    'b2stbr_reflection_on'     '1'                    Physics
*    'b2stbr_sna_corr'          '0.0'                  Numerics
*    'b2stbr_sput_chem_alpha'   '1.0e-2'               Numerics
*    'b2stbr_sput_chem_cutoff_alpha' '1.0'             Physics
*    'b2stbr_sput_chem_cutoff_beta' '3.0'              Physics
*    'b2stbr_sput_chem_model'   '0'                    Physics
*    'b2stbr_sput_dst'          '-1'                   Physics
*    'b2stbr_sput_dst2'         '-1'                   Physics
*    'b2stbr_sput_dst3'         '-1'                   Physics
*    'b2stbr_sput_frac_flag'    '0'                    Physics
*    'b2stbr_sput_frc'          '0.0'                  Physics
*    'b2stbr_sput_mixed_alpha'  '1.0'                  Physics
*    'b2stbr_sput_mixed_beta'   '1.0'                  Physics
*    'b2stbr_sput_phys'         '0.0'                  Physics
*    'b2stbr_sput_phys_alpha'   '1.0e-2'               Numerics
*    'b2stbr_sput_phys_col'     '3'                    Physics
*    'b2stbr_sput_phys_model'   '1'                    Physics
*    'b2stbr_sput_res'          '0.0'                  Physics
*    'b2stbr_sput_src'          '1'                    Physics
*    'b2stbr_sputter_energy_on' '1'                    Physics
*    'b2stbr_taumax'            '0.05'                 Numerics
*    'b2stbr_therm_evap'        '0.0'                  Physics
*    'b2stcx_rg0'               '1.0'                  Numerics
*    'b2stcx_styl0'             '0'                    Numerics
*    'b2stel_phm0'              '1.0'                  Physics
*    'b2stel_rg0'               '1.0'                  Numerics
*    'b2stel_rg1'               '1.0'                  Numerics
*    'b2stel_rxm0'              '0.0'                  Numerics
*    'b2stel_rxm1'              '0.0'                  Numerics
*    'b2stel_rxm2'              '0.0'                  Numerics
*    'b2stel_rxm3'              '0.0'                  Numerics
*    'b2stel_rxm4'              '0.0'                  Numerics
*    'b2stel_styl0'             '0'                    Numerics
*    'b2tanml_anml_all_ions'    '1'                    Physics
*    'b2tanml_anomalous'        '1.0'                  Physics
*    'b2tfcc_xfac'              '1.0'                  Numerics
*    'b2tfhe_PSch'              '1.0'                  Physics
*    'b2tfhe_alfTeEh'           '0.0'                  Physics
*    'b2tfhe_anomalous'         '1.0'                  Physics
*    'b2tfhe_conduction_only'   '0'                    Physics
*    'b2tfhe_dia_cur'           '1.0'                  Physics
*    'b2tfhe_fch_pTe'           '1.0'                  Physics
*    'b2tfhe_hybr2'             '0'                    Numerics
*    'b2tfhe_lim_flux'          '0'                    Physics
*    'b2tfhe_mdf'               '0'                    Numerics
*    'b2tfhe_mode_ehx'          '0'                    Numerics
*    'b2tfhe_neutral'           '0.0'                  Physics
*    'b2tfhe_no_current'        '0'                    Numerics
*    'b2tfhe_no_hybr'           '0'                    Numerics
*    'b2tfhe_prl_cur'           '1.0'                  Numerics
*    'b2tfhe_stochastic'        '0.0'                  Physics
*    'b2tfhe_upwind'            '0'                    Numerics
*    'b2tfhe_vdia_par'          '1.0'                  Physics
*    'b2tfhe_vis_par'           '1.0'                  Physics
*    'b2tfhe_vis_per'           '1.0'                  Numerics
*    'b2tfhe_vis_q'             '1.0'                  Physics
```

```
*    'b2tfhi_Zhdanov_vel_heat'  '1'                    Physics
*    'b2tfhi_hybr2'             '0'                    Numerics
*    'b2tfhi_lim_flux'          '0'                    Physics
*    'b2tfhi_mdf'               '0'                    Numerics
*    'b2tfhi_no_hybr'           '0'                    Numerics
*    'b2tfhi_upwind'            '0'                    Numerics
*    'b2tfnb_PSch'              '1.0'                  Physics
*    'b2tfnb_PSch_style'        See description        Numerics
*    'b2tfnb_alpha'             '0.0'                  Physics
*    'b2tfnb_dia_lim_all_ions'  '1'                    Physics
*    'b2tfnb_drift_style'       See description        Numerics
*    'b2tfnb_flux_limit_min_ti' '0.0'                  Physics
*    'b2tfnb_gamma'             '2.0'                  Physics
*    'b2tfnb_mdf'               '0'                    Numerics
*    'b2tfnb_no_hybr'           '0'                    Numerics
*    'b2tfnb_pflux_cor'         '1'                    Numerics
*    'b2tfnb_poldidr'           '1.0'                  Numerics
*    'b2tfnb_poleldr'           '1.0'                  Numerics
*    'b2tfnb_raddidr'           '1.0'                  Numerics
*    'b2tfnb_radeldr'           '1.0'                  Numerics
*    'b2tfnb_vis_per'           '1.0'                  Physics
*    'b2tfnb_vis_q'             '1.0'                  Physics
*    'b2tfnb_xcur'              '0.0'                  Physics
*    'b2tfnb_xfrhie'            '1.0'                  Numerics
*    'b2tfnb_ycur'              '1.0'                  Physics
*    'b2tfrn_xvecrb'            '1.0'                  Physics
*    'b2tfrn_xwdia'             '1.0'                  Physics
*    'b2tiner_fch_inert_style'  See description        Numerics
*    'b2tiner_inert'            '1.0'                  Physics
*    'b2tiner_inert_all_ions'   '1'                    Numerics
*    'b2tinnt_fchin_in_core'    '0'                    Physics
*    'b2tinnt_kinetic'          '0'                    Physics
*    'b2tinnt_style'            '1'                    Numerics
*    'b2tlc0_alpha'             '1.0'                  Physics
*    'b2tlc0_gamma'             '1.0'                  Physics
*    'b2tlh0_alpha'             '1.0'                  Physics
*    'b2tlh0_flux_limit_min_ti' '0.0'                  Physics
*    'b2tlh0_flux_limit_style'  '2'                    Numerics
*    'b2tlh0_gamma'             '1.0'                  Physics
*    'b2tlh0_hcimx_flag'        '1'                    Numerics
*    'b2tlmv_style'             '1'                    Physics
*    'b2tlnl_ee'                '0'                    Physics
*    'b2tlnl_ei'                '0'                    Physics
*    'b2tlnl_ii'                '0'                    Physics
*    'b2tqca_model'             '1'                    Physics
*    'b2tqca_phm0'              '1.0'                  Physics
*    'b2tqce_fke_Zhdanov'       '1'                    Physics
*    'b2tqce_model'             '1'                    Physics
*    'b2tqin_csigin_style'      '1'                    Physics
*    'b2tqna_ballooning'        '0.0'                  Physics
*    'b2tqna_ballooning_rescale' '1.0'                 Physics
*    'b2tqna_bb_ref'            See description (real) Physics
*    'b2tqna_diagno'            '0'                    Output
*    'b2tqna_divsol_rescale'    '1.0'                  Physics
*    'b2tqna_inputfile'         '0'                    Run
*    'b2tqna_ixref'             See description (integer) Physics
*    'b2tqna_jsep_cnv'          jsep                   Physics
*    'b2tqna_jxa_cnv'           jxa                    Physics
*    'b2tqna_max_df0'           '1e30'                 Physics
*    'b2tqna_min_df0'           '0.0'                  Physics
*    'b2tqna_model_sig'         '0'                    Physics
*    'b2tqna_new_df0'           '0'                    Physics
*    'b2tqna_pfr_rescale'       '1.0'                  Physics
*    'b2tqna_pw0_cnv'           '1.0'                  Physics
*    'b2tqna_pw1_cnv'           '5.0'                  Physics
*    'b2tqna_pw2_cnv'           '2.0'                  Physics
*    'b2tqna_transport_namelist' '0'                   Run
```

```
*    'b2tqna_user_transport'    '0'                  Physics
*    'b2tqna_vout_cnv'          '0.0'                Physics
*    'b2tral_Zhdanov_closure'   '0'                  Physics
*    'b2tral_Zhdanov_cond'      '1'                  Physics
*    'b2tral_Zhdanov_nc'        '0'                  Physics
*    'b2tral_Zhdanov_nc_min'    '6'                  Physics
*    'b2tral_Zhdanov_nc_spec'   '2'                  Physics
*    'b2tral_Zhdanov_vish'      '1'                  Physics
*    'b2tral_Zhdanov_visu'      '1'                  Physics
*    'b2tral_amfact_***'        '1.0'                Physics
*    'b2tral_amfact_on'         '0'                  Physics
*    'b2tral_cvsa_style'        '1'                  Physics
*    'b2tral_mode'              '1'                  Numerics
*    'b2tral_zh_analyt_mdf'     '0'                  Physics
*    'b2tral_zh_imp_analyt'     '0'                  Physics
*    'b2tral_zh_tf_toff'        '0'                  Physics
*    'b2tral_zh_vis_type'       '0'                  Physics
*    'b2tral_zhcscorr'          '1'                  Physics
*    'b2tral_zhcscorr_vq'       '1'                  Physics
*    'b2tral_zhd_corr'          '1'                  Physics
*    'b2tral_zhd_corrT'         '1'                  Physics
*    'b2tral_zhflcorr'          '3.0'                Physics
*    'b2tral_zhflcorrh'         '0.4'                Physics
*    'b2trcl_conductive_limit'  '1'                  Numerics
*    'b2trcl_core_cond_limit'   '0'                  Numerics
*    'b2trcl_csig_mltpl'        '1.0'                Numerics
*    'b2trcl_cthe'              '0.0'                Physics
*    'b2trcl_cthi'              '2.65'               Physics
*    'b2trcl_cvsa_mltpl'        '1.0'                Numerics
*    'b2trcl_lambda'            '-5.0'               Physics
*    'b2trcl_lluciani'          '3'                  Physics
*    'b2trcl_lthf21'            '0'                  Physics
*    'b2trcl_luc_et_form'       '1'                  Physics
*    'b2trcl_lvis21'            '0'                  Physics
*    'b2treq_phm0'              '1.0'                Physics
*    'b2trno_con_e_stochastic'  '1.0'                Physics
*    'b2trno_csig_an_style'     '1'                  Physics
*    'b2trno_flux_limit_to_dpa' '1'                  Numerics
*    'b2trno_pol_anom_scale'    '1.0'                Physics
*    'b2tstch_delta'            '0.0'                Physics
*    'b2tstch_sig'              '1.0'                Physics
*    'b2tvspr_dia_lim_all_ions' '1'                  Physics
*    'b2tvspr_visper_all_ions'  '1'                  Physics
*    'b2upht_stylec'            '0'                  Numerics
*    'b2upht_te_min'            '0.1'                Numerics
*    'b2upht_ti_min'            '0.1'                Numerics
*    'b2ush9_cor_diag'          '1.0'                Numerics
*    'b2usht_cor_diag'          '1.0'                Numerics
*    'b2usmo_cfc0'              '1.0'                Numerics
*    'b2ux11p_acpar'            '8.0'                Numerics
*    'b2ux11p_cpu'              '0'                  Output
*    'b2ux11p_mult_nonzero'     '15'                 Numerics
*    'b2ux11p_mult_solvdim'     '15'                 Numerics
*    'b2ux11p_mult_solvdim1'    '10'                 Numerics
*    'b2ux11p_nltrsol'          '2'                  Output
*    'b2ux11p_style'            '2'                  Numerics
*    'b2ux5p_acpar'             '8.0'                Numerics
*    'b2ux5p_cpu'               '0'                  Output
*    'b2ux5p_mult_nonzero'      '10'                 Numerics
*    'b2ux5p_mult_solvdim'      '15'                 Numerics
*    'b2ux5p_mult_solvdim1'     '10'                 Numerics
*    'b2ux5p_nltrsol'           '2'                  Output
*    'b2ux5p_style'             '2'                  Numerics
*    'b2ux7p_acpar'             '8.0'                Numerics
*    'b2ux7p_cpu'               '0'                  Output
*    'b2ux7p_mult_nonzero'      '10'                 Numerics
*    'b2ux7p_mult_solvdim'      '15'                 Numerics
```

```
*    'b2ux7p_mult_solvdim1'     '10'                Numerics
*    'b2ux7p_nltrsol'           '2'                 Output
*    'b2ux7p_style'             '2'                 Numerics
*    'b2ux9p_acpar'             '8.0'               Numerics
*    'b2ux9p_cpu'               '0'                 Output
*    'b2ux9p_mult_nonzero'      '15'                Numerics
*    'b2ux9p_mult_solvdim'      '15'                Numerics
*    'b2ux9p_mult_solvdim1'     '10'                Numerics
*    'b2ux9p_nltrsol'           '2'                 Output
*    'b2ux9p_style'             '2'                 Numerics
*    'b2wdat_append'            '0'                 Output
*    'b2wdat_iout'              '0'                 Output
*    'b2xzdd_zero_dead_and_core' '1'                Numerics
*    'b2yrdr_ns'                ns                  Output
*    'b2ytdr_jsep1'             nx1/2               Run
*    'b2ytdr_ndepth1'           ndepth_nml          Run
*    'b2ytdr_non_commensurate'  '0'                 Run
*    'b2ytdr_ns'                ns                  Run
*    'b2ytdr_rescale_neutrals'  '1.0'               Run
*    'balance_average'          '0'                 Output
*    'balance_netcdf'           '0'                 Output
*    'cdfmovie_b25_fluxes'      '0'                 Output
*    'cdfmovie_b25_sources'     '0'                 Output
*    'cdfmovie_b25_state_variables' '1'             Output
*    'cdfmovie_eirene_fluxes'   '0'                 Output
*    'cdfmovie_eirene_sources'  '0'                 Output
*    'cdfmovie_eirene_state_variables' '0'          Output
*    'cdfmovie_triangles'       '0'                 Output
*    'eirene_ank_mods'          '0'                 Numerics
*    'eirene_averaging_multiple_calls' '1'          Numerics
*    'eirene_background'        '1'                 Physics
*    'eirene_dpc_fix'           '1'                 Numerics
*    'eirene_extrap'            '1'                 Numerics
*    'eirene_format'            'iter'              Output
*    'eirene_ionising_core'     '0'                 Physics
*    'eirene_lhalpha'           '1'                 Physics
*    'eirene_lvib'              '1'                 Physics
*    'eirene_mc_linearisation'  '1'                 Numerics
*    'eirene_mc_output_style'   '1'                 Output
*    'eirene_na_max'            '1e30'              Numerics
*    'eirene_na_min'            '0.0'               Numerics
*    'eirene_nesepm_istra'      '-1'                Physics
*    'eirene_neutr_avg'         '0'                 Numerics
*    'eirene_print_minmax'      '0'                 Numerics
*    'eirene_repeat_first_call' '1'                 Physics
*    'eirene_savef30'           '0'                 Output
*    'eirene_savef31'           '0'                 Output
*    'eirene_sheath_pot'        '1'                 Physics
*    'eirene_te_max'            '1e30'              Numerics
*    'eirene_te_min'            '0.0'               Numerics
*    'eirene_ti_max'            '1e30'              Numerics
*    'eirene_ti_min'            '0.0'               Numerics
*    'eirene_ua_max'            '+c'                Numerics
*    'eirene_ua_min'            '-c'                Numerics
*    'eirene_underrelax'        '0'                 Numerics
*    'eirene_use_recyceir'      '1'                 Physics
*    'eirene_uub_style'         '0'                 Numerics
*    'get_residuals'            '0'                 Output
*    'heatdiff1D_linlog'        '1'                 Numerics
*    'heatdiff1D_ratio'         '100.0'             Numerics
*    'ids_from_43'              '0'                 Output
*    'ma28_nwrite'              '0'                 Output
*    'my_out_digits'            6 or 15             Output
*    'neoclassical_ic'          '3'                 Physics
*    'nrings_for_no_speedup_averaging' '0'          Numerics
*    'set_transport_eta'        '2.0'               Physics
*    'set_transport_eta_alpha'  '0.5'               Physics
```

```
*    'set_transport_eta_ceiling' '10.0'                Physics
*    'set_transport_eta_floor'  '0.1'                  Physics
*    'set_transport_ixref'      See description (integer) Physics
*    'set_transport_iyref'      See description (integer) Physics
*    'set_transport_required_te_gradient' '5.0e4'      Physics
*    'solps_version'            'iter'                 Output
*    'tallies_netcdf'           '0'                    Output
*  ------------------------------------------------------------------
*
*
*
*  1. Purpose of individual parameters
*
*   *   1.1. Geometry switches
*
*    'b2agfs_geometry'          'upgrade.geometry'     Geometry
*    'b2agfs_lineardevice'      linear.geometry        Geometry
*    contains the file name of the geometry file to be read.
*    The file will be looked for in the run directory, in the ../baserun
*    directory, in $SOLPSTOP/data.local/meshes, and in
*    $SOLPSTOP/modules/Carre/meshes/$DEVICE. To be used in b2ag.dat.
*
*    'b2mwti_jxa'               See description (integer) Geometry
*    Cell index, on the basis mesh, of the outer midplane. Default value
*    depends on geometry:
*    Lower Single-null  : jxa=rightcut(1)-(rightcut(1)-leftcut(1))/4, i.e.
*    three quarters of the way between the two cuts.
*    Upper Single-null  : jxa=leftcut(1)+(rightcut(1)-leftcut(1))/4, i.e.
*    one quarter of the way between the two cuts.
*    Stellarator island : jxa=leftcut(1)+(rightcut(1)-leftcut(1))/2, i.e.
*    midway between the two cuts.
*    Double-null        : jxa=(rightcut(1)+rightcut(2))/2, i.e. halfway
*    between the two outer cuts.
*    Limiter geometry   : jxa=nx/2
*    Straight geometry  : jxa=nx/2
*
*    'b2mwti_jxi'               See description (integer) Geometry
*    Cell index, on the basis mesh, of the inner midplane. Default value
*    depends on geometry:
*    Lower Single-null  : jxi=leftcut(1)+(rightcut(1)-leftcut(1))/4, i.e.
*    one quarter of the way between the two cuts.
*    Upper Single-null  : jxi=rightcut(1)-(rightcut(1)-leftcut(1))/4, i.e.
*    three quarters of the way between the two cuts.
*    Stellarator island : jxi=leftcut(1)+(rightcut(1)-leftcut(1))/4, i.e.
*    one quarter of the way between the two cuts.
*    Double-null        : jxi=(leftcut(1)+leftcut(2))/2, i.e. halfway
*    between the two inner cuts.
*    Limiter geometry   : jxi=nx/4
*    Straight geometry  : jxi=nx/4
*
*    'b2mwti_jsep'              See description (integer) Geometry
*    Cell index, on the basis mesh, of the last poloidal ring before the
*    separatrix. Can only be provided for slab geometries, otherwise
*    calculated from geometry:
*    Single-null : jsep=topcut(1)-1
*    Stellarator : jsep=topcut(1)-1
*    Double-null : jsep=min(topcut(1),topcut(2))-1
*    Default for straight geometries : jsep=ny/2
*
*    'b2agmx_pbs_from_basis_mesh' '1'                  Geometry
*    If pbs_from_basis_mesh.eq.0, the pbs array is computed from values on
*    the actual working mesh (old treatment).
*    If pbs_from_basis_mesh.eq.1, the pbs array is computed from values on
*    the (possibly) finer basis mesh from the geometry file and the
*    contributions from each basis cell are added to obtain the value on
*    the working mesh.
*    In both cases, mind the value of 'b2news_area_fix', which should be
```

```
*    also declared in b2ag.dat if the non-default behaviour is wanted.
*
*    'b2agmt_qc_style'          '1'                    Geometry
*    When b2agmt_qc_style.eq.1, the cosine of the angle between the left
*    face of cells and the magnetic field direction is computed using
*    information from cells to both the left and right of the face, based
*    on interpolations of the qz array (cosine at cell centers).
*    When b2agmt_qc_style.eq.2, the cosine of the angle between the left
*    face of cells and the magnetic field direction is computed assuming
*    that it is the angle between the left cell face and the bisector of
*    the poloidal direction in the left and right cells sharing the common
*    face.
*    When b2agmt_qc_style.eq.0, we only use information from the cell at
*    the right of the face. This is the treatment used up to version 3.0.6,
*    inclusively.
*
*    'b2agdr_redef_qc'          '0'                    Geometry
*    When b2agdr_redef_qc.eq.1, the cosine of the angle between the left
*    face of a cell and the magnetic field direction is interpolated from
*    the values at the cell centers.
*    When b2agdr_redef_qc.eq.0, we only use information from the cell at
*    the right of the face. This is the treatment used up to version 3.0.6,
*    inclusively.
*    This switch is only active if b2agmt_qc_style.eq.0.
*
*    'b2agdr_redef_gmtry'       '1'                    Geometry
*    When b2agdr_redef_gmtry.eq.1, geometrical quantities are adjusted so
*    as to ensure that the poloidal flux between two neighbouring flux
*    surfaces remains constant and that toroidal magnetic field times major
*    radius is the same constant for all cells independently of whether it
*    is computed at cell centers, cell faces, or cell corners.
*
*    'b2agfs_periodic_bc'       '0'                    Geometry
*    periodic_bc specifies if this is either an island or limiter geometry.
*    If periodic_bc.eq.1 then island/limiter treatment is turned on. We
*    differentiate between the two case through nncut:
*    nncut.eq.0 = limiter case
*    nncut.ge.1 = island divertor case (there should be nncut islands then)
*    The case periodic_bc.eq.-1 is used to remove tranverse physics in 1-D
*    cases.
*    If periodic_bc.eq.0 then we have the usual treatment. No other values
*    are currently allowed. To be used in b2ag.dat.
*
*    'b2agsi_isymm'             '1'                    Geometry
*    isymm specifies the type of symmetry of the geometry: isymm.eq.0
*    implies a slab geometry,
*    isymm.eq.1.or.isymm.eq.2 imply rotational symmetry about the crx=0
*    axis, and isymm.eq.3.or.isymm.eq.4 indicate rotational symmetry about
*    the cry=0 axis.
*    Other values are not allowed.
*    isymm.eq.1.or.isymm.eq.3 mean a toroidal geometry case, i.e. with a
*    toroidal (out-of-plane) magnetic field component.
*    isymm.eq.2.or.isymm.eq.4 mean a cylindrical geometry case, i.e. no
*    toroidal magnetic field component. To be used in b2ag.dat.
*
*    'b2stbc_coreregno'         See description        Geometry
*    'b2stbc_coreregn2'         '4'                    Geometry
*    coreregno is the boundary index of the core boundary in the input
*    files b2ah.dat and b2mn.dat. For standard single-null and double-null
*    cases, the default value of coreregno is 1. For a straight geometry or
*    limiter case, the default value of coreregno is 0.
*    coreregn2 is the boundary index of the second core boundary in case of
*    a double-null geometry, and is not used otherwise.
*
*    'b2stbc_pfrregno1'         '0'                    Geometry
*    'b2stbc_pfrregno2'         '2'                    Geometry
*    pfrregno1 is the boundary index of the first half of the private flux
```

```
*     region boundary on which feedback is being applied (through
*     b2stbc_ndes or b2stbc_private_flux_puff) in the input files b2ah.dat
*     and b2mn.dat. pfrregno2 is the second half of the same private flux
*     region. For a standard single null, the pair should be (0,2). For a
*     standard double-null, the lower private flux region is labelled with
*     the pair (0,5), and the upper private flux region with the pair (2,3).
*     One can use only one private flux region by setting
*     pfrregno1.eq.pfrregno2.
*
*     'b2stbc_solregno'          '3'                    Geometry
*     solregno is the boundary index of the SOL North boundary in the input
*     files b2ah.dat and b2mn.dat. For a standard single-null case, solregno
*     is 3. For a standard double-null case, solregno for the outer SOL
*     North boundary would be 7.
*
*     'b2agmt_1d_width'          '1.0'                  Geometry
*     For 1-D cases, gives the width of the domain (in m) in the third
*     (toroidal) dimension.
*
*     'b2stbr_first_flight_no_of_start_points' 2*(nx+2)+2*max(nncut,1)*(ny+2) Geometry
*     When the first flight model is turned on, this number must be greater
*     than or equal to the num
*     ber of boundary cells on the mesh. Only in cases with special
*     geometries (i.e. limiters, islands, ...) need it be increased beyond
*     the default value above. This variable is only used within the first
*     flight module.
*
*     'b2agfs_geom_match_dist'   '1.0e-6'               Geometry
*     Distance used as the matching criterion when reading the geometry
*     file.
*
*     'b2agfs_Bt_adjust'         '0'                    Geometry
*     If the mesh is offset (See b2agfs_xoffset, b2agfs_yoffset below) and
*     Bt_adjust.eq.1, then the magnetic field is recomputed for toroidal
*     geometries, keeping the pitch constant but changing the toroidal field
*     magnitude according to Bt = Bt0/R. To be used in b2ag.dat.
*
*     'b2agfs_Bt_rescale'        '1.0'                  Geometry
*     The magnetic field will be multiplied by Bt_rescale. All components of
*     the field are scaled together. To be used in b2ag.dat.
*
*     'b2agfs_pit_rescale'       '1.0'                  Geometry
*     The magnetic field line pitch will be multiplied by pit_rescale.
*     This means that the poloidal field component is multiplied by
*     pit_rescale, while the toroidal field component is left unchanged. To
*     reverse the plasma current direction, set pit_rescale to -1.0.
*     The sign convention used is that a positive poloidal field points in
*     the direction of increasing <ix>. Be mindful however that, when
*     inverting the sign of the poloidal magnetic field, you are also
*     inverting the direction of the parallel velocity. You will then need
*     to use the 'b2mndr_inverse_ua' switch to correct for that. To be used
*     in b2ag.dat.
*
*     'b2agfs_Bt_reversal'       '0'                    Geometry
*     If Bt_reversal.eq.1, then the sign of the toroidal field component
*     only is reversed. The sign convention used is that a positive toroidal
*     field leads to a Bx(grad(B)) direction pointing down, so B is out of
*     the page when looking at the usual view of the poloidal plane with the
*     center line on the left. To be used in b2ag.dat.
*
*     'b2agfs_min_pitch'         '1.0'                  Geometry
*     Minimum allowed value for the pitch angle (in degrees) at the plates.
*
*     'b2agfs_xoffset'           '0.0'                  Geometry
*     'b2agfs_yoffset'           '0.0'                  Geometry
*     xoffset and yoffset are offsets of the basis mesh in the x- and
*     y-direction respectively. The mesh will be translated by
```

```
*    (xoffset,yoffset) if any of the two is non-zero from its position
*    given in the local_sonnet (See 'b2agfs_geometry' above) file.
*    To be used in b2ag.dat.
*
*    'b2agfs_xrescale'          '1.0'                 Geometry
*    'b2agfs_yrescale'          '1.0'                 Geometry
*    Rescaling factors of the x- and y- coordinates of the basis mesh.
*    To be used in b2ag.dat.
*
*    'b2agfs_nncut'             See description (integer) Geometry
*    Number of topological cuts in geometry. Needed only to force a
*    different value from the one computed automatically by b2ag.
*
*    'b2agfs_leftcut'           See description (integer) Geometry
*    'b2agfs_rightcut'          See description (integer) Geometry
*    'b2agfs_bottomcut'         See description (integer) Geometry
*    'b2agfs_topcut'            See description (integer) Geometry
*    Parameters describing the position and extent of the first topological
*    cut, i.e. corresponding to the first X-point. Needed only to force
*    different values from the ones computed automatically by b2ag.
*    leftcut is the poloidal index of cells directly to the RIGHT of the
*    left branch of the cut.
*    rightcut is the poloidal index of cells directly to the RIGHT of the
*    right branch of the cut.
*    bottomcut is the radial index of cells directly below the cut.
*    topcut is the radial index of cells directly above the cut.
*
*    'b2agfs_leftcut2'          See description (integer) Geometry
*    'b2agfs_rightcut2'         See description (integer) Geometry
*    'b2agfs_bottomcut2'        See description (integer) Geometry
*    'b2agfs_topcut2'           See description (integer) Geometry
*    Parameters describing the position and extent of the second
*    topological cut, i.e. corresponding to the second X-point. Needed only
*    to force different values from the ones computed automatically by
*    b2ag.
*    leftcut2 is the poloidal index of cells directly to the RIGHT of the
*    left branch of the second cut.
*    rightcut2 is the poloidal index of cells directly to the RIGHT of the
*    right branch of the second cut.
*    bottomcut is the radial index of cells directly below the second cut.
*    topcut is the radial index of cells directly above the second cut.
*
*    'b2agdr_nxiso1'            '-2'                  Geometry
*    'b2agdr_nxiso2'            '-2'                  Geometry
*    'b2agdr_nyiso1'            '-2'                  Geometry
*    'b2agdr_nyiso2'            '-2'                  Geometry
*    Range of an optional isolated region to be included in the geometry.
*    The isolated region will extend over the cell range spanned by
*    (nxiso1:nxiso2,nyiso1:nyiso2).
*    If only some of the four coordinates are given, the region is made to
*    extent from the given coordinates to the end of the grid.
*    Neighbourhood arrays and region indices are automatically adjusted.
*
*   *   1.2. Run switches
*
*    'b2mndr_ids_path'          See description       Run
*    'b2mndr_pulse_number'      '0'                   Run
*    'b2mndr_shot_number'       '0'                   Run
*    'b2mndr_run_number'        '0'                   Run
*    'b2mndr_database'          $(DEVICE)             Run
*    'b2mndr_device'            $(DEVICE)             Run
*    'b2mndr_user'              $(USER)               Run
*    These switches serve as identification for the simulation. They can be
*    inherited from the SOLPS-GUI:
*    IDS path : Only available if using IMAS Access Layer 5. The directory
*    path where the IMAS data entry will be saved. Supersedes the pulse
*    (previously shot) and run numbers. May begin with "$HOME", "$IMASDIR",
```

```
*     or "$SOLPSTOP", that will be parsed by the code. If it begins with a
*     '/', is understood as an absolute path, otherwise will be prefixed
*     with the $IMASDIR environment variable. Defaults to the last line in
*     the 'shotnumber.history' file if present, and to
*     $IMASDIR/${pulse_number}/${run_number} otherwise or if $IMASDIR is
*     modified from its default value by the switches below.
*     Pulse number : Pulse number identifying the run (supersedes
*     b2mndr_shot_number). Defaults to the last number found in
*     shotnumber.history, or 0 if the file is not found. Must be positive
*     and non-zero if an IMAS data entry is requested. If using IMAS Access
*     Layer 4, cannot exceed 214748.
*     Run number : The number of the run. Must be positive or zero. If using
*     IMAS Access Layer 4, limited to 5 digits (i.e. span from 0 to 99999).
*     Database : IMAS IDS database name (supersedes b2mndr_device). Defaults
*     to 'solps-iter' if $DEVICE if undefined. If $DEVICE is 'iter', the
*     code internally changes it to 'ITER' to follow the IMAS convention.
*     Can be used to rebuild the IMASDIR path.
*     User : The user who ran the simulation. Can be used to rebuild the
*     IMASDIR path.
*
*     'b2aidr_read_b2fstate'     '0'                    Run
*     If read_b2fstate.ne.0, b2aidr will read an existing b2fstate file and
*     append to it a flat state description for additional new species being
*     introduced in the run.
*
*     'b2mndr_ntim'              '1'                    Run
*     Number of timesteps desired in this run. If ntim.eq.0, the code simply
*     reads the input file and creates the b2fplasma and b2fstate files
*     corresponding to the present plasma state.
*
*     'b2mndr_dtim'              '1.0'                  Run
*     Timestep (in seconds).
*
*     'b2mndr_delta_max'         '0.0'                  Run
*     'b2mndr_delta_min'         '0.0'                  Run
*     'b2mndr_dt_change_dec'     '1.0'                  Run
*     'b2mndr_dt_change_inc'     '1.0'                  Run
*     'b2mndr_dt_max'            '1.0e+01'              Run
*     'b2mndr_dt_min'            '1.0e-30'              Run
*     One of the dynamic timestep control parameters. Only active if both
*     delta_max and delta_min are nonzero. The timestep is increased by a
*     factor dt_change_inc or decreased by a factor dt_change_dec, within
*     the bounds [dt_min, dt_max], according to whether the largest
*     instantaneous change in all equations is smaller than delta_min or
*     larger than delta_max, respectively.
*
*     'b2mndr_stim'              '0.0'                  Run
*     stim specifies the initial time --- default 0.
*     If set to a positive or zero value, this overwrites the time value
*     read from b2fstati.
*     If set to a negative value, the run continues from the time read in
*     b2fstati. In that case, the tracing data is appended to the existing
*     files, otherwise the tracing files are overwritten.
*
*     'b2mndr_etim'              '0.0'                  Run
*     etim specifies the end time. Only active if etim > stim.
*
*     'b2mndt_nstg0'             '1'                    Run
*     'b2mndt_nstg1'             '1'                    Run
*     'b2mndt_nstg2'             '1'                    Run
*     Time-dependent mode and iterative mode switches. The basic code
*     timestep proceeds as follows:
*      ..test input arguments
*      ..compute auxiliary quantities
*      ..prepare source computation
*      ..do i0=1,nstg0
*       ..compute log-log linearised rate coefficients
```

```
*      ..do i1=1,nstg1
*       ..compute source linearisation
*       ..do i2=1,nstg2
*        ..perform one inner iteration
*        ..re-compute auxiliary quantities
*        ..produce monitoring output
*       ..enddo
*      ..enddo
*     ..enddo
*    This can be completed by the the 'b2mndt_nstg_ares??' switches.
*    See 'Numerics' section for details.
*
*    'b2news_no_solve'          '0'                    Run
*    If no_solve.eq.1, then the code is run without actually solving any
*    equations. All secondary and dependent data is computed.
*    The nstg(0:2) array is overwritten to '1's. The simulation time will
*    not be updated. The code will compute fluxes, sources, transport
*    coefficients, etc... 'ntim' times but not update the basic plasma
*    quantities. Additionally, if no_solve is set to a negative value, then
*    only some equations are solved (simulation time and nstg arrays will
*    behave normally).
*    no_solve.eq.-1 will activate the parallel momentum equations only.
*    no_solve.eq.-2 will activate the density equations only.
*    no_solve.eq.-4 will activate the potential equation only. (Active only
*    if poteq.eq.1, otherwise, the behaviour dictated by the poteq setting
*    applies).
*    no_solve.eq.-8 will activate the heat equations only. These can be
*    combined. For example, no_solve.eq.-3 will activate the parallel
*    momentum and particle conservation equations.
*    If the no_solve switch requires an equation not to be solved, the
*    settings in the corresponding solvexx arrays from
*    b2.numerics.parameters are moot.
*    The latter are reserved for fine-tuning numerical diagnostics.
*    If wishing to toggle whether to solve or not to solve each equation
*    separately, one should set, in the b2.numerics.parameters input file,
*    the SOLVEMO, SOLVEMT, SOLVECO, SOLVEE, SOLVEEI, SOLVEET and SOLVEPO
*    arrays, respectively, for each of the parallel momentum equations, the
*    total momentum equation, each density equation, the electron and ion
*    heat equations, and the total energy equation.
*    When running in time-dependent mode, the total energy and total
*    momentum equations are not solved.
*    The table below shows which combination of equations will be solved
*    depending on the value of switch 'b2news_no_solve'
*    *-------------------------------------------------
*    *        | value |  co  |  mo  |  he  |  hi  |  po  |
*    *-------------------------------------------------
*    *        |   0   |  +   |  +   |  +   |  +   |  +   |
*    *-------------------------------------------------
*    *        |   1   |  -   |  -   |  -   |  -   |  -   |
*    *-------------------------------------------------
*    *        |  -1   |  -   |  +   |  -   |  -   |  -   |
*    *-------------------------------------------------
*    *        |  -2   |  +   |  -   |  -   |  -   |  -   |
*    *-------------------------------------------------
*    *        |  -3   |  +   |  +   |  -   |  -   |  -   |
*    *-------------------------------------------------
*    *        |  -4   |  -   |  -   |  -   |  -   |  +   |
*    *-------------------------------------------------
*    *        |  -5   |  -   |  +   |  -   |  -   |  +   |
*    *-------------------------------------------------
*    *        |  -6   |  +   |  -   |  -   |  -   |  +   |
*    *-------------------------------------------------
*    *        |  -7   |  +   |  +   |  -   |  -   |  +   |
*    *-------------------------------------------------
*    *        |  -8   |  -   |  -   |  +   |  +   |  -   |
*    *-------------------------------------------------
*    *        |  -9   |  -   |  +   |  +   |  +   |  -   |
```

```
*    *-----------------------------------------------------
*    *        | -10   |  +   |  -   |  +   |  +   |  -   |
*    *-----------------------------------------------------
*    *        | -11   |  +   |  +   |  +   |  +   |  -   |
*    *-----------------------------------------------------
*    *        | -12   |  -   |  -   |  +   |  +   |  +   |
*    *-----------------------------------------------------
*    *        | -13   |  -   |  +   |  +   |  +   |  +   |
*    *-----------------------------------------------------
*    *        | -14   |  +   |  -   |  +   |  +   |  +   |
*    *-----------------------------------------------------
*
*    'b2mndr_cpu'               '0.0'                 Run
*    CPU limit. If cpu.eq.0.0, no CPU limit is enforced. Otherwise, the
*    code stops smoothly after it has been running for at least 'cpu' CPU
*    seconds.
*
*    'b2mndr_elapsed'           '0.0'                 Run
*    Elapsed time limit. Is elapsed.eq.0.0, no time limit is enforced.
*    Otherwise, the code stops smoothly after is has been running for at
*    least 'elapsed' wall clock seconds.
*
*    'b2mndr_savecpu'           '3600.0'              Run
*    If savecpu.gt.0.0, CPU time interval after which save files
*    plasmastate.xxxx are written. Other additional options for writing of
*    the checkpointing files are available by means of the b2mndt_ntim_save
*    and/or b2mndr_plasmatim switches.
*    These can be used, in conjunction with the b2co.exe instruction, to
*    recover a crashed run.
*
*    'b2mndr_ismain'            '1'                   Run
*    ismain identifies the index of the main plasma species.
*    It must hold that ismain is not a neutral species.
*
*    'b2news_facdrift_dec'      '0.0'                 Run
*    'b2news_facdrift_inc'      '1.0'                 Run
*    'b2news_facdrift_start'    '0.0'                 Run
*    'b2news_facdrift_target'   '0.0'                 Run
*    Ramping parameters for facdrift, which multiplies the diamagnetic
*    terms as well as the ion inertia current. The code is started on the
*    first time step with facdrift=facdrift_start. If
*    facdrift_target.ne.facdrift_start, then, on each time step, facdrift
*    is multiplied by facdrift_inc. If the code does not converge on the
*    timestep, facdrift is decreased by facdrift_dec. A facdrift profile is
*    also possible, see Numerics section for details.
*    The ion-neutral friction current requires either facdrift or fac_ExB
*    to be turned on as well.
*
*    'b2news_facExB_dec'        '0.0'                 Run
*    'b2news_facExB_inc'        '1.0'                 Run
*    'b2news_facExB_start'      '0.0'                 Run
*    'b2news_facExB_target'     '0.0'                 Run
*    Ramping parameters for facExB, which multiplies the ExB terms. Same
*    treatment as above for facdrift. Can be superseded by b2news_ExB to
*    set a constant value throughout the run.
*
*    'b2news_facvis_dec'        '0.0'                 Run
*    'b2news_facvis_inc'        '1.0'                 Run
*    'b2news_facvis_start'      '0.0'                 Run
*    'b2news_facvis_target'     '0.0'                 Run
*    Ramping parameters for facvis, which multiplies the drift viscosity
*    terms. Same treatment as above for facdrift. Can be superseded by
*    b2news_vis to set a constant value throughout the run.
*
*    'b2stbc_boundary_namelist' See description       Run
*    'b2stbr_neutrals_namelist' '0'                   Run
*    'b2srdt_numerics_namelist' '0'                   Run
```

```
*    'b2tqna_transport_namelist' '0'                    Run
*    Namelist file indicators. If xxx_namelist.eq.1, then the code looks
*    for the file b2.xxx.parameters, in which it expects to find the xxx
*    namelist.
*    In the case of boundary_namelist, defaults to 1 for double-null and
*    snowflake geometries, 0 otherwise.
*
*    'b2sral_inputfile'         '0'                    Run
*    Profiles file indicator. If b2sral_inputfile.eq.1, then the code looks
*    for the file b2.sources.profile in which it expects to find the
*    'profile' namelist.
*    This namelist will contain profiles of sources measured from the
*    location (ixref,iyref) (default is outer midplane separatrix), which
*    is set with the 'set_transport_i[xy]ref' switches.
*
*    'b2tqna_inputfile'         '0'                    Run
*    Profiles file indicator. If b2tqna_inputfile.eq.1, then the code looks
*    for the file b2.transport.inputfile, in which it expects to find the
*    transport namelist.
*    This namelist will contain profiles of transport parameters measured
*    from the location (ixref,iyref) (default is outer midplane
*    separatrix), which is set with the 'set_transport_i[xy]ref' switches.
*
*    'b2mndr_eirene'            '0'                    Run
*    Turns on coupling with the Eirene Monte Carlo neutral code if nonzero.
*    To be used, the code must be compiled with the -DB25_EIRENE option.
*
*    'b2mndr_astra'             '0'                    Run
*    Turns on coupling with the ASTRA core transport code if nonzero.
*    To be used, the code must be compiled with the -DASTRA option.
*
*    'b2mndr_rescale_neutrals_sources' '1.0'           Run
*    Multiplier to the neutral sources (either computed by Eirene or the
*    corresponding rate coefficients).
*
*    'b2mndr_rescale_neutrals'  '1.0'                  Run
*    Multiplier to the neutral density on the first timestep.
*
*    'b2mndr_density_rescale'   '1.0'                  Run
*    Multiplier of all densities on the first timestep.
*    If set to a negative value, the rescaling (by |density_rescale|) will
*    only be applied to species with zn.eq.zn_rescale.
*
*    'b2mndr_zn_rescale'        '0'                    Run
*    Nuclear charge number of the species to be rescaled if density_rescale
*    is set to a negative value.
*
*    'b2stbr_core_sources_rescale' '1.0'               Run
*    Multiplier to the totally ionised species sources at the core
*    boundary.
*
*    'b2mndt_density_control'   '0'                    Run
*    Feedback on the total heavy particle density. If density_control.ne.0,
*    the sum of all densities is kept constant.
*
*    'b2stbc_feedback'          '0'                    Run
*    If feedback.eq.1, turns on feedback mode for the boundary conditions.
*    See Physics section for more details. Equivalent to LFEEDBACK=.true.
*    in the b2.boundary.parameters namelist.
*
*    'b2stbc_ncallfeedback'     '0'                    Run
*    Timestep index after which the feedback in b2stbc is activated.
*
*    'b2stbr_first_flight'      '0'                    Run
*    If first_flight.ne.0, turns on the first flight model. See Physics
*    section for additional details.
*
```

```
*    'b2ytdr_ns'               ns                      Run
*    New number of species desired when creating a new initial state file
*    using b2yt.exe. Must be specified within b2yt.dat.
*
*    'b2ytdr_jsep1'            nx1/2                   Run
*    Radial index of separatrix in the converted grid. Must be specified
*    within b2yt.dat. Only applies when converting linear geometries.
*
*    'b2ytdr_ndepth1'          ndepth_nml              Run
*    New resolution of target depth desired when creating a new initial
*    state file using b2yt.exe. Must be specified within b2yt.dat.
*
*    'b2ytdr_rescale_neutrals' '1.0'                   Run
*    Rescaling of neutral densities by rescale_neutrals when creating a new
*    initial state file using b2yt.exe. Must be specified within b2yt.dat.
*
*    'b2ytdr_non_commensurate' '0'                     Run
*    When set to 1, b2yt will attempt to interpolate a plasma solution and
*    create new input files for a new grid which is not commensurate to the
*    original grid, but still has the same topology.
*    Must be specified within b2yt.dat.
*
*   *   1.3. Physics switches
*
*    'b2siav_addvis'           '1.0'                   Physics
*    Multiplier to heat flux contribution to divergence of viscosity tensor
*    in the momentum equation.
*
*    'b2siav_addvis1'          '1.0'                   Physics
*    When not equal to '0.0', adds contribution to divergence of viscosity
*    tensor coming from x-variations in B.
*
*    'b2siav_style_qip'        '0'                     Physics
*    If style_qip.eq.1, adds a classical ion heat conductivity term to the
*    heat flux used to compute the heat viscosity current (see manual for
*    full details).
*
*    'b2npmo_b2sifr_'          '1'                     Physics
*    If b2sigp_style is set to '2', this switch has no effect.
*    When set to '1', the new correct form of the friction force is used,
*    applicable for non-hydrogenic plasmas or hydrogenic mixtures.
*    The value '0' corresponds to the old SOLPS5.0 treatment. Not
*    recommended unless wanting to recover older 5.0 results.
*
*    'b2sihs_istyle_Joule_heating' '1'                 Physics
*    If b2sigp_style is set to '2', this switch has no effect.
*    When set to '1', there is no radial contribution in Joule heating
*    because there are no physical reasons to take it into account.
*    The value '0' corresponds to the old SOLPS5.0 treatment.
*    If set to '2' or '3', will have the same physics meaning as the '0' or
*    '1' settings, respectively, but use area interpolation instead of
*    volume interpolation to compute cell face quantities.
*
*    'b2sian_phm0'             '1.0'                   Physics
*    Multiplier to the parallel momentum source term associated with the
*    anomalous current. It is recommended '1.0'. Only active if both the
*    ExB and diamagnetic drifts are turned on.
*    The value '0.0' corresponds to the old SOLPS5.0 treatment.
*
*    'b2sicf_phm0'             '1.0'                   Physics
*    Multiplier of the centrifugal force term. It is recommended '1.0'.
*    The value '0.0' corresponds to the old SOLPS5.0 treatment.
*
*    'b2sicf_phm1'             '1.0'                   Physics
*    Multiplier of the centrifugal force correction term due to
*    linearization.
*    It is recommended '1.0'.
```

```
*    The value '0.0' corresponds to the old SOLPS5.0 treatment.
*
*    'b2tfhe_anomalous'         '1.0'                  Physics
*    'b2tanml_anomalous'        '1.0'                  Physics
*    Real parameter which determines anomalous current.
*    Both b2tfhe_anomalous and b2tanml_anomalous point to the same switch
*    (backward compatibility with SOLPS5.2): b2tfhe_anomalous is checked
*    first and only if it is not found is b2tanml_anomalous looked for.
*    If b2tfhe_anomalous is 0 then anomalous current is switched off
*    otherwise anomalous current is switched on, and the numerical value of
*    b2tfhe_anomalous acts as a multiplier to the anomalous electrical
*    conductivity and thermo-electric coefficients provided.
*
*    'b2tanml_anml_all_ions'    '1'                    Physics
*    When set to 1 the subdivision between all ions of the anamalous
*    current is applied. The SOLPS5.2 physics model is used.
*    Otherwise the total anamalous current (summed over all ions) is
*    assigned to main ions.
*
*    'b2news_ExB'               '0.0'                  Physics
*    Real parameter which multiplies ExB flows. If b2news_ExB.eq.0 and
*    b2news_facExB_start.eq.0 then ExB flows are switched off. If
*    b2news_ExB is nonzero, then ExB flows are multiplied by that constant
*    throughout the run.
*    See also Run section on switches b2news_facExB_... for more details. A
*    spatial fac_ExB profile is also possible, see Numerics section for
*    details.
*
*    'b2news_vis'               '0.0'                  Physics
*    Real parameter which multiplies the viscous drift flows. If
*    b2news_vis.eq.0 and b2news_facvis_start.eq.0 then the viscous drift
*    flows are switched off. If b2news_vis is nonzero, then the viscous
*    drift flows are multiplied by that constant throughout the run.
*    See also Run section on switches b2news_facvis_... for more details. A
*    spatial fac_vis profile is also possible, see Numerics section for
*    details.
*
*    'b2tiner_inert'            '1.0'                  Physics
*    Real parameter which multiplies the ion inertial current.
*
*    'b2tfhe_dia_cur'           '1.0'                  Physics
*    Real parameter which multiplies the diamagnetic current.
*
*    'b2tfhe_vdia_par'          '1.0'                  Physics
*    Real parameter which multiplies the convective heat flux due to grad
*    B-drift of guiding centers in non-modified heat fluxes of electrons
*    and ions.
*
*    'b2tfrn_xvecrb'            '1.0'                  Physics
*    Real parameter which multiplies the contribution from the ExB drift
*    velocity to the flows passed to Eirene.
*
*    'b2tfrn_xwdia'             '1.0'                  Physics
*    Real parameter which multiplies the contribution from the diamagnetic
*    drift velocity to the flows passed to Eirene.
*
*    'b2tfhe_neutral'           '0.0'                  Physics
*    Real parameter which multiplies the ion-neutral current.
*    If b2tfhe_neutral is 0 then the ion-neutral current is switched off
*    otherwise the ion-neutral current is switched on.
*    The ion-neutral current also requires either the diamagnetic or ExB
*    drifts to be turned on as well.
*
*    'b2tinnt_kinetic'          '0'                    Physics
*    Integer switch to activate the usage of kinetic vectorial momentum
*    sources due to ion-neutral friction computed based on Eirene tallies.
*    If b2tinnt_kinetic.eq.0 (default), then the friction force is computed
```

```
*     from explicit expressions for CX cross-section and ion drift
*     velocities. However, the CX cross-section is available for hydrogen
*     isotopes only, the expression might not be fully valid for arbitrary
*     plasma composition, and the current is assumed to be carried by
*     electrons, i.e. no contribution to ion flux is computed.
*     If b2tinnt_kinetic.eq.1, then the friction force is computed from
*     vectorial momentum sources coming from Eirene. These are species-
*     resolved, so the contribution from this current to the ion flux is
*     computed and transferred to the particle balance equations for each
*     species separately. The switch is meaningful for coupled mode only and
*     requires 'use_eirene' to be set to '1', otherwise the default approach
*     is applied.
*
*     'b2tinnt_fchin_in_core'    '0'                    Physics
*     Integer switch to turn off or on the ion-neutral current in the core
*     region (applies to coupled runs only). This is recommended in cases
*     where the neutral densities are very low in the core region and the
*     ion-neutral current is likely to vary widely from one iteration to the
*     next as a result of Monte Carlo noise.
*     If fchin_in_core.eq.0 (default), then fchin is set to zero in the
*     core.
*     If fchin_in_core.eq.1, then fchin is unchanged.
*
*     'b2tfhe_PSch'              '1.0'                  Physics
*     Real parameter which multiplies the Pfirsch-Schlueter electron heat
*     flux and conductivity.
*
*     'b2tfhe_vis_par'           '1.0'                  Physics
*     Real parameter which multiplies the current driven by parallel
*     viscosity.
*     If b2tfhe_vis_par is 0 then the viscosity-driven current is switched
*     off otherwise the viscosity-driven current is switched on.
*
*     'b2tfhe_vis_q'             '1.0'                  Physics
*     Real parameter which multiplies the current driven by heat viscosity
*     effects.
*
*     'b2tfhe_stochastic'        '0.0'                  Physics
*     Real parameter which turns on stochastic current.
*     If b2tfhe_stochastic is 0 then stochastic current is switched off
*     otherwise stochastic current is switched on.
*
*     'b2tstch_delta'            '0.0'                  Physics
*     Width of the stochastic current layer (in metres), measured from the
*     separatrix inward, along the poloidal index ixref (given by
*     b2tqna_ixref).
*     If b2tfhe_stochastic.ne.0, then b2tstch_delta must be greater than
*     zero.
*
*     'b2tstch_sig'              '1.0'                  Physics
*     Multiplier to the magnetic field line stochastic diffusion
*     coefficient, describing the stochastic conductivity.
*
*     'b2trno_con_e_stochastic'  '1.0'                  Physics
*     Multiplier to the stochastic conductivity.
*
*     'b2trcl_lluciani'          '3'                    Physics
*     If lluciani.ne.0, then transport coefficients on cells belonging to
*     closed field lines are modified according to the Luciani model.
*     If lluciani.eq.1, the standard connection length formulation is used.
*     If lluciani.eq.2, the old SOLPS4.0 formulation is used for backward
*     compatibility.
*     If lluciani.eq.3, Spb's new form Luciani's coefficient.
*
*     'b2trcl_luc_et_form'       '1'                    Physics
*     Only active if b2trcl_lluciani.eq.3 (default).
*     If luc_et_form.eq.0, the SPb form of the Luciani coefficient is
```

```
*     applied. This is correct only for hydrogenic majority plasmas.
*     If luc_et_form.eq.1 (default), the generalized form of the SPb Luciani
*     coefficient is used. If b2tral_Zhdanov_closure.eq.0 (default), then
*     the result is the same as if the SPb form of the Luciani coefficient
*     is used (and correct only for hydrogenic majority plasmas). If
*     b2tral_Zhdanov_closure.eq.1, then the multi-ion generalized form of
*     the SPb Luciani coefficient is used (correct for arbitrary plasma
*     mixtures).
*
*     'b2trcl_lthf21'            '0'                    Physics
*     If lthf21.ne.0, then the heat transfer model is modified according to
*     the 21-moment approach. This should only be used for multi-fluid runs.
*
*     'b2trcl_lvis21'            '0'                    Physics
*     If lvis21.ne.0, then the viscosities are modified according to the
*     21-moment approach. This should only be used for multi-fluid runs.
*     This switch should not be used along with 'b2tfhe_vis_par' to avoid
*     double-counting of classical viscosity effects.
*
*     'b2tral_Zhdanov_closure'   '0'                    Physics
*     If b2tral_Zhdanov_closure.eq.1, the Zhdanov-Grad module is enabled,
*     and fully multi-ion collisional closure for the components along the
*     magnetic field of the vector and tensor moments of the distribution
*     function is applied for all the ions (without specifying main ions and
*     impurities) based on the works of Zhdanov [V. M. Zhdanov, 2002] and
*     Makarov et al. [S. O. Makarov et al., PoP 2021]. Otherwise the
*     standard SOLPS-ITER model is applied.
*
*     'b2tral_zh_imp_analyt'     '0'                    Physics
*     If b2tral_zh_imp_analyt.eq.0, the Zhdanov-Grad closure method is
*     applied by means of the explicit matrix inversion methed (EMIM).
*     If b2tral_zh_imp_analyt.eq.1, the improved analytical method (IAM) is
*     applied. See [S. O. Makarov et al., PoP 2021] for details.
*     This switch is subservient to b2tral_Zhdanov_closure.
*
*     'b2tral_zh_analyt_mdf'     '0'                    Physics
*     If b2tral_zh_analyt_mdf.eq.1, the MDF friction force calculation is
*     applied. See [S. O. Makarov et al., PoP 2021] for details.
*     This switch is subservient to b2tral_Zhdanov_closure.
*
*     'b2tral_Zhdanov_cond'      '1'                    Physics
*     If b2tral_Zhdanov_cond.eq.0, the Braginskii expression for the ion
*     conductivity is used.
*     This switch is subservient to b2tral_Zhdanov_closure.
*
*     'b2tral_Zhdanov_vish'      '1'                    Physics
*     If b2tral_Zhdanov_vish.eq.0, the Braginskii expression for the heat
*     flux dependent viscous stress is used.
*     This switch is subservient to b2tral_Zhdanov_closure.
*
*     'b2tral_Zhdanov_visu'      '1'                    Physics
*     If b2tral_Zhdanov_visu.eq.0, the Braginskii expression for the
*     velocity-dependent viscous stress is used.
*     This switch is subservient to b2tral_Zhdanov_closure.
*
*     'b2tral_zhflcorr'          '3.0'                  Physics
*     Flux-limiting coefficient multiplier.
*
*     'b2tral_zhflcorrh'         '0.4'                  Physics
*     Additional flux-limiting coefficient multiplier for the heat flux.
*
*     'b2tral_zhd_corr'          '1'                    Physics
*     Multiplier to the corrections due to the difference in definition for
*     temperature and Joule heating between Zhdanov and Braginskii.
*     This switch is subservient to b2tral_Zhdanov_closure.
*
*     'b2tral_zhd_corrT'         '1'                    Physics
```

```
*    Multiplier to the corrections due to the difference in definition for
*    temperature between Zhdanov and Braginskii in the thermal force
*    calculation.
*    This switch is subservient to b2tral_Zhdanov_closure.
*
*    'b2tral_zhcscorr'          '1'                    Physics
*    If b2tral_zhcscorr.eq.0, the corrections for each individual ion
*    charge state from the isonuclear sequence value are disabled.
*    This switch is subservient to b2tral_Zhdanov_closure.
*
*    'b2tral_zhcscorr_vq'       '1'                    Physics
*    If b2tral_zhcscorr_vq.eq.0, the corrections for each individual ion
*    charge state from the isonuclear sequence value are disabled for the
*    heat stress-viscosity.
*    This switch is subservient to b2tral_Zhdanov_closure.
*
*    'b2sifr_Zhdanov_tf_fr'     '1'                    Physics
*    If b2sifr_Zhdanov_tf_fr.eq.1, the Zhdanov expressions for the thermal
*    and friction forces are used.
*    This switch is subservient to b2tral_Zhdanov_closure.
*
*    'b2sifr_Zhdanov_test'      '0'                    Physics
*    If b2sifr_Zhdanov_test.eq.1, the Zhdanov-Grad calculation takes place,
*    but is not used. To be used for testing purposes.
*
*    'b2tfhi_Zhdanov_vel_heat'  '1'                    Physics
*    If b2tfhi_Zhdanov_vel_heat.eq.1, the Zhdanov velocity-dependent ion
*    heat flux is used.
*    This switch is subservient to b2tral_Zhdanov_closure.
*
*    'b2tral_zh_tf_toff'        '0'                    Physics
*    String of integers with (nspecies*(nspecies+1))/2-nspecies digits. If
*    a digit is {\tt 1}, turn off the thermal force between corresponding
*    species starting from 1 and 2; 1 and 3;....; up to nspecies-1 and
*    nspecies.
*    This switch is subservient to b2tral_Zhdanov_closure.
*
*    'b2tral_zh_vis_type'       '0'                    Physics
*    If b2tral_zh_vis_type.eq.0, the Braginskii-like velocity-dependent
*    stress-viscosity is used.
*    If b2tral_zh_vis_type.eq.1, the Zhdanov-like velocity-dependent
*    stress-viscosity is used.
*    If b2tral_zh_vis_type.eq.2, the Makarov-like velocity-dependent
*    stress-viscosity is used.
*
*    'b2npmo_smbvi_factor'      '1.0'                  Physics
*    This switch defines the fraction of the cross-ion part of the stress-
*    viscosity term which is applied. The cross-ion part of the stress-
*    viscosity term can lead to numerical problems. If so, it is
*    recommended to start from '0.0' and increase it step by step.
*    This switch is subservient to b2tral_Zhdanov_closure.
*
*    'b2tral_amfact_on'         '0'                    Physics
*    If b2tral_amfact_on.eq.1, the mass correction for the thermal and
*    friction force calculation is applied.
*
*    'b2tral_amfact_***'        '1.0'                  Physics
*    If b2tral_amfact_on.eq.1, this multiplier is applied for the mass of
*    the species corresponding to isonuclear sequence number '***'. Recall
*    the sequence numbering starts at zero (e.g. b2tral_amfact_000,
*    b2tral_amfact_001, ...).
*
*    'b2npmo_impr_form_fr'      '0'                    Physics
*    If b2npmo_impr_form_fr.eq.1, an improved analytical form for the
*    friction force is computed instead of the form corrresponding to
*    b2sigp_style.eq.2.
*    Used only if b2tral_Zhdanov_closure is equal to 0.
```

```
*
*    'b2npmo_impr_form_tf'      '0'                    Physics
*    If b2npmo_impr_form_tf.eq.1, an improved analytical form for the
*    thermal force is computed instead of the form corresponding to
*    b2sigp_style.eq.2.
*    Used only if b2tral_Zhdanov_closure is equal to 0.
*
*    'b2npmo mass multiplicator' '1.0'                 Physics
*    Using this number the impurity mass can be artificialy increased.
*    Used only if b2tral_Zhdanov_closure is equal to 0.
*
*    'b2sqel_artificial_radiation' '0.0'               Physics
*    If art_rad.ne.0, then an artificial radiation loss term is added to
*    the electron cooling rate. Art_rad represents the percent fraction of
*    impurities in the plasma.
*
*    'b2tral_Zhdanov_nc'        '0'                    Physics
*    If b2tral_Zhdanov_nc.eq.1, neoclassical corrections for impurity
*    transport are applied for a chosen impurity isonuclear sequence, with
*    index b2tral_Zhdanov_nc_spec.
*
*    'b2tral_Zhdanov_nc_spec'   '2'                    Physics
*    Index of the isonuclear sequence for which the impurity neoclassical
*    corrections will be applied. For Ne in case of a D+Ne plasma,
*    b2tral_Zhdanov_nc_spec should be set to 2.
*    The corrections are more accurate for impurity species with at least
*    Z=10.
*
*    'b2tral_Zhdanov_nc_min'    '6'                    Physics
*    b2tral_Zhdanov_nc_min sets the minimum ion charge Z^{min}_a, above
*    which (inclusively) neoclassical corrections will be applied for the
*    chosen impurity species by the switch b2tral_Zhdanov_nc_spec.
*    Lower values than 6 are not recommended.
*
*    'b2stbc_fheycore'          '0.0'                  Physics
*    'b2stbc_fhiycore'          '0.0'                  Physics
*    'b2stbc_fhiycore_kinetic_energy' '0'              Physics
*    'b2stbc_fchycore'          '-1.0e30'              Physics
*    'b2stbc_fnaycore'          '-1.0e30'              Physics
*    'b2stbc_isfeedback'        '0'                    Physics
*    'b2stbc_iyped'             jsep/2                 Physics
*    'b2stbc_ndes'              '0.0'                  Physics
*    'b2stbc_ndes_sol'          '0.0'                  Physics
*    'b2stbc_nepedm_sol'        '0.0'                  Physics
*    'b2stbc_nesepm'            '0.0'                  Physics
*    'b2stbc_nesepm_overshoot'  '0.0'                  Physics
*    'b2stbc_nesepm_pfr'        '0.0'                  Physics
*    'b2stbc_nesepm_sol'        '0.0'                  Physics
*    'b2stbc_private_flux_puff' '0.0'                  Physics
*    'b2stbc_volrec'            '0.0'                  Physics
*    'b2stbc_volrec_overshoot'  '0.0'                  Physics
*    'b2stbc_nesepm_minpuff'    '0.0'                  Physics
*    'b2stbc_nesepm_maxpuff'    '0.0'                  Physics
*    'eirene_nesepm_istra'      '-1'                   Physics
*    The 20 switches above are all feedback switches and require that
*    b2stbc_feedback.ne.0 (See Run section for details) or that
*    lfeedback=.true. in b2.boundary.parameters. The feedback is done as a
*    boundary condition and is under-relaxed using the b2stbc_...._alpha
*    switches (See Numerics section for details).
*    fheycore is the radial electron heat flow entering the core boundary.
*    fhiycore is the radial ion heat flow entering the core boundary.
*    fchycore is the radial current entering the core boundary. fnaycore is
*    the radial flux of species "isfeedback" entering the core boundary.
*    If fhiycore_kinetic_energy.eq.1, the parallel kinetic energy flux is
*    counted as part of the ion energy flux.
*    For double-null cases, the f..ycore variables contain the total flow
*    through both core boundaries taken together. The core boundaries are
```

```
*     identified by the coreregno and coreregn2 switches (see Geometry
*     section).
*     The other seven switches are density feedback switches: nesepm is the
*     outer midplane separatrix electron density. (See b2mwti_jxa switch in
*     Geometry section for setting of midplane position). This is controlled
*     through particle input of main plasma species isfeedback through the
*     core boundary (See Geometry section for the core boundary indices
*     coreregno and coreregn2).
*     nesepm_sol is the outer midplane separatrix electron density, but this
*     time controlled through a gas puff outside the SOL region. (See
*     b2mwti_jxa switch in Geometry section for setting of midplane
*     position). This is done through particle input of neutrals species
*     isfeedback through the SOL North boundary (See Geometry section for
*     the SOL North boundary index solregno). If nesepm_overshoot.gt.1.0,
*     the gas puff is turned off when the midplane separatrix density
*     reaches nesepm*nesepm_overshoot. Minimum and maximum values of the
*     feedback gas puff can be set using b2stbc_nesepm_minpuff and
*     b2stbc_nesepm_maxpuff
*     nepedm_sol is similar to nesepm_sol but the feedback is exerted on the
*     electron density at the top of the pedestal, along the outer midplane,
*     at radial position b2stbc_iyped. ndes_sol is the total particle
*     content from the isonuclear sequence of species 'isfeedback' over the
*     entire simulation domain.
*     It is governed, like nesepm_sol, by nesepm_overshoot and nesepm_alpha
*     and corresponds to a SOL boundary feedback. volrec_sol is the total
*     particle source due to volume recombination coming from Eirene. Its
*     feedback is governed by volrec_overshoot and volrec_alpha and
*     volrec_beta parameters (See Numerics section for the latter two).
*     nesepm_pfr is the outer midplane separatrix electron density, but this
*     time controlled through a gas puff outside the PFR region. (See
*     b2mwti_jxa switch in Geometry section for setting of midplane
*     position). This is done through particle input of neutrals species
*     isfeedback through the private flux boundaries (See Geometry section
*     for the PFR South boundary indices pfrregno1 and pfrregno2).
*     ndes is the total particle (neutrals + ions) content of the species
*     whose neutral species has index isfeedback. This is controlled through
*     particle input of neutrals species isfeedback through the
*     private flux boundaries (See Geometry section for the private flux
*     region boundary indices pfrregno1 and pfrregno2). private_flux_puff is
*     the particle flow of species isfeedback through the private flux
*     region boundary. This is controlled through particle input of neutrals
*     species isfeedback through the private flux boundaris (See Geometry
*     section for the private flux region boundary indices pfrregno1 and
*     pfrregno2).
*     All but the first of these feedback schemes ("nesepm") can also be run
*     though an Eirene neutral gas puff. In that case, one should set an
*     appropriate gas puffing stratum in the Eirene input file input.dat, as
*     well as in the b2.neutrals.parameters file. The code will assume,
*     unless told otherwise through the use of "eirene_nesepm_istra", that
*     the first stratum of type 'C' found in b2.neutrals.parameters is the
*     stratum used for the feedback. If "eirene_nesepm_istra" is set to a
*     positive number, the code will consider the feedback stratum to be the
*     stratum in the "eirene_nesepm_istra"-th position in
*     b2.neutrals.parameters.
*
*     'b2stbc_type13_ref'        '1'                   Physics
*     'b2stbc_type16_ref'        '1'                   Physics
*     'b2stbc_type20_ref'        '1'                   Physics
*     'b2stbc_type21_ref'        '1'                   Physics
*     'b2stbc_type16_kinetic_energy' '0'               Physics
*     These switches are associated with the boundary condition of type 13,
*     16, 17, 20 or 21 for particle flux, or electron and ion energy fluxes,
*     i.e. constant density or temperature giving a prescribed flux through
*     a radial surface, located type13_ref or type16_ref radial steps away
*     from the boundary of the computational domain, respectively, and
*     applied type20_ref steps away. See code and description in b2cdcn for
*     details.
```

```
*     Type 21 also applies to the electric potential boundary condition.
*     It scales the feedback strength according to the value of the variable
*     on the ring type21_ref steps inside.
*     When type16_ref.gt.1, then the position where the core flux tallies
*     are computed moves along with the boundary condition.
*     If type16_kinetic_energy.eq.1, then the energy flux due to kinetic
*     energy transport and viscous terms is included in the ion heat flux
*     component of type 16 and 17 boundary conditions.
*
*     'b2stbc_bcene_15_style'    '1'                    Physics
*     If bcene_15_style.eq.0, the electron sheath boundary condition
*     computes the electron current as the difference between the ion flow
*     and the poloidal current.
*     If bcene_15_style.eq.1 (recommended, default), the electron current at
*     the sheath is computed directly from the sheath potential drop. This
*     method has proven numerically more stable, especially in cases with
*     drifts.
*
*     'b2stbc_bcene_16_style'    '0'                    Physics
*     If bcene_16_style.eq.1, the boundary condition is enforced using the
*     modified fluxes fhe_mdf. Recommended when running cases with drifts.
*
*     'b2stbc_bceni_16_style'    '0'                    Physics
*     If bceni_16_style.eq.1, the boundary condition is enforced using the
*     modified fluxes fhi_mdf. Recommended when running cases with drifts.
*
*     'b2stbc_bc_ref'            '0.01'                 Physics
*     Under-relaxation factor used for feedback scheme in boundary condition
*     types BCCON = 16, 19, 22, 23, 26, and 27.
*
*     'b2stbc_bc_ref_te'         '0.01'                 Physics
*     Under-relaxation factor used for feedback scheme in boundary condition
*     type BCENE = 16 (with bcene_16_style.eq.1).
*
*     'b2stbc_bc_ref_ti'         '0.01'                 Physics
*     Under-relaxation factor used for feedback scheme in boundary condition
*     types BCENI = 16 (with bceni_16_style.eq.1), 24, and 27.
*
*     'b2stbc_secmodel'          '0'                    Physics
*     If secmodel.eq.1, then the secondary electron emission coefficient at
*     the plates is computed locally according to a kinetic model, otherwise
*     the default values of cbsch(7,ireg) and/or gammae are used.
*
*     'b2stbr_plate_model'       '0'                    Physics
*     'b2stbr_plate_option'      '3'                    Physics
*     'b2stbr_sput_chem_model'   '0'                    Physics
*     'b2stbr_sput_chem_cutoff_alpha' '1.0'             Physics
*     'b2stbr_sput_chem_cutoff_beta' '3.0'              Physics
*     'b2stbr_sput_mixed_alpha'  '1.0'                  Physics
*     'b2stbr_sput_mixed_beta'   '1.0'                  Physics
*     'b2stbr_sput_phys_model'   '1'                    Physics
*     'b2stbr_sputter_energy_on' '1'                    Physics
*     'b2stbr_sput_res'          '0.0'                  Physics
*     'b2stbr_therm_evap'        '0.0'                  Physics
*     'b2stbr_sput_dst'          '-1'                   Physics
*     'b2stbr_sput_dst2'         '-1'                   Physics
*     'b2stbr_sput_dst3'         '-1'                   Physics
*     'b2stbr_sput_frac_flag'    '0'                    Physics
*     'b2stbr_sput_frc'          '0.0'                  Physics
*     'b2stbr_sput_phys'         '0.0'                  Physics
*     'b2stbr_sput_src'          '1'                    Physics
*     'b2stbr_sput_phys_col'     '3'                    Physics
*     'b2stbr_alpha'             '0.25'                 Physics
*     'b2stbr_plate_temp'        '300.0'                Physics
*     'b2stbr_plate_thick'       '0.00'                 Physics
*     'b2stbr_redep_alpha'       '0.00'                 Physics
*     Sputtering model switches. See the code in b2stbr and the
```

```
*     b2mod_sputter module for specific implementation details and
*     references.
*     The default values represent the case of Graphite plates. Sput_src is
*     the atomic number of the plasma species which causes chemical
*     sputtering.
*     Sput_dst (when .ge.0) is the species index of the destination species
*     produced by physical sputtering, chemical sputtering and/or RES. If
*     either of the latter two processes are included, the chemical
*     sputtering and RES production rate calculations provided in the code
*     assume that sput_dst points to a Carbon species.
*     Sput_dst2 and sput_dst3 (when .ge.0) represent other species produced
*     by wall interactions for mixed materials scenarios.
*     Sput_frac_flag is the switch to turn on mixed materials scenarios
*     (when sput_frac_flag.eq.1).
*     Plate_model.eq.0 means the 0-D time-independent plate heating model
*     while plate_model.eq.1 indicates the 1-D time-dependent plate heating
*     treatment. Plate_model.eq.2 gives acces to a 2-D time-dependent model.
*     Plate_option chooses the initialisation of the plate temperature
*     profile. If plate_option.eq.1, the profile is set to the constant
*     given in plate_temp. If plate_option.eq.2, the profile is computed to
*     be the 0-D equilibrium profile. If plate_option.eq.3, the profile is
*     read from results of the previous run.
*     Sput_phys_model is a switch for choosing between the TRIM tables
*     (model 1, default) or an empirical formula (model 0). When TRIM data
*     is not available, the empirical formula is automatically used.
*     Extrapolations of low and high energy ranges beyond the TRIM table
*     data is done using the same physical dependencies as the empirical
*     formula. Sput_chem_model is a switch for the chemical sputtering model
*     used.
*     If sput_chem_model.eq.0 (default), the empirical formula is used.
*     If sput_chem_model.eq.1, a constant with a low energy cut-off is used.
*     The cutoff occurs at approximately sput_chem_cutoff_alpha and the
*     width is determined by sput_chem_cutoff_beta (the larger the value,
*     the narrower the width over which the transition from 0 to 1 occurs).
*     For model 0, sput_frc is a multiplier to the empirical formula, while
*     for model 1, sput_frc is the constant chemical sputtering yield.
*     Sput_frc is superseded by the chem_sput array from
*     b2.neutrals.namelist if the latter is used.
*     For neutrals species, we add a factor of \alpha*na*vbar to the
*     particle flux to the plate (used to compute chemical and RES
*     sputtering), where vbar is the average neutral particle speed.
*     Sput_phys turns on physical sputtering when sput_phys.gt.0.0 and
*     contains a (real) multiplier for the physical sputtering rate looked
*     up in the TRIM tables. It is superseded by the phys_sput array in
*     b2.neutrals.parameters if the latter is used.
*     sput_phys_col indicates which column to use in the TRIM table; the
*     default (3) corresponds to 30 degrees incidence. The columns are for
*       0 15 30 45 55 65 75 80 85
*     degrees. This angle is also used in the empirical formula. Plate_temp
*     is the temperature (in Kelvin) of the COOLED end of the divertor
*     plates. The surface temperature is computed self-consistently (using a
*     1-D description) according to the incident heat fluxes from the
*     plasma. The plate is assumed to have a thickness of plate_thick,
*     measured in metres. When plate_thick.eq.0 (default), the plate surface
*     temperature is assumed to be the same as plate_temp. Sput_res turns on
*     radiation enhanced sublimation (RES) when .gt.0.0 and is a multiplier
*     to the RES rate.
*     The switch sputter_energy_on turns on the energy contribution from the
*     sputtered particles (physical and chemical). If
*     sputter_energy_on.eq.0, the particles are sputtered cold, with no
*     energy contribution to the ion heat equation.
*     If sputter_energy_on.ne.0 (default), the chemically sputtered
*     particles are re-introduced into the plasma with the thermal energy
*     corresponding to the surface temperature of the material, while
*     physically sputtered particles are re-introduced with the TRIM-
*     calculated energy sputtering yield (if the TRIM model is used).
*     Therm_evap turns on thermal evaporation when .gt.0.0 and is a
```

```
*    multiplier to the thermal evaporation rate.
*    Redep_alpha is the multiplier to the reduction of the sputtering yield
*    due to prompt redeposition. The promptly redeposited species is
*    counted as eroded-then-deposited species. If redep_alpha is positive,
*    the formula used for the prompt redeposition fraction is from R. Dux
*    et al., Nucl. Fusion 51 (2011) 053002:
*      f_{prompt} = 1/(1 + \alpha*(l_{ion}/\rho_W)^2).
*    If redep_alpha is negative, the formula used is from D. Tskhakaya et
*    al., JNM 463 (2015) 624:
*      f_{prompt} = 1/(1 + abs(\alpha)*(l_{ion}/\rho_i)*(T_i/T_e) ).
*    In this case, the recommended value is abs(redep_alpha)=0.01. This
*    formula is meant for tungsten when l_{ion} is less than 5 * \rho_i.
*
*    'b2stbr_refl_model'        '1'                   Physics
*    'b2stbr_reflection_on'     '1'                   Physics
*    Reflection model switches. If the recycling coefficients (particle and
*    energy) are set to zero, then the code attempts to use, for surfaces
*    in contact with walls, the TRIM reflection data for particle and
*    energy reflection coefficients, when refl_model.eq.1. If
*    refl_model.eq.0, an empirical formula is used instead. The model
*    switch is subservient to reflection_on.ne.0. If reflection_on.eq.0,
*    then no reflection is used.
*    If reflection_on.eq.1 (default), then the incident species is
*    reflected unchanged. Otherwise, the reflected flux is of the
*    associated neutral species to the incident particles.
*
*    'b2mndr_coronal_model'     '0'                   Physics
*    Turns on the coronal model if coronal.ne.0. I.e., the reaction rates
*    of all reactions are made density-independent, and the value computed
*    at the lowest density (given in b2ar.dat) is used.
*
*    'b2mndr_hz'                '1.0'                 Physics
*    hz has been introduced into the new form of the parallel momentum
*    balance equation. If fac_hz = 0.0 then hz = 1 and old form of
*    equations is used. If fac_hz = 1.0 then new form of equations is used.
*
*    'b2stbr_bas_recycled_neutrals_contr' '1.0'       Physics
*    Introduced for nulling recycling energy when it is zero.
*
*    'b2tfhe_alfTeEh'           '0.0'                 Physics
*    When set to '1.0', the old form of the electron heat flux calculation
*    is used and b2tfhe_fch_pTe should be set to '0.0'. It is recommended
*    to use 0.0.
*    Cannot be set to 1.0 when using the SOLPS5.2 physics model.
*
*    'b2tfhe_fch_pTe'           '1.0'                 Physics
*    When set to '1.0', the new form of the electron heat flux calculation
*    is used and b2tfhe_alfTeEh should be set to '0.0'. It is recommended
*    to use 1.0.
*    Cannot be set to 0.0 when using the SOLPS5.2 physics model.
*
*    'b2tfnb_xcur'              '0.0'                 Physics
*    Xcur is a multiplier to the parallel viscosity, perpendicular
*    viscosity, heat viscosity, ion inertial and anomalous current
*    contributions to the ion poloidal flows (particle and energy).
*    If set to 0 (default), these currents are carried by electrons. If set
*    to 1, these poloidal currents are carried by the main hydrogenic ions.
*    The latter option can only be used if the plasma has a single majority
*    hydrogenic species. Otherwise, an error will be returned.
*
*    'b2tfnb_ycur'              '1.0'                 Physics
*    Ycur is a multiplier to the parallel viscosity, perpendicular
*    viscosity, heat viscosity, ion inertial and anomalous current
*    contributions to the ion radial flows (particle and energy).
*    If set to 0, these currents are carried by electrons. If set to 1
*    (default), these radial currents are carried by the main hydrogenic
*    ions. The latter option can only be used if the plasma has a single
```

```
*    majority hydrogenic species. Otherwise, an error will be returned.
*
*    'b2tfnb_vis_per'           '1.0'                  Physics
*    vis_per is a multiplier to the perpendicular viscosity current
*    contributions to the ion poloidal flows (particle and energy).
*    Subservient to b2tfnb_xcur and b2tfnb_ycur.
*
*    'b2tfnb_vis_q'             '1.0'                  Physics
*    vis_q is a multiplier to the heat viscosity current contributions to
*    the ion poloidal flows (particle and energy). Subservient to
*    b2tfnb_xcur and b2tfnb_ycur.
*
*    'b2tqce_fke_Zhdanov'       '1'                    Physics
*    When set to '1', the Zhdanov expression is used in the electron
*    thermal conductivity. It is recommended to use 'b2tqce_fke_Zhdanov'
*    '1'.
*    This switch is only active if, simultaneously, one has
*    b2tqce_model.eq.1 and b2tfhe_fch_pTe.eq.1.0.
*
*    'b2tqna_ixref'             See description (integer) Physics
*    Poloidal index (on the basis mesh) of the reference surface for the
*    flux-scaled transport model. The default value is the outer midplane,
*    computed as such (see b2mwti_jxa in Geometry section or
*    set_transport_ixref below):
*    Lower Single-null  : ixref=rightcut(1)-(rightcut(1)-leftcut(1))/4
*    Upper Single-null  : ixref=leftcut(1)+(rightcut(1)-leftcut(1))/4
*    Stellarator island : ixref=leftcut(1)+(rightcut(1)-leftcut(1))/2
*    Double-null        : ixref=(rightcut(1)+rightcut(2))/2
*    Limiter geometry   : ixref=nx/2
*    Straight geometry  : ixref=nx/2
*    The flux-scaled transport model is used in conjunction with scaling
*    factors listed in column (7) of the transport coefficients description
*    in input files b2ah.dat and b2mn.dat.
*
*    'b2tqna_user_transport'    '0'                    Physics
*    'set_transport_eta'        '2.0'                  Physics
*    'set_transport_eta_alpha'  '0.5'                  Physics
*    'set_transport_eta_floor'  '0.1'                  Physics
*    'set_transport_eta_ceiling' '10.0'                Physics
*    'set_transport_ixref'      See description (integer) Physics
*    'set_transport_iyref'      See description (integer) Physics
*    'set_transport_required_te_gradient' '5.0e4'      Physics
*    The last switch is subservient to b2tqna_user_transport, and only used
*    if user_transport.ne.0. When this is the case, the electron
*    perpendicular heat diffusivity and anomalous perpendicular particle
*    transport coefficient are scaled so as to maintain the
*    required_te_gradient (in units of eV/m), at the location (ixref,iyref)
*    on the basis mesh. The default position of the reference cell is on
*    the outer midplane, sligthly inside the separatrix, as follows:
*    |Configuration|     ixref                     |     iyref
*    --------------+-------------------------------+----------------------
*    |Single-null  | rightcut(1)-                  | 2*topcut(1)/3
*    | (lower)     | (rightcut(1)-leftcut(1))/4    |
*    |             |                               |
*    |Single-null  | leftcut(1)+                   | 2*topcut(1)/3
*    | (upper)     | (rightcut(1)-leftcut(1))/4    |
*    |             |                               |
*    |Stellarator  | leftcut(1)+                   | 2*topcut(1)/3
*    | island      | (rightcut(1)-leftcut(1))/2    |
*    |             |                               |
*    |Double-null  | (rightcut(1)+rightcut(2))/2   | 2*min(topcut(1),
*    |             |                               |       topcut(2))/3
*    |             |                               |
*    |Limiter      | nx/2                          | 2*topcut(1)/3
*    |             |                               |
*    |Straight     | nx/2                          | ny/2
*
```

```
*     The transport_eta switches are activated when user_transport.eq.3 and
*     allow modifications of the electron heat conductivity according to a
*     constant eta (Grad Te / Grad Ne) model. See routine set_transport_eta
*     code for details.
*     A model for disruption transport coefficients is available with
*     user_transport.eq.7. See routine set_transport_disruption code for
*     details.
*     The value user_transport.eq.8 activates the neoclassical model for
*     computation of the transport coefficients in the core. See routine
*     set_transport_neo and subroutines called for details. Implemented from
*     the NEOART package.
*     The location given by ixref and iyref can also serve as the anchor
*     point for the transport and/or sources profiles provided by the user
*     in the b2.transport.inputfile and b2.sources.profile files.
*
*     'b2tqna_max_df0'           '1e30'                 Physics
*     'b2tqna_min_df0'           '0.0'                  Physics
*     Upper and lower limits of the diffusivity coefficients df0 computed in
*     b2tqna for each neutral species.
*
*     'b2tqna_new_df0'           '0'                    Physics
*     When new_df0.eq.1, the neutral diffusivity is computed according to
*     the local charge exchange, ionisation and elastic collision rates,
*     instead of the standard form with constant cross-sections.
*
*     'b2tqna_ballooning'        '0.0'                  Physics
*     'b2tqna_ballooning_rescale' '1.0'                 Physics
*     'b2tqna_bb_ref'            See description (real) Physics
*     Ballooning model switches. If ballooning.ne.0, then all transport
*     coefficients in cell (i) are rescaled by a factor of
*     ballooning_rescale*abs(bb_ref/bb(i))**ballooning .
*     The default value for bb_ref is the arithmetic average of the total
*     magnetic field strength over the entire computational domain.
*
*     'b2tqna_vout_cnv'          '0.0'                  Physics
*     'b2tqna_pw0_cnv'           '1.0'                  Physics
*     'b2tqna_pw1_cnv'           '5.0'                  Physics
*     'b2tqna_pw2_cnv'           '2.0'                  Physics
*     'b2tqna_jsep_cnv'          jsep                   Physics
*     'b2tqna_jxa_cnv'           jxa                    Physics
*     Model parameters for the anomalous radial convection velocity
*     mimicking "blob" transport. The radial convection velocity for all
*     species in the SOL is calculated with the formula:
*     vla0=vout_cnv*y^pw0_cnv*(1-x^pw1_cnv)^pw2_cnv where x is the
*     normalized poloidal coordinate such that at the X-point it is equal to
*     1, and at jxa_cnv is 0, and y is the normalized radial coordinate,
*     such that at jsep_cnv it is equal to 0, and at the North boundary 1.
*     The six *_CNV parameters are only activated if 'b2tqna_user_transport'
*     is set to '6'.
*
*     'b2tqna_pfr_rescale'       '1.0'                  Physics
*     Scaling factor for all ion and electron transport coefficients inside
*     private flux regions.
*
*     'b2tqna_divsol_rescale'    '1.0'                  Physics
*     Scaling factor for all ion and electron transport coefficients inside
*     divertor SOL regions.
*
*     'b2tvspr_visper_all_ions'  '1'                    Physics
*     When set to 1 the subdivision of the viscosity ion perpendicular
*     current is applied for all ions. The SOLPS5.2 physics model is used.
*     Otherwise the viscosity ion perpendicular current will be non zero
*     only for main ions.
*
*     'b2tvspr_dia_lim_all_ions' '1'                    Physics
*     When set to 1 the limitation of the diamagnetic and ExB velocities is
*     applied in perpendicular viscosity current calculation. The SOLPS5.2
```

```
*     physics model is used.
*     Otherwise the limitation is not applied.
*
*     'b2sifr_phm0'              '1.0'                  Physics
*     Multiplier of the friction term between charged species (only ions in
*     case b2sigp_style.eq.2).
*
*     'b2sifr_phm1'              '1.0'                  Physics
*     If b2sigp_style.eq.2, multiplier to the friction force term between
*     electrons and ions.
*     Otherwise, multiplier of the ehxp term in the older expression for the
*     thermal force.
*
*     'b2sifr_phm2'              '1.0'                  Physics
*     Multiplier of the electron thermal gradient term in the thermal force
*     term and parallel current
*
*     'b2sifr_phm3'              '1.0'                  Physics
*     Multiplier of the ion thermal gradient term in the thermal force term.
*
*     'b2sifr_limthee'           '0.3'                  Physics
*     'b2sifr_limthii'           '0.3'                  Physics
*     'b2trcl_cthe'              '0.0'                  Physics
*     'b2trcl_cthi'              '2.65'                 Physics
*     Parameters for the computation of the thermal force term.
*
*     'b2trcl_lambda'            '-5.0'                 Physics
*     'b2tlnl_ee'                '0'                    Physics
*     'b2tlnl_ei'                '0'                    Physics
*     'b2tlnl_ii'                '0'                    Physics
*     If lambda is positive, the Coulomb logarithm is set to lambda.
*     If lambda is negative, then abs(lambda) is the lower bound to be used
*     when computing the Coulomb logarithm according to the NRL formulae.
*     The computation of the Coulomb logarithm takes place in b2tlnl.
*     The default formula is to use the Braginskii formulation, which will
*     give the same answer for electron-electron (ee), electron-ion (ei),
*     and ion-ion (ii) collisions. Respectively, non-default values of '1',
*     '2' and '3' for b2tlnl_ii, _ee, and _ei, respectively, make use of the
*     calculation according to Wesson.
*
*     'b2tfnb_PSch'              '1.0'                  Physics
*     Multiplier to the Pfirsch-Schlueter flows.
*
*     'b2news_BoRiS'             '0.0'                  Physics
*     The poloidal convective heat flux contains a prefactor of (3/2+BoRiS).
*     The default gives us the internal energy equation. The value BoRiS =
*     1.0 gives us the total energy equation.
*     Only for use to benchmark with codes using the total energy equation.
*     When used, the meaning of fhe and fhi changes from internal energy
*     fluxes to total energy fluxes.
*
*     'b2tfhe_conduction_only'   '0'                    Physics
*     When conduction_only.eq.1, convective heat transfer terms are turned
*     off. Only to be used when benchmarking against pure conduction cases.
*
*     'b2tfnb_alpha'             '0.0'                  Physics
*     'b2tfnb_gamma'             '2.0'                  Physics
*     'b2tfnb_flux_limit_min_ti' '0.0'                  Physics
*     Parameters for the flux limit to the convective neutral flow.
*     Alpha is a multiplier to the classical flux limit value.
*     The larger \alpha is, the weaker the flux limit is.
*     \gamma is the exponent used in the flux-limiting formula.
*     The smaller \gamma is, the stronger the flux limit is.
*     If alpha.eq.0, no flux limit is applied.
*     flux_limit_min_ti specifies the minimum ti to be used (in eV).
*
*     'b2tfnb_dia_lim_all_ions'  '1'                    Physics
```

```
*    When set to 1 the limitation of the diamagnetic velocity is applied
*    for all ions. The SOLPS5.2 physics model is used.
*    Otherwise the restriction is applied for all ions except of main ions
*    and the limit velocity is by the factor bx smaller.
*
*    'b2tlc0_alpha'             '1.0'                 Physics
*    'b2tlc0_gamma'             '1.0'                 Physics
*    Parameters for the flux limit to dpa0 - pressure driven neutral
*    diffusion.
*    Alpha is a multiplier to the classical flux limit value.
*    \gamma is the exponent used in the flux-limiting formula. If
*    alpha.eq.0, no flux limit is applied.
*
*    'b2tlh0_alpha'             '1.0'                 Physics
*    'b2tlh0_gamma'             '1.0'                 Physics
*    'b2tlh0_flux_limit_min_ti' '0.0'                 Physics
*    Parameters for the flux limit to the heat conductivity of the
*    neutrals. Alpha is a multiplier to the classical flux limit value.
*    The larger \alpha is, the weaker the flux limit is.
*    \gamma is the exponent used in the flux-limiting formula.
*    The smaller \gamma is, the stronger the flux limit is.
*    If alpha.eq.0, no flux limit is applied.
*    flux_limit_min_ti specifies the minimum ti to be used (in eV).
*
*    'b2tlmv_style'             '1'                   Physics
*    If style = 0 then the original form of the viscosity flux limit is
*    used, otherwise the SPb flux limit is used.
*
*    'b2sihs_phm0'              '1.0'                 Physics
*    Multiplier of the contribution to electron heat sources from
*    divergence(ue,ve). This term is superseded by the BoRiS switch if
*    invoked.
*
*    'b2sihs_phm1'              '1.0'                 Physics
*    Multiplier of the contribution to ion heat sources from
*    divergence(ua,va). This term is superseded by the BoRiS switch if
*    invoked.
*
*    'b2sihs_phm2'              '1.0'                 Physics
*    Multiplier of the contribution to ion heat sources from viscous
*    heating due to poloidal velocity differences. This term is superseded
*    by the BoRiS switch if invoked.
*
*    'b2sihs_phm3'              '1.0'                 Physics
*    Multiplier of the contribution to electron heat sources from Joule
*    heating (electron-ion friction).
*
*    'b2sihs_phm4'              '1.0'                 Physics
*    Multiplier of the contribution to ion heat sources from atom-atom
*    friction. This term is superseded by the BoRiS switch if invoked.
*
*    'b2sihs_phm5'              '1.0'                 Physics
*    Multiplier of the contribution to electron heat sources from
*    divergence of the electrical drift. That contribution is also
*    multiplied by facExB.
*
*    'b2sihs_phm6'              '1.0'                 Physics
*    Multiplier of the contribution to ion heat sources from divergence of
*    the electrical drift. That contribution is also multiplied by facExB.
*
*    'b2sihs_phm7'              '0.0'                 Physics
*    Multiplier of the contribution to heat sources from friction due to
*    diamagnetic velocities. Normally already included in 'phm3' term
*    above.
*
*    'b2sihs_phm8'              '1.0'                 Physics
*    Multiplier of the contribution to heat sources from viscous heating
```

```
*     due to radial velocity differences. This term is only included when
*     using b2nph9_style.eq.1 or b2npht_style.eq.1 (defaults). This term is
*     superseded by the BoRiS switch if invoked.
*
*     'b2sdia_facgt'             '0.0'                  Physics
*     Multiplier of the contribution to electron and ion heat sources from
*     divergence of the gradTxB heat flux. That contribution is also
*     multiplied by facdrift. This term is no longer needed as it has been
*     transferred from being a source to being included in the flows.
*
*     'b2sral_style'             '2'                    Physics
*     When set to '0' or '2', the code calls the standard b2stbc routine,
*     which uses the particle flux with drift terms included in the
*     expression of the electron particle flux (fne). When set to '1', the
*     code calls the b2stbc_spb routine instead, which uses the particle
*     flux without drift terms in fne. It is recommended '2'.
*
*     'b2sqcx_styl0'             '0'                    Physics
*     'b2sqcx_phm0'              '1.0'                  Physics
*     phm0 : Multiplier to the charge exchange rate coefficient. Subservient
*     to styl0: only used if styl0.eq.0 or is.lt.2. If styl0.ne.0, the
*     charge exchange rate coefficients for species of index 2 and above are
*     set to zero.
*
*     'b2sqel_phm0'              '1.0'                  Physics
*     Multiplier to the ionisation rate coefficient.
*
*     'b2sqel_phm1'              '1.0'                  Physics
*     Multiplier to the recombination rate coefficient.
*
*     'b2sqel_phm2'              '1.0'                  Physics
*     Multiplier to the heat loss rate coefficient.
*
*     'b2stel_phm0'              '1.0'                  Physics
*     Multiplier of the sources due to atomic physics.
*
*     'b2tfhe_lim_flux'          '0'                    Physics
*     If '0' and 'b2trcl_conductive_limit' is '1', the flux limit is not
*     applied directly to the electron heat flux but is applied through the
*     transport coefficients. This is the recommended default behaviour.
*     If '1' and 'b2trcl_conductive_limit' is '0', the flux limit is applied
*     directly to the electron heat flux.
*     If '0' and 'b2trcl_conductive_limit' is '0', no flux limit is applied.
*
*     'b2tfhi_lim_flux'          '0'                    Physics
*     If '0' and 'b2trcl_conductive_limit' is '1', the flux limit is not
*     applied directly to the ion heat flux but is applied through the
*     transport coefficients. This is the recommended default behaviour.
*     If '1' and 'b2trcl_conductive_limit' is '0', the flux limit is applied
*     directly to the ion heat flux.
*     If '0' and 'b2trcl_conductive_limit' is '0', no flux limit is applied.
*
*     'b2treq_phm0'              '1.0'                  Physics
*     Multiplier to the temperature equipartition term.
*
*     'b2tqca_phm0'              '1.0'                  Physics
*     Multiplier for the classical parallel viscosity.
*
*     'b2tqca_model'             '1'                    Physics
*     If model.eq.1, use the Balescu formulation from SOLPS5.2 classical
*     parallel ion heat diffusivity.
*     If model.eq.2, use the older Braginskii SOLPS4.0 model.
*     Note: old option model.eq.3 removed, replaced with model.eq.1., but
*     numerical treatment w.r.t. factor 4/3 according to old model.eq.3.
*
*     'b2tqce_model'             '1'                    Physics
*     If model.eq.1, use the fitted Balescu formulation from SOLPS5.0
```

```
*     classical parallel electron heat diffusivity.
*     If model.eq.2, use the older Braginskii SOLPS4.0 model.
*     If model.eq.3, use the 21-moment Balescu results.
*
*     'b2tqna_model_sig'         '0'                    Physics
*     If '1', use constant density in core region to calculate anomalous
*     conductivity sig0=dfsig*qe*ne(nmdpl,-1), nmdpl - number of midplane
*     cell. If '0', use sig0=dfsig*qe*ne(x,y)
*
*     'b2tral_cvsa_style'        '1'                    Physics
*     If '1', only the parallel ion classical contribution to the viscosity
*     is multiplied by 4/3. If '0', all components (ions and neutrals,
*     parallel and perpendicular) are multiplied by 4/3.
*
*     'b2trno_csig_an_style'     '1'                    Physics
*     If '1', the anomalous contributions in the parallel direction to the
*     electrical conductivity csig and the thermo-electric coefficient calf
*     are zeroed out.
*
*     'b2tqin_csigin_style'      '1'                    Physics
*     Integer switch to choose the expression for the ion-neutral
*     conductivity.
*     If '1' (default), the ion-neutral conductivity is computed from
*     hydrogen gas diffusivity (SOLPS5.2 treatment).
*     If '0', it is computed from the local CX rate (SOLPS5.0/5.1
*     treatments).
*
*     'b2trno_pol_anom_scale'    '1.0'                  Physics
*     If pol_anom_scale.ne.1, the poloidal anomalous transport coefficients
*     are scaled by the number. To get the same treatment as setting the old
*     switch (radial_only) to 1, set the new switch to 0.0.
*     This multiplication is to only take place for charged species.
*
*     'eirene_lhalpha'           '1'                    Physics
*     'eirene_lvib'              '1'                    Physics
*     Deprecated. Use "DEFINE_LINES" in block 12 of Eirene input file
*     instead.
*
*     'eirene_repeat_first_call' '1'                    Physics
*     If > 0 then repeats the first call to Eirene in eirene_mc so many
*     times.
*     Useful if a density feedback gas puff is being used because the flux
*     passed to Eirene on its first call is ignored (instead the value in
*     the Eirene input file is used).
*
*     'eirene_use_recyceir'      '1'                    Physics
*     If > 0 use recyceir (non species dependent) to specify the recycling
*     coefficients, else if 0 use recyc (species dependent).
*
*     'eirene_ionising_core'     '0'                    Physics
*     If <> 0 then recycles the neutral flux having crossed the core
*     boundary within Eirene as ions. The recycling is modulated as per the
*     poloidal density distribution of the ions, and neutrals come back as
*     fully-stripped ions.
*     'eirene_ionizing_core' is an alias for this switch.
*     If the value = 1, then the flux is added by direct modification of the
*     sources in the guard cells --- this will only work if a standard flux
*     boundary condition is applied at that boundary.
*     If the value is < 0, then the absolute value specifies which boundary
*     in b2.boundary.parameters is to be used. This will only work for the
*     BCCON = 13 type boundary condition, for which a negative value for
*     ionizing_core is required.
*     Cannot be used in conjunction with 'eirene_ank_mods'.
*     Use of this switch is not allowed if the core boundary condition is
*     not of a flux type. Moreover, certain boundary conditions expressly
*     include the flux of neutrals already (BCCON = 6, 19, 21, 23, 26, or
*     27), in which case use of this switch is redundant and should be
```

```
*    avoided.
*
*    'eirene_background'        '1'                    Physics
*    If eirene_background.eq.0, the ion velocities passed to Eirene to be
*    used for the collisions are based on grad-B and ExB drifts (vadia +
*    vaecrb).
*    If eirene_background.eq.1, these velocities contain the full
*    diamagnetic and ExB drifts (wadia + vaecrb).
*    Note: recycling fluxes are always computed based on grad-B and ExB
*    drifts only (and are not affected by this switch), because diamagnetic
*    drift flows largely close within the sheath.
*
*    'eirene_sheath_pot'        '1'                    Physics
*    If eirene_sheath_pot.eq.1, the sheath potential drop as computed by
*    B2.5 (i.e. including effects of parallel currents, secondary electron
*    emission, etc.) is passed to EIRENE to compute ion acceleration in the
*    sheath.
*    If eirene_sheath_pot.eq.0, the sheath potential drop is recomputed by
*    EIRENE, usually assuming zero current and secondary electron emission.
*
*    'b2mndr_atomic_physics_rescale' '0'               Physics
*    If atomic_physics_rescale is not 0, then the code will rescale the
*    rates stored in b2frates according to the multipliers given in the
*    b2.atomic_physics_rescale.parameters inputfile before making use of
*    them.
*
*    'neoclassical_ic'          '3'                    Physics
*    ..set the contribution ic in NEOART
*    0 --- classical particle flux
*    1 --- banana plateau contribution
*    2 --- Pfirsch-Schlueter contribution
*    3 --- both banana and PS
*    4 --- all contributions
*    mind that B2.5 already calculates the classical transport !
*    avoid double transport, 0+4 for cross-checks only !
*
*   *   1.4. Output switches
*
*    'b2mndr_b2time'            '1'                    Output
*    Specifies the number of timesteps between writes of the b2time.nc
*    time-dependent file. If b2time.gt.0, always writes out on the last
*    timestep. If b2mndr_stim.lt.0, data from the current run is appended
*    to the existing b2time.nc file, otherwise the file is overwritten.
*
*    'b2mndr_tally'             '1'                    Output
*    Specifies the number of timesteps between writes of tallies. If
*    tally.gt.0, always writes out on the last timestep.
*
*    'b2mndt_moitlv'            '-1'                   Output
*    Controls detailed monitoring output. moitlv.eq.-1 means no output,
*    moitlv.eq.0 means output once per timestep, moitlv.eq.1 means output
*    once per nstg0 iteration, moitlv.eq.2 means output once per nstg1
*    iteration and moitlv.eq.3 means output once per nstg2 iteration. See
*    'b2mndr_nstg?' description in Run section for more details.
*
*    'b2mndt_moqtlv'            '3'                    Output
*    Controls quick trace output. moqtlv.eq.-1 means no output, moqtlv.eq.0
*    means output once per timestep, moqtlv.eq.1 means output once per
*    nstg0 iteration, moqtlv.eq.2 means output once per nstg1 iteration and
*    moqtlv.eq.3 means output once per nstg2 iteration. See 'b2mndr_nstg?'
*    description in Run section for more details.
*
*    'b2mndr_mvnum'             '0'                    Output
*    Specifies the maximum number of instances at which movie data will be
*    output.
*
*    'b2mndr_mvinc'             '1'                    Output
```

```
*     Specifies the number of timesteps between movie output. If greater
*     than 1 and the last timestep does not correspond to a planned frame,
*     an extra endstate frame is added.
*
*     'b2mndr_plasnum'           '0'                    Output
*     Specifies the maximum number of instances at which extra writes of
*     b2fplasmf.xxxx will occur.
*
*     'b2mndr_plasinc'           '1'                    Output
*     Specifies the number of timesteps between b2fplasmf.xxxx writes.
*
*     'b2mndr_cdfmovietim'       '0.0'                  Output
*     If non-zero, the movie files b2movies.nc and eirenemovies.nc are
*     generated (the second one only in case of coupled runs), containing
*     several multidimensional quantities. Gives the real-time interval
*     between writes of movie frames in b2movies.nc and eirenemovies.nc. If
*     non-zero, must be at least as large as the simulation timestep.
*
*     'cdfmovie_b25_state_variables' '1'                Output
*     Controls the quantities outputted to b2movies.nc. If
*     cdfmovie_b25_state_variables.ge.1 then the plasma state variables (na,
*     ne, Te, Ti, etc.) over the B2.5 grid are outputted.
*
*     'cdfmovie_b25_fluxes'      '0'                    Output
*     Controls the quantities outputted to b2movies.nc. If
*     cdfmovie_b25_fluxes.ge.1 then the plasma fluxes (fna(x/y), fhe(x/y),
*     fhi(x/y), etc.) over the B2.5 grid are outputted.
*
*     'cdfmovie_b25_sources'     '0'                    Output
*     Controls the quantities outputted to b2movies.nc. If
*     cdfmovie_b25_sources.ge.1 then the sources calculated by B2.5 (i.e. by
*     atomic processes, electron cooling and line radiation) over the B2.5
*     grid are outputted.
*
*     'cdfmovie_eirene_state_variables' '0'             Output
*     Controls the quantities outputted to b2movies.nc. If
*     cdfmovie_eirene_state_variables.ge.1 then neutrals state variables
*     (densities and temperatures for atoms and molecules) over the B2.5
*     grid are outputted.
*
*     'cdfmovie_eirene_fluxes'   '0'                    Output
*     Controls the quantities outputted to b2movies.nc. If
*     cdfmovie_eirene_fluxes.ge.1 then the neutrals fluxes (particle and
*     energy fluxes for atoms and molecules) over the B2.5 grid are
*     outputted.
*
*     'cdfmovie_eirene_sources'  '0'                    Output
*     Controls the quantities outputted to b2movies.nc. If
*     cdfmovie_eirene_sources.ge.1 then the plasma sources due to plasma-
*     neutrals interactions (particle and energy sources for each fluid
*     species, and neutrals radiation from atoms and molecules) over the
*     B2.5 grid are outputted.
*
*     'cdfmovie_triangles'       '0'                    Output
*     Controls the quantities outputted to eirenemovies.nc. If
*     cdfmovie_triangles.ge.1 then particle density, energy density and x-,
*     y- and z-directed momentum density for atoms and molecules on the
*     Eirene triangular grid are outputted, in case of coupled runs.
*
*     'b2mndr_ntim_save'         '0'                    Output
*     Another option for plasma state file output. If greater than 0, gives
*     the number of B2.5 full interations between successive write-ups of
*     plasmastate files, in addition to the ones written at regular CPU
*     intervals (see b2mndr_savecpu and b2mndr_plasmatim). Should be used,
*     at the exclusion of other plasmastate write-up frequency settings, in
*     conjunction with the Leuven Monte Carlo averaging scheme.
*
```

```
*     'b2mndt_av'                '0'                    Output
*     If b2mndt_av.gt.0, turns on computation of running averages.
*
*     'b2mndt_av_continue'       '1'                    Output
*     If b2mndt_av_continue.gt.0, continuously perform running averages.
*
*     'b2mndt_av_ntim_batch'     '500'                  Output
*     If ntim_batch.gt.0, number of iterations used to compute batch
*     averages, written out in 'b2batch.nc'.
*
*     'b2mndt_av_ntim_run'       '1000'                 Output
*     If ntim_run.gt.0, number of iterations used for writing running
*     averages.
*
*     'b2mndt_av_batch_all'      '0'                    Output
*     If b2mndt_av_batch_all.ge.0, produces standard output for batch
*     averages.
*     If b2mndt_av_batch_all.eq.1, entire 2D fields of batch averages and
*     variances for ne, te, ... are written. Expert use only, check
*     $SOLPSTOP/doc/SOLPS-ITER_Eirene_averaging.pdf.
*
*     'b2mndr_plasmatim'         '0.0'                  Output
*     Another option for plasma state file output. If greater than 0, gives
*     the real-time interval between successive write-ups of plasmastate
*     files, in addition to the ones written at regular CPU intervals (see
*     b2mndr_savecpu and b2mndr_ntim_save). Must then be at least as large
*     as the simulation timestep.
*
*     'b2mndr_ids_save'          '0'                    Output
*     Frequency, in timesteps, at which an IDS is saved containing the
*     plasma state. Can be used in conjunction with b2mndr_ids_time and
*     b2mndr_ids_av.
*     If used, an IDS time slice will be written at the end of run.
*     If negative, a single IDS will be written at the end of the run.
*     Will not work if running a "0 timesteps" simulation, in which case the
*     use of b2_ual_write is recommended.
*
*     'b2mndr_ids_time'          '0.0'                  Output
*     Frequency, in simulation seconds, at which an IDS is saved containing
*     the plasma state. Can be used in conjunction with b2mndr_ids_save and
*     b2mndr_ids_av.
*     If non-zero, must be at least as large as the simulation timestep.
*     If used, an IDS time slice will be written at the end of run.
*     Will not work if running a "0 timesteps" simulation, in which case the
*     use of b2_ual_write is recommended.
*
*     'b2mndr_ids_av'            '0'                    Output
*     Frequency, in number of batch averages, at which an IDS is saved
*     containing the averaged plasma state. Can be used in conjunction with
*     b2mndr_ids_save and b2mndr_ids_time.
*     The averaged plasma state is saved as a separate occurrence of the
*     edge_profiles and edge_sources IDSs.
*
*     'b2wdat_iout'              '0'                    Output
*     If iout.eq.1, a large set of *.dat output files will be produced
*     containing the values of a variety of code quantities, evaluated at
*     the end of the B2.5 iteration.
*     If iout.eq.2, one gets a set of profiles at the midplane and the
*     targets.
*     If iout.eq.3, one gets a set of files providing the principal code
*     variables.
*     If iout.eq.4, an even larger set of output files will be produced, for
*     the purposes of a full run analysis, evaluated in the individual
*     routines where the quantities are used.
*     For even more detailed debugging analysis, one should instead use
*     "procedure_name"_iout.eq.1.
*     The files and their content are fully described in the
```

```
*     Output_description.pdf file in the $SOLPSTOP/doc directory and
*     Appendix G of the SOLPS-ITER manual.
*
*     'get_residuals'            '0'                    Output
*     If get_residuals.eq.1, then the residuals of unsolved equations will
*     be outputted into b2ftrace. Equations are unsolved if
*     'b2news_no_solve' is nonzero or any elements of the solveco, solvemo,
*     solveee, solveei, or solvepo are set to .false. .
*     By default, get_residuals.eq.0 and the residuals for unsolved
*     equations are set to zero.
*     The residual of the total momentum equation is nonzero when all
*     momentum equations for parallel velocities are solved together.
*     The residual of the total energy equation is nonzero when Te and Ti
*     equations are solved together.
*
*     'b2wdat_append'            '0'                    Output
*     If append.ne.0, the output files produced by the b2wdat_iout.eq.4
*     switch are appended upon every write, instead of being rewritten every
*     time.
*     For b2wdat_iout values 1 to 3, the value of append indicates the
*     frequency, in number of main B2.5 timesteps, at which the files are
*     appended. Data from the last timestep is always included.
*
*     'my_out_digits'            6 or 15                Output
*     Specifies the number of significant digits with which the *.dat files
*     are written out. Defaults to 6 in normal mode and 15 in debug mode.
*     Must be positive.
*
*     'b2mndr_old_style'         '0'                    Output
*     If old_style.gt.0, old-fashioned (SOLPS4 style) output is added at the
*     end of the b2mn.prt file.
*
*     'b2mndr_av_read'           '0'                    Output
*     If b2mndr_av_read.gt.0, the initial plasma is the averaged solution
*     read from the b2faveri file. Check $SOLPSTOP/doc/SOLPS-
*     ITER_Eirene_averaging.pdf for details.
*
*     'b2mndr_na_eps'            '1.0e19'               Output
*     'b2mndr_po_eps'            '1.0e+1'               Output
*     'b2mndr_te_eps'            '1.0e+1'               Output
*     'b2mndr_ti_eps'            '1.0e+1'               Output
*     'b2mndr_ua_eps'            '1.0e+4'               Output
*     The five switches above are safeguards numbers for when printing
*     changes after a time step. The change is computed as: deltaX =
*     abs((X(t)-X(t-1))/(X(t)+X_eps))
*
*     'b2mndr_trantim'           '0.0'                  Output
*     Produces a numbered 'tran' file every trantim real-time seconds. An
*     endstate file is written if it falls between scheduled write-up times.
*     Only available within the -DJET environment. If non-zero, must be at
*     least as large as the simulation timestep.
*
*     'b2mwti_target_offset'     '1'                    Output
*     The diagnostic values from b2time.nc use guard cell values if
*     target_offset.eq.0, and values from the neighbouring real cell if
*     target_offset.eq.1. Fluxes are not affected.
*
*     'b2mwti_2dwrite'           '0'                    Output
*     Controls additional output to b2time.nc.  If 2dwrite.ge.1 then a few
*     2d arrays (ne, Te, Ti) are written with each write to b2time.nc. If
*     2dwrite.ge.2 then fluxes, electric potential, kinetic energy, and
*     fluid particle and energy source terms are also included (e.g., rsana,
*     rsahi, rqrad).
*
*     'b2mwti_ismain0'           '0'                    Output
*     Index of the species used to create the 'dp3d?.last10' diagnostic
*     files.
```

```
*     Defaults to the neutral species associated with ismain. If no such
*     species is declared, the default will be set to ismain. Is expected to
*     either correspond to a neutral species or to ismain.
*     Can also be used in b2ar.dat to specify the index of the main neutral
*     CX species when resorting to the AMNS database (starting from version
*     1.3.2).
*     If ismain is also defaulted, then will be 0.
*
*     'b2mwqt_style'             '1'                    Output
*     Specifies the amount of data that is written out to b2ftrace. See the
*     manual (Section on b2yq) for full details.
*
*     'tallies_netcdf'           '0'                    Output
*     'b2stbr_b2wall_netcdf'     '0'                    Output
*     'balance_netcdf'           '0'                    Output
*     'balance_average'          '0'                    Output
*     'b2mndr_cdf_default'       '0'                    Output
*     If tallies_netcdf.ne.0, the file 'b2tallies.nc' is created, which
*     contains the regional tallies in CDF format. If b2mndr_stim.lt.0, data
*     from the current run is appended to the existing b2tallies.nc file,
*     otherwise the file is overwritten.
*     If b2mndt_av_ntim_batch.gt.0, the file 'b2batch.nc' is created, which
*     contains the batch averages of the tallies.
*     If b2wall_netcdf.ne.0, the file 'b2wall.nc' is created, which contains
*     the wall tallies in CDF format [written every b2wall_netcdf main
*     calls]. If b2mndr_stim.lt.0, data from the current run is appended to
*     the existing b2wall.nc file, otherwise the file is overwritten.
*     If balance_netcdf.ne.0, the file 'balance.nc' is created, which
*     contains all of the arrays required by the balance post-processing
*     routines, in CDF format.
*     If balance_average.ne.0, the balance arrays are averaged over the
*     number of completed timesteps (itim).
*     The NetCDF files are written using the format selected by means of the
*     b2mndr_cdf_default switch. The possible formats are:
*     0: local NetCDF library default
*     1: NetCDF classic format
*     2: 64 bit format
*     3: NetCDF4 format
*     4: NetCDF4 classic format
*     5: 64 bit data format.
*     Format availability depends on local NetCDF library implementation.
*
*     'ank_tracing'              '0'                    Output
*     If ank_tracing.ge.1, additional tracing output from Andrei Kukushkin
*     is produced, in files to be found in the tracing/ directory inside the
*     run directory. The traces will be written every ank_tracing iteration.
*     If b2mndr_stim.lt.0, data from the current run is appended to the
*     existing files, otherwise the files are overwritten.
*
*     'b2stbc_diagno'            '0'                    Output
*     Controls level of output in b2stbc and subservient routines.
*     Level 1 (diagno.ge.1) output includes wrong_flow warnings, transport
*     coefficients at the (nx/2,ny) basis mesh position, the solution found
*     for the given heat flux at constant temperature bc, and all feedback
*     diagnostics.
*     Level 2 output (diagno.ge.2) includes the integral current correction,
*     the boundary sources on the first call, excessive sound speeds, and
*     the iterative progress to the given heat flux at constant temperature
*     bc.
*     Level 3 output (diagno.ge.3) includes the boundary sources on each
*     call, and the boundary.prt file listing all boundary cells.
*
*     'b2stbr_output'            '0'                    Output
*     Output flag for the first_flight model. If output.ge.1, the files
*     start, end, and fort.88 are created, which contain information about
*     the chords used in the model.
*
```

```
*     'b2npmo_vlct_diagno'       '1'                    Output
*     Output flag to control diagnostics related to use of ion_vlct_restrict
*     switch.
*     If vlct_diagno.eq.0, the velocity restrictions are applied silently.
*     If vlct_diagno.eq.1 (default), the user only gets a count of how many
*     times the velocity restriction has been applied for each species, if
*     any.
*     If vlct_diagno.eq.2, the user gets the full details of where and how
*     large the applied velocity restriction was, if any.
*
*     'eirene_savef30'           '0'                    Output
*     'eirene_savef31'           '0'                    Output
*     For coupled runs, the code will create two files named fort.30 and
*     fort.31, containing the plasma geometry and plasma state,
*     respectively, for use by Eirene. This happens irrespective of the
*     setting of the two switches above. If however, the user wishes to
*     obtain these files for a non-coupled run, then these two switches can
*     be set to 1 and the files will then be produced at the end of the run.
*     If savef31.lt.0, then a fort.31.###### file is produced after every
*     B2.5 time step. Additionally, if savef31.ne.0, the fort.31 file will
*     be updated before every Eirene call for coupled runs.
*
*     'b2mndr_inverse_ua'        '0'                    Output
*     If inverse_ua.eq.1, the code will produce a 'b2fstati_with-ua' file
*     that contains the same plasma information, but with the opposite sign
*     convention for the parallel velocity. The run will then proceed with
*     the new velocities with their sign inverted.
*
*     'b2yrdr_ns'                ns                     Output
*     New number of species desired when reading a new atomic rates file
*     using b2yr.exe. Must be specified within b2yr.dat.
*     To be used for checking atomic rates after bundling and before any
*     plasma state files have been created or converted, i.e. when the
*     number of species in the atomic rates file b2frates does not match the
*     number of species declared in the run parameters file b2fpardf.
*
*     'b2srsm_diagno'            '0'                    Output
*     Controls level of output in b2srsm. If diagno.ge.2, output will be
*     given on every call. If diagno.eq.1, output will be given on main
*     calls only.
*
*     'b2ux5p_cpu'               '0'                    Output
*     'b2ux7p_cpu'               '0'                    Output
*     'b2ux9p_cpu'               '0'                    Output
*     'b2ux11p_cpu'              '0'                    Output
*     If cpu.gt.0, prints out the time spent in the matrix solver.
*
*     'b2ux5p_nltrsol'           '2'                    Output
*     'b2ux7p_nltrsol'           '2'                    Output
*     'b2ux9p_nltrsol'           '2'                    Output
*     'b2ux11p_nltrsol'          '2'                    Output
*     Output flag for the matrix solvers. Larger numbers mean increasing
*     output level.
*
*     'eirene_mc_output_style'   '1'                    Output
*     If nonzero, prints sources computed by Eirene. If greater than 1,
*     prints them on every use of the recycling sources, not just when they
*     are computed.
*
*     'ma28_nwrite'              '0'                    Output
*     If nwrite.gt.0, prints the content of the sparse matrix in the
*     b2_matrix file, for the first nwrite calls.
*
*     'b2news_ncallout'          '-1'                   Output
*     If the iteration number is equal to ncallout, then several output
*     files 'b2ne_npmo', 'b2ne_xppb', 'b2ne_npco', 'b2ne+nppo' which detail
*     the convergence behaviour and residuals.
```

```
*
*    'b2tqna_diagno'            '0'                    Output
*    If diagno.gt.0, outputs details of the calculations of neutral
*    transport coefficients when using the new_df0 model. See switch
*    b2tqna_new_df0 for more details.
*
*    'b2mndr_idout0'            'pgnl;pgmm;pzmm'       Output
*    'b2mndr_idout1'            'pzmm'                 Output
*    idout0 and idout1 specify the desired selection of output segments;
*    idout0 for output before the calculation and idout1 for output at the
*    conclusion of the calculation. The generic description of either of
*    these 'idout' variables follows. idout specifies the desired selection
*    of output segments for printed and graphical output. Associated with
*    each segment is a four-character word, which is identified on the
*    printed output. (The same word may be associated with several
*    segments.) A particular segment will be produced only if its
*    associated word has a match, as defined by the routine strmas, in
*    idout.
*
*    'eirene_format'            'iter'                 Output
*    This switch is to be provided within b2yt.dat to indicate to the b2yt
*    converter the format of the original Eirene input.dat file to be
*    modified. The accepted values are (case-insensitive):
*    'old' for input files from SOLPS4.0 and SOLPS5.0 runs using 'old'
*    Eirene_96
*    'new' for input files from SOLPS4.0 and SOLPS5.0 runs using 'new'
*    Eirene_99
*    'facelift' for input files from SOLPS5.1 runs
*    'juelich' for input files from Juelich Eirene versions (from 2008 to
*    2022)
*    'iter' for input files from SOLPS4.2/4.3 and SOLPS-ITER runs
*    'msv' for input files from the Eirene Milestone Version
*
*    'solps_version'            'iter'                 Output
*    This switch is to be provided within b2yt.dat to indicate to the b2yt
*    converter the format of the original fort.44 file to be modified. The
*    accepted values are (case-insensitive):
*    '4.3' for a fort.44 file produced from SOLPS4.3 runs
*    '5.0' for a fort.44 file produced from SOLPS5.0 runs
*    '5.1' for a fort.44 file produced from SOLPS5.1 runs
*    '5.2' for a fort.44 file produced from SOLPS5.2 runs
*    'iter' for input files from SOLPS-ITER runs (no conversion necessary)
*
*    'ids_from_43'              '0'                    Output
*    This switch is to be used when running the b2_ual_write utility
*    program to obtain an IDS from the SOLPS run results.
*    If set to 1, the IDS will indicate that the results come from a
*    converted SOLPS4.3 run with no additional calculation. By default, the
*    IDS will indicate a SOLPS-ITER run.
*
*   *   1.5. Numerics switches
*
*    'b2news_potit'             '50'                   Numerics
*    'b2news_potitmin'          '0'                    Numerics
*    Maximum and minimum number of iterations in the potential equation. It
*    must hold that potitmin < potit.
*
*    'b2news_potok'             '1.0e-2'               Numerics
*    Target residual for the potential equation.
*
*    'b2news_ramp_slow'         '0'                    Numerics
*    If ramp_slow.eq.0 (default), the ExB and diamagnetic drifts
*    multipliers are ramped on each call of b2news, otherwise they are only
*    changed on the first call of the nstg loop on the timestep.
*
*    'b2mndr_min_areshe'        '0.0'                  Numerics
*    'b2mndr_min_areshi'        '0.0'                  Numerics
```

```
*     'b2mndr_min_aresco'        '0.0'                  Numerics
*     Minimum residuals for stopping a run. All criteria must be met
*     simultaneously for the run to stop. The continuity equation residual
*     applies to the "ismain" species (See "Run" section).
*
*     'b2mndt_nstg_areshe'       '0.0'                  Numerics
*     'b2mndt_nstg_areshi'       '0.0'                  Numerics
*     'b2mndt_nstg_aresco'       '0.0'                  Numerics
*     Minimum residuals for an internal solution loop to stop.
*     Used in conjunction with (not in replacement of) the nstg(0:2)
*     numbers.
*     All nonzero criteria must be met simultaneously. The continuity
*     equation residual applies to the 'ismain' species (See 'Run' section).
*
*     'b2news_nsmin'             '0'                    Numerics
*     'b2news_nsmax'             ns                     Numerics
*     Allows for solving only for the species range [nsmin:nsmax-1].
*
*     'eirene_na_max'            '1e30'                 Numerics
*     'eirene_na_min'            '0.0'                  Numerics
*     'eirene_te_max'            '1e30'                 Numerics
*     'eirene_te_min'            '0.0'                  Numerics
*     'eirene_ti_max'            '1e30'                 Numerics
*     'eirene_ti_min'            '0.0'                  Numerics
*     'eirene_ua_max'            '+c'                   Numerics
*     'eirene_ua_min'            '-c'                   Numerics
*     Upper and lower bounds used when writing out data for Eirene.
*
*     'eirene_print_minmax'      '0'                    Numerics
*     Print min and max values of te, ti, na & ua
*
*     'eirene_extrap'            '1'                    Numerics
*     If eirene_extrap.eq.1, then guard cell plasma parameter values are
*     calculated from the neighbouring real cell values. If
*     eirene_extrap.eq.0, the guard cell values are used unchanged.
*
*     'eirene_ank_mods'          '0'                    Numerics
*     If ank_mods.ne.0, then uses an additional scheme to ensure particle
*     balance as the B2.5 solution evolves, due to the internal iteration
*     scheme, away from the plasma background on which the Eirene sources
*     were originally computed at the beginning of the time step. The user
*     is referred to the text in $SOLPSTOP/doc/Source_Scaling_in_B2.pdf for
*     a full description of the method used.
*     Cannot be used in conjunction with 'eirene_ionising_core'.
*     Should also not be used if Eirene is run in time-dependent mode.
*
*     'eirene_dpc_fix'           '1'                    Numerics
*     If dpc_fix.eq.1, then uses the true particle source from the stratum.
*     If dpc_fix.eq.2, sets this particle source to zero. The equivalent of
*     the old behaviour is dpc_fix.eq.0 and is wrong!
*
*     'b2stbr_sna_corr'          '0.0'                  Numerics
*             Now obsolete. This speed-up scheme using artificial sources is
*     now species-resolved, and its parameters are to be provided through
*     the numerics namelist. Please refer to the corresponding part of the
*     documentation.
*             Multiplier to the neutral source differential between puffing
*     and pumping strata for code speed-up.
*             See E. Kaveeva et al., Nucl. Fusion 58 (2018) 126018 for
*     details.
*
*     'b2stbr_eir_src_nhist'     '1'                    Numerics
*     If b2stbr_eir_src_nhist.gt.1, then Eirene sources are accumulated and
*     a moving average is computed. Note that only one of
*     b2stbr_eir_src_nhist, eirene_neutr_avg, eirene_underrelax or
*     eirene_averaging_multiple_calls may be used.
*
```

```
*     'eirene_neutr_avg'         '0'                    Numerics
*     If eirene_neutr_avg.gt.0, then Eirene sources are accumulated and
*     averaged until eirene_neutr_avg+1 steps, at which point the averaging
*     process is reset. Note that only one of b2stbr_eir_src_nhist,
*     eirene_neutr_avg, eirene_underrelax or eirene_averaging_multiple_calls
*     may be used.
*
*     'eirene_underrelax'        '0'                    Numerics
*     If eirene_underrelax.gt.0, then an underrelaxation scheme is used for
*     the Eirene sources, with an underrelaxation ratio of
*     1/eirene_underrelax. Note that only one of b2stbr_eir_src_nhist,
*     eirene_neutr_avg, eirene_underrelax or eirene_averaging_multiple_calls
*     may be used.
*
*     'eirene_averaging_multiple_calls' '1'             Numerics
*     If eirene_averaging_multiple_calls.gt.1, then Eirene is called
*     multiple times within a single time step, corresponding to the value
*     of this switch, before a new plasma solution is computed by B2.5, and
*     the Eirene sources are averaged over all performed calls before being
*     used to compute the new plasma solution. Note that this is different
*     from setting b2mndt_nstg1.gt.1, since in the latter case a new plasma
*     solution is computed immediately after one call of Eirene, with this
*     cycle being repeated b2mndt_nstg1 times. Also note that only one of
*     b2stbr_eir_src_nhist, eirene_neutr_avg, eirene_underrelax or
*     eirene_averaging_multiple_calls may be used.
*
*     'eirene_uub_style'         '0'                    Numerics
*     If eirene_uub_style.eq.0, compute poloidal velocities passed to Eirene
*     from ua, vadia, wadia, vaecrb arrays. If eirene_uub_style.eq.1,
*     compute poloidal velocities passed to Eirene from fna and fna_eir
*     fluxes.
*
*     'b2news_area_fix'          '3'                    Numerics
*     When area_fix.eq.0, recovers old SOLPS5.0 behaviour.
*     If area_fix.ge.1, then certain computations of poloidal velocities are
*     done by dividing flows by the area normal to the flux tube as opposed
*     to the area of contact.
*     If area_fix.ge.2, then additionally this treatment is used for the
*     parallel contact area used in determining plasma flux limiters.
*     If area_fix.ge.3, then additionally this treatment is used for all
*     parallel contact areas.
*
*     'b2npmo_ion_vlct_restrict' '0'                    Numerics
*     If ion_vlct_restrict.ne.0, the parallel speed of ions is limited to be
*     no more than 3 times the global plasma sound speed. In addition, if
*     ion_vlct_restrict.eq.2, the ExB ion speeds are limited to be no more
*     than twice the local ion sound speed times the local pitch angle.
*
*     'b2tfhe_vis_per'           '1.0'                  Numerics
*     vis_per is a multiplier to the perpendicular viscosity current.
*     If vis_per.ne.0.0, the electric potential equation is solved by the
*     subroutine b2npp7 using a 7-point stencil.
*     The value '1.0' is recommended for runs with drifts when perpendicular
*     viscosity and corresponding perpendicular viscosity current is taken
*     into account.
*
*     'b2stbc_sheath_drift_fix'  '1'                    Numerics
*     If sheath_drift_fix.eq.0, then the drift velocity used in the sheath
*     boundary conditions is the sum of diamagnetic and ExB contributions
*     (old, but likely wrong, treatment). If sheath_drift_fix.eq.1, then the
*     drift velocity used in the sheath boundary conditions is the ExB
*     velocity only (recommended).
*
*     'b2stbc_fix_fch_in_fhe_sheath' '2'                Numerics
*     If fix_fch_in_fhe_sheath.eq.0, then the wrong (old) implementation of
*     the FCH contribution to FHE used;
*     If fix_fch_in_fhe_sheath.eq.1, then the wrong (second) implementation
```

```
*    of the FCH contribution to FHE used;
*    If fix_fch_in_fhe_sheath.eq.2, then the correct implementation of the
*    FCH contribution to FHE used;
*
*    'b2stbr_potential_at_guard_cell' '1'              Numerics
*    If potential_at_guard_cell.eq.1, the electric potential used for the
*    computation of the incident energy for sputtering yields is taken as
*    the value in the guard cell. If potential_at_guard_cell.eq.0, the
*    value from the neighbouring real cell is used instead.
*
*    'b2trcl_conductive_limit'  '1'                    Numerics
*    When set to '1' (recommended default), the flux limit to the parallel
*    electron and ion heat fluxes is applied through the associated
*    transport coefficients.
*    If 'b2trcl_conductive_limit' is '0', then the keys 'b2tfhe_lim_flux'
*    and 'b2tfhi_lim_flux' are used to decide whether a flux limit is
*    applied, to the fluxes directly.
*
*    'b2trcl_core_cond_limit'   '0'                    Numerics
*    If core_cond_limit.eq.0, then the heat flux limits due to chvemx and
*    chvimx are not applied in the core. Switch only active if
*    'b2trcl_conductive_limit' .ne. 0.
*
*    'b2tlh0_flux_limit_style'  '2'                    Numerics
*    If '0', use the SOLPS5.0 scheme for neutral heat conductivity flux
*    limits.
*    If '1', SPb form to calculate parallel and perpendicular neutral heat
*    conductivity flux limit (does not preserve symmetry, kept for backward
*    compatibility reasons).
*    If '2', modifies the SPb treatment for the flux limits to be applied
*    on the transport coefficients directly.
*    It is recommended '2'.
*
*    'b2tral_mode'              '1'                    Numerics
*    Switch to choose between various interpolation schemes for transport
*    coefficients (Does not apply to pinch velocities vla or to the
*    temperature-driven electric conductivity alf).
*    mode.eq.-1: harmonic averaging
*    mode.eq.0 : geometric averaging
*    mode.eq.1 : arithmetic averaging (SOLPS5.0 formulation)
*    mode.eq.2 : arithmetic averaging (SOLPS4.0 formulation)
*
*    'b2tfhe_no_hybr'           '0'                    Numerics
*    'b2tfhi_no_hybr'           '0'                    Numerics
*    'b2tfnb_no_hybr'           '0'                    Numerics
*    'b2tfhe_hybr2'             '0'                    Numerics
*    'b2tfhi_hybr2'             '0'                    Numerics
*    Switches to toggle between the standard 5.0 hybrid scheme
*    (no_hybr.eq.0) and the old 4.0 upwind scheme (no_hybr.eq.1).
*    These apply separately to the electron heat, ion heat and particle
*    conservation equations.
*    If hybr2.ne.0, then a mixed scheme is used where only the conductivity
*    is computed according to the 4.0 scheme.
*
*    'b2tfhe_upwind'            '0'                    Numerics
*    'b2tfhi_upwind'            '0'                    Numerics
*    If upwind.ne.0, then the SOLPS4 upwind scheme in b2tfhe_ and b2tfhi_
*    to obtain the heat fluxes.
*
*    'b2tfnb_pflux_cor'         '1'                    Numerics
*    If pflux_cor.eq.1 and fna_mdf is used, enforces that the integral
*    particle flux across the domain boundaries computed from fna is equal
*    to the same integral computed from the fna_mdf fluxes.
*
*    'b2trcl_cvsa_mltpl'        '1.0'                  Numerics
*    Artificial coefficient providing faster convergence to the
*    neoclassical electric field but giving a distortion of the flows in
```

```
*     the SOL as a side-effect.
*     Can be applied <>1 during the convergence and turned off for the final
*     stage of calculations. Use with caution.
*
*     'b2trcl_csig_mltpl'        '1.0'                  Numerics
*     Multiplier to the electrical conductivity between guard cells along
*     the core boundary. A large value helps to damp numerical instabilities
*     in the potential when drifts are turned on and the electron density
*     and temperature are kept constant along the core boundary.
*
*     'b2ux5p_mult_nonzero'      '10'                   Numerics
*     'b2ux7p_mult_nonzero'      '10'                   Numerics
*     'b2ux9p_mult_nonzero'      '15'                   Numerics
*     'b2ux11p_mult_nonzero'     '15'                   Numerics
*     Number of expected nonzero matrix elements per matrix row.
*
*     'b2ux5p_mult_solvdim'      '15'                   Numerics
*     'b2ux7p_mult_solvdim'      '15'                   Numerics
*     'b2ux9p_mult_solvdim'      '15'                   Numerics
*     'b2ux11p_mult_solvdim'     '15'                   Numerics
*     'b2ux5p_mult_solvdim1'     '10'                   Numerics
*     'b2ux7p_mult_solvdim1'     '10'                   Numerics
*     'b2ux9p_mult_solvdim1'     '10'                   Numerics
*     'b2ux11p_mult_solvdim1'    '10'                   Numerics
*     Multipliers to the number of nonzero elements in the solution matrix
*     for workspace arrays in the matrix solver.
*
*     'b2ux5p_style'             '2'                    Numerics
*     Choose the type of 5-point stencil matrix solver. Style.eq.0 = iluter,
*     Style.eq.1 = slv5pt, Style.eq.2 = MA28.
*     NOTE: Only style.eq.2 will give good results. Other values are NOT
*     recommended! style.eq.0 and style.eq.1 are only applicable to linear
*     geometries with no cuts and no isolated regions.
*
*     'b2ux7p_style'             '2'                    Numerics
*     Choose the type of 7-point stencil matrix solver. Style.eq.0 = iluter,
*     Style.eq.1 = slv5pt, Style.eq.2 = MA28copy3. Style.eq.3 = SDRV from
*     YSMP.
*     NOTE: Only style.eq.2 will give good results. Applying style.eq.3
*     should be corrected and is no longer recommended. It might give slow
*     convergence or even divergence of the potential equation. Other values
*     are NOT recommended! style.eq.0 .or. style.eq.1 will return an error.
*
*     'b2ux9p_style'             '2'                    Numerics
*     Choose the type of 9-point stencil matrix solver. Style.eq.0 = iluter,
*     Style.eq.1 = slv5pt, Style.eq.2 = MA28copy.
*     NOTE: Only style.eq.2 will give good results. Other values are NOT
*     recommended! style.eq.0 and style.eq.1 will return an error.
*
*     'b2ux11p_style'            '2'                    Numerics
*     Choose the type of 11-point stencil matrix solver. Style.eq.0 =
*     iluter, Style.eq.1 = slv5pt, Style.eq.2 = MA28copy4.
*     NOTE: Only style.eq.2 will give good results. Other values are NOT
*     recommended! style.eq.0 and style.eq.1 will return an error.
*
*     'b2ux5p_acpar'             '8.0'                  Numerics
*     'b2ux7p_acpar'             '8.0'                  Numerics
*     'b2ux9p_acpar'             '8.0'                  Numerics
*     'b2ux11p_acpar'            '8.0'                  Numerics
*     Paremeter needed for iluter matrix solver.
*
*     'b2stbc_fchy_dia'          '0.0'                  Numerics
*     Multiplier to the neoclassical current and diamagnetic heat flux
*     convective boundary conditions, also multiplied by facdrift.
*     Adds a poloidal variation consistent with neoclassics and diamagnetic
*     contributions to the heat flux boundary conditions.
*     Should only be used with the 5.0 model and heat flux boundaries.
```

```
*     Allows use b2stbc_integral_current if fchy_dia.eq.0, forbids it
*     otherwise.
*
*     'b2stbc_fchy_dia_coreonly' '1'                    Numerics
*     If fchy_dia_coreonly.eq.1, the neoclassical current and diamagnetic
*     heat flux convective boundary conditions are only applied on the core
*     boundary.
*     If fchy_dia_coreonly.eq.0, those are applied to all boundaries,
*     including PFR, North SOL edge and diverted island center boundary.
*
*     'b2stbc_neoclassical'      '0.0'                  Numerics
*     Real parameter which establishes boundary conditions for radial
*     component of the current for North and South boundaries when these lie
*     on closed flux surfaces.
*     If b2stbc_neoclassical is 0 then the radial component of the current
*     is zero.
*     If b2stbc_neoclassical is not 0 the radial component of the current is
*     given by the neoclassical value. b2stbc_neoclassical is superseded if
*     facdrift.ne.0.
*     This cannot be used in conjunction with b2stbc_integral_current below.
*
*     'b2stbc_cbc'               '1.0'                  Numerics
*     Multiplier to the ExB velocity for sheath boundary conditions in
*     b2stbc_spb and BCMOM=13 case of b2stbc_phys.
*
*     'b2stbc_integral_current'  '0.0'                  Numerics
*     If integral_current.gt.0, corrects current sources so that there be no
*     surface current at the North and South edges of the edge plasma.
*     The value of integral_current multiplies the correction term added to
*     the current source.
*     This setting is incompatible with the normal use of diamagnetic drift
*     terms (facdrift.gt.0.0 .and. b2stbc_fchy_dia.ne.0.0) or neoclassical
*     boundary conditions (b2stbc_neoclassical.gt.0.0).
*
*     'b2stbr_first_flight_dl'   '0.001'                Numerics
*     Step length (in metres) for computing the first flight model chords.
*
*     'b2stbr_taumax'            '0.05'                 Numerics
*     Now obsolete. The speed-up scheme is now species-resolved, and its
*     parameters are to be provided through the numerics namelist.
*     Maximum allowed fraction of source to be added or substracted to the
*     real source in the method of effective sources for code speed-up.
*
*     'b2stbr_c_pumpavr'         '0.02'                 Numerics
*     Underrelaxation parameter to compute the artificial source in the
*     method of effective sources for code speed-up.
*
*     'b2stbr_first_flight_no_of_flights' '9'           Numerics
*     Number of chords started from each start point in the first flight
*     model.
*
*     'b2stbr_first_flight_table_size' '200000'         Numerics
*     Workspace size given to the first flight table.
*
*     'b2stbc_she0ep'            '1.0e-36'              Numerics
*     'b2stbc_shi0ep'            '1.0e-36'              Numerics
*     'b2stbc_sna0ep'            '1.0e-36'              Numerics
*     Small level sources introduced in all cells.
*
*     'b2stbc_bcpot_16_step'     '50'                   Numerics
*     Frequency (in number of calls to b2stbc_phys) at which the constant
*     value of the potential at the boundaries where BCPOT=16 is applied
*     will be recomputed.
*
*     'b2stbc_stab_coeff_sheath_te' '0.0'               Numerics
*     Stabilizing term to the source associated with boundary condition
*     BCENE=15, when bcene_15_style.eq.1 (default).
```

```
*    To help suppress electron temperature oscillations, this parameter can
*    be set to a positive value. The range 1-100 is recommended.
*
*    'b2stbc_stab_coeff_sheath_ti' '0.0'               Numerics
*    Stabilizing term to the source associated with boundary condition
*    BCENI=15.
*    To help suppress ion temperature oscillations, this parameter can be
*    set to a positive value. The range 1-100 is recommended.
*
*    'b2stbc_stab_coeff_sheath_ni' '0.0'               Numerics
*    Stabilizing term to the source associated with boundary condition
*    BCCON=14.
*    To help suppress ion density oscillations, this parameter can be set
*    to a positive value. The range 1-100 is recommended.
*
*    'b2mndr_na_min'            '1.0e4'                Numerics
*    Minimal density maintained in all cells for all species (in m^{-3}).
*    It must be true that na_min is smaller than na_new, as well as any of
*    the initial densities provided in b2ai.dat.
*    This switch replaces 'b2mndr_na0eps' from SOLPS5.x.
*
*    'b2mndr_na_new'            '1.0e14'               Numerics
*    Initial density (in m^{-3}) put in all cells for all new species if
*    not overwritten by initial state file. This switch replaces
*    'b2mndr_na0eps' from SOLPS5.x.
*
*    'b2upht_te_min'            '0.1'                  Numerics
*    Sets the minimum electron temperature value (in eV).
*
*    'b2upht_ti_min'            '0.1'                  Numerics
*    Sets the minimum ion temperature value (in eV).
*
*    'b2news_guard_flows'       '2'                    Numerics
*    If guard_flows.eq.0, flows between neighbouring guard cells are
*    blocked. The blocking fully applies only if b2mndt_style.eq.0.
*    Otherwise, use of b2news_l_fbound should be favoured.
*    If guard_flows.eq.1, flows between neighbouring guard cells are kept.
*    If guard_flows.eq.2, particle flows between neighbouring guard cells
*    are blocked for the density equations and the incorrect corner values
*    are replaced by interpolated values. Time derivative sources in guard
*    cells are turned off. It is recommended 2.
*
*    'b2news_l_fbound'          '0'                    Numerics
*    'b2news_l_enebound'        '0'                    Numerics
*    'b2news_l_timbound'        '0'                    Numerics
*    Flags to internally identify and isolate boundary cells to limit flows
*    between neighbouring guard cells.
*    If l_fbound.ne.0, flows between boundary conditions cells are turned
*    off.
*    If l_enebound.ne.0, remove non-boundary energy sources in guard cells.
*    If l_timbound.ne.0, time derivative sources are not added to guard
*    cells.
*
*    'b2news_fac_ref'           See description (integer) Numerics
*    'b2news_facdrift_tanh_a'   '0.0'                  Numerics
*    'b2news_facdrift_tanh_b'   '0.0'                  Numerics
*    'b2news_fac_ExB_tanh_a'    '0.0'                  Numerics
*    'b2news_fac_ExB_tanh_b'    '0.0'                  Numerics
*    'b2news_fac_vis_tanh_a'    '0.0'                  Numerics
*    'b2news_fac_vis_tanh_b'    '0.0'                  Numerics
*    'b2news_iy_nocoreExB'      '-2'                   Numerics
*    Parameters for spatial variation of the diamagnetic, ExB and viscosity
*    drift term multipliers. These are allowed to have tanh profiles
*    centered at tanh_a and of width tanh_b, where both quantities are in
*    metres and measured along the radial coordinate for the cell column
*    ix.eq.fac_ref (default value is jxa, see Geometry section for its
*    computation). The maximum value of facdrift and fac_ExB is controlled
```

```
*     by their respective _start and _target switches, see Run section for
*     details.
*     For consistency with the naming convention of related variables:
*     b2news_facExB_tanh_a is an alias for b2news_fac_ExB_tanh_a,
*     b2news_facExB_tanh_b is an alias for b2news_fac_ExB_tanh_b,
*     b2news_facvis_tanh_a is an alias for b2news_fac_vis_tanh_a, and
*     b2news_facvis_tanh_b is an alias for b2news_fac_vis_tanh_b.
*     Additionally, for all core cell rows where iy.le.iy_nocoreExB,
*     fac_ExB(ix,iy) is set to zero. This means that ExB contributions and
*     flows will be zeroed out in those cells.
*     For consistency with the boundary condition treatment, the value
*     iy_nocoreExB.eq.-1 is NOT recommended!
*
*     'nrings_for_no_speedup_averaging' '0'             Numerics
*     Number of poloidal rings in the confined region (counted from the
*     separatrix towards the core boundary) where the speed-up method of the
*     partial flux surface averaging (CORR_CORE_DN and CORR_CORE_DT) should
*     not be used.
*
*     'b2stbc_istyle_cur_contr_on_S_and_N' '2'          Numerics
*     When set to '2', SPb form of adding currents on the South core
*     boundary is included by using BCPOT=12, and on the South PFR and North
*     boundaries by using BCPOT=13. It is recommended '2', in conjunction
*     with the BCPOT=12 and BCPOT=13 settings, respectively.
*     When set to '1', SPb form of adding currents on South and North
*     boundaries is done explicitly in addition to other existing boundary
*     conditions for the potential equation.
*     The old 5.0 calculation is recovered by using the value '0'.
*
*     'b2stbc_istyle_fchi'       '0'                    Numerics
*     If '1', explicitly use the expression (bx*cs*na) of particle flux
*     (fna) from boundary condition instead of fna, for boundary condition
*     type BCPOT=11.
*
*     'b2news_recalculate_contributions' '1'            Numerics
*     If recalculate_contributions.eq.0, turns off recomputation of sources
*     when wrong_flow flag is active.
*
*     'b2news_no_b2sral_call'    '1'                    Numerics
*     If no_b2sral_call.eq.0, add an additional call to recompute the
*     sources in b2news_ to reproduce the behaviour from SOLPS4.
*
*     'b2news_do_2nd_b2npco_call' '0'                   Numerics
*     If do_2nd_b2npco_call.eq.1, perform a second call to the density
*     equation solve to improve particle balance, as done in SOLPS4.
*
*     'b2news_re_eval_prtls_fluxes' '0'                 Numerics
*     If re_eval_prtls_fluxes.eq.1, the particle fluxes are recomputed at
*     the end of b2news_. This is necessary for rescaling of the Eirene
*     sources during coupled runs, so this switch is superceded by
*     use_eirene, and also needed to reproduce SOLPS4 runs.
*
*     'b2mndt_style'             '1'                    Numerics
*     If  0, the SOLPS5.0 physics model is used, by calling the b2news
*     routine.
*     If  1, the SOLPS5.2 physics model is used, and b2news_ is called
*     instead.
*     If  2, the SOLPS5.2 physics model is used, and b2news_m is called
*     instead to run the code in fully self-consistent time-dependent mode.
*     B2news_m uses new order of solving equations: first the density
*     equations, then the parallel velocity, electrical potential and Te and
*     Ti equations.
*     If -1, corresponds to a special conversion run for SOLPS4.x data where
*     new fluxes are not computed and equations are not solved.
*
*     'b2mndt_ntim_step_out'     '1'                    Numerics
*     When set to a value of 'K', residuals at each K-th step will be
```

```
*     written in b2ftrace. It helps to avoid a huge b2ftrace files during
*     long time run.
*
*     'b2stbc_cbsnafac'          '0.01'                Numerics
*     'b2stbc_fchycore_alpha'    '0.1'                 Numerics
*     'b2stbc_fheycore_alpha'    '0.1'                 Numerics
*     'b2stbc_fhiycore_alpha'    '0.1'                 Numerics
*     'b2stbc_fnaycore_alpha'    '0.1'                 Numerics
*     'b2stbc_nesepm_alpha'      '1.0e-3'              Numerics
*     'b2stbc_nesepm_beta'       '0.0'                 Numerics
*     'b2stbc_nesepm_gamma'      '0.99'                Numerics
*     'b2stbc_volrec_alpha'      '1.0e-3'              Numerics
*     'b2stbc_volrec_beta'       '1.0'                 Numerics
*     'b2stbr_sput_chem_alpha'   '1.0e-2'              Numerics
*     'b2stbr_sput_phys_alpha'   '1.0e-2'              Numerics
*     Feedback relaxation parameters. See Physics section for individual
*     feedback quantities.
*
*     'b2stbc_type13_fac'        '1.0'                 Numerics
*     'b2stbc_type13_norm'       '1.0e15'              Numerics
*     Additional feedback strength adjustment parameters for BCCON=13
*     boundary condition, density adjusted to match a specific flux.
*     The density is adjusted by a factor of:
*     (1.0_R8+CONPAR(IS,IB,2)*(DFS-FS)/
*     (abs(DFS)+abs(FS)+abs(type13_norm)+abs(NAS*type13_fac)))
*     where FS is the current particle flux, DFS is the desired particle
*     flux, and NAS the current volume-averaged density. The desired flux
*     DFS is computed as
*     CONPAR(IS,IB,1)+CONPAR(IS,IB,3)
*     where the first number is provided directly by the user in
*     b2.boundary.parameters and the second is internally set to correspond
*     to the re-entry of ionised neutrals that have crossed into the core
*     for cases run with Eirene where eirene_ionising_core is activated.
*     See also the description of 'eirene_ionising_core'.
*
*     'heatdiff1D_linlog'        '1'                   Numerics
*     'heatdiff1D_ratio'         '100.0'               Numerics
*     Parameters related to solving the temperature equation for the target
*     plate elements in depth. If linlog.eq.1, a linear subdividing of the
*     plate element is used. If linlog.eq.2, a logarithmic subdividing is
*     used, with the surface layer being ratio times thinner than the last
*     layer in the bulk.
*     ratio must be larger than 1.
*
*     'b2tlh0_hcimx_flag'        '1'                   Numerics
*     When set to -1, only the gradient to the left/bottom is used to
*     compute the conductive neutral flux limits.
*     When set to 0, only the gradient to the left/bottom is used to compute
*     the conductive neutral flux limits, except when these do not exist
*     (the other face value is used).
*
*     'b2trno_flux_limit_to_dpa' '1'                   Numerics
*     If '1', flux limit to neutrals contribution to dpa0 - the diffusion
*     coefficient is applied in b2tlc0.F. It is recommended '1'.
*     b2tlc0.F has the flux limit parameters \alpha and \gamma which are
*     given by 'b2tlc0_alpha' and 'b2tlc0_gamma'. 'b2tfnb_alpha' and
*     'b2tlc0_alpha' cannot be different from zero simultaneously.
*     'b2tfnb_alpha' gives another form of flux limit which is applied to
*     the whole particle flux.
*
*     'b2mndr_isfb'              ismain                Numerics
*     'b2mndr_ixfb'              See description       Numerics
*     'b2mndr_iyfb'              ny/2                  Numerics
*     'b2mndr_ne_wanted'         '0.0'                 Numerics
*     'b2mndr_ne_wanted_time'    '0.0'                 Numerics
*     Experimental: allows for density feedback of the number of electrons
*     accompanying species 'isfb' at location (ixfb,iyfb).
```

```
*     For USN topologies, the default value for ixfb is 3*nx/8.
*     For all others, it is 5*nx/8.
*     Note that ixfb and iyfb are real numbers!
*
*     'b2mndt_use_b2srst'        '1'                   Numerics
*     Switches off the stabilization of the source coefficients.
*
*     'b2mndt_rxf'               See description       Numerics
*     Main under-relaxation parameter.
*     Should be set to 1.0 (default) if running in time-dependent mode
*     (b2mndt_style.eq.2).
*     Otherwise, default value is 0.5.
*     Setting this switch to zero prevents updating of the principal code
*     quantities.
*
*     'b2news_xfm0'              '1.0'                 Numerics
*     'b2news_xfm1'              '1.0'                 Numerics
*     'b2news_xfm2'              '1.0'                 Numerics
*     'b2news_xfm3'              '1.0'                 Numerics
*     Under-relaxation parameters for the parallel momentum, continuity,
*     potential and energy equations respectively.
*     Setting these switches to zero prevents updating of the associated
*     code quantities.
*
*     'b2npco_pcm0'              '1.0'                 Numerics
*     'b2npco_pcm1'              '1.0'                 Numerics
*     pcm0 specifies an additional under-relaxation factor that is applied
*     to the velocity correction. It is required that
*     0.le.pcm0.and.0.le.pcm1.
*     Not used if running in time-dependent mode (b2mndt_style.eq.2).
*
*     'b2npco_rxg'               '1.0'                 Numerics
*     rxg specifies a special under-relaxation parameter. A term of the form
*     "abs(residual)/rxg" is added to the diagonal of the matrix of the
*     correction equation, with the effect of limiting the computed
*     correction. rxg is dimensionless and of order unity; smaller values of
*     rxg imply stronger damping.
*     Not used if running in time-dependent mode (b2mndt_style.eq.2).
*
*     'b2npht_pcm0'              '1.0'                 Numerics
*     'b2npht_pcm1'              '1.0'                 Numerics
*     pcm0 specifies an additional under-relaxation factor that is applied
*     to the temperature correction.
*     It is required that 0.le.pcm0.and.0.le.pcm1.
*     Not used if running in time-dependent mode (b2mndt_style.eq.2).
*
*     'b2npht_rxg'               '1.0'                 Numerics
*     rxg specifies a special under-relaxation parameter. A term of the form
*     "abs(residual)/rxg" is added to the diagonal of the matrix of the
*     correction equation, with the effect of limiting the computed
*     correction. rxg is dimensionless and of order unity; smaller values of
*     rxg imply stronger damping.
*     Not used if running in time-dependent mode (b2mndt_style.eq.2).
*
*     'b2npht_style'             '1'                   Numerics
*     'b2nph9_style'             '1'                   Numerics
*     When set to '1', SPb form of the program b2sihs_ is called. It is
*     recommended '1'.
*
*     'b2npmo_rxg'               '1.0e6'               Numerics
*     Normalisation factor for the parallel momentum equation.
*     Not used when running in time_dependent mode (b2mndt_style.eq.2).
*
*     'b2news_poteq'             '1'                   Numerics
*     If poteq.eq.0, the potential equation is jumped over and not solved.
*     If poteq.eq.2, the potential is set to 3.1*Te/qe as per SOLPS4.0.
*     If poteq.eq.1, the potential equation is solved according to the
```

```
*    no_solve switch settings.
*    If poteq.ne.1, then 'b2tfhe_no_current' must be set to '1'.
*
*    'b2nxdv_style'             '1'                   Numerics
*    When set to '1', the total friction force cancel is not calculated at
*    the guard boundary cells.
*    It is recommended '1'.
*
*    'b2nxfc_style'             '1'                   Numerics
*    style specifies the precise form of interpolation used in the
*    computation of flcb and cvcb. When set to '1', SPb form of the
*    transport terms in the momentum correction equation is used.
*    It is recommended '1'.
*
*    'b2nxfx_style'             '1'                   Numerics
*    When set to '1', SPb form of an expression that occurs in the
*    electron-atom thermal force is used. Only used if b2sigp_style.ne.2.
*    It is recommended '1'.
*
*    'b2sifr_styl0'             '0'                   Numerics
*    'b2sifr_styl1'             '0'                   Numerics
*    Specify the type of linearisation used in the thermal force term.
*
*    'b2tfhe_mode_ehx'          '0'                   Numerics
*    Switch to choose between various discretization schemes when computing
*    the poloidal electric field.
*    See code for details.
*
*    'b2sigp_style'             '2'                   Numerics
*    When set to '1', SPb form of the pressure gradient term on the right
*    hand of the momentum balance equation is used.
*    When set to '2', the parallel current term contains a correction due
*    to impurities, when Z_eff is not equal to the average plasma ion
*    charge. Only usable if the potential equation is solved simultaneously
*    (b2news_poteq.eq.1).
*    The default value of '2' is recommended for cases with one main
*    hydrogenic species (or at least where the main species is the lightest
*    one) and any amount of impurities. It is not guaranteed to yield
*    correct results for hydrogen isotopic mixtures or non-hydrogenic
*    plasmas.
*
*    'b2sigp_pressure_restriction' '1'                Numerics
*    When set to '1', a CFL-like restriction is applied to the pressure
*    gradient term for minority ions and for neutrals species.
*
*    'b2xzdd_zero_dead_and_core' '1'                  Numerics
*    If 1 then zero passed sources in dead regions,
*    If 2 zero passed sources in dead regions and core boundary cells
*    If 0 then SKIP
*
*    'b2sihs_rf0'               '1.0'                 Numerics
*    'b2sihs_rf1'               '1.0'                 Numerics
*    'b2sihs_rf2'               '1.0'                 Numerics
*    'b2sihs_rf3'               '1.0'                 Numerics
*    'b2sihs_rf4'               '1.0'                 Numerics
*    rf0-4 are numerical parameters, see the code. All must be non-negative
*    and of order unity; larger values are meant to provide more
*    stabilisation. Some or all may be set to zero and removed eventually.
*    I have had some experience to indicate that rf0, rf2, rf4 may have a
*    useful function; however, later changes in the code may have made all
*    these parameters superfluous.
*
*    'b2sihs_style'             '0'                   Numerics
*    style determines the form of the strange electron-atom energy transfer
*    term.
*
*    'b2srdt_phm0'              '1.0'                 Numerics
```

```
*    'b2srdt_phm1'              '1.0'                  Numerics
*    'b2srdt_phm3'              '1.0'                  Numerics
*    'b2srdt_phm4'              '0.0'                  Numerics
*    'b2srdt_phm5'              '1.0'                  Numerics
*    Multipliers to the density, parallel momentum, heat, potential, and
*    electron particle time-derivative source terms, respectively.
*    'phm4' is normally zero since the potential equation contains no
*    source terms.
*
*    'b2srst_rf0'               '1.0'                  Numerics
*    'b2srst_rf1'               '1.0'                  Numerics
*    'b2srst_rf2'               '1.0'                  Numerics
*    'b2srst_rf3'               '1.0'                  Numerics
*    (rf0/1/2/3 for numerical stabilisation)
*
*    'b2stcx_rg0'               '1.0'                  Numerics
*    (rg0 for numerical stabilisation; needs experiments.)
*
*    'b2stcx_styl0'             '0'                    Numerics
*    Specifies the type of linearisation used in the charge exchange
*    momentum source term.
*
*    'eirene_mc_linearisation'  '1'                    Numerics
*    Specifies the type of linearisation used in the sources derived from
*    the Monte Carlo neutrals. If linearisation.eq.0, all sources are fed
*    to the constant term, if linearisation.eq.1, positive terms go in the
*    constant term, and negative terms in the proportional term.
*    'eirene_mc_linearization' is an alias for this switch.
*
*    'b2stbm_linearisation'     '1.0'                  Numerics
*    Same as above, but applied to the sources coming for an externally
*    coupled code providing plasma sources, for instance a kinetic
*    treatment of trace impurity ions. 'b2stbm_linearization' is an alias
*    for this switch.
*
*    'b2stbm_impgyro_mod'       '0'                    Numerics
*    Specifies the frequency (in units of full B2.5 timesteps) at which an
*    externally coupled code providing additional source (for instance a
*    kinetic treatment of trace impurity ions such as IMPGYRO) should be
*    called.
*
*    'b2stel_rg0'               '1.0'                  Numerics
*    'b2stel_rg1'               '1.0'                  Numerics
*    'b2stel_rxm0'              '0.0'                  Numerics
*    'b2stel_rxm1'              '0.0'                  Numerics
*    'b2stel_rxm2'              '0.0'                  Numerics
*    'b2stel_rxm3'              '0.0'                  Numerics
*    'b2stel_rxm4'              '0.0'                  Numerics
*    Damping coefficient applied to the atomic physics sources.
*
*    'b2stel_styl0'             '0'                    Numerics
*    Specifies the type of linearisation used in the charge exchange
*    momentum source term.
*
*    'b2tfcc_xfac'              '1.0'                  Numerics
*    Multiplier to the pressure force term.
*
*    'b2tfhe_mdf'               '0'                    Numerics
*    If '1', SPb new form of calculating electron heat flux is used. It is
*    recommended '1' for runs with drifts.
*
*    'b2tfhe_no_current'        '0'                    Numerics
*    If no_current.eq.1, all currents are set to zero. The setting
*    no_current.eq.1 must be used if the potential equation is not
*    explicitly solved for, i.e. for b2news_poteq.ne.1.
*
*    'b2tfhe_prl_cur'           '1.0'                  Numerics
```

```
*    Multiplier to the parallel current.
*
*    'b2tfhi_mdf'              '0'                   Numerics
*    If '1', SPb new form of calculating ion heat flux is used.
*    It is recommended '1' for runs with drifts.
*
*    'b2tfnb_drift_style'      See description       Numerics
*    When set to '0', drift velocities are calculated at cell centers (old
*    SOLPS5.0 style).
*    When set to '1', drift velocities are calculated at cell faces
*    (SOLPS5.2 style).
*    When set to '2', drift velocities are calculated at cell faces too,
*    but a better approximation of the drift velocities and corresponding
*    fluxes is used. This approximation guarantees that the divergences of
*    the ExB drift velocity and diamagnetic drift fluxes are numerically
*    zero in a constant B-field, which simplifies the physical analysis.
*    Setting 'b2tfnb_drift_style' to '2' requires that (and may be
*    activated only if) the b2fgmtry file was generated with
*    'b2agdr_redef_gmtry' set to '1' (default). If 'b2agdr_redef_gmtry' was
*    set to '0', the default value for 'b2tfnb_drift_style' is '1'.
*    It is recommended '2'.
*
*    'b2tfnb_PSch_style'       See description       Numerics
*    Controls the numerical approximation for the so-called Pfirsch-
*    Schlueter fluxes of particles and internal energy (divergent part of
*    diamagnetic drift in the SOLPS5.2 physics model).
*    Default value is '2', unless the b2fgmtry file was generated with
*    'b2agdr_redef_gmtry' set to '0' (old style), in which case, the value
*    is '1'.
*
*    'b2tiner_fch_inert_style' See description       Numerics
*    Controls the numerical approximation for the inertia current.
*    Default value is '2' unless the b2fgmtry file was generated with
*    'b2agdr_redef_gmtry' set to '0' (old style), in which case, the value
*    is '1'.
*    A value of '0' recovers the older behaviour (versions 3.0.6 or less).
*
*    'b2tiner_inert_all_ions'  '1'                   Numerics
*    When set to 1 the subdivision between all ions of the inertia current
*    is applied. The SOLPS5.2 physics model is used.
*    Otherwise the total inertia current (summed over all ions) is assigned
*    to main ions.
*
*    'b2tfnb_poleldr'          '1.0'                 Numerics
*    Multiplier to the poloidal electrical drift velocity.
*
*    'b2tfnb_radeldr'          '1.0'                 Numerics
*    Multiplier to the radial electrical drift velocity.
*
*    'b2tfnb_poldidr'          '1.0'                 Numerics
*    Multiplier to the poloidal diamagnetic drift velocity.
*
*    'b2tfnb_raddidr'          '1.0'                 Numerics
*    Multiplier to the radial diamagnetic drift velocity.
*
*    'b2tfnb_mdf'              '0'                   Numerics
*    If '1', SPb new form of calculating particle flux is used.
*    It is recommended '1' for runs with drifts.
*
*    'b2tfnb_xfrhie'           '1.0'                 Numerics
*    Multiplier to the Rhie and Chow upwind correction.
*
*    'b2upht_stylec'           '0'                   Numerics
*    (stylec is a numerical switch, needs experiments)
*
*    'b2usmo_cfc0'             '1.0'                 Numerics
*    Linearisation constant.
```

```
*
*    'b2usht_cor_diag'          '1.0'                  Numerics
*    'b2ush9_cor_diag'          '1.0'                  Numerics
*    Linearisation constant for the temperature equilibration term.
*
*    'b2srsm_enable'            '0'                    Numerics
*    If enable.ne.0, then the sources for particle, parallel momentum,
*    electron and ion heat are rescaled, for non-boundary cells, if the
*    local timestep exceeds the physics timescale implied by those sources.
*
*    'b2tinnt_style'            '1'                    Numerics
*    Integer switch to choose the numerical treatment of the ion pressure
*    gradient calculation for the ion-neutral current.
*    If b2tinnt_style.eq.0, then the chain rule is first applied to the
*    product of density and temperature, and (grad n)/n is rewritten as
*    grad(log(n)).
*    If b2tinnt_style.eq.1 (default), then the gradient of the product of
*    density and temperature is calculated directly.
*
*   *   1.6. Atomic Physics switches
*
*    'b2ardr_fix_cx'            '1'                    Atomic Physics
*    It is used to 'correct' the CX data
*    0 => do not fix
*    1 => only fix H if CX data is < 1e-40 [default]
*    2 => fix if CX data is < 1e-40
*    3 => fix H
*    4 => fix all
*    At the moment the only species with CX data in ADAS is C. Be careful
*    with options that use the fit formula other than for H.
*    See the comments in ratstr.F for the origin of the fit formula used to
*    'fix' the CX data.
*
*    'b2ardr_no_weisheit'       '0'                    Atomic Physics
*    When set to 1, disables the correction of the hydrogen atomic rate
*    data with the Weisheit data using ratwei.
*    *** Use with caution! ***
*
*    'b2ardr_no_smoothing'      '0'                    Atomic Physics
*    When set to 1, disables the tail smoothing for of rate coefficients to
*    represent the effect of a power-law tail on the distribution function.
*    It has been added for compatibility with SOLPS4.3
*
*    'adas_extrap'              '1'                    Atomic Physics
*    Switch for choosing extrapolation method of ADAS data:
*    If adas_extrap.eq.0, a simple 2-D linear extrapolation in log-log
*    space is used (may lead to spurious results).
*    If adas_extrap.eq.1, use the extrapolated result, but growing no
*    faster than the parameter variation to the 3/2 power (default).
*    If adas_extrap.eq.2, use the nearest domain edge value.
*
*    'b2ardr_fix_recomb'        '0'                    Atomic Physics
*    'b2ardr_zmax_recomb'       '1'                    Atomic Physics
*    When fix_recomb is changed from 0, and if the ADAS option is in use,
*    then uses the PRB file to add the contribution arising from is-->is-1
*    processes.
*    This term includes the bremsstrahlung.
*    The '0' option only contains the bremsstrahlung for the is-->is-1
*    process.
*    The correction is only applied for isonuclear sequences with nuclear
*    charge up to zmax_recomb, inclusive.
*    Use of this option is still experimental!
*
*    'b2ardr_rtnt'              '40'                   Atomic Physics
*    'b2ardr_rtnn'              '32'                   Atomic Physics
*    The number of intervals in the discretisation of the density (rtnn)
*    and temperature (rtnt) ranges for the atomic physics tables.
```

```
*
*    'b2mndr_dpc_mod_rates_ne_hot_frac' '0.0'          Atomic Physics
*    'b2mndr_dpc_mod_rates_te_hot' '0.0'               Atomic Physics
*    Modify the atomic rates by including a hot electron population of
*    temperature te_hot (in eV) and a population faction ne_hot_frac.
*    Still experimental.
*
*-----------------------------------------------------------------------
*.end b2cdci

      end subroutine b2cdci
```

A.5 b2cdcn.F

```
*-----------------------------------------------------------------------
* Generated by b2cdcn.py from b2input.xml. DO NOT EDIT THIS FILE!
*.specification

      subroutine b2cdcn ()
      implicit none

*-----------------------------------------------------------------------
*.documentation
*
*  1. purpose
*
*     B2CDCN provides information about namelists of the B2.5 code.
*     No computation is performed in this routine.
*
*     The namelists are listed in no particular order. Each description
*     will begin with the file from which the code expects to read the
*     namelist and its general purpose. Then, each namelist variable
*     will be listed with its use and size.
*
*-----------------------------------------------------------------------
*.text
*
* NAMELIST /NEUTRALS/
*    Found in b2.neutrals.parameters.
*    This namelist is used to either set parameters for the B2.5 neutral fluid
*    model or for the Eirene Monte Carlo neutrals run.
*    When coupling to Eirene, it is essential that the neutral boundaries
*    conditions declared here be in the same exact order than they are in the
*    Eirene input file.
*    For coupled cases, some of the parameters will become active only on the
*    second call to Eirene, while values inherited from the Eirene input file
*    will be used for the first call. In such cases, use of the switch
*    'eirene_repeat_first_call' is recommended.
*    This namelist is read if either 'b2stbr_neutrals_namelist' or
*    'b2mndr_eirene' are set to 1.
*    Some of the arrays are dimensioned by NTRACK, which is the maximum number
*    of tracked species in the mixed materials model. This parameter is set in
*    b2mod_layer and currently holds the value 4.
*  NSTRAI - integer. Number of neutral sources, or 'strata'. Must not be larger
*     than DEF_NSTRA from the
*     [$SOLPSTOP/modules/B2.5/]src/modules(.local)/b2mod_dimensions.F file.
*     Need not include the time-dependent stratum. If a time-dependent stratum
*     is used, is incremented internally as needed. The incremented value is
*     referred to as NSTRAT below.
*     Defaults to 0.
*  RCPOS - integer array of length (NSTRAT). Position in the B2.5 grid of the
*     strata. Similar use as BCPOS from /BOUNDARY/ namelist.
*     Defaults to -2.
*  RCSTART - integer array of length (NSTRAT). Start coordinate on the B2.5
*     grid the strata. Similar use as BCSTART from /BOUNDARY/ namelist.
*     Defaults to -2.
*  RCEND - integer array of length (NSTRAT). End coordinate on the B2.5 grid of
*     the strata. Similar use as BCEND from /BOUNDARY/ namelist.
*     Defaults to -2.
*  RC_LIST_SIZE - integer array of length (NSTRAT). Contains the size of the
*     recycling boundary lists. Similar use as BC_LIST_SIZE from /BOUNDARY/
*     namelist.
*     Defaults to 0.
*  RC_LIST_X - integer array of length (2*(NXD+NYD),NSTRAT). Contains the
*     X-coordinate of the cells where recycling boundaries are applied. Similar
*     use as BC_LIST_X from /BOUNDARY/ namelist.
*     Defaults to -2.
*  RC_LIST_Y - integer array of length (2*(NXD+NYD),NSTRAT). Contains the
```

```
*     Y-coordinate of the cells where recycling boundaries are applied. Similar
*     use as BC_LIST_Y from /BOUNDARY/ namelist.
*     Defaults to -2.
*  TARGSP - integer array of size (NSTRAT,NTRACK). Identifies the base
*     material(s) of this stratum wall. The number corresponds to the B2.5
*     species index produced by sputtering. Check b2cdci for details. If the
*     bulk species is not sputtered, should contain -1.
*     Defaults to b2stbr_sput_dst.
*  CHEMSP - logical array of length NSTRAT. Indicates whether chemical
*     sputtering is allowed from this wall stratum.
*     Defaults to '.false.'.
*  RECYC - real array of size (0:NS-1,NSTRAT). Stores the particle recycling
*     coefficients of species (is) on stratum (istra). Defaults to 1.0 for non-
*     carbon species and 0.0 for carbon species. Particles recycle into the
*     neutral species associated to their isonuclear sequence.
*     Applies to B2.5 neutral fluid species.
*     Also multiplies Eirene recycling fluxes if 'eirene_use_recyceir' is set
*     to 0 (see b2cdci for details).
*  MRECYC - real array of size (0:NS-1,NSTRAT). Stores the parallel momentum
*     recycling coefficients of species (is) on stratum (istra). The parallel
*     momentum is carried back by the recycled neutrals. Applies only to B2.5
*     neutral fluid species.
*     Defaults to 0.0.
*  ERECYC - real array of size (0:NS-1,NSTRAT). Stores the energy recycling
*     coefficients of species (is) on stratum (istra). Defaults to 0.3 for non-
*     carbon species and 0.0 for carbon species. The energy is carried back by
*     the recycled neutrals.
*     Applies only to B2.5 neutral fluid species.
*  RCION - real array of size (0:NS-1,NSTRAT). Stores the particle-into-ion
*     recycling coefficients of species (is) on stratum (istra). Particles
*     recycle into the next ionised species associated to their isonuclear
*     sequence. Applies only to B2.5 neutral fluid species.
*     Defaults to 0.0.
*  RECYCEIR - real array of size (NSTRAT). Multiplier to the Eirene recycling
*     fluxes from stratum (istra). Only used if 'eirene_use_recyceir' is set to
*     1 (default).
*     Defaults to 1.0.
*  USERFLUXPARM - real array of size (NSTRAT,2). The element (istra,1) contains
*     the strength of constant gas puff strata (type 'C') for Eirene in
*     particles/second. Its values overwrite the values provided by the FLUX
*     variables from the strata description in block 7 of the Eirene input file.
*     Defaults to 0.
*  CRCSTRA - character*1 array of length (NSTRAT). Contains the type of stratum
*     for Eirene. Possible options include:
*     'N','S','W','E' - topological mesh boundaries (same use as BCCHAR in
*     /BOUNDARY/ namelist)
*     'V' - volume recombination source (related RCPOS, RCSTART, RCEND and
*     RC_LIST variables are ignored).
*     'C' - constant or feedback gas puff source (related RCPOS, RCSTART, RCEND
*     and RC_LIST variables are ignored). Gas puffs for B2.5 fluid neutrals are
*     handled via b2.boundary.parameters. Unless specified otherwise by use of
*     'eirene_nesepm_istra', it is the first 'C' stratum that is used for
*     feedback puff schemes. See b2cdci for details.
*     'T' - time-dependent source for Eirene (related RCPOS, RCSTART, RCEND and
*     RC_LIST variables are ignored). Long-lived Eirene neutrals are stored in
*     this stratum. See EIRENE_STEP_DT below.
*     Defaults to ' '.
*  RF_NEUT - real array of size (4). Relaxation parameters multiplying the
*     Eirene particle, parallel momentum, electron energy and ion energy sources
*     respectively before use in B2.
*     Defaults to 1.0.
*  PHYS_SPUT - real array of size (0:NS-1,NSTRAT). Stores the physical
*     sputtering multiplier for B2.5 species (is) sputtering on the wall of
*     stratum (istra). Must be non-zero for physical sputtering to occur. Only
*     applies if using B2.5 fluid neutral model.
*     Defaults to 0.0.
*  CHEM_SPUT - real array of size (0:NS-1,NSTRAT). Stores the chemical
```

```
*     sputtering multiplier for B2.5 species (is) sputtering on the wall of
*     stratum (istra). Must be non-zero for chemical sputtering to occur. Only
*     applies if using B2.5 fluid neutral model.
*     Defaults to 0.0.
*  EIRENE_STEP_CPU - real. Length of CPU time devoted to Eirene calls after the
*     first one (in s). If zero, the value given in the Eirene input file will
*     be used.
*     Defaults to 0.0.
*  EIRENE_STEP_DT - real. Physical time (in s) the Eirene particles are to be
*     followed before being passed to the time-dependent stratum. If zero, the
*     value given by DTIMV in block 13 of the Eirene input file will be used.
*     Defaults to 0.0.
*  EIRENE_MOD - integer. Frequency of Eirene calls. Eirene is called every
*     EIRENE_MOD full B2.5 iterations.
*     Defaults to 1.
*  VOLRECSTART - real array of size (NSTRAT). Initial volume recombination rate
*     from stratum (istra) to be used for the volume recombination scaling
*     regulation algorithm. Rendered obsolete by 'eirene_dpc_fix'.
*     Defaults to 1.e21.
*  VOLRECINC - real. Maximum rate of increase of the volume recombination
*     source strength. The volume recombination regulation scaling scheme is
*     activated if VOLRECINC is neither 0 nor 1.
*     Rendered obsolete by 'eirene_dpc_fix'.
*  VOLRECWT - real. Weight of new volume recombination strength relative to
*     that of previous iteration in the volume recombination regulation scaling
*     scheme.
*     Rendered obsolete by 'eirene_dpc_fix'.
*     Defaults to 0.1.
*  SPECIES_START - integer array of size (NSTRAT). Specifies the index of the
*     first B2.5 species involved in stratum (istra) for EIRENE.
*     Defaults to 0.
*  SPECIES_END - integer array of size (NSTRAT). Specifies the index of the
*     last B2.5 species involved in stratum (istra) for EIRENE.
*     Defaults to ns-1.
*  NEUTRALS_FILENAME - character*256. Name of the next file to use for reading
*     a new /NEUTRALS/ namelist.
*     Defaults to b2.neutrals.parameters.
*  NEUTRALS_TIME_MOD - real. When the B2.5 run simulation time, in seconds,
*     modulo(NEUTRALS_TIME_MOD), reaches or exceeds NEUTRALS_TIME_SWITCH, reads
*     the new namelist from NEUTRALS_FILENAME. Also switches to the new namelist
*     if the ELM count (here time/NEUTRALS_TIME_MOD) changes. Only active if
*     NEUTRALS_TIME_MOD is greater than 0.
*     Defaults to 0.0.
*  NEUTRALS_TIME_SWITCH - real. If NEUTRALS_TIME_MOD is greater than 0, time
*     (in seconds) within an ELM cycle at which a new NEUTRALS namelist is read.
*     Otherwise, B2.5 simulation time, in seconds, when to read the next
*     NEUTRALS namelist file from NEUTRALS_FILENAME.
*     Only active if NEUTRALS_TIME_SWITCH is greater than 0.
*     Defaults to 0.0.
*  L_NEUTRAD - integer. If l_neutrad >= 0, then the radiated power due to the
*     neutrals atoms is taken directly from the Eirene calculation, instead of
*     being recomputed by B2.
*     Defaults to 0.
*  L_NEUTFLUX - integer. If l_neutflux >=0, then correct treatment of the
*     incident fluxes in B2.5 and b2plot; if <0, then old (approximate)
*     treatment
*     Defaults to 0 for coupled cases, -1 otherwise.
*  LSTRASCL - integer array of size (NSTRAT,0:natm). Indicates with which
*     Eirene atomic species to scale the Eirene stratum (istra) (0 means no
*     scaling, default). Atomic species 0 stands for electrons.
*  B2EATCR - integer array of size (0:NS-1). Contains the index of the Eirene
*     atomic species corresponding to the B2.5 species (is).
*     Defaults to ordering of the B2.5 isonuclear sequences.
*  B2ESPCR - integer array of size (0:NS-1). Contains the isonuclear sequence
*     index of the B2.5 species (is).
*     Defaults to ordering of the B2.5 isonuclear sequences.
*  EB2ATCR - integer array of size (NATM). Contains the index of the B2.5
```

```
*     neutral fluid species corresponding to the Eirene atomic species (iatm).
*     Defaults to first B2.5 species of each isonuclear sequence.
*  EB2SPCR - integer array of size (NSPECIES). Contains the index of the first
*     B2.5 fluid for each species.
*     Defaults to first B2.5 species of each isonuclear sequence.
*  LATMSCL - integer array of size (NATM). Contains the index of the B2.5
*     isonuclear sequence with which the Eirene atomic species (IATM) should be
*     scaled.
*     Defaults to assuming one-to-one match between Eirene and B2.5 atomic
*     species.
*  LMOLSCL - integer array of size (NMOL). Contains the index of the Eirene
*     atomic species with which the Eirene molecular species (IMOL) should be
*     scaled.
*     Defaults to 0.
*  MLCMP - integer array of size (NATM,NMOL). Contains the number of atoms from
*     Eirene atomic species IATM within each molecule of Eirene molecular
*     species IMOL.
*     Defaults to 0.
*  LIONSCL - integer array of size (NION). Contains the index of the Eirene
*     atomic species with which the Eirene test ion species (IION) should be
*     scaled. If not provided, LIONSCL will map to LMOLSCL for molecular ions
*     that match an existing declared molecule, or to the first species listed
*     in the name of the molecular ion as declared in the Eirene input file if
*     no matching molecule is found.
*     Defaults to 0.
*  LCNS - integer array of size (NSTS). Contains the indices of Eirene non-
*     standard surfaces (block 3A in input.dat) corresponding to core
*     boundaries.
*     Defaults to 0.
*  LTNS - integer array of size (NSTS). Contains the indices of Eirene non-
*     standard surfaces (block 3A in input.dat) corresponding to target
*     boundaries.
*     Defaults to 0.
*  LSNS - integer array of size (NSTRAT*NSRFS). Contains the indices of Eirene
*     surfaces related to the recycling sources.
*     Defaults to 0.
*  KSNS - integer array of size (NSTRAT). Contains the number of Eirene
*     surfaces for each Eirene recycling stratum.
*     Defaults to 0.
*  GPFC - real array of size (NATM,NSTRAT). Specifies the fraction of Eirene
*     atomic species (iatm) in one particle puffed from stratum (istra).
*     Defaults to 0.0.
*  DBG_EIR_MC - integer. Debug output control for eirene_mc routine. See code
*     for usage.
*     Defaults to 0.
*  DEBUG_FLAGS - integer array of size (100). Non-zero elements yield
*     additional print-out data for debugging B2-Eirene coupling. See code for
*     specific uses. Many slots still available for user-specific needs.
*     If debug_flags(81).gt.0, will print out target fluxes in files
*     'targb2.datv', 'targb2n.datv' and 'targb2pl.datv'.
*     If debug_flags(90).gt.0, will print output about Eirene non-standard
*     surfaces [debug_flags(90):debug_flags(91)] and strata
*     [debug_flags(92):debug_flags(93)] (0 means sum over all strata).
*     Additional output can be obtained for strata
*     [debug_flags(94):debug_flags(95)].
*     If only debug_flags(90) is specified, the output will include all
*     individual Eirene strata, for all non-standard surfaces, starting from
*     debug_flags(90).
*     Defaults to 0.
*  NEUT_SCL_LIM - real. Maximum number by which Eirene neutral sources may be
*     scaled in either direction within the ank-mods scheme. See explaining text
*     in $SOLPSTOP/doc/Source_Scaling_in_B2.pdf for details.
*     Defaults to 2.0.
*  TRACK_INDEX - integer array of size (0:NS-1). Specifies the mixed material
*     species index related to B2.5 species (is).
*     Defaults to 1 for species sput_dst, 0 for all others.
*  TRACK_CHEM_SPUT - logical array of size (NTRACK). Indicates whether mixed
```

```
*     material species (itrack) participates in chemical sputtering. Defaults to
*     .true. for first tracked species if sput_dst is carbon, and to .false. for
*     all other cases.
*  CHEMICAL_EROSION_REDEP_FAC - real. Multiplier to the chemical sputtering
*     coefficient of carbon for computing the rate used for redeposited carbon.
*     Defaults to 1.0.
*  CHEMICAL_EROSION_BE_FAC - logical. Indicates whether the presence of
*     beryllium is to impact on the chemical sputtering yield of carbon.
*     Defaults to '.false.'.
*  CHEMICAL_EROSION_BE_FAC_A - real. The chemical sputtering yield of carbon is
*     multiplied by
*     (1.0-C/2*(tanh((frac-A)/B)-tanh((-A)/B)))
*     where frac is the fractional content of Be in the surface layer material.
*     Defaults to 0.2.
*  CHEMICAL_EROSION_BE_FAC_B - real. See above.
*     Defaults to 0.05.
*  CHEMICAL_EROSION_BE_FAC_C - real. See above.
*     Defaults to 0.9.
*  N_SPCSRF - integer. Number of special surfaces groups for diagnostics.
*     Defaults to 0.
*  L_SPCSRF - integer array of length (NLIM+NSTS). List of surface segments
*     (non-default standard surfaces [NDSS] or additional surfaces in Eirene
*     notation) included in the groups. Negative numbers correspond to NDSS.
*     Defaults to 0.
*  SPS_ID - character*8 array of length (N_SPCSRF). Labels of groups of special
*     surfaces.
*  I_SPCSRF - integer array of length (N_SPCSRF). Index of the first Eirene
*     surface belonging to a special surface group in the L_SPCSRF list.
*     Defaults to 0.
*  J_SPCSRF - integer array of length (N_SPCSRF). Index of the last Eirene
*     surface belonging to a special surface group in the L_SPCSRF list.
*     Defaults to 0.
*  SPS_ABSR - real array of length (N_SPCSRF). Particle absorption on the
*     surface. If positive, will supercede the setting from the Eirene input
*     file: RECYCT = 1-SPS_ABSR. Ignored if negative.
*     Defaults to -1.0.
*  SPS_TRNO - real array of length (N_SPCSRF). Surface transparency in positive
*     direction. If positive, will supercede the setting from the Eirene input
*     file: TRANSP(1,) = SPS_TRNO. Ignored if negative.
*     Defaults to -1.0.
*  SPS_TRNI - real array of length (N_SPCSRF). Surface transparency in negative
*     direction. If positive, will supercede the setting from the Eirene input
*     file: TRANSP(2,) = SPS_TRNI. If negative, the setting from SPS_TRNO is
*     used.
*     Defaults to -1.0.
*  SPS_MTRI - real array of length (N_SPCSRF). Surface material in Eirene
*     notation (e.g., 1206 for C). If positive, will supercede the setting from
*     the Eirene input file: ZNML = SPS_MTRI. Ignored if negative.
*     Defaults to 0.
*  SPS_MTRL - character*8 array of length (N_SPCSRF). Surface material in human
*     notation (e.g., 'C').
*  SPS_TMPR - real array of length (N_SPCSRF). Surface temperature (in eV). If
*     set larger to 1.e10, will be ignored. Otherwise, will supercede the
*     setting from the Eirene input file: EWALL = SPS_TMPR.
*     Defaults to 1.e15.
*  SPS_SPPH - real array of length (N_SPCSRF). Fudge factor for physical
*     sputtering. If positive, will supercede the setting from the Eirene input
*     file: RECYCS = SPS_SPPH. Ignored if negative.
*     Defaults to -1.0.
*  SPS_SPCH - real array of length (N_SPCSRF). Fudge factor for chemical
*     sputtering. If positive, will supercede the setting from the Eirene input
*     file: RECYCC = SPS_SPCH. Ignored if negative.
*     Defaults to -1.0.
*  SPS_SGRP - integer array of length (N_SPCSRF). Sputtered particle species
*     flag for chemical sputtering. If positive, will supercede the setting from
*     the Eirene input file: ISRC = SPS_SGRP. Ignored if negative.
*     Defaults to -1.
```

```
*  WRITE_NML_NEUT - logical. If .true. (default), writes the content of the
*     namelist to stdout after it has been read.
*     Defaults to .true..
*  MAXPOIN - integer. Obsolete. Maximum number of points needed to describe a
*     region contour in Eirene.
*     Defaults to 2000.
*  TIME_DEP_PUFF - logical array of length (NSTRAT). Indicates whether a
*     stratum is a time-dependent gas puff. Should only be .true. for strata
*     defined as gas puffs.
*     Defaults to .false. for all strata.
*  TIME_DEP_PUFF_FUNC - logical array of length (NSTRAT). Indicates whether a
*     stratum is using a time-dependent gas puff function. Should only be .true.
*     for strata defined as gas puffs. For use with TIME_DEP_PUFF_CASE and
*     TIME_DEP_PUFF_PARAM.
*     Defaults to .false. for all strata.
*  TIME_DEP_PUFF_CASE - integer array of length (NSTRAT). Used to select the
*     functional form of the time-dependent gas puff function. For use with
*     TIME_DEP_PUFF_FUNC and TIME_DEP_PUFF_PARAM.
*     Defaults to -1 for all strata.
*  TIME_DEP_PUFF_PARAM - real array of length (NSTRAT). Used to set the
*     parameters for the functional form of the time-dependent gas puff
*     function. For use with TIME_DEP_PUFF_FUNC and TIME_DEP_PUFF_CASE.
*     Defaults to 0 for all strata.
*  NGPDATA - integer data of size (NSTRAT). Number of points over which the
*     time profile of the strength of gas puff (istra) is given. Defaults to 0.
*     Current maximum value is 100.
*     Defaults to 0.
*  GPDATA - real data of size (NGPDATA,2,NSTRAT). For (i,ik,istra) in
*     (1:NGPDATA(istra),1:2,1:NSTRAT),
*     GPDATA(i,1,istra) contains the time point (i) for stratum (istra) (in s).
*     GPDATA(i,2,istra) contains the gas puff strength of stratum (istra) at
*     time point (i) (in particles/s).
*     The gas puff strength before the first time point is given by
*     USERFLUXPARM(istra,1).
*     The gas puff strength after the last time point is given by the GPDATA
*     value of the last time point.
*     Otherwise, the gas puff strength is linearly interpolated from the given
*     data.
*     Defaults to 0.0.
*
* The following are active only if NO_B2_CHEM_SPUT is not defined among the
* compiler options and are used to override the Eirene chemical sputtering
* settings:
*
*  CHEMICAL_SPUTTER_YIELD - real array of size (0:NLIM+NSTS). Passed to Eirene.
*     Chemical sputter yield of wall surface (ilim). Element 0 corresponds to
*     the default setting for all surfaces.
*     Defaults to 0.0.
*  FCHAR_CHEMICAL - real. Nuclear charge of atomic species causing the
*     sputtering. If set to its default value of 0, then the chemical sputtering
*     switches from this block are not used and the settings from the Eirene
*     input file are used.
*     Defaults to 0.
*  IGASS_CHEMICAL - integer. Passed to Eirene. Eirene atomic species index of
*     the sputtered particle. If igass_chemical > natmi, the data from
*     chemical_sputter_yield is not used and the yield from the Eirene surface
*     blocks is used instead.
*     Defaults to 0.
*  ITSPUT_CHEMICAL - integer. Passed to Eirene. Eirene type index of the
*     sputtered particle. Atoms = 1, Ions = 4. Defaults to 0, meaning 1 eV atom
*     chemical sputtering.
*     Defaults to 0.
*  ISSPUT_CHEMICAL - integer. Passed to Eirene. Mass*100 + nuclear charge of
*     the sputtered particle. Defaults to 0, internally changed to 1206 =
*     carbon.
*     Defaults to 0.
*
```

```
* NAMELIST /WALL_SAVE/
*    Found in b2.wall_save.parameters.
*    Read if present in the run directory.
*    Automatically generated by B2.5 but can be modified by user.
*    Contains data describing the state of the walls lining the vacuum vessel.
*    These data are used for smooth restarts and are not really intended for
*    human use.
*  NDEPTH_NML - integer. Dimension NDEPTH used for the arrays within this
*     namelist. Represents the number of depth layer discretising the wall
*     elements for the wall model. If using the 0-D model or the time-
*     independent 1-D model, will contain 1 (default). Should not exceed the
*     value of the parameter NDEPTH declared in b2mod_wall.F.
*     Defaults to 1.
*  IMAPX - integer array of size (NWALL). Indicates the (ix) position in the
*     grid of wall element (iwall). Default ordering of the wall elements
*     proceeds clockwise in physical space from the lower left corner.
*  IMAPY - integer array of size (NWALL). Indicates the (iy) position in the
*     grid of wall element (iwall). Default ordering of the wall elements
*     proceeds clockwise in physical space from the lower left corner.
*  XYMAP - integer array of size (-1:NX,-1:NY). XYMAP(ix,iy) contains the wall
*     index (iwall) of the wall element located a grid locaion (ix,iy). If there
*     is no wall element at this position, contains 0.
*     Default ordering of the wall elements proceeds clockwise in physical
*     space from the lower left corner.
*  SURFACE_MATERIAL_NAME - character*6 of size (NWALL). Contains the filename
*     from which to extract the surface material properties for wall element
*     (iwall). The files are to be found in
*     $SOLPSTOP/data.local/Surface_properties/ or
*     $SOLPSTOP/modules/B2.5/Database/Surface_properties/.
*     Defaults to 'C'.
*  COATING_MATERIAL_NAME - character*6 of size (NWALL). Contains the filename
*     from which to extract the bulk material properties of the eventual coating
*     for wall element (iwall).
*     Defaults to ' '.
*  BULK_MATERIAL_NAME - character*6 of size (NWALL). Contains the filename from
*     which to extract the bulk material properties for wall element (iwall).
*     The files are to be found in $SOLPSTOP/data.local/Bulk_properties/ or
*     $SOLPSTOP/modules/B2.5/Database/Bulk_properties/.
*     Defaults to 'C'.
*  LAYER_ALLOYS - character*6 of size (NALLOYS). Contains the filename from
*     which to extract the material properties for alloy (nalloy) which may be
*     present in mixed materials deposited layers. Not yet operational. The
*     files are to be found in $SOLPSTOP/data.local/Bulk_properties/,
*     $SOLPSTOP/data.local/Surface_properties/,
*     $SOLPSTOP/modules/B2.5/Database/Surface_properties/, or
*     $SOLPSTOP/modules/B2.5/Database/Bulk_properties/.
*  TARGET_TEMP - real array of size (NWALL,NDEPTH). Contains the temperature
*     (in Kelvin) of wall element (iwall) at depth layer (idepth).
*     If plate_option.eq.1, will be set to backplate_temp(iwall).
*     If plate_option.eq.2 and empty, will be set to equilibrium 1-D profile
*     deduced from plasma incoming fluxes.
*     If plate-option.eq.3, must be set.
*  INERTIAL_COOLING - logical array of size (NWALL). Indicates whether wall
*     element (iwall) is inertially cooled instead of actively cooled.
*     Defaults to '.false.'.
*  BACKPLATE_TEMP - real array of size (NWALL). Contains the temperature (in
*     Kelvin) maintained by cooling at the back end of wall element (iwall).
*     Defaults to b2stbr_plate_temp.
*  PLATE_THICKNESS - real array of size (NWALL). Contains the thickness (in
*     metres) of wall element (iwall).
*     Defaults to b2stbr_plate_thick.
*  COATING_THICKNESS - real array of size (NWALL). Contains the thickness (in
*     metres) of the eventual coating on wall element (iwall).
*     Defaults to 0.
*  PLATE_TIME_FACTOR - real array of size (NWALL). Multiplier to the time for
*     the equations for temperature and composition evolution of wall element
*     (iwall).
```

```
*     Defaults to 1.0.
*  DEPOSITION - real array of size (NWALL,NTRACK). Contains the amount of
*     deposited material (in atoms) from species (itrack) onto wall element
*     (iwall).
*     Defaults to 0.0.
*  EROSION - real array of size(NWALL,NTRACK). Contains the amount of eroded
*     material (in atoms) of species (itrack) from wall element(iwall).
*     Defaults to 0.0.
*  CHEMICAL_SPUTTERING - real array of size (NWALL,0:NS-1). Contains the
*     chemical sputter yield of 'sput_dst' or 'sput_dst_bulk' species on wall
*     element (iwall) caused by B2.5 species (is).
*     Defaults to 0.0.
*  PHYSICAL_SPUTTERING - real array of size (NWALL,0:NS-1). Contains the
*     physical sputter yield of 'sput_dst' or 'sput_dst_bulk' species on wall
*     element (iwall) caused by B2.5 species (is).
*     Defaults to 0.0.
*  PHYSICAL_SPUTTERING_ENERGY - real array of size (NWALL,0:NS-1). Contains the
*     fraction of returned energy carried by sputtered particles of species
*     'sput_dst' or 'sput_dst_bulk' species from wall element (iwall) caused by
*     B2.5 species (is).
*     Defaults to 0.0.
*  RES_SPUTTERING - real array of size (NWALL,0:NS-1). Contains the RES sputter
*     yield of 'sput_dst' or 'sput_dst_bulk' species on wall element (iwall)
*     caused by B2.5 species (is).
*     Defaults to 0.0.
*  THERMAL_EVAPORATION - real array of size(NWALL,0:NS-1). Contains the rate of
*     thermal evaporation of species (is) (in particles/second) from wall
*     element (iwall).
*     Defaults to 0.0.
*  BACKSCATTERING - real array of size (NWALL,0:NS-1). Contains the
*     backscattering fraction for incoming B2.5 species (is) onto wall element
*     (iwall).
*     Defaults to 0.0.
*  BACKSCATTERING_ENERGY - real array of size (NWALL,0:NS-1). Contains the
*     backscattered energy fraction for incoming B2.5 species (is) onto wall
*     element (iwall).
*     Defaults to 0.0.
*  PLATE_TIME - real array of size (NWALL). Indicates how much simulation time
*     has elapsed for wall element (iwall) (in seconds).
*     Defaults to 0.0.
*  PLATE_AREA - real array of size (NWALL). Indicates the wall area (in square
*     metres) for wall element (iwall). Defaults to the area computed from the
*     B2.5 grid information.
*  MONOLAYER_DEPOSITION - real array of size (NWALL,NTRACK). Contains the
*     amount of deposited material (in monolayers) from species (itrack) onto
*     wall element (iwall).
*     Defaults to 0.0.
*  MONOLAYER_EROSION - real array of size (NWALL,NTRACK). Contains the amount
*     of eroded material (in monolayers) of species (itrack) from wall element
*     (iwall).
*     Defaults to 0.0.
*  LAYER_NCONSTITUENTS - integer array of size (NWALL). Indicates the number of
*     elemental constituents within the surface layer of wall element (NWALL).
*     Defaults to 1.
*  LAYER_NZCONSTITUENTS - integer array of size (NWALL,6+NTRACK). Contains the
*     atomic numbers Z of the various elements present within the surface layer
*     of wall element (iwall). Defaults to 6 for the first value, 0 otherwise.
*  LAYER_RELCONSTITUENTS - real array of size (NWALL,6+NTRACK). Contains the
*     relative atomic abundances of the various elements present within the
*     surface layer of wall element (iwall). Defaults to 1.0 for the first
*     value, 0.0 otherwise.
*     Replaces LAYER_NRELCONSTITUENTS.
*
* NAMELIST /B2MD_NAMELIST/
*    Found in b2md.dat and read by b2md.
*    Contains data useful for cataloguing the run (done from 'save_mds' and
*    'resave_mds' scripts).
```

```
*  EXP - character*128. Name of the experiment being modelled.
*     Defaults to 'NOT_SET'.
*  SHOT - integer. Shot number identifying the run. Defaults to the last number
*     found in shotnumber.history, or 0 if the file is not found.
*  TIME - real. Time point of the experimental shot being simulated.
*     Defaults to 0.0.
*  COMMENT - character*128. Label for the run.
*     Defaults to 'NOT_SET'.
*  TIMEDEP - logical. If .true. (default), saves data from b2time.nc.
*  SNAPSHOT - logical. If .true. (default), saves data from b2fplasma.
*  TALLIES - logical. If .true. (default), saves data from b2tallies.nc.
*  MOVIES - logical. If .true. (default), saves data from b2movies.nc.
*  OVERWRITE_SHOTNUMBER - integer. Indicate the shot number to overwrite (to be
*     used only when updating an already saved run with 'resave_mds' script).
*     Defaults to the last number found in shotnumber.history, or 0 if the file
*     is not found.
*
* NAMELIST /BOUNDARY/
*    Found in b2.boundary.parameters.
*    Contains data used to specify the boundary conditions.
*    This namelist is read if 'b2stbc_boundary_namelist' is set to 1.
*    Necessary for double-null geometries or impinging limiting structures.
*    It is recommended but not strictly necessary that the boundary conditions
*    declared here match in number and order any neutral boundary conditions
*    declared in the b2.neutrals.parameters file.
*    This description is copied from the SOLPS manual.
*  NBC - integer. Number of boundary segments. Defaults to 0.
*  BCCHAR - character*1 array of length (NBC). Specifies the nature of the
*     boundary segment
*     N = 'North' boundary
*     S = 'South' boundary
*     W = 'West' boundary
*     E = 'East' boundary
*     X = 'X' boundary used for specifying a fixed value on a row of cells
*     Y = 'Y' boundary used for specifying a fixed value on a column of cells
*     Defaults to ' '.
*  CONPAR - real array of size (0:NS-1,NBC,3). Contains parameters helping to
*     define the boundary conditions for the continuity equation of species
*     (is). See description of BCCON below for details.
*     Defaults to 0.0.
*  MOMPAR - real array of size (0:NS-1,NBC,2). Contains parameters helping to
*     define the boundary conditions for the parallel momentum equation of
*     species (is). See description of BCMOM below for details.
*     Defaults to 0.0.
*  ENEPAR - real array of size (NBC,2). Contains parameters helping to define
*     the boundary conditions for the electron energy equation. See description
*     of BCENE below for details.
*     Defaults to 0.0.
*  ENIPAR - real array of size (NBC,2). Contains parameters helping to define
*     the boundary conditions for the ion energy equation. See description of
*     BCENI below for details.
*     Defaults to 0.0.
*  POTPAR - real array of size (NBC,2). Contains parameters helping to define
*     the boundary conditions for the potential equation. See description of
*     BCPOT below for details.
*     Defaults to 0.0.
*  BCPOS - integer array, length (NBC). For North, South or X boundary
*     conditions, it specifies the row index; for West, East and Y boundary
*     conditions, it specifies the column index.
*     Defaults to -2.
*  BCSTART - integer array, length (NBC). For North, South or X boundary
*     conditions, it specifies the start column index; for West, East and Y
*     boundary conditions, it specifies the start row index.
*     Defaults to -2.
*  BCEND - integer array, length (NBC). For North, South X boundary conditions,
*     it specifies the end column index; for West, East and Y boundary
*     conditions, it specifies the end row index.
```

```
*     Defaults to -2.
*  BC_LIST_SIZE - integer array of length (NBC). Contains the size of the list
*     of cells where a boundary condition is applied.
*     Defaults to 0.
*  BC_LIST_X - integer array of length (2*(NXD+NYD),NBC). Contains the
*     X-coordinate of the cells where boundaries conditions are applied.
*     Defaults to -2.
*  BC_LIST_Y - integer array of length (2*(NXD+NYD),NBC). Contains the
*     Y-coordinate of the cells where boundaries conditions are applied.
*     Defaults to -2.
*  BCCON - integer array, length (0:NS-1,NBC). Specifying the type of density
*     boundary condition for each segment and species (fastest varying index is
*     species); makes use of CONPAR to specify additional information, as
*     indicated:
*     0 : default, no boundary condition is applied
*     1 : prescribe the value of the density, CONPAR(,,1) specifies the
*     required density in m^{-3}
*     2 : prescribe the gradient of the density, CONPAR(,,1) specifies the
*     required density gradient in m^{-4}
*     3 : sheath conditions, CONPAR(,,1) not used (zero gradient is used)
*     4 : prescribe the value of the density, weakly a mixed boundary
*     condition, CONPAR(,,1) specifies the required density in m^{-3} and
*     CONPAR(,,2) specifies the 'strength' of the boundary condition
*     5 : prescribe the particle flux per unit area, CONPAR(,,1) specifies the
*     required particle flux density in m^{-2}.s^{-1}
*     6 : prescribe the total particle flux for a constant density, CONPAR(,,1)
*     specifies the particle flux in s^{-1}
*     7 : prescribe the density as a function of other plasma parameters [not
*     yet available]
*     8 : prescribe the total particle flux with constant flux density,
*     CONPAR(,,1) specifies the particle flux in s^{-1}
*     9 : prescribe the decay length for the density, CONPAR(,,1) specifies the
*     gradient length in metres (should use type 15 instead when drifts are
*     turned on)
*     10 : leakage option for density, recommended for cases with drifts,
*     CONPAR(,,1) specifies the leakage factor, \alpha in \Gamma_{loss} =
*     \alpha C_s n_a. CONPAR(,,2) is an optional stabilization parameter that
*     can be set to a positive value to help with the linearization of the
*     boundary condition.
*     11 : particle flux feedback boundary condition, CONPAR(,,1) not used,
*     derived from CBSNA(0,IS,IREG). For feedback schemes not specified via
*     b2.feedback_control.parameters, the species used must be declared using
*     the 'b2stbc_isfeedback' switch.
*     12 : particle density feedback boundary condition, as above, CONPAR(,,1)
*     not used, derived from CBSNA(0,IS,IREG). The species used must be
*     declared using the 'b2stbc_isfeedback' switch.
*     13 : particle density to achieve specified total flux,
*     CONPAR(,,1) is the specified flux crossing the flux surface
*     'b2stbc_type13_ref' steps away from the boundary,
*     CONPAR(,,2) is the strength of the feedback,
*     CONPAR(,,3), when running with Eirene and the 'ionising core' switch is
*     used, is set internally to match the re-entering flux of ionised neutrals
*     that crossed the core boundary (one must then have 'ionising_core'.eq.-IB
*     where IB is the boundary index).
*     The feedback scheme can be further tweaked with the switches
*     'b2stbc_type13_norm' and 'b2stbc_type13_fac'. See code for details.
*     14 : if MOMPAR(:,:,2) < 0.5, collective sound speed velocity flux, and
*     marginal imposition of the Bohm-Chodura sheath entrance condition;
*     CONPAR(,,1) is a multiplier to the outgoing collective sound speed C_s;
*     to be used in conjunction with BCENE/I=15, BCPOT=11, and BCMOM=13. If
*     MOMPAR(:,:,2) >= 0.5, single species sound speed velocity flux, non-
*     marginal imposition of the Bohm-Chodura sheath entrance condition;
*     MOMPAR(,,1) is a multiplier to the outgoing specific species sound speed
*     C_{s,a}; to be used in conjunction with BCMOM=13, BCENE=15 with
*     bcene_15_style=1, BCENI=15, BCPOT=3 or BCPOT=11 with istyle_fchi=0.
*     Recommended for cases with drifts.
*     15 : prescribe a radial leakage velocity, CONPAR(,,1) specifies the
```

```
*     leakage velocity in units of the local thermal velocity.
*     16 : particle density to achieve specified total flux, used with ASTRA
*     coupling. The total desired flux is summed over all BCCON=16 core
*     boundaries. CONPAR(,,1) specifies the desired particle flux in s^{-1}.
*     18 : prescribe total main ion particle flux, used with ASTRA coupling.
*     19 : particle flux feedback boundary condition, flux is summed over
*     neutrals and ions, for coupling with ASTRA. The total desired flux is
*     summed over all BCCON=19 core boundaries. This boundary condition type is
*     applied to ions in their highest ionisation stage. CONPAR(,,1) specifies
*     the desired particle flux in s^{-1}.
*     20 : constant density feedback condition, CONPAR(,,1) specifies the
*     desired density in m^{-3}.
*     21 : prescribe the value of the density and add a poloidal density
*     variation to get a solution which is as close as possible to neoclassical
*     theory. It is recommended to use this boundary condition together with
*     corresponding condition on ion temperature (BCENI=24). The total desired
*     density is summed over all BCCON=21 core boundaries. CONPAR(,,1)
*     specifies the desired density in m^{-3}.
*     22 : feedback boundary condition: given total particle flux with constant
*     average density. The poloidal density variation is added to make the
*     solution as close as possible to the neoclassical value derived for pure
*     H/D/T plasma. It is recommended to use this boundary condition with
*     corresponding boundary condition on ion temperature (BCENI=23,24). The
*     total desired flux is summed over all BCCON=22 core boundaries.
*     CONPAR(,,1) specifies the desired particle flux in s^{-1}.
*     23 : feedback boundary condition: given sum of integrated neutrals and
*     main ion particle fluxes with constant average density. The poloidal
*     density variation is added to make the solution as close as possible to
*     the neoclassical value derived for pure H/D/T plasma. It is recommended
*     to use this boundary condition with corresponding boundary condition on
*     ion temperature (BCENI=23,24). The total desired flux is summed over all
*     BCCON=23 core boundaries. CONPAR(,,1) specifies the desired particle flux
*     in s^{-1}.
*     24 : constant density feedback scaled by density on the ring
*     'bc_type21_ref' away. CONPAR(,,1) specifies the desired density in
*     m^{-3}. CONPAR(,,2) is the strength of the feedback
*     25 : feedback boundary condition: prescribe the average value of the
*     density and add the poloidal density variation from the neighbouring
*     radial ring. This boundary condition is suitable for any species of a
*     multi-species plasma (i.e. in the case when the condition used in
*     BCCON=21 fails). It is recommended to use this boundary condition
*     together with the corresponding condition on ion temperature
*     (BCENI=26,27). CONPAR(,,1) specifies the desired average density in
*     m^{-3}.
*     26 : feedback boundary condition: prescribe the total ion flux and find
*     the average density. The poloidal density variation from the neighbouring
*     radial ring is applied on top of the average density. This boundary
*     condition is suitable for any species of a multi-species plasma (i.e. in
*     the case when the condition used in BCCON=22 fails). It is recommended to
*     use this boundary condition together with a corresponding condition on
*     the ion temperature (BCENI=26,27). The total desired flux is summed over
*     all BCCON=26 core boundaries. CONPAR(,,1) specifies the desired particle
*     flux in s^{-1}.
*     27 : feedback boundary condition: prescribe the particle flux sum for all
*     neutrals and ions belonging to a given isonuclear sequence and find the
*     average density of the ionisation state to which the BCCON=27 condition
*     is applied. The poloidal density variation from the neighbouring radial
*     ring is applied on top of the average density. It is recommended to use
*     this boundary condition together with corresponding condition on ion
*     temperature (BCENI=25,26). The total desired flux is summed over all
*     BCCON=27 core boundaries. CONPAR(,,1) specifies the desired particle flux
*     in s^{-1}. For main ions and low Z impurities, it is recommended to set
*     BCCON=27 for the highest ionisation state. For high Z impurities, if the
*     ionisation potential of the highest ionisation state of the given
*     isonuclear sequence is much larger than the expected core plasma
*     temperature, the BCCON=27 condition should be set for the ionisation
*     state for which the density is expected to be maximal. For ionisation
```

```
*     states higher than that one, it is recommended to set BCCON=2 with
*     CONPAR=0.0, since their densities are expected to be negligible, and
*     BCCON=2 with CONPAR=0.0 seems to be the safest boundary condition for low
*     density species.
*     28 : density is read from the CON_FN file. If CONPAR(,,2) > 0 the profile
*     is scaled so CONPAR(,,1) gives the maximum density. Values for the
*     density are given in m^{-3}.
*     29 : presure is prescribed from the CONPAR(,,1) in Pa.
*     30 : The particle flux per unit area is read from the CON_FN file. If
*     CONPAR(,,2) > 0 the profile is scaled so CONPAR(,,1) gives the maximum
*     particle flux per unit area. Values for the density are given in
*     m^{-2}s^{-1}.
*  BCMOM - integer array, length NS * NBC. Specifying the type of parallel
*     momentum or velocity boundary condition for each segment and species
*     (fastest varying index is species); makes use of MOMPAR to specify
*     additional information, as indicated
*     0 : default, no boundary condition is applied
*     1 : prescribe the value of the parallel velocity, MOMPAR(,,1) specifies
*     the parallel velocity in m.s^{-1}
*     2 : prescribe the gradient of the parallel velocity, MOMPAR(,,1)
*     specifies the parallel velocity gradient in s^{-1}
*     3 : sheath conditions, Mach number as input,
*     if MOMPAR(,,2) < 0.5, then the velocity is set to exactly
*     MOMPAR(,,1) * C_{s,collective}, otherwise the velocity is set to be at
*     least MOMPAR(,,1) * C_{s,a}, the thermal velocity of that species
*     4 : prescribe the value of the velocity, weakly a mixed boundary
*     condition, MOMPAR(,,1) specifies the parallel velocity in m.s^{-1} and
*     MOMPAR(,,2) specifies the 'strength' of the boundary condition
*     5 : prescribe the parallel momentum flux per unit area, MOMPAR(,,1)
*     specifies the parallel momentum flux density in N.m^{-2}
*     6 : prescribe the total parallel momentum flux for a constant parallel
*     velocity [not yet available]
*     7 : prescribe the parallel momentum as a function of other plasma
*     parameters [not yet available]
*     8 : special : limited shear, imposes zero gradient for the Mach number.
*     [[[Eventually intended to have MOMPAR(,,1) specify the gradient of the
*     Mach number in m^{-1}]]]
*     9 : prescribe the total parallel momentum flux with constant flux
*     density, MOMPAR(,,1) specifies the parallel momentum flux in N
*     10 : prescribe the decay length for the parallel momentum, MOMPAR(,,1)
*     specifies the decay length in m
*     11 : Rozhansky viscosity condition for the parallel momentum, MOMPAR(,,1)
*     is not used
*     12 : condition from b2stbc_spb for the parallel momentum
*     13 : if MOMPAR(:,:,2) < 0.5, collective sound speed velocity flux, and
*     marginal imposition of the Bohm-Chodura sheath entrance condition;
*     MOMPAR(,,1) is a multiplier to the outgoing collective sound speed C_s;
*     to be used in conjunction with BCCON=14, BCENE/I=15, and BCPOT=11. If
*     MOMPAR(:,:,2) >= 0.5, single species sound speed velocity flux, non-
*     marginal imposition of the Bohm-Chodura sheath entrance condition;
*     MOMPAR(,,1) is a multiplier to the outgoing specific species sound speed
*     C_{s,a}; to be used in conjunction with BCCON=14, BCENE=15 with
*     bcene_15_style=1, BCENI=15, BCPOT=3 or BCPOT=11 with istyle_fchi=0.
*     Recommended for cases with drifts.
*     14 : condition from b2stbc_spb for the parallel momentum
*     15 : prescribe the value of the parallel velocity, scaled with
*     B_average/B_local
*     16 : prescribe the average value of the parallel velocity MOMPAR(,,1)
*     specifies the parallel velocity in m.s^{-1}
*     17 : leakage option for parallel momentum, MOMPAR(,,1) specifies the
*     leakage factor, \alpha in \Gamma_{loss} = \alpha C_{s,a} m_a n_a u_a
*     18 : reads parallel velocity profile in units of m.s^{-1} from the
*     specified file in MOM_FN. If MOMPAR(,,2) > 0 the profile is scaled so the
*     maximum parallel velocity is given by MOMPAR(,,1) in m.s^{-1}. Only
*     available for E and W boundaries.
*     19 : reads the momentum fllux profile from MOM_FN in units of
*     m^{-2}.s^{-1}. This has to be used in combination with BCCON = 1, 28, 30
```

```
*     and BCENI = 15. Only available for E and W boundaries.
*     20 : reads the Mach number from the MOM_FN file. Velocity is calculated
*     with the temperature at the boundary and \gamma_i = 1. Only available for
*     E and W boundaries.
*     21 : reads the parallel velocity gradient from the MOM_FN file in units
*     of s^{-1}.  If MOMPAR(,,2) > 0 the profile is scaled so the maximum
*     parallel velocity gradient is given by MOMPAR(,,1). Only available for E
*     and W boundaries.
*  BCENE - integer array, length NBC. Specifying the type of electron energy or
*     temperature boundary condition for each segment; makes use of ENEPAR to
*     specify additional information, as indicated
*     0 : default, no boundary condition is applied
*     1 : prescribe the value of the electron temperature, ENEPAR(,1) specifies
*     the temperature in eV
*     2 : prescribe the gradient of the electron temperature, ENEPAR(,1)
*     specifies the temperature gradient in eV.m^{-1}
*     3 : sheath conditions, electron energy transmission, ENEPAR(,1) specifies
*     an additional contribution to the energy transmission coefficient in
*     addition to that of the potential difference [the sound speed used
*     depends on settings of MOMPAR(,ISMAIN,2)]
*     4 : prescribe the value of the electron temperature, weakly a mixed
*     boundary condition, ENEPAR(,1) specifies the temperature in eV and
*     ENEPAR(,2) specifies the 'strength' of the boundary condition
*     5 : prescribe the electron energy flux per unit area, ENEPAR(,1)
*     specifies the energy flux density in W.m^{-2}
*     6 : prescribe the total electron energy flux for a constant electron
*     temperature, ENEPAR(,1) specifies the energy flux in W (no longer
*     supported, use types [16] or [17] instead)
*     7 : prescribe the electron temperature as a function of other plasma
*     parameters [not yet available]
*     8 : prescribe the total electron heat flux with constant flux density,
*     ENEPAR(,1) specifies the energy flux in W
*     9 : prescribe the decay length for the electron temperature, ENEPAR(,1)
*     specifies the decay length in m (can also use type [19] instead)
*     10 : feedback option for core, ENEPAR(,1) not used, derived from
*     cbshe(0,coreregno)
*     11 : not used
*     12 : sheath conditions, electron energy transmission coefficient,
*     ENEPAR(,1) specifies an energy transmission factor, delta_e in Q_e =
*     delta_e \Gamma_e T_e
*     13 : prescribe the electron energy flux per unit area proportional to
*     temperature, ENEPAR(,1) specifies the energy flux density per temperature
*     in W.m^{-2}.J^{-1} (the temperature here in J)
*     14 : leakage option for electron energy, ENEPAR(,1) specifies the leakage
*     factor, \alpha in \Gamma_{loss} = \alpha C_s, collective n_e T_e
*     15 : sheath boundary condition, from b2stbc_spb, recommended when using
*     drifts. Linked to using BCCON=14 and BCMOM=13 for all ion species,
*     BCENI=15, and BCPOT=11. When bcene_15_style=1, the electron flux is
*     computed via the exponential of the sheath potential drop, and
*     ENEPAR(:,1) is used to artificially increase or decrease the sheath
*     transmission coefficient for electrons (default = 0).
*     16 : feedback boundary condition with constant temperature, ENEPAR(,1)
*     specifies the power flux in W across the flux surface with index
*     'b2stbc_type16_ref' (default=-1), ENEPAR(,2) should be something like 0.1
*     and specifies the strength of the feedback. Also see type [17] below.
*     Available if bcene_16_style=0 (default). If bcene_16_style=1, integrated
*     electron heat flux with constant electron temperature, summed over all
*     core boundaries with BCENE=16.
*     17 : feedback boundary condition with constant shared temperature for
*     both electrons and ions, with ENEPAR(,1) + ENIPAR(,1) giving the total
*     power flux in W across the flux surface with index 'b2stbc_type16_ref'
*     (default=-1), ENEPAR(,2) should be something like 0.1 and specifies the
*     strength of the feedback. Replaces [16] for high densities and large
*     values of 'b2stbc_type16_ref'.
*     18 : fractional drop condition. Not yet working.
*     19 : same as [9] but to be used when simultaneously setting BCCON=1 on
*     the same boundary.
```

```
*     20 : feedback boundary condition with constant temperature, as per type
*     [16] but with 'type20_' switches and a different feedback scheme.
*     21 : constant temperature feedback scaled by temperature on the ring
*     bc_type21_ref away.
*     ENEPAR(,1) specifies the desired electron temperature in eV.
*     ENEPAR(,2) is the strength of the feedback.
*     22 : radial leakage condition for the electron temperature. ENEPAR(,1)
*     specifies the leakage velocity in units of the electron thermal velocity.
*     A temperature gradient such that the diffusive flux is set to match this
*     leakage is imposed.
*     23 : electron temperature profile in eV is read from the ENE_FN file. If
*     ENEPAR(,2) > 0 the profile is scaled so ENEPAR(,1) provides the maximum
*     electron temperature in eV.
*  BCENI - integer array, length NBC. Specifying the type of ion energy or
*     temperature boundary condition for each segment; makes use of ENIPAR to
*     specify additional information, as indicated
*     0 : default, no boundary condition is applied
*     1 : prescribe the value of the ion temperature, ENIPAR(,1) specifies the
*     temperature in eV
*     2 : prescribe the gradient of the ion temperature, ENIPAR(,1) specifies
*     the temperature gradient in eV.m^{-1}
*     3 : sheath conditions, ion energy transmission, ENIPAR(,1) specifies the
*     contribution to the energy transmission coefficient [the sound speed used
*     depends on settings of MOMPAR(,ISMAIN,2)]
*     4 : prescribe the value of the ion temperature, weakly a mixed boundary
*     condition, ENIPAR(,1) specifies the temperature in eV and ENIPAR(,2)
*     specifies the 'strength' of the boundary condition
*     5 : prescribe the ion energy flux per unit area, ENIPAR(,1) specifies the
*     energy flux density in W.m^{-2}
*     6 : prescribe the total ion energy flux for a constant ion temperature,
*     ENIPAR(,1) specifies the energy flux in W (no longer supported, use types
*     [16] or [17] instead)
*     7 : prescribe the ion temperature as a function of other plasma
*     parameters [not yet available]
*     8 : prescribe the total ion heat flux with constant flux density,
*     ENIPAR(,1) specifies the energy flux in W
*     9 : prescribe the decay length for the ion temperature, ENIPAR(,1)
*     specifies the decay length in m (can also use type [19] instead)
*     10 : feedback option for core, ENIPAR(,1) not used, derived from
*     cbshi(0,ISMAIN,coreregno)
*     11 : sheath conditions, ion energy transmission coefficient, ENIPAR(,1)
*     specifies an energy transmission factor, delta_i in Q_i = delta_i T_i
*     sum_a n_a C_{s,a}
*     12 : sheath conditions, ion energy transmission coefficient, ENIPAR(,1)
*     specifies an energy transmission factor, delta_i in Q_i = delta_i T_i
*     sum_a \Gamma_a
*     13 : prescribe the ion energy flux per unit area proportional to
*     temperature, ENIPAR(,1) specifies the energy flux density per temperature
*     in W.m^{-2}.J^{-1} (the temperature here in J)
*     14 : leakage option for ion energy, ENIPAR(,1) specifies the leakage
*     factor, \alpha in \Gamma_{loss} = \alpha C_sT_i
*     15 : sheath boundary condition, from b2stbc_spb, recommended when using
*     drifts. If MOMPAR(ismain,:,2) < 0.5, collective sound speed velocity
*     flux, and marginal imposition of the Bohm-Chodura sheath entrance
*     condition; the ion heat flux is computed using the collective sound speed
*     and ENIPAR(:,1) sets the ion sheath transmission coefficient (def: 1.5);
*     to be used in conjunction with BCENE=15, BCPOT=11, BCCON=14, and
*     BCMOM=13. If MOMPAR(ismain,:,2) >= 0.5, single species sound speed
*     velocity flux, non-marginal imposition of the Bohm-Chodura sheath
*     entrance condition; the ion heat flux is computed using the target
*     particle flux and ENIPAR(:,1) sets the ion sheath transmission
*     coefficient (def: 1.5); to be used in conjunction with BCMOM=13, BCENE=15
*     with bcene_15_style=1, BCCON=14, BCPOT=3 or BCPOT=11 with istyle_fchi=0.
*     Recommended for cases with drifts.
*     16 : feedback boundary condition with constant temperature, ENIPAR(,1)
*     specifies the power flux in W across the flux surface with index
*     'b2stbc_type16_ref' (default=-1), ENIPAR(,2) should be something like 0.1
```

```
*     and specifies the strength of the feedback. Also see type [17] below.
*     Available if bceni_16_style=0 (default). If bceni_16_style=1, integrated
*     ion heat flux with constant ion temperature, summed over all core
*     boundaries with BCENI=16. ENIPAR(,1) specifies the power flux in W
*     17 : feedback boundary condition with constant shared temperature for
*     both electrons and ions, see BCENE=17 above for description.
*     18 : fractional drop condition. Not yet working.
*     19 : same as [9] but to be used when simultaneously setting BCCON=1 on
*     the same boundary.
*     20 : feedback boundary condition with constant temperature, as per type
*     [16] but with 'type20_' switches and a different feedback scheme.
*     21 : from b2stbc_spb
*     22 : radial leakage condition for the ion temperature. ENIPAR(,1)
*     specifies the leakage velocity in units of the collective ion thermal
*     velocity. A temperature gradient such that the diffusive flux is set to
*     match this leakage is imposed.
*     23 : prescribe the poloidally averaged value of the ion temperature and
*     introduce a poloidal variation as close as possible to neoclassical
*     solution. It is recommended to use this boundary condition together with
*     corresponding condition on ion density (BCCON=21,22,23). The average is
*     taken over all core boundaries with BCENI=23. ENIPAR(,1) specifies the
*     temperature in eV
*     24 : feedback boundary condition with prescribed total ion flux, constant
*     poloidally averaged ion temperature and a poloidal variation as close as
*     possible to neoclassical solution. It is recommended to use this boundary
*     condition together with corresponding condition on ion density
*     (BCCON=21,22,23). The flux is summed over all core boundaries with
*     BCENI=24. ENIPAR(,1) specifies the energy flux in W
*     25 : constant temperature feedback scaled by temperature on the ring
*     bc_type21_ref away. ENIPAR(,1) specifies the desired ion temperature in
*     eV. ENIPAR(,2) is the strength of the feedback
*     26 : prescribe the poloidally averaged value of the ion temperature and
*     introduce a poloidal variation in a simplified manner. This boundary
*     condition is suitable for any plasma composition (i.e. when BCENI=23
*     fails). It is recommended to use this boundary condition together with
*     corresponding condition on ion density (BCCON = 25,26,27). The average is
*     taken over all core boundaries with BCENI=26. ENIPAR(,1) specifies the
*     temperature in eV
*     27 : feedback boundary condition with prescribed total ion heat flux,
*     constant poloidally averaged ion temperature and a poloidal variation in
*     a simplified manner. This boundary condition is suitable for any plasma
*     composition (i.e. when BCENI=24 fails). It is recommended to use this
*     boundary condition together with corresponding condition on ion density
*     (BCCON = 25,26,27). The flux is summed over all core boundaries with
*     BCENI=27. ENIPAR(,1) specifies the energy flux in W
*     28 : ion temperature profile in eV is read from the ENI_FN file. If
*     ENIPAR(,2) > 0 the profile is scaled so ENIPAR(,1) provides the maximum
*     ion temperature in eV.
*  BCPOT - integer array, length NBC. Specifying the type of electric potential
*     or current boundary condition for each segment; makes use of POTPAR to
*     specify additional information, as indicated
*     0 : default, no boundary condition is applied
*     1 : prescribe the value of the potential, POTPAR(,1) specifies the
*     potential in V
*     2 : prescribe the gradient of the potential, POTPAR(,1) specifies the
*     potential gradient in V.m^{-1}
*     3 : sheath conditions, POTPAR(,2) used for biasing [see code for details]
*     4 : prescribe the value of the potential weakly a mixed boundary
*     condition, POTPAR(,1) specifies the potential in V and POTPAR(,2)
*     specifies the 'strength' of the boundary condition
*     5 : prescribe the current flux density per unit area, POTPAR(,1)
*     specifies the electric current flux density in A.m^{-2}
*     6 : prescribe the total current flux density for a constant potential
*     [not yet available]
*     7 : prescribe the potential as a function of other plasma parameters [not
*     yet available]
*     8 : prescribe the total electric current with constant flux density,
```

```
*     POTPAR(,1) specifies the electric current in A
*     9 : prescribe the decay length for the potential, POTPAR(,1) specifies
*     the decay length in m
*     10 : feedback option for core [not yet tested!!!!!!!!!] (based on using
*     cbsch(0,coreregno))
*     11 : sheath conditions, electron energy transmission from b2stbc_spb,
*     POTPAR(,2) specifies the bias potential in V. Recommended for use in
*     cases with drifts, along with BCENE/I=15, BCCON=14, and BCMOM=13.
*     12 : imposes the currents due to drifts for the South core boundary. Must
*     be used in conjunction with istyle_cur_contr_on_S_and_N.eq.2
*     13 : imposes the currents due to drifts for the South private flux and
*     North boundaries. Must be used in conjunction with
*     istyle_cur_contr_on_S_and_N.eq.2
*     14 : prescribe the value of the radial sheath potential, POTPAR(,1), in V
*     15 : potential profile is read from POT_FN file. Only available for E and
*     W boundaries.
*     16 : constant electric potential feedback on imposed total current. The
*     current prescribed is given by the sum of the POTPAR(IB,1) (in A) over
*     all the BCPOT=16 boundaries. Still experimental, will not work for drift
*     cases.
*     17 : sheath condition for floating target. This BC provides a floating
*     target with a near zero current through the target. First, the total
*     target current is calculated. Then, the potential is corrected by a
*     factor POTPAR(,1) * current to reduce the current close to zero.
*     POTPAR(,2) is used to provide an initial guess in V of the target
*     potential and to store the value, in V, during the iterative process.
*     21 : constant potential feedback scaled by potential on the ring
*     'bc_type21_ref' away. POTPAR(,1) specifies the desired potential in V.
*     POTPAR(,2) is the strength of the feedback
*  GAMMAI - real. Adiabatic coefficient multiplying the ion temperature when
*     computing the sound speeds.
*     Defaults to 1.0.
*  GAMMAE - real. Secondary electron emission coefficient.
*     Defaults to 0.5.
*  LBNDUSR - logical. If .true. will also call b2stbc_bas boundary condition
*     routine after the end of the physics boundary condition routine (governed
*     by the data from b2ah.dat and b2mn.dat).
*     Defaults to '.false.'.
*  LFEEDBACK - logical. Indicates whether a feedback scheme is used. Obsolete.
*     Superceded by 'b2stbc_feedback'.
*     Defaults to '.false.'.
*  NNISO - integer. Number of dead (or isolated) regions.
*     Defaults to 0.
*  NIISO - real array of size (0:NS-1). Density of species (is) in (m-3) to
*     impose in isolated regions.
*  TEISO - real. Electron temperature (in eV) to impose in isolated regions.
*     Defaults to 1.0.
*  TIISO - real. Ion temperature (in eV) to impose in isolated regions.
*     Defaults to 1.0.
*  PHIISO - real. Electric potential (in V) to impose in isolated regions.
*     Defaults to 0.0.
*  NXISO1 - integer array of size NNISO. Column number of bottom left corner of
*     the dead region (iiso).
*     Defaults to -2.
*  NXISO2 - integer array of size NNISO. Column number of top right corner of
*     the dead region (iiso).
*     Defaults to -2.
*  NYISO1 - integer array of size NNISO. Row number of bottom left corner of
*     the dead region (iiso).
*     Defaults to -2.
*  NYISO2 - integer array of size NNISO. Row number of top right corner of the
*     dead region (iiso).
*     Defaults to -2.
*  BOUNDARY_FILENAME - character*256. Name of the next file to use for reading
*     a new /BOUNDARY/ namelist.
*     Defaults to b2.boundary.parameters.
*  BOUNDARY_TIME_MOD - real. When the B2.5 run simulation time, in seconds,
```

```
*     modulo(BOUNDARY_TIME_MOD), reaches or exceeds BOUNDARY_TIME_SWITCH, reads
*     the new namelist from BOUNDARY_FILENAME. Also switches to the new namelist
*     as the ELM count (here time/BOUNDARY_TIME_MOD) changes. Only active if
*     BOUNDARY_TIME_MOD is greater than 0.
*     Defaults to 0.0.
*  BOUNDARY_TIME_SWITCH - real. If BOUNDARY_TIME_MOD is greater than 0, time
*     (in seconds) within an ELM cycle at which a new BOUNDARY namelist is read.
*     Otherwise, B2.5 simulation time, in seconds, when to read the next
*     BOUNDARY namelist file from BOUNDARY_FILENAME.
*     Only active if BOUNDARY_TIME_SWITCH is greater than 0.
*     Defaults to 0.0.
*  LCBS - integer array of size (NBC). List of core boundary segments,
*     according to their numbering in the current /BOUNDARY/ namelist.
*     For cases with closed field lines, the code will attempt to build the
*     array from the topology information available, i.e., it will contain the
*     list of 'South' boundaries forming a closed contour.
*     For linear cases, in case a "core" type boundary condition is desired,
*     then LCBS must be specified explicitly.
*     Defaults to 0.
*  WRITE_NML_BND - logical. If .true. (default), writes the content of the
*     namelist to stdout after it has been read.
*     Defaults to .true..
*  CON_FN - character*256 array of length NS * NBC. List of boundary profile
*     files to be imposed as boundary conditions for the density.
*     Defaults to b2.con.profile.
*  MOM_FN - character*256 array of length NS * NBC. List of boundary profile
*     files to be imposed as boundary conditions for the parallel velocity.
*     Defaults to b2.mom.profile.
*  ENI_FN - character*256 array of length NBC. List of boundary profile files
*     to be imposed as boundary conditions for the ion energy.
*     Defaults to b2.eni.profile.
*  ENE_FN - character*256 array of length NBC. List of boundary profile files
*     to be imposed as boundary conditions for the electron energy.
*     Defaults to b2.ene.profile.
*  POT_FN - character*256 array of length NBC. List of boundary profile files
*     to be imposed as boundary conditions for the electric potential.
*     Defaults to b2.pot.profile.
*
* NAMELIST /FEEDBACK/
*    Found in b2.feedback_save.parameters.
*    Read if present in the run directory.
*    Used internally by B2.5 to regulate feedback loops and restarts.
*    Not really intended for human use.
*    See b2cdci and SOLPS manual for description of the feedback schemes.
*  SAVED_CBSHE_CORE - real. Last value used for the core electron energy radial
*     flux feedback.
*     Defaults to 0.0.
*  SAVED_CBSHI_CORE - real. Last value used for the core ion energy radial flux
*     feedback.
*     Defaults to 0.0.
*  SAVED_CBSNA_CORE - real. Last value used for the core particle radial flux
*     feedback. Corresponds to 'isfeedback' B2.5 species.
*     Defaults to 0.0.
*  SAVED_CBSCH_CORE - real. Last value used for the core radial electric
*     current feedback.
*     Defaults to 0.0.
*  SAVED_CBSNA_PFR1 - real. Last value used for the radial inner PFR boundary
*     flux feedback. Corresponds to 'isfeedback' B2.5 species.
*     Defaults to 0.0.
*  SAVED_CBSNA_PFR2 - real. Last value used for the radial outer PFR boundary
*     flux feedback. Corresponds to 'isfeedback' B2.5 species.
*     Defaults to 0.0.
*  SAVED_CBSNA_SOL - real. Last value used for the radial particle flux
*     feedback in the SOL. Corresponds to 'isfeedback' B2.5 species.
*     Defaults to 0.0.
*  SAVED_NA_FEEDBACK_ACTUATOR - real array of size (NSPECIES). Last value used
*     for the feedback control actuator for each isonuclear sequence.
```

```
*     Defaults to 0.0.
*  SAVED_NA_FEEDBACK_PREV - real array of size (NSPECIES). Last value of the
*     density used for the feedback control for each isonuclear sequence.
*     Defaults to 0.0.
*
* NAMELIST /FEEDBACK_CONTROL/
*    Found in b2.feedback_control.parameters.
*    Allows for more complex feedback schemes.
*    This namelist is read if 'b2stbc_feedback' is set to '1' in the b2mn.dat
*    file or if LFEEDBACK=.true. in b2.boundary.parameters (and that latter file
*    is also read).
*  VACUUM_COMMUNICATION - integer. If > 0, allows for a
*     communication of particle fluxes across vacuum regions. This option only
*     applies to neutrals. The density boundary condition is based on the
*     difference between the average pressure and the local pressure.
*     Defaults to 0.
*  VACUUM_COMMUNICATION_NREG - integer array of size (NVAC). Number of
*     communicating vacuum regions.
*     Defaults to 0.
*  VACUUM_COMMUNICATION_METHOD - integer array of size (NVAC). Option for
*     resorbing the pressure difference.
*     1: Try to set a flux. Corr = (beta*ave_pressure - pressure)/temp * alpha
*     2: Try to set a density based on pressure equality.
*     Corr = \alpha * beta * pressure / Ti
*     Defaults to 0.
*  VACUUM_COMMUNICATION_IY - integer array of size (NVACREG,NVAC). Radial index
*     of the ring on which the neutral pressure is computed for region IREG.
*     Defaults to -2.
*  VACUUM_COMMUNICATION_IX1 - integer array of size (NVACREG,NVAC). Poloidal
*     lower bound of the range over which the neutral pressure is computed for
*     region IREG.
*     Defaults to -2.
*  VACUUM_COMMUNICATION_IX2 - integer array of size (NVACREG,NVAC). Poloidal
*     upper bound of the range over which the neutral pressure is computed for
*     region IREG.
*     Defaults to -2.
*  VACUUM_COMMUNICATION_ALPHA - real array of size (0:NSPECIES-1,NVAC).
*     Parameter for setting the pressure correction. See above.
*     Defaults to 0.0.
*  VACUUM_COMMUNICATION_BETA - real array of size (0:NSPECIES-1,NVAC).
*     Parameter for setting the pressure correction. See above.
*     Defaults to 1.0.
*  NA_FEEDBACK_TARGET - real array of size (0:NSPECIES-1). Sets the target
*     value of the quantity associated with species (ISPECIES) for the feedback
*     scheme defined by NA_FEEDBACK_CHOICE(ISPECIES).
*     Defaults to 0.0.
*  NA_FEEDBACK_TIME - real array of size (0:NSPECIES-1). Sets the time of
*     reference (in s) for the feedback of species (ISPECIES).
*     Defaults to 0.0.
*  NA_FEEDBACK_CHOICE - integer array of size (0:NSPECIES-1). Choice of
*     quantity on which the feedback is computed:
*     0: no action
*     1: local species density (averaged over the rectangle of cells
*     [IX1:IX2,IY1:IY2]). Summed over all charge states of that species
*     (including contributions from Eirene species).
*     2: local electron density (averaged over the rectangle of cells
*     [IX1:IX2,IY1:IY2]).
*     3: outer midplane separatrix electron density.
*     4: total particle content for that species summed over the rectangle of
*     cells [IX1:IX2,IY1:IY2], including contributions from Eirene species.
*     5: total ion content for that species (not including neutrals) summed
*     over the rectangle of cells [IX1:IX2,IY1:IY2].
*     6: neutral particle flux through the core boundary.
*     7: relative average concentration of this species at the separatrix.
*     8: relative average concentration of this species averaged over the
*     rectangle of cells [IX1:IX2,IY1:IY2].
*     9: inner midplane separatrix electron density.
```

```
*     10: neutral pressure at PFR southern boundary (by A. Kukushkin).
*     11: total particle content for that species outside the separatrix,
*     including contributions from Eirene species.
*     12: radiated power in Watts within the rectangle of cells
*     [IX1:IX2,IY1:IY2].
*     13: radiated power in Watts from the zone outside the separatrix,
*     including the PFR.
*     14: power flux density at outer bottom target. Usually
*     NA_FEEDBACK_INVERSE should be set to .true. for this actuator to work
*     properly.
*     15: total ion saturation current at the target. NA_FEEDBACK_INVERSE
*     should be set accordingly, depending on where the solution is with
*     respect to detachment threshold.
*     16: peak ion saturation current at the target. NA_FEEDBACK_INVERSE should
*     be set accordingly, depending on where the solution is with respect to
*     detachment threshold.
*     17: radiated power from this isonuclear sequence in Watts within the
*     rectangle of cells [IX1:IX2,IY1:IY2].
*     18: radiated power from this isonuclear sequence in Watts from the zone
*     outside the separatrix, including the PFR.
*     Defaults to 0.
*  NA_FEEDBACK_OPTION - integer array of size (0:NSPECIES-1). Option for
*     computing the new fedback quantity.
*     0: no action
*     1: rescale slowed by na_feedback_alpha
*     2: pure rescale
*     3: rescaling slowed by tanh_log
*     4: rescale done according to SOLPS4 formula
*     The target waveform for the particle content is
*     N = C + V*(time-T)
*     and the current puffing rate S is adjusted
*     S --> max(0, min(X,S + D)), where D = F*((N - <N>)/dt + (<N>_prev -
*     <N>)/dt_prev)
*     5: rescale done according to SOLPS4 formula: N = C*exp((time-T)*V)
*     6: rescale slowed by na_feedback_alpha (SOLPS4 style)
*     7: combinations of cases 2 and 3: if difference between current and
*     target value is within NA_FEEDBACK_GAMMA, then slowing down by tanh(log)
*     is applied, pure rescaling applied otherwise.
*     8: rescaling calculated similarly to option 7, however it is then
*     multiplied by the current pumping/puffing balance (S_pump - S_core -
*     S_ext). Only applicable to gas puff actuator, making it follow pumping
*     that suppresses the oscillations around the target value typical for gas
*     puff actuator.
*     9: rescaling formula from option 4 is used, with V=0 and target assumed
*     linearly proportional to the total particle content for this isonuclear
*     sequence (including contributions from Eirene species) within the
*     rectangle of cells [IX1:IX2,IY1:IY2].
*     10: rescaling formula from option 4 is used, with V=0 and target assumed
*     linearly proportional to the total particle content for this isonuclear
*     sequence outside the separatrix (including contributions from Eirene
*     species).
*     Defaults to 0.
*  NA_FEEDBACK_ACTUATOR - integer array of size (0:NSPECIES-1). Choice for the
*     actuator used for the feedback.
*     0: no action
*     1: gas puff strength, imposed via boundary condition for standalone cases
*     or via EIRENE stratum flux for coupled cases. See NA_FEEDBACK_IB for
*     setting which boundary condition or stratum this applies to.
*     2: rescale na [[Experimental!! Does not guarantee stable code runs]]
*     3: core boundary flux condition
*     4: absorption coefficient in Eirene (for coupled mode only)
*     Defaults to 0.
*  NA_FEEDBACK_ALPHA - real array of size (0:NSPECIES-1). Factor by which the
*     rescaling is slowed. Rescaling factor is :
*     Option 1: (1 + alpha*target/current) / (1 + alpha)
*     Option 3: 2**(tanh(log(x)/beta)*log(alpha)/log(2))
*     Options 4 and 5: Corresponds to parameter F
```

```
*     Defaults to 0.001.
*  NA_FEEDBACK_BETA - real array of size (0:NSPECIES-1). Factor by which the
*     rescaling is slowed. See above. Options 4 and 5: Corresponds to parameter
*     V (ffb_rtvn)
*     Defaults to 1.0.
*  NA_FEEDBACK_GAMMA - real array of size (0:NSPECIES-1). Ratio of current to
*     target feedback value at which linear rescaling transitions to tanh slowed
*     rescaling for feedback options 7 and 8.
*     Defaults to 0.01.
*  NA_FEEDBACK_CONST - real array of size (0:NSPECIES-1). Options 4 and 5:
*     Corresponds to parameter C. If negative, C is computed as the initial
*     total particle content of the sequence.
*     Defaults to 0.0.
*  NA_FEEDBACK_IX1 - integer array of size (0:NSPECIES-1). Lower poloidal bound
*     for the region over which the quantity used for the feedback of species
*     (ISPECIES) is computed, if applicable.
*     Defaults to -2.
*  NA_FEEDBACK_IX2 - integer array of size (0:NSPECIES-1). Upper poloidal bound
*     for the region over which the quantity used for the feedback of species
*     (ISPECIES) is computed, if applicable.
*     Defaults to -2.
*  NA_FEEDBACK_IY1 - integer array of size (0:NSPECIES-1). Lower radial bound
*     for the region over which the quantity used for the feedback of species
*     (ISPECIES) is computed, if applicable.
*     Defaults to -2.
*  NA_FEEDBACK_IY2 - integer array of size (0:NSPECIES-1). Upper poloidal bound
*     for the region over which the quantity used for the feedback of species
*     (ISPECIES) is computed, if applicable.
*     Defaults to -2.
*  NA_FEEDBACK_IB - integer array of size (0:NSPECIES-1). Index of boundary
*     condition through which the feedback is being applied. If negative refers
*     to the EIRENE stratum index instead. Specifically for actuator 4 it refers
*     to the index of special surface group.
*     Defaults to 0.
*  NA_FEEDBACK_IT - integer array of size (0:NSPECIES-1). Index of divertor
*     target to which feedback is applied.
*     Defaults to 0.
*  NA_FEEDBACK_ACTUATOR_MIN - real array of size (0:NSPECIES-1). Minimum value
*     of actuator being applied.
*     Defaults to 0.0.
*  NA_FEEDBACK_ACTUATOR_MAX - real array of size (0:NSPECIES-1). Maximum value
*     of actuator being applied.
*     Defaults to 0.0.
*  NA_FEEDBACK_PUFF_MIN - real array of size (0:NSPECIES-1). Minimum gas puff
*     being applied. Obsolete.
*     Defaults to 0.0.
*  NA_FEEDBACK_PUFF_MAX - real array of size (0:NSPECIES-1). Maximum gas puff
*     being applied. Obsolete.
*     Defaults to 0.0.
*  NA_FEEDBACK_OVERSHOOT - real array of size (0:NSPECIES-1). If the density is
*     larger than target*overshoot, the gas puff is turned off.
*     Defaults to 0.0.
*  NA_FEEDBACK_INVERSE - logical array of size (0:NSPECIES-1). If .true. both
*     feedback target and feedback current are inverted. If .false. actuator
*     value increases in order to increase current feedback value. If .true.
*     actuator value decreases in order to increase current feedback value.
*     Defaults to '.false.'.
*
* NAMELIST /PROFILE/
*    Found in b2.sources.profile.
*    Allows the user to set external sources to use within the code.
*    Unless specifying a divertor heat source, the source will be located in the
*    core and main SOL regions only.
*    Used if 'b2sral_inputfile' is set to 1.
*  NSDATA - integer array of size (NKIND_DATA,NKIND_SOURCE,0:NS). Number of
*     points over which the source profile of (kind_data,kind_source,is) is
*     defined. Should not exceed NY+2.
```

```
*     If KIND_DATA=1, the data is expressed as a profile in physical distance
*     from the separatrix (in metres) along the outer midplane.
*     The user can change this default reference location by use of the
*     'i[xy]ref_profile' switches.
*     If KIND_DATA=2, the data is expressed as a profile in flux distance from
*     the separatrix (not yet available).
*     Defaults to 0.
*  SDATA - real data of size (2,NY+2,NKIND_SOURCE,0:NS). For (i,ir,ik,is) in
*     (1:2,1:NY+2,1:NKIND_SOURCE,0:NS),
*     SDATA(1,ir,:,:) contains the radial location of the profile point (ir).
*     SDATA(2,ir,:,:) contains the source profile value at point (ir).
*     KIND_SOURCE=1 means particle source of species (is) (in particles/s/m3)
*     KIND_SOURCE=2 means parallel momentum source for species (is) (in N/m3)
*     KIND_SOURCE=3 means electron heat source (in Watts/m3)
*     KIND_SOURCE=4 means ion heat source (in Watts/m3)
*     KIND_SOURCE=5 means electric charge source (in Amperes/m3)
*     KIND_SOURCE=6 means non-ambipolar electron particle source (in e/s/m3)
*     Defaults to 0.0.
*  NXDATA - integer array of size (NKIND_DATA,NKIND_SOURCE,0:NS). Number of
*     points over which the axial profile of (kind_data,kind_source,is) is
*     defined. Should not exceed NX+2. Defaults to 0.
*     If KIND_DATA=1, the data is expressed as a profile in physical distance,
*     here connection length, rescaled from 0.0 to 1.0.
*     For closed field lines, the reference location for the zero of distance
*     is set by use of the 'i[xy]ref_profile' switches.
*     If KIND_DATA=2, the data is expressed as a profile in (ix) cell index,
*     again normalized from 0.0 to 1.0 to match the [0:nx-1] interval.
*  XDATA - real data of size (2,NY+2,NKIND_SOURCE,0:NS). Multiplier to the
*     poloidal source profile in the axial direction. Same convention for
*     KIND_SOURCE as above.
*     Defaults to 1.0.
*  DIVHEAT - real. Additional divertor ion heat source (in Watts/m3)
*     Defaults to 0.0.
*  IXREF_PROFILE - integer. Poloidal location along which the sources radial
*     profile is measured.
*     If not provided, the value from 'set_transport_ixref' is used.
*     Defaults to -2.
*  IYREF_PROFILE - integer. Radial location along which the sources axial
*     profile is measured.
*     If not provided, the value from 'set_transport_iyref' is used.
*     Defaults to -2.
*  SOURCES_FILENAME - character*256. Name of the next file to use for reading a
*     new /PROFILE/ namelist. Quantities not present in the new file will be
*     inherited from the old one.
*     Defaults to b2.sources.profile.
*  SOURCES_TIME_MOD - real. When the B2.5 run simulation time, in seconds,
*     modulo(SOURCES_TIME_MOD), reaches or exceeds SOURCES_TIME_SWITCH, reads
*     the new namelist from SOURCES_FILENAME. Also switches to the new namelist
*     if the ELM count (here time/SOURCES_TIME_MOD) changes. Only active if
*     SOURCES_TIME_MOD is greater than 0.
*     Defaults to 0.0.
*  SOURCES_TIME_SWITCH - real. If SOURCES_TIME_MOD is greater than 0, time (in
*     seconds) within an ELM cycle at which a new PROFILE namelist is read.
*     Otherwise, B2.5 simulation time, in seconds, when to read the next
*     PROFILE namelist file from SOURCES_FILENAME.
*     Only active if SOURCES_TIME_SWITCH is greater than 0.
*     Defaults to 0.0.
*
* NAMELIST /TRANSPORT/
*    Found in b2.transport.inputfile.
*    Allows the user to set transport coefficients profiles.
*    Only used is 'b2tqna_inputfile' is set to 1.
*    Overwrites b2ah.dat, b2mn.dat and b2.transport.parameters.
*    See SOLPS manual for details.
*  NDATA - integer array of size (NKIND_DATA,NCOEF,0:NS). Number of points over
*     which the source profile of (kind_data,kind_coef,is) is defined. Should
*     not exceed NY+2.
```

```
*     If KIND_DATA=1, the data is expressed as a profiles in physical distance
*     from the separatrix (in metres).
*     If KIND_DATA=2, the data is expressed as a profile in flux distance from
*     the separatrix (not yet available).
*     Defaults to 0.
*  TDATA - real data of size (3,NY+2,NKIND_COEFF,0:NS). For (i,ir,ik,is) in
*     (1:3,1:NY+2,1:NKIND_COEFF,0:NS), TDATA(1,ir,:,:) contains the radial
*     location of the profile point (ir).
*     By default, these are measured at the outer midplane, as distance to the
*     separatrix in metres. The user can change this by setting the
*     'set_transport_i[xy]ref' switches to choose a different location for the
*     distance reference.
*     TDATA(2,ir,:,:) contains the transport profile value at point (ir).
*     TDATA(3,ir,:,:) contains the ELM transport profile value at point (ir).
*     KIND_COEFF=1 means density-driven particle diffusivity for species (is)
*     KIND_COEFF=2 means pressure-driven particle diffusivity for species (is)
*     KIND_COEFF=3 means ion heat diffusivity for species (is)
*     KIND_COEFF=4 means electron heat diffusivity
*     KIND_COEFF=5 means poloidal pinch velocity for species (is)
*     KIND_COEFF=6 means radial pinch velocity for species (is)
*     KIND_COEFF=7 means viscosity for species (is)
*     KIND_COEFF=8 means field-driven radial current conductivity
*     KIND_COEFF=9 means temperature-driven radial current conductivity
*     Defaults to 0.0.
*  ADDSPEC - integer array of size (NS,NKIND_COEFF,0:NS). If
*     ADDSPEC(is,ikind,spec).ge.0, then the transport coefficient profile of
*     type (ikind) from species (spec) is also used for species with index
*     ADDSPEC(is,ikind,spec).
*     Defaults to -5.
*  TRANSPORT_FILENAME - character*256. Name of the next file to use for reading
*     a new /TRANSPORT/ namelist.
*     Defaults to b2.transport.inputfile.
*  TRANSPORT_TIME_MOD - real. When the B2.5 run simulation time, in seconds,
*     modulo(TRANSPORT_TIME_MOD), reaches or exceeds TRANSPORT_TIME_SWITCH,
*     reads the new namelist from TRANSPORT_FILENAME. Also switches to the new
*     namelist if the ELM count (here time/TRANSPORT_TIME_MOD) changes. Only
*     active if TRANSPORT_TIME_MOD is greater than 0.
*     Defaults to 0.0.
*  TRANSPORT_TIME_SWITCH - real. If TRANSPORT_TIME_MOD is greater than 0, time
*     (in seconds) within an ELM cycle at which a new TRANSPORT namelist is
*     read.
*     Otherwise, B2.5 simulation time, in seconds, when to read the next
*     TRANSPORT namelist file from TRANSPORT_FILENAME.
*     Only active if TRANSPORT_TIME_SWITCH is greater than 0.
*     Defaults to 0.0.
*  REGION_FLAGS - logical array of size (NREG,NKIND_COEFF). If
*     region_flags(ireg,ikind) is .true. (default), then the transport
*     parameters profiles for region (ireg) and kind (ikind) are used.
*     Defaults to .true..
*  NO_PFLUX - logical. If .true., transport coefficients profiles are not
*     implemented in the private flux regions.
*     Defaults to '.false.'.
*  NO_DIV - logical. If .true., transport coefficients profiles are not
*     implemented in the divertor regions.
*     Defaults to '.false.'.
*  POLOIDAL_SCALING - logical array of size (10). If .true., then the transport
*     coefficients profiles from the current b2.transport.inputfile are
*     increased by a factor of 1.0+Gaussian where Gaussian is a Gaussian profile
*     in the poloidal direction of amplitude SCALING_STRENGTH extending from
*     SCALING_IX_BEGIN to SCALING_IX_END inclusively. The profile has a decay
*     length of SCALING_WIDTH (in units of the number of poloidal cells).
*     Defaults to '.false.'.
*  SCALING_STRENGTH - real array of size 10. See above.
*     Defaults to 0.
*  SCALING_WIDTH - real array of size 10. See above. Defaults to about 1/3 of
*     the interval over which the scaling is to be done.
*     Defaults to 1/3 interval.
```

```
*  SCALING_IX_BEGIN - integer array of size 10. See above.
*     Defaults to -2.
*  SCALING_IX_END - integer array of size 10. See above.
*     Defaults to -2.
*  ELM_TIME_BEGIN - real. Time (in seconds) modulo ELM_TIME_PERIOD at which the
*     ELM phase begins and the ELM data must be used.
*     Defaults to 0.0.
*  ELM_TIME_END - real. Time (in seconds) modulo ELM_TIME_PERIOD at which the
*     ELM phase ends and the ELM data is no longer used.
*     Defaults to 0.0.
*  ELM_TIME_PERIOD - real. Indicates the real frequency of simulated ELMs. See
*     above for usage.
*     If zero, no ELM profiles are used.
*     Defaults to 0.0.
*  ELM_IX_BEGIN - integer. Poloidal position at which the ELM profile starts to
*     be applied.
*     Defaults to -2.
*  ELM_IX_END - integer. Poloidal position at which the ELM profile ends being
*     applied.
*     Defaults to -2.
*
* NAMELIST /NEUTRALS_SAVE/
*    Found in b2.neutrals_save.parameters.
*    Read if file is present.
*    Used internally for regulation and rescaling of volume recombination
*    sources and restarts. Superceded by use of 'eirene_dpc_fix' switch. See
*    code in eirene_mc and description in b2cdci for details.
*  SAVED_VOLREC - real array of size (NSTRAT). Contains the last value of the
*     strength of volume recombination sources from stratum (istra).
*     Defaults to 0.0.
*
* NAMELIST /NUMERICS/
*    Found in b2.numerics.parameters.
*    Used to specifiy time multipliers per equation and per region for the B2.5
*    solver.
*    Active if 'b2srdt_numerics_namelist' is set to 1 or greater.
*    It is recommended to use the same multiplier for all equations within one
*    region.
*    The time multipliers should not be used if running in time-dependent mode
*    (b2mndt_style.eq.2).
*  DTCO - real array of size (0:NS-1,0:NREG). Multiplier to the time used in
*     solving the continuity equation of species (is) in region (ireg).
*     Defaults to 1.0.
*  DTMO - real array of size (0:NS-1,0:NREG). Multiplier to the time used in
*     solving the parallel momentum equation of species (is) in region (ireg).
*     Defaults to 1.0.
*  DTEE - real array of size (0:NREG). Multiplier to the time used in solving
*     the electron heat equation in region (ireg).
*     Defaults to 1.0.
*  DTEI - real array of size (0:NREG). Multiplier to the time used in solving
*     the ion heat equation in region (ireg).
*     Defaults to 1.0.
*  DTCO_SOL - real array of size (0:NS-1). Multiplier to the time used in
*     solving the continuity equation of species (is) in the far SOL defined by
*     iy >= iy_SOL_dtXX.
*     Defaults to 1.0.
*  DTMO_SOL - real array of size (0:NS-1,0:NREG). Multiplier to the time used
*     in solving the parallel momentum equation of species (is) in the far SOL
*     defined by iy >= iy_SOL_dtXX.
*     Defaults to 1.0.
*  DTEE_SOL - real. Multiplier to the time used in solving the electron heat
*     equation in the far SOL defined by iy >= iy_SOL_dtXX).
*     Defaults to 1.0.
*  DTEI_SOL - real array of size (0:NREG). Multiplier to the time used in
*     solving the ion heat equation in the far SOL defined by iy >= iy_SOL_dtXX.
*     Defaults to 1.0.
*  IY_SOL_DTXX - integer. Defines the region of the SOL (iy >= iy_SOL_dtXX)
```

```
*     where the time multipliers dt[XX]_SOL should be applied. If
*     iy_SOL_dtXX.eq.-2 (default), no time multiplication is applied for any
*     equation independently of dt[XX]_SOL values. If iy_SOL_dtXX points to a
*     closed flux surface (i.e. if iy_SOL_dtXX.le.jsep), the code produces an
*     error message and stops.
*     Defaults to -2.
*  SOLVECO - logical array of size (0:NS-1,0:NREG). Indicates whether the
*     continuity equation for species (is) is to be solved in region (ireg).
*     Subservient to 'no_solve' switch.
*     Defaults to .true..
*  SOLVEMO - logical array of size (0:NS-1,0:NREG). Indicates whether the
*     parallel momentum equation for species (is) is to be solved in region
*     (ireg). Subservient to 'no_solve' switch.
*     Defaults to .true..
*  SOLVEMT - logical array of size (0:NREG). Indicates whether the total
*     parallel momentum equation is to be solved in region (ireg). For the total
*     parallel momentum equation to be solved in a region, all momentum
*     equations in the range defined by 'b2news_nsmin' and 'b2news_nsmax' must
*     have SOLVEMO true in that region. Subservient to 'no_solve' switch.
*     It is important to note that the 'b2news_no_solve' switch allows for
*     solving or not all momentum equations together as a whole, along with the
*     total equation.
*     Defaults to .true. unless using the time-dependent mode, in which case it
*     is set to .false. and should not be used.
*  SOLVEPO - logical array of size (0:NREG). Indicates whether the potential
*     energy equation is to be solved in region (ireg). Subservient to
*     'no_solve' switch.
*     Defaults to .true..
*  SOLVEEE - logical array of size (0:NREG). Indicates whether the electron
*     energy equation is to be solved in region (ireg). Subservient to
*     'no_solve' switch.
*     Defaults to .true..
*  SOLVEEI - logical array of size (0:NREG). Indicates whether the ion energy
*     equation is to be solved in region (ireg). Subservient to 'no_solve'
*     switch.
*     Defaults to .true..
*  SOLVEET - logical array of size (0:NREG). Indicates whether the total energy
*     equation is to be solved in region (ireg). For the equation to be solved
*     in region (ireg), both SOLVEEE and SOLVEEI must be true in that region.
*     Subservient to 'no_solve' switch.
*     It is important to note that the 'b2news_no_solve' switch allows to solve
*     or not to solve the electron and ion heat equations together with the
*     total energy equation, as a group.
*     Defaults to .true. unless using the time-dependent mode, in which case it
*     is set to .false. and should not be used.
*  TIME_FACTOR_REQUIRED - real. Minimum time scale of evolution allowed for all
*     equations. Only active is 'b2srsm_enable' is set to 1.
*     Defaults to 0.1.
*  CORE_DT_SUPPRESSION - real. De-multiplier to the timestep in the core. Only
*     active if less than 1. Should be larger than 0. Applies fully to the
*     innermost core ring of cells (IY .eq. -1). See CORE_DT_FACTOR for further
*     use.
*     Defaults to 1.0.
*  CORE_DT_FACTOR - real. Multiplier to the timestep in the core. Only active
*     is less than 1. Should be larger than 0. Multiplies each successive core
*     ring of cells (increasing IY) by CORE_DT_FACTOR, until the local time step
*     multiplier is equal to 1.
*     Defaults to 1.0.
*  CORR_CORE_DN - real array of size (0:NS-1). Pressure correction speed-up
*     parameter \alpha_a, acting on the density contribution from species a.
*     Will apply in the core region, except for the nrings outer surfaces. See
*     nrings_for_no_speedup_averaging for details.
*     See Pressure_correction_speed-up.pdf in $SOLPSTOP/doc for a full
*     description. Should be roughly equal to corr_core_dt below. Does not
*     apply to neutral species.
*     Defaults to 1.0.
*  CORR_CORE_DT - real. Pressure correction speed-up parameter \alpha_T, acting
```

```
*     on the temperature contributions. Will apply in the core region, except
*     for the nrings outer surfaces. See nrings_for_no_speedup_averaging for
*     details.
*     See Pressure_correction_speed-up.pdf in $SOLPSTOP/doc for a full
*     description. Should be roughly equal to corr_core_dn above.
*     Defaults to 1.0.
*  SNA_CORR - real array of size (1:NSPECIES). Multiplier to the neutral source
*     differential between puffing (plus sputtering plus core boundary flux plus
*     any external sources) and pumping, which is further added to the
*     ionization source for code speed-up. One multiplier must be provided per
*     isonuclear sequence. Such settings might be relevant for e.g. a mixture of
*     main ion and trace impurity.
*     See E. Kaveeva et al., Nucl. Fusion 58 (2018) 126018 for details and keep
*     in mind that in that paper the scheme was described in its original
*     species-unresolved form.
*     Defaults to 0.0.
*  TAUMAX - real array of size (1:NSPECIES). Maximum allowed fraction of
*     ionization source for a given isonuclear sequence from its neutral state
*     (coming from Eirene) to be added or subtracted to the real Eirene source
*     in the method of effective sources for code speed-up.
*     Defaults to 0.05.
*  DO_SNA_CORR_CORE - logical array of size (1:NSPECIES). Parameter which
*     controls exclusion/inclusion of confined region from/in the method of
*     effective sources for code speed-up for each isonuclear sequence. If
*     .true., then the flux through the core boundary of that species is taken
*     into account during the source correction computation, and the source
*     correction is applied on closed flux surfaces (default behavior). If
*     .false., then the particle imbalance is computed as puffing plus
*     sputtering plus any external sources minus pumping without account of the
*     flux through the core boundary, and no source correction is applied on
*     closed surfaces (such a setting might be helpful if the core flux is much
*     bigger than the puffing rate and the core is isolated from the SOL by a
*     strong transport barrier).
*     Defaults to .true..
*  NUMERICS_FILENAME - character*256. Name of the next file to use for reading
*     a new /NUMERICS/ namelist.
*     Defaults to b2.numerics.namelist.
*  NUMERICS_TIME_MOD - real. When the B2.5 run simulation time, in seconds,
*     modulo(NUMERICS_TIME_MOD), reaches or exceeds NUMERICS_TIME_SWITCH, reads
*     the new namelist from NUMERICS_FILENAME. Also switches to the new namelist
*     as the ELM count (here time/NUMERICS_TIME_MOD) changes. Only active if
*     NUMERICS_TIME_MOD is greater than 0.
*     Defaults to 0.0.
*  NUMERICS_TIME_SWITCH - real. If NUMERICS_TIME_MOD is greater than 0, time
*     (in seconds) within an ELM cycle at which a new NUMERICS namelist is read.
*     Otherwise, B2.5 simulation time, in seconds, when to read the next
*     NUMERICS namelist file from NUMERICS_FILENAME.
*     Only active if NUMERICS_TIME_SWITCH is greater than 0.
*     Defaults to 0.0.
*  WRITE_NML_NUM - logical. If .true. (default), writes the content of the
*     namelist to stdout after it has been read.
*     Defaults to .true..
*  ADD_TE_CORR_TO_PO - logical array of size (0:NREG). If .true. (default),
*     adds dte(ix,iy)/qe to the potential correction after the internal energy
*     balance equations are solved, where dte(ix,iy) is the electron temperature
*     correction on the time step. Individually set for each region index.
*     Defaults to .true..
*
* NAMELIST /TRANSPORT_MODELS/
*    Found in b2.transport_models_save.parameters.
*    Read if file is present.
*  ETA_HCE_MULT - real array of size (-1:NY). Used by the user specified
*     set_transport_eta transport model. See code for details.
*     Defaults to 1.0.
*
* NAMELIST /TRANSPORT/
*    Found in b2.transport.parameters.
```

```
*    Allows user to choose between different anomalous transport models.
*    Read if 'b2tqna_transport_namelist' is set to 1.
*    Overwrites data from b2ah.dat and b2mn.dat.
*    Does not apply to fluid neutral species.
*  FLAG_DNA - integer. Transport model flag for density-driven diffusion.
*  FLAG_DPA - integer. Transport model flag for pressure-driven dffusion.
*  FLAG_VLA - integer. Transport model flag for pinch velocity.
*  FLAG_VSA - integer. Transport model flag for viscosity.
*  FLAG_HCI - integer. Transport model flag for ion heat diffusivity.
*  FLAG_HCE - integer. Transport model flag for electron heat diffusivity.
*  FLAG_SIG - integer. Transport model flag for field-driven current radial
*     conductivity.
*  FLAG_ALF - integer. Transport model flag for temperature-driven current
*     radial conductivity.
*
*     All flags follow:
*     FLAG=0: Use model from b2ah.dat and b2mn.dat. Default.
*     FLAG=1: Constant, value set to PARM.
*     FLAG=2: 1/N model, multiplied by PARM.
*     FLAG=3: Bohm model, multiplied by PARM.
*     FLAG=4: flux-scaled model, multiplied by PARM.
*  PARM_DNA - real array of size (0:NS-1). Parameter for the density-driven
*     particle diffusion coefficient for species (is).
*  PARM_DPA - real array of size (0:NS-1). Parameter for the pressure-driven
*     particle diffusion coefficient for species (is).
*  PARM_VLA - real array of size (0:NS-1). Parameter for the anomalous radial
*     pinch velocity for species (is).
*  PARM_VSA - real array of size (0:NS-1). Parameter for the viscosity for
*     species (is).
*  PARM_HCI - real array of size (0:NS-1). Parameter for the heat diffusivity
*     coefficient for species (is).
*  PARM_HCE - real. Parameter for the electron heat diffusivity coefficient.
*  PARM_SIG - real. Parameter for the anomalous radial field-driven current
*     conductivity.
*  PARM_ALF - real. Parameter for the anomalous radial temperature-driven
*     current conductivity.
*  TRANSPORT_FILENAME - character*256. Name of the next file to use for reading
*     a new /TRANSPORT/ namelist.
*     Defaults to b2.transport.parameters.
*  TRANSPORT_TIME_MOD - real. When the B2.5 run simulation time, in seconds,
*     modulo(TRANSPORT_TIME_MOD), reaches or exceeds TRANSPORT_TIME_SWITCH,
*     reads the new namelist from TRANSPORT_FILENAME. Also switches to the new
*     namelist if the ELM count (here time/TRANSPORT_TIME_MOD) changes. Only
*     active if TRANSPORT_TIME_MOD is greater than 0.
*     Defaults to 0.0.
*  TRANSPORT_TIME_SWITCH - real. If TRANSPORT_TIME_MOD is greater than 0, time
*     (in seconds) within an ELM cycle at which a new TRANSPORT namelist is
*     read.
*     Otherwise, B2.5 simulation time, in seconds, when to read the next
*     TRANSPORT namelist file from TRANSPORT_FILENAME.
*     Only active if TRANSPORT_TIME_SWITCH is greater than 0.
*     Defaults to 0.0.
*  CFLME - real. Main multiplier to the electron heat flux limit.
*     Defaults to value inherited from b2ah.dat (or b2mn.dat if specified in
*     b2cmpt block).
*  CFLMI - real. Main multiplier to the ion heat flux limit.
*     Defaults to value inherited from b2ah.dat (or b2mn.dat if specified in
*     b2cmpt block).
*  CFLMV - real. Main multiplier to the viscous heat flux limit.
*     Defaults to value inherited from b2ah.dat (or b2mn.dat if specified in
*     b2cmpt block).
*  CFLAL - real. Multiplier to the thermo-electric coefficient flux limit.
*     Defaults to value inherited from b2ah.dat (or b2mn.dat if specified in
*     b2cmpt block).
*  CFLAB - real. Multiplier to the friction force flux limit.
*     Defaults to value inherited from b2ah.dat (or b2mn.dat if specified in
*     b2cmpt block).
```

```
*  CFLME_CORE - real. Limit to multiplier to the electron heat flux limit in
*     the deep core.
*     Defaults to 0.0.
*  CFLME_SOL - real. Limit to multiplier to the electron heat flux limit in the
*     far SOL.
*     Defaults to 0.0.
*  CFLME_TANH_A - real. Radial location, measured (in metres) as a radial
*     distance along the OMP away from the separatrix, for the center of the
*     tanh profile multiplying the electron heat flux limit.
*     Defaults to 0.0.
*  CFLME_TANH_B - real. Radial width, measured (in metres) as a radial distance
*     along the OMP, for the tanh profile multiplying the electron heat flux
*     limit. Must be positive.
*     Defaults to 0.0.
*  CFLMI_CORE - real. Limit to multiplier to the ion heat flux limit in the
*     deep core.
*     Defaults to 0.0.
*  CFLMI_SOL - real. Limit to multiplier to the ion heat flux limit in the far
*     SOL.
*     Defaults to 0.0.
*  CFLMI_TANH_A - real. Radial location, measured (in metres) as a radial
*     distance along the OMP away from the separatrix, for the center of the
*     tanh profile multiplying the ion heat flux limit.
*     Defaults to 0.0.
*  CFLMI_TANH_B - real. Radial width, measured (in metres) as a radial distance
*     along the OMP, for the tanh profile multiplying the ion heat flux limit.
*     Must be positive.
*     Defaults to 0.0.
*  CFLMV_CORE - real. Limit to multiplier to the viscous heat flux limit in the
*     deep core.
*     Defaults to 0.0.
*  CFLMV_SOL - real. Limit to multiplier to the viscous heat flux limit in the
*     far SOL.
*     Defaults to 0.0.
*  CFLMV_TANH_A - real. Radial location, measured (in metres) as a radial
*     distance along the OMP away from the separatrix, for the center of the
*     tanh profile multiplying the viscous heat flux limit.
*     Defaults to 0.0.
*  CFLMV_TANH_B - real. Radial width, measured (in metres) as a radial distance
*     along the OMP, for the tanh profile multiplying the viscous heat flux
*     limit. Must be positive.
*     Defaults to 0.0.
*     For the electron, ion, and viscous heat flux limit coefficients, if the
*     main multiplier is not specified but CFLM*_CORE and CFLM*_SOL are set to
*     equal non-zero values, their value will be used. If CFLM*_CORE and
*     CFLM*_SOL are different and non-zero, then a radial tanh profile,
*     centered at radial OMP position CFLM*_TANH_A (in metres) and of width
*     CFLM*_TANH_B is applied. The flux limit coefficients in the PFR are set
*     to a constant value equal to that at the separatrix. The OMP is located
*     at the poloidal index provided by 'b2news_fac_ref', or, if not specified,
*     'b2mwti_jxa'.
*     For all flux limit multipliers, setting them to zero turns off the
*     corresponding flux limiter. Otherwise, the multipliers must be positive.
*  WRITE_NML_TRANSP - logical. If .true. (default), writes the content of the
*     namelist to stdout after it has been read.
*     Defaults to .true..
*  VOUT_CNV - real array of size (0:NS-1). Value of the "blob" convection
*     velocity for species (is) at the outer grid edge (in m/s).
*     Defaults to 0.0.
*  PW0_CNV - real array of size (0:NS-1). Exponent in radial profile of the
*     "blob" velocity for species (is).
*     Defaults to 1.0.
*  PW1_CNV - real array of size (0:NS-1). First exponent in poloidal profile of
*     the "blob" velocity for species (is).
*     Defaults to 5.0.
*  PW2_CNV - real array of size (0:NS-1). Second exponent in a poloidal profile
*     of the "blob" velocity for species (is).
```

```
*     Defaults to 2.0.
*     The four *_CNV parameters are only activated if 'b2tqna_user_transport'
*     is set to '6'.
*
* NAMELIST /USER/
*    Found in b2.user.parameters.
*    Contains additional data for user-specific diagnostics. Always read if
*    present.
*    This version originally provided by Andrei Kukushkin for ITER cases,
*    however, it may be used (at least partially) for arbitrary machine.
*  LHETRGTS - integer array of size (NLIM). List of surface indices (EIRENE
*     notation) which are used for calculation of helium enrichment.
*     Defaults to Eirene recycling target surfaces defined in the LTNS array.
*  LPFRB_I - integer. B2.5 x-cell index corresponding to the first edge of the
*     bypass between the inner and outer divertor (Western divertor, edge
*     closest to target). If non-positive, counted backwards from the X-point
*     location in the lower PFR, from the inner upper target in the upper PFR,
*     from the lower outer target in the outer SOL, and from the inner upper
*     target in the inner SOL.
*     Defaults to 0.
*  LPFRB_O - integer. B2.5 x-cell index corresponding to the first edge of the
*     bypass between the inner and outer divertor (Eastern divertor, edge
*     closest to target). If non-positive, counted backwards from the outer
*     lower target in the lower PFR, from the lower outer target in the outer
*     SOL, from the upper outer target in the upper PFR, and from the inner
*     upper target in the inner SOL.
*     Defaults to 0.
*  LPFRT_I - integer. B2.5 x-cell index corresponding to the second edge of the
*     bypass between the inner and outer divertor (Western divertor, edge
*     furthest to target). If non-positive, counted backwards from the X-point
*     location in the PFR, from the outer target in the outer SOL for a SN, from
*     the top targets in the SOL for a DN.
*     Defaults to 0.
*  LPFRT_O - integer. B2.5 x-cell index corresponding to the second edge of the
*     bypass between the inner and outer divertor (Eastern divertor, edge
*     furthest to target). If non-positive, counted backwards from the X-point
*     location in the PFR, from the outer target in the outer SOL for a SN, from
*     the top targets in the SOL for a DN.
*     Defaults to 0.
*  J_HE_AT - integer. Species index of the helium atoms in Eirene. The code
*     attempts to find a match by default.
*  J_NE_AT - integer. Species index of the neon atoms in Eirene. The code
*     attempts to find a match by default.
*  J_H_AT - integer array of size (3). Species indices of the hydrogen isotopes
*     in Eirene. If using only one hydrogen species, only the first element
*     needs to be provided. Otherwise the 3 elements correspond to H/D/T. The
*     code attempts to find a match by default.
*  L_H_MOL - integer array of size (NMOL,3). Number of hydrogen isotope nuclei
*     for molecules in Eirene. If using only one hydrogen species, only the
*     first element needs to be provided. Otherwise the 3 elements correspond to
*     H/D/T. The code attempts to find a match by default.
*  FUSION_POWER - real. Fusion power occuring in core (including neutrons, in
*     Megawatts).
*     Defaults to 0.0.
*  SPMP_HE_TO_D - real. Ratio of He to DT pumping speeds (typically, 0.8).
*     Defaults to 1.0.
*  LPFRS_PMP - integer. Location of the pump. 0 no pump at all (default), 1 -
*     SN of lower DN PFR, 2 - (outer) SOL, 3 - upper DN PFR, 4 - inner SOL for
*     DN
*     Defaults to 0.
*  NPFRGRP - integer. Actual number of surface groups for PFR flows.
*     Defaults to 0.
*  LPFRGRP - integer array of size (NLIM). List of surface segments for PFR
*     flows.
*  IPFRGRP - integer array of size (NPFRGRP). First positions in this list for
*     each group.
*     Defaults to 0.
```

```
*  JPFRGRP - integer array of size (NPFRGRP). Last positions in this list for
*     each group.
*     Defaults to 0.
*  GPFRGRP - character*8 array of size (NPFRGRP). Labels for surface groups for
*     PFR flows.
*     Defaults to ' '.
*  NNTRGRP - integer. Actual number of surface groups for neutral data.
*  LNTRGRP - integer array of size (NLIM). List of surface segments for neutral
*     data.
*  INTRGRP - integer array of size (NNTRGRP). First positions in this list for
*     each group.
*  JNTRGRP - integer array of size (NNTRGRP). Last positions in this list for
*     each group.
*  GNTRGRP - character*8 array of size (NNTRGRP). Labels for the neutral data
*     surface groups.
*  SPMP_NOM - real. Nominal pumping speed.
*     Defaults to 0.0.
*  TE_DET_THRESHOLD - real. Electron temperature at the target (in eV) which is
*     used a threshold identifying the transition of the local magnetic flux
*     tube to detachment. The width of the zone with Te less than
*     TE_DET_THRESHOLD (counted from the strike point towards the SOL side) is
*     outputted into user_SPb.trc. This distance monotonically increases with
*     increasing of radiated power fraction or with decreasing of peak target
*     energy loads, therefore it may be used as a measure of the degree of
*     detachment.
*     Defaults to 2.0.
*  Q95_ALBLL - real. Safety factor used to estimate the ballooning "alpha"
*     parameter, if not assigned will be estimated as L_conn/(2*pi*R) at the
*     core interface.
*     Defaults to 0.0.
*  R0_ALBLL - real. Major radius used to estimate the ballooning "alpha"
*     parameter. If not assigned, will be estimated as 0.5*(Rmin-Rmax) at the
*     core interface.
*     Defaults to 0.0.
*  B0_ALBLL - real. Magnetic field used to estimate the ballooning "alpha"
*     parameter. If not assigned, will be estimated as Bomp*R0/Romp at the core
*     interface.
*     Defaults to 0.0.
*  TRGSHP - real array of size (NTRGTS). Target shaping factor applied to
*     plasma heat loads in peak heat flux estimate calculated by b2mod_usertrc.
*     Defaults to 1.0.
*  FILEDATA - logical array of size 12. Switches to turn on/off the tracing
*     files controlled by the ank_tracing switch. The files are defined in
*     b2mod_diag, in order:
*     1:  ntrc_srt  :  tracing/intshrt.trc    :  integral data (short version)
*     2:  ntrc_tst  :  tracing/test.trc       :  test data (can be absent)
*     3:  ntrc_res  :  tracing/residuals.trc  :  residuals by regions
*     4:  ntrc_src  :  tracing/sources.trc    :  integral sources
*     5:  ntrc_bln  :  tracing/blnn.trc       :  particle balance - old style
*     by A.Kukushkin
*     6:  ntrc_ble  :  tracing/blne.trc       :  energy balance
*     7:  ntrc_int  :  tracing/integral.trc   :  additional integrals
*     8:  ntrc_usr  :  tracing/user.trc       :  user-specific data
*     9:  ntrc_blm  :  tracing/blnm.trc       :  parallel momentum balance
*     10:  ntrc_sep  :  tracing/sepdata.trc    :  separatrix data
*     11:  ntrc_bls  :  tracing/blnn_SPb.trc   :  particle balance - new SPb
*     style compatible with speed-up schemes (Kaveeva et al., Nucl. Fusion 58
*     126018)
*     12:  ntrc_uss  :  tracing/user_SPb.trc   :  user-specific data introduced
*     by SPb group - contains alternative to user.trc set of data and is more
*     straightforward in programming
*
*     If there is no b2.user.parameters file present, the flags for user.trc
*     and user_SPb.trc are set to .false..
*     If the tracing files are to be appended but a reading error occurs when
*     opening them, the corresponding filedata element is overwritten to
*     .false..
```

```
*     Defaults to .true..
*  USER_FILENAME - character*80. Filename where /USER/ namelist is stored.
*     Defaults to b2.user.parameters.
*
* NAMELIST /SPUTTER_SAVE/
*    Found in b2.sputter_save.parameters.
*    Read if file is present.
*    Used to regulate the temporal evolution of sputtering and restarts.
*    Not intended for human use.
*  NX_SP - integer. Array dimension NX used in this namelist.
*     Defaults to 'nx'.
*  NY_SP - integer. Array dimension NY used in this namelist.
*     Defaults to 'ny'.
*  NS_SP - integer. Array dimension NS used in this namelist.
*     Defaults to ns.
*  SPUTTER_YIELD - real array of size (-1:NX,-1:NY,0:NS-1,1:2). Contains the
*     chemical sputtering yield of species (is) at position (ix,iy) in element
*     (:,:,:,1) and the physical sputtering yield in element (:,:,:,2).
*     Defaults to 0.0.
*  SPUTTER_YIELD2 - real array of size (-1:NX,-1:NY,0:NS-1,1:2). Contains the
*     energy chemical sputtering yield of species (is) at position (ix,iy) in
*     element (:,:,:,1) and the energy physical sputtering yield in
*     element(:,:,:,2).
*     Defaults to 0.0.
*
* NAMELIST /ATOMIC_PHYSICS_RESCALE/
*    Found in b2.atomic_physics_rescale.parameters.
*    Read if 'b2mndr_atomic_physics_rescale' is set to 1.
*    Contains rescaling multipliers for atomic physics rates.
*  RESCALE_SA - real array of size(0:NS-1). Scaling factors for rtsa:
*     ionisation rates of species (is).
*     Defaults to 1.0.
*  RESCALE_RA - real array of size(0:NS-1). Scaling factors for rtra:
*     recombination rates of species (is).
*     Defaults to 1.0.
*  RESCALE_QA - real array of size(0:NS-1). Scaling factors for rtqa: electron
*     cooling rates of species (is).
*     Defaults to 1.0.
*  RESCALE_CX - real array of size(0:NS-1). Scaling factors for rtcx: charge
*     exchange rates of species (is).
*     Defaults to 1.0.
*  RESCALE_RD - real array of size(0:NS-1). Scaling factors for rtrd: line
*     radiation rates of species (is).
*     Defaults to 1.0.
*  RESCALE_BR - real array of size(0:NS-1). Scaling factors for rtbr:
*     bremsstrahlung radiation rates of species (is).
*     Defaults to 1.0.
*
* NAMELIST /GRID/
*    Found in linear.geometry.
*    These switches are used to produce a geometry for a linear device.
*  NCOIL - integer. Number of coils in the device. For each coils the switches
*     R1, Z1, R2, Z2, CURRENT, windings and nlayers have to be provided.
*     Defaults to 20.
*  R1 - real array of length NCOIL. Inner radius of the coil in units of m.
*     Defaults to 0.0.
*  Z1 - real array of length NCOIL. Axial start position of the coil in units
*     of m.
*     Defaults to 0.0.
*  R2 - real array of length NCOIL. Outer radius of the coil in units of m.
*     Defaults to 0.0.
*  Z2 - real array of length NCOIL. Axial end position of the coil in units of
*     m.
*     Defaults to 0.0.
*  CURRENT - real array of length NCOIL. Current in the coil in units of A.
*     Defaults to 0.0.
*  WINDINGS - integer array of length NCOIL. Number of windigs of each coil.
```

```
*     Defaults to 0.
*  LAYERS - integer array of length NCOIL. Number of layers to divide each coil
*     in the radial direction.
*     Defaults to 5.
*  STEPSIZE - real. Initial spacial step size used in the calculation of the
*     field in units of m.
*     Defaults to 1.0e-4.
*  BTOR - real. Toroidal magnetic field in units of T.
*     Defaults to 0.0.
*  FN_Z - character. Two column file with the profile of axial cell size in the
*     numerical space. First line of the file determine the number of rows to
*     read.
*     Defaults to 'z.dat'.
*  FN_R - character. Two column file with the profile of radial cell size in
*     the numerical space. First line of the file determine the number of rows
*     to read.
*     Defaults to 'r.dat'.
*  R_NE - real. Radial position in m of the northeastmost point of the plasma
*     region.
*     Defaults to 0.0.
*  Z_NE - real. Axial position in m of the northeastmost point of the plasma
*     region.
*     Defaults to 0.0.
*  R_SE - real. Radial position in m of the southeastmost point of the plasma
*     region.
*     Defaults to 0.0.
*  Z_SE - real. Axial position in m of the southeastmost point of the plasma
*     region.
*     Defaults to 0.0.
*  R_SW - real. Radial position in m of the southwestmost point of the plasma
*     region.
*     Defaults to 0.0.
*  Z_SW - real. Axial position in m of the southwestmost point of the plasma
*     region.
*     Defaults to 0.0.
*  R_NW - real. Radial position in m of the northwestmost point of the plasma
*     region.
*     Defaults to 0.0.
*  Z_NW - real. Axial position in m of the northwestmost point of the plasma
*     region.
*     Defaults to 0.0.
*  R_START - real. Starting radial point in m for the distribution of cells
*     provided by fn_r.
*     Defaults to 0.0.
*  R_END - real. End radial point in m for the distribution of cells provided
*     by fn_r.
*     Defaults to 0.0.
*  Z_START - real. Starting axial point in m for the distribution of cells
*     provided by fn_z.
*     Defaults to 0.0.
*  Z_END - real. End axial point in m for the distribution of cells provided by
*     fn_z.
*     Defaults to 0.0.
*  GC_N - real. Guard cell size for the North boundary in m.
*     Defaults to 1.0e-4.
*  GC_S - real. Guard cell size for the South boundary in m.
*     Defaults to 1.0e-4.
*  GC_W - real. Guard cell size for the West boundary in m.
*     Defaults to 1.0e-4.
*  GC_E - real. Guard cell size for the East boundary in m.
*     Defaults to 1.0e-4.
*  nExternalRegions - integer. Number of external regions to be used in the
*     grid.
*     Defaults to 10.
*  nNodesInRegion - integer array of size nExternalRegions. Number of nodes per
*     region. Minimum of 3 is required per region.
*     Defaults to 0.
```

```
*  regionR - real array of size nNodesInRegion,nExternalRegions. Radial
*     positions in m of the nodes defininig the external region
*     Defaults to 0.0.
*  regionZ - real array of size nNodesInRegion,nExternalRegions. Axial
*     positions in m of the nodes defininig the external region
*     Defaults to 0.0.
*  startCellR - integer array of size nExternalRegions. Radial starting index
*     of the node on the B2.5 boundary for the region.
*     Defaults to -1.
*  startCellZ - integer array of size nExternalRegions. Axial starting index of
*     the node on the B2.5 boundary for the region.
*     Defaults to -1.
*  endCellR - integer array of size nExternalRegions. Radial final index of the
*     node on the B2.5 boundary for the region.
*     Defaults to -1.
*  endCellZ - integer array of size nExternalRegions. Axial final index of the
*     node on the B2.5 boundary for the region.
*     Defaults to -1.
*  WRITE_NML_LD - logical. If .true., writes the content of the namelist to
*     stdout after it has been read.
*     Defaults to '.false.'.
*
*-----------------------------------------------------------------------
*.end b2cdcn

      end subroutine b2cdcn
```

Appendix B

The physics of B2.5

An extensive review of `B2-Eirene` has been published as [16]. It provides a full description of the **SOLPS5.0** physics model, which can be obtained in the code by setting the `b2mndt_style` switch to `0`. It is asked that authors using `SOLPS` simulation results cite that work when referring to the code. This review, only some short sections of which are detailed below, can be found as part of the `SOLPS` package at `${SOLPSTOP}/doc/CPP_2006.pdf`. Most of the discussion and definitions presented here apply to both the **SOLPS5.0** and **SOLPS 5.2** physics models. In cases where the models differ, the use of the **SOLPS 5.2** model is recommended, although the **SOLPS5.0** model remains available in the code for backward compatibility.

The equations are written on a curvilinear orthogonal coordinate system. The x-coordinate varies along flux surfaces and the y-coordinate varies perpendicular to flux surfaces (see figure B.1), z is the toroidal direction.

The metric coefficients are $h_x = 1/\|\nabla x\|$, $h_y = 1/\|\nabla y\|$ and $\sqrt{g} = h_x h_y h_z$. One can replace h_z by $2\pi R$, where R is the local major radius. Components of a vector are the projections on unit basis vectors ; they are not the contravariant or covariant components ($b_x = B_x/B$, $b_z = B_z/B$). Ion velocities $V_\perp$ (velocity perpendicular both to magnetic field B and the y-axis) and V_y are determined from the two components of the momentum balance equation for ions perpendicular to a magnetic field. Using the Braginskii expression for the friction force, one obtains:

$$V_\perp = V_\perp^{(a)} - \frac{1}{enB}\frac{1}{h_y}\frac{\partial nT_i}{\partial y} + V_\perp^{(in)} + V_\perp^{(vis)} + V_\perp^{(s)} \tag{B.1}$$

$$V_y = V_y^{(a)} + \frac{B_z}{enB^2}\frac{1}{h_x}\frac{\partial nT_i}{\partial x} + V_y^{(in)} + V_y^{(vis)} + V_y^{(s)} \tag{B.2}$$

Velocities $V_{\perp,y}^{(a)}$ are the sum of the $\mathbf{E}\times\mathbf{B}$ drifts and diffusive and thermo-diffusive velocities. This part corresponds to the ambipolar flux. It is the same both for ions and electrons. The second term in the r.h.s. of Eqs. (B.1) and (B.2) corresponds to diamagnetic flow of ions, while the other terms are caused by inertial, viscous and ion-neutral friction forces. These non-ambipolar terms are connected with currents: $\vec{V}^{(in)} = \vec{j}^{(in)}/en, \vec{V}^{(vis)} = \vec{j}^{(vis)}/en, \vec{V}^{(s)} = \vec{j}^{(s)}/en$.
In the presence of anomalous transport (D is the classical diffusion coefficient, $D = \frac{T_e+T_i}{eb}\frac{\nu_{ei}}{\omega_{ce}}$, with electron cyclotron frequency ω_{ce} and electron-ion collision frequency ν_{ei})

$$V_\perp^{(a)} = -\frac{1}{B}\frac{1}{h_y}\frac{\partial\Phi}{\partial y} - \frac{D}{T_e+T_i}\frac{b_z}{h_x}\left(\frac{\partial p}{n\partial x} - \frac{3}{2}\frac{\partial T_e}{\partial x}\right) - D_n^{(AN)}\frac{1}{h_x n}\frac{\partial n}{\partial x} - D_p^{(AN)}\frac{1}{h_x n}\frac{\partial p_i}{\partial x} + V_\perp^{(AN)} \tag{B.3}$$

$$V_y^{(a)} = \frac{B_z}{B^2}\frac{1}{h_x}\frac{\partial\Phi}{\partial x} - \frac{D}{T_e+T_i}\frac{1}{h_y}\left(\frac{\partial p}{n\partial y} - \frac{3}{2}\frac{\partial T_e}{\partial y}\right) - D_n^{(AN)}\frac{1}{h_y n}\frac{\partial n}{\partial y} - D_p^{(AN)}\frac{1}{h_y n}\frac{\partial p_i}{\partial y} + V_y^{(AN)} \tag{B.4}$$

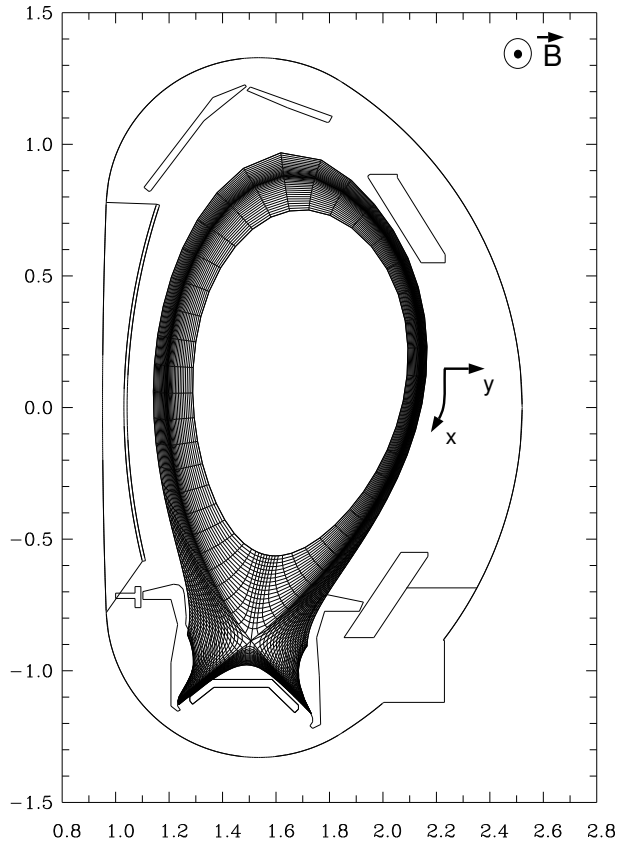


Figure B.1: Coordinate system: x is the poloidal coordinate, y is the radial coordinate orthogonal to the flux surfaces. The coordinate $\perp$ corresponds to the direction perpendicular both to the magnetic field and to the y-axis. The directions of magnetic field and plasma current correspond to normal operation conditions of ASDEX–Upgrade (ion ∇B drift directed towards the X-point).

The poloidal velocity is a sum $V_x = b_z V_\perp + b_x V_\parallel$.

In the particle balance the fact that the diamagnetic velocity is almost divergence free is taken into account. Therefore, instead of the diamagnetic velocity it is possible to introduce an effective velocity $\vec{\tilde{V}}^{(dia)}$, which gives the same contribution to the divergence of the particle flux:

$$\frac{\partial n}{\partial t} + \frac{1}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x} n \left(b_x V_\parallel + b_z V_\perp^{(0)}\right)\right) + \frac{1}{\sqrt{g}}\frac{\partial}{\partial y}\left(\frac{\sqrt{g}}{h_y} n V_y^{(0)}\right) = S^n \tag{B.5}$$

$$V_\perp^{(0)} = V_\perp^{(a)} + V_\perp^{(in)} + V_\perp^{(vis)} + V_\perp^{(s)} + \tilde{V}_\perp^{(dia)} \tag{B.6}$$

$$V_y^{(0)} = V_y^{(a)} + V_y^{(in)} + V_y^{(vis)} + V_y^{(s)} + \tilde{V}_y^{(dia)} \tag{B.7}$$

$$\tilde{V}_\perp^{(dia)} = \frac{T_i B_z}{e}\frac{\partial}{h_y \partial y}\left(\frac{1}{B^2}\right) \tag{B.8}$$

$$\tilde{V}_y^{(dia)} = -\frac{T_i B_z}{e}\frac{\partial}{h_x \partial x}\left(\frac{1}{B^2}\right) \tag{B.9}$$

The velocity $\tilde{V}^{(dia)}$ is an effective velocity. It represents vertical guiding centre drift of ions caused by ∇B.

Combining the inertia and gyroviscosity terms in the Braginskii equations, one finds

$$m_i\left[\frac{\partial nV_\parallel}{\partial t}+\frac{1}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x}n\left(V_\perp^{(0)}b_z+V_\parallel b_x\right)V_\parallel\right)+\frac{1}{\sqrt{g}}\frac{\partial}{\partial y}\left(\frac{\sqrt{g}}{h_y}nV_y^{(0)}V_\parallel\right)\right]=$$
$$-\frac{b_x}{h_x}\frac{\partial nT_i}{\partial x}-b_x\frac{en}{h_x}\frac{\partial\Phi}{\partial x}+F_k+\frac{4}{3}b_xB^{3/2}\frac{\partial}{h_x\partial x}\left(\frac{\eta_0 b_x}{B^2}\frac{\partial\left(\sqrt{B}\left(V_\parallel+\frac{B}{B_x}\left(V_x^{(dia)}+V_x^{(E)}\right)\right)\right)}{h_x\partial x}\right)$$
$$+B^{3/2}b_x\frac{\partial}{h_x\partial x}\left(\frac{b_x}{\nu_{ii}B^2}\frac{\partial\left(\sqrt{B}q_{i\parallel}^{(0)}\right)}{h_x\partial x}\right)+\frac{1}{\sqrt{g}}\frac{\partial}{\partial y}\left(\frac{\sqrt{g}}{h_y^2}\eta_2\frac{\partial V_\parallel}{\partial y}\right)$$
$$+\frac{1}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x^2}\eta_2\frac{\partial V_\parallel}{\partial x}\right)+S_{i\parallel}^m+R_{ie\parallel}. \tag{B.10}$$

The diamagnetic terms in the l.h.s. of (B.10) are almost cancelled [17, 18, 19, 20], so that velocities normal to the flux surface in the second and third term, as in the particle balance equation, are determined by (B.1) and (B.2). In [20] the divergent part of diamagnetic velocity has been neglected. Here, it is kept, because the corresponding flux is still relatively large and can be comparable with the diffusive flux even for anomalous diffusion coefficients. The authors of [17, 18, 19] also kept this flux. In contrast to [17, 18, 19] there is an additional term of the same kind in the r.h.s. of Eq. (B.10):

$$F_k=-m_i\left(\frac{1}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x}nV_\parallel\tilde{V}_\perp^{(dia)}b_z\right)+\frac{1}{\sqrt{g}}\frac{\partial}{\partial y}\left(\frac{\sqrt{g}}{h_y}nV_\parallel\tilde{V}_y^{(dia)}\right)\right) \tag{B.11}$$

This parallel force is the Coriolis force, connected with the diamagnetic drift velocity and toroidal (parallel) rotation. This force can be obtained in a different way by averaging the drift kinetic equation [21] and is of the same order as the inertia terms.

Parallel viscosity is represented by the fourth and fifth terms in the r.h.s. The former is the Balescu parallel viscosity with coefficient $\eta_0=\frac{4}{3}0.96\sqrt{2}nT_i/\nu_{ii}$. A detailed derivation is given at the end of this section, see B.1. The latter corresponds to the parallel viscosity driven by parallel ion heat flux $q_{i\parallel}^{(0)}=-\kappa_{i,\parallel}b_x\frac{1}{h_x}\frac{\partial T_i}{\partial x}$. This viscosity is absent in the Braginskii equations, however, in a tokamak, where parallel particle flux multiplied by temperature is of the order of the ion heat flux, these two viscosities are of the same order. On closed flux surfaces far from the separatrix, the flux-surface-averaged parallel viscosity coincides with the well-known neoclassical expression for the Pfirsch-Schlüter regime [22]. Thus, parallel viscosity in Eq. (B.10) provides smooth transition to neoclassical theory. The classical perpendicular viscosity coefficient can be replaced by an anomalous value as in [17, 18, 19]: $\eta_2=nm_iD_n^{(AN)}$. Perpendicular viscosity and inertia terms with the anomalous particle flux are responsible for the transport of the parallel momentum in radial and poloidal directions. The radial transport can be rather important despite the fact that $\eta_2\ll\eta_0$, since the scale in the radial direction is rather small.

The term $S_{i\parallel}^m$ represents the parallel momentum loss due to ion-neutral friction and parallel momentum input from neutral beam injection. $R_{ie\parallel}$ is the ion-electron friction.

The parallel momentum balance for electrons has the standard form:

$$j_\parallel=\sigma_\parallel\left(\frac{b_x}{e}\frac{1}{h_x}\left(\frac{\partial nT_e}{n\partial x}+0.71\frac{\partial T_e}{\partial x}\right)-\frac{b_x}{h_x}\frac{\partial\Phi}{\partial x}\right). \tag{B.12}$$

Expressions for $j_\perp$ and j_y are obtained from radial and poloidal projections of the total momentum balance equation (the sum of electron and ion momentum balance equations). The resulting current is a sum of contributions from pressure gradient (diamagnetic current) $\vec{j}^{(dia)}$, inertia and gyroviscosity $\vec{j}^{(in)}$, viscosity $\vec{j}^{(vis)}$ and ion-neutral friction

$\vec{j}^{(s)}$. Similar to the particle continuity equation, the effective current $\tilde{\vec{j}}$ is introduced, which gives the same contribution to $\nabla\vec{j}$, as the ordinary current. This gives the current continuity equation

$$\frac{1}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x}\tilde{j}_x\right)+\frac{1}{\sqrt{g}}\frac{\partial}{\partial y}\left(\frac{\sqrt{g}}{h_y}\tilde{j}_y\right)=0\,. \tag{B.13}$$

$$\tilde{j}_x = b_z\tilde{j}_\perp + b_x\tilde{j}_\parallel\,. \tag{B.14}$$

$$\vec{j} = \tilde{\vec{j}}^{(dia)} + \vec{j}^{(in)} + \vec{j}^{(vis)} + \vec{j}^{(s)} + \vec{j}_\parallel\,. \tag{B.15}$$

The effective diamagnetic current can be chosen in a form

$$\tilde{j}_\perp^{(dia)} = \frac{1}{b_z}\frac{n\left(T_e+T_i\right)B_z}{h_y}\frac{\partial}{\partial y}\left(\frac{1}{B^2}\right)\,, \tag{B.16}$$

$$\tilde{j}_y^{(dia)} = -\frac{n\left(T_e+T_i\right)B_z}{h_x}\frac{\partial}{\partial x}\left(\frac{1}{B^2}\right)\,. \tag{B.17}$$

Current driven by parallel viscosity should be taken into account in Eqs. (B.13), (B.14) and (B.15). This current is a small correction to the diamagnetic current Eqs. (B.16) and (B.17). However, on closed flux surfaces, where the pressure is almost constant, the diamagnetic current averaged over the flux surfaces tends to zero. The resulting remaining part can become comparable to the parallel viscosity-driven current, which is necessary to satisfy the condition that the averaged net radial current should be zero. The terms, which contain the plasma viscosity, produce almost divergence-free currents. Similar to the case of diamagnetic currents, it can be reduced to

$$\tilde{j}_\perp^{(vis\parallel)} = -\frac{B_x\eta_0}{3\sqrt{B}}\frac{\partial\left(\sqrt{B}\left(V_\parallel+b_x\left(V_\perp^{(dia)}+V_\perp^{(E)}\right)\right)\right)}{h_x\partial x}\frac{\partial}{h_y\partial y}\left(\frac{1}{B^2}\right) \tag{B.18}$$

$$\tilde{j}_y^{(vis\parallel)} = b_z\frac{B_x\eta_0}{3\sqrt{B}}\frac{\partial\left(\sqrt{B}\left(V_\parallel+b_x\left(V_\perp^{(dia)}+V_\perp^{(E)}\right)\right)\right)}{h_x\partial x}\frac{\partial}{h_x\partial x}\left(\frac{1}{B^2}\right) \tag{B.19}$$

The current produced by components of viscosity tensor connected with heat fluxes is:

$$\tilde{j}_\perp^{(visq)} = -\frac{0.24B_x}{\sqrt{B}\nu_{ii}}\frac{\partial\left(q_{i\parallel}^{(0)}\sqrt{B}\right)}{h_x\partial x}\frac{\partial}{h_y\partial y}\left(\frac{1}{B^2}\right) \tag{B.20}$$

$$\tilde{j}_y^{(visq)} = b_z\frac{0.24B_x}{\sqrt{B}\nu_{ii}}\frac{\partial\left(q_{i\parallel}^{(0)}\sqrt{B}\right)}{h_x\partial x}\frac{\partial}{h_x\partial x}\left(\frac{1}{B^2}\right) \tag{B.21}$$

The second part of the current contains contributions from both inertia and gyroviscosity forces. In contrast to the parallel momentum balance, the cancellation is not complete (as has been noticed in [20]). Expressions for this part of the current are

$$\begin{aligned} j_\perp^{(in)} = -\frac{m_i}{B}\Bigg[&\frac{\partial nV_y}{\partial t}+\frac{1}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x}n\left(V_\perp^{(0)}b_z+V_\parallel b_x\right)V_y\right)+\frac{1}{\sqrt{g}}\frac{\partial}{\partial y}\left(\frac{\sqrt{g}}{h_y}nV_y^{(0)}V_y\right)\Bigg] \\ &-\frac{2b_x}{B}\frac{1}{h_x}\frac{\partial}{\partial x}\left(\eta_3\frac{\partial V_\parallel}{h_x\partial x}\right)+\frac{m_i}{B}nV_\parallel^2\frac{\partial\ln h_z}{h_y\partial y} \end{aligned} \tag{B.22}$$

$$j_y^{(in)} = \frac{m_i}{B}\left[\frac{\partial n V_\perp}{\partial t} + \frac{1}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x} n \left(V_\perp^{(0)} b_z + V_\parallel b_x\right) V_\perp\right) + \frac{1}{\sqrt{g}}\frac{\partial}{\partial y}\left(\frac{\sqrt{g}}{h_y} n V_y^{(0)} V_\perp\right)\right] - \frac{2b_x}{B}\frac{1}{h_x}\frac{\partial}{\partial x}\left(\eta_3 \frac{\partial V_\parallel}{h_y \partial y}\right) - \frac{m_i}{B} n V_\parallel^2 \frac{\partial \ln h_z}{h_x \partial x} \tag{B.23}$$

where $\eta_3 = \frac{nT_i}{2\omega_{ci}}, \eta_1 = \frac{\eta_2}{4}$.

The last terms proportional to the gradient of B correspond to the vertical guiding centre drift of ions caused by the centrifugal force. This current can be simply added to the diamagnetic current. Other terms seem to be less important.

The contribution from perpendicular viscosity is

$$j_\perp^{(vis\perp)} = \frac{1}{B}\left[\frac{1}{h_x}\frac{\partial}{\partial x}\left(\eta_1 \frac{\partial V_y}{h_x \partial x}\right) + \frac{1}{h_y}\frac{\partial}{\partial y}\left(\eta_1 \frac{\partial V_y}{h_y \partial y}\right) + \frac{1}{h_x h_y}\frac{\partial \eta_1}{\partial x}\frac{\partial V_\perp}{\partial y} - \frac{1}{h_x h_y}\frac{\partial \eta_1}{\partial y}\frac{\partial V_\perp}{\partial x}\right] + \frac{4b_x}{B}\frac{1}{h_x h_y}\left(\frac{\partial \eta_1}{\partial x}\frac{\partial V_\parallel}{\partial y} - \frac{\partial \eta_1}{\partial y}\frac{\partial V_\parallel}{\partial x}\right) \tag{B.24}$$

$$j_y^{(vis\perp)} = -\frac{1}{B}\left[\frac{1}{h_y}\frac{\partial}{\partial y}\left(\eta_1 \frac{\partial V_\perp}{h_y \partial y}\right) + \frac{1}{h_x}\frac{\partial}{\partial x}\left(\eta_1 \frac{\partial V_\perp}{h_x \partial x}\right) - \frac{1}{h_x h_y}\frac{\partial \eta_1}{\partial x}\frac{\partial V_y}{\partial y} + \frac{1}{h_x h_y}\frac{\partial \eta_1}{\partial y}\frac{\partial V_y}{\partial x}\right]. \tag{B.25}$$

Here, the largest term is the first one in the r.h.s. of the expression for the radial current, since the radial derivatives are the largest. This current corresponds to a pure cylindrical approximation. It is responsible, for example, for closing currents near flush-mounted probes [23]. It was taken into account in [19, 24]. However, in the problem of global current structure in the edge plasma, the currents driven by parallel viscosity seem to be more important.

The total current caused by viscosity is

$$\vec{j}^{(vis)} = \vec{j}^{(vis\parallel)} + \vec{j}^{(vis\perp)} + \vec{j}^{(visq)} \ . \tag{B.26}$$

The contribution from ion neutral friction is

$$j_\perp^{(s)} = \frac{S_{iy}^m}{B} \ . \tag{B.27}$$

$$j_y^{(s)} = -\frac{S_{i\perp}^m}{B} \ , \tag{B.28}$$

where S_{iy}^m and $S_{i\perp}^m$ are the momentum sinks due to ion-neutral friction.

$$S_{i\perp}^m = -n\mu_{iN}\nu_{iN}\left(V_\perp - V_{\perp N}\right) \ , \tag{B.29}$$

$$S_{iy}^m = -n\mu_{iN}\nu_{iN}\left(V_y - V_{yN}\right) \ , \tag{B.30}$$

where $\mu_{iN} = m_i/2$ is the effective mass, $\nu_{iN} = n_N \langle V_{iN}\sigma_{cx}\rangle$ is the ion-neutral collision frequency, n_N is the neutral density and $\vec{V_{iN}}$ is the neutral mean velocity. For the ion velocity used above, only electric and diamagnetic drifts are left, neglecting smaller terms, to obtain

$$\tilde{j}_x^{(s)} = \tilde{j}_\perp^{(s)} b_z = -\sigma_{iN} b_z^2 \frac{\partial \Phi}{h_x \partial x} - \sigma_{iN} b_z^2 \frac{1}{en}\frac{\partial n T_i}{h_x \partial x} + \sigma_{iN} B_z V_{yN} \ , \tag{B.31}$$

$$\tilde{j}_y^{(s)} = -\sigma_{iN}\frac{\partial \Phi}{h_y \partial y} - \sigma_{iN}\frac{1}{en}\frac{\partial n T_i}{h_y \partial y} - \sigma_{iN} B V_{\perp N} \ , \tag{B.32}$$

where σ_{iN} is the ion-neutral perpendicular conductivity

$$\sigma_{iN} = \frac{nm_i \langle V_{iN}\sigma_{cx}\rangle n_N}{2B^2} \ . \tag{B.33}$$

The rate of charge-exchange $\langle V_{iN}\sigma_{cx}\rangle$ can be expressed through the mobility K_0 measured in the absence of a magnetic field for room temperature and atmospheric pressure for deuterium [25]:

$$K_0 = \frac{2e}{m_i \langle V_{iN}\sigma_{cx}\rangle n_N^{(0)}} \ , \tag{B.34}$$

with $n_N^{(0)} = 2.687 \cdot 10^{25} m^{-3}$ and $K_0 = 11.2 \cdot 10^{-4} \frac{m^2}{sV}$. Assuming that the charge-exchange cross-section is approximately constant one gets

$$\langle V_{iN}\sigma_{cx}\rangle = 3.2 \cdot 10^{-15} \sqrt{\frac{T_i}{0.026}} \frac{m^3}{s} \ , \tag{B.35}$$

where T_i is measured in eV. A similar expression, which gives approximately the same result, is [26]:

$$\langle V_{iN}\sigma_{cx}\rangle = \frac{32\sigma_{cx}}{3\sqrt{\pi}} \sqrt{\frac{T_i}{m_i}} \frac{m^3}{s} \ . \tag{B.36}$$

.

In the energy balance equations appears a special problem, which is the possible inclusion of an anomalous radial current. The basic guidelines are as follows:

- The expression must be as simple as possible, and the most convenient format is an Ohm-like expression;
- The system of equations must be informed about this non-classical contribution, but
- The classical core of the equations must not be disturbed.

Apparently, the radial current would enter the electron energy equation via the Joule term, but this manipulation is inappropriate since one cannot be sure that this term directly heats the electrons, which is the case in a purely classical setup. The proper thing to do is to keep only the non-radial classical contribution in the Joule term, and construct additional heat exchange terms containing the radial current. In this way, the system decides how to thermodynamically allocate the "alien" heat term, among electrons and ion species.
For this reason, one cannot simply use the standard Ohm's law in the radial direction with anomalous coefficients, or any other prescribed expression whatsoever, as a constructive procedure is necessary for simultaneously producing the radial current as well as the heat exchange terms. At this point it has to be mentioned that this peculiarity arises only in the multi-fluid case, whereas in the single species case these heat exchange terms can indeed be constructed even using a prescribed expression for the radial current.
The trick for carrying out this program is to correct the electron friction density $\vec{R}^e$ by adding the term $n_e e\tilde{\eta}_\perp \vec{J}_\perp$ similar to the classical one, only using the *ad hoc* parameter $\tilde{\eta}_\perp$.
Since $\vec{R}^e = -\sum_a \vec{R}^a$, one has a modified total ion momentum equation at hand which in conjuction with the pressure equilibrium (not affected by the additional term) $\vec{J} \times \vec{B} = \nabla P$, gives the diffusive part of the radial current, namely

$$J_y^{diff} = \frac{\tilde{\eta}_\perp e n_e}{B^2} \frac{\partial P_i}{\partial y} \tag{B.37}$$

where P_i is the total ion pressure. This dependency is also found in [17]. Moreover, an additional dependence on the radial electric field can potentially be added without interfering with the procedure.
The final step, namely the construction of the heat exchange terms, is performed by claiming that the well known relation [27] (adapted for the multi-fluid case)

$$Q_e = -\sum_a \left(Q_a + (\vec{u}^a - \vec{u}^e) \cdot \vec{R}^a \right) \ , \tag{B.38}$$

which stems from the fundamental null space of the collision operator, is still true while carrying the modified friction. Claiming this, after performing the proper manipulations in the energy equations, one reveals the *ad hoc* contribution as the exchange term that one wants, which reads $\sum_a (\vec{u}^a - \vec{u}^e)_\perp n_a \frac{Z_a^2}{Z_{eff}} e\tilde{\eta}_\perp \vec{J}_\perp$ added in the total ion energy equation and subtracted from the electron equation. Note that in the single species case, this term boils down to $\tilde{\eta}_\perp J_\perp^2$ removing energy from the electrons, depositing it to the ions, a feature predicted by neoclassical theory [28].

In the final form of the energy balance equations, the convective heat flux and the diamagnetic heat flux are combined as in [17, 18, 19, 20], see also [23].

The energy balance equation for the electrons is

$$\begin{aligned}
&\frac{3}{2}\frac{\partial nT_e}{\partial t} + \frac{1}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x}\tilde{q}_{ex}\right) + \frac{1}{\sqrt{g}}\frac{\partial}{\partial y}\left(\frac{\sqrt{g}}{h_y}\tilde{q}_{ey}\right) + \frac{nT_e}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x}\left(V_\parallel - j_\parallel/en\right)b_x\right) \\
&\qquad = Q_e + nT_e B\frac{1}{h_x h_y}\left(\frac{\partial\Phi}{\partial y}\frac{\partial}{\partial x}\left(\frac{1}{B^2}\right) - \frac{\partial\Phi}{\partial x}\frac{\partial}{\partial y}\left(\frac{1}{B^2}\right)\right) ,
\end{aligned} \tag{B.39}$$

where

$$\begin{aligned}
\tilde{q}_{ex} &= \frac{3}{2}nT_e\left(-\frac{b_z}{Bh_y}\frac{\partial\Phi}{\partial y} + b_x V_\parallel - b_x j_\parallel/en\right) \\
&+ \frac{5}{2}nT_e b_z\left(-\frac{D}{T_e+T_i}\frac{b_z}{h_x}\left(\frac{\partial p}{n\partial x} - \frac{3}{2}\frac{\partial T_e}{\partial x}\right) - D_n^{(AN)}\frac{1}{h_x n}\frac{\partial n}{\partial x} - D_p^{(AN)}\frac{1}{h_x n}\frac{\partial p_i}{\partial x} + V_{e\perp}^{(AN)}\right) \\
&- 0.71 b_x j_\parallel T_e/e - \kappa_{e\parallel}\frac{b_x^2}{h_x}\frac{\partial T_e}{\partial x} - \kappa_{e\perp}b_z^2\frac{1}{h_x}\frac{\partial T_e}{\partial x} \\
&+ \frac{3}{2}\frac{T_e}{eB}\frac{\nu_{ei}}{\omega_{ce}}\frac{b_z^2}{h_x}\frac{\partial n\left(T_e+T_i\right)}{\partial x} - \frac{5}{2}nT_e^2\frac{B_z}{e}\frac{\partial}{h_y\partial y}\left(\frac{1}{B^2}\right) ,
\end{aligned} \tag{B.40}$$

$$\begin{aligned}
\tilde{q}_{ey} &= \frac{3}{2}nT_e\left(\frac{B_z}{B^2}\frac{1}{h_x}\frac{\partial\Phi}{\partial x}\right) \\
&+ \frac{5}{2}nT_e\left(-\frac{D}{T_e+T_i}\frac{1}{h_y}\left(\frac{\partial p}{n\partial y} - \frac{3}{2}\frac{\partial T_e}{\partial y}\right) - D_n^{(AN)}\frac{1}{h_y n}\frac{\partial n}{\partial y} - D_p^{(AN)}\frac{1}{h_y n}\frac{\partial p_i}{\partial y} + V_{ey}^{(AN)}\right) \\
&- \kappa_{e\perp}\frac{1}{h_y}\frac{\partial T_e}{\partial y} + \frac{3}{2}\frac{T_e}{eB}\frac{\nu_{ei}}{\omega_{ce}}\frac{1}{h_y}\frac{\partial n\left(T_e+T_i\right)}{\partial y} + \frac{5}{2}nT_e^2\frac{B_z}{e}\frac{\partial}{h_x\partial x}\left(\frac{1}{B^2}\right) ,
\end{aligned} \tag{B.41}$$

$$Q_\Delta = \frac{3m_e}{m_i}n\nu_{ei}\left(T_e - T_i\right) , \tag{B.42}$$

$$Q_u = R_{ie\parallel}j_\parallel/en , \tag{B.43}$$

$$Q_{AN} = \sum_a \left(\vec{u}^a - \vec{u}^e\right)_\perp n_a \frac{Z_a^2}{Z_{eff}} e\tilde{\eta}_\perp \vec{J}_\perp , \tag{B.44}$$

$$Q_e = -Q_\Delta - Q_u - Q_{AN} . \tag{B.45}$$

Here $\kappa_{e\parallel}$ is the Balescu (not Braginskii as stated in the CPP review paper [29] or in `${SOLPSTOP}/doc/CPP_2006.pdf`) heat conductivity with a possible heat flux limit [30]. The leading numerical coefficient is then $\frac{5}{2}\frac{2.16z}{1+0.27z}$ as opposed to 3.16 for the Braginskii formula [31]. The perpendicular heat conductivity $\kappa_{e\perp}$ can be either

classical or anomalous. In the last term of the energy balance equation, it is taken into account that the $\mathbf{E}\times\mathbf{B}$ drift velocity is almost divergence-free.

The ion energy balance equation is

$$\begin{aligned}&\frac{3}{2}\frac{\partial nT_i}{\partial t}+\frac{1}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x}\tilde{q}_{ix}\right)+\frac{1}{\sqrt{g}}\frac{\partial}{\partial y}\left(\frac{\sqrt{g}}{h_y}\tilde{q}_{iy}\right)+\frac{nT_i}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x}V_{\parallel}b_x\right)\\ &=Q_{\Delta}+Q_{AN}+\frac{\eta_0}{3}\left(2b_x\frac{\partial V_{\parallel}}{h_x\partial x}\right)^2+nT_iB\frac{1}{h_xh_y}\left(\frac{\partial\Phi}{\partial y}\frac{\partial}{\partial x}\left(\frac{1}{B^2}\right)-\frac{\partial\Phi}{\partial x}\frac{\partial}{\partial y}\left(\frac{1}{B^2}\right)\right),\end{aligned}\tag{B.46}$$

where

$$\begin{aligned}\tilde{q}_{ix}&=\frac{3}{2}nT_i\left(-\frac{b_z}{Bh_y}\frac{\partial\Phi}{\partial y}+b_xV_{\parallel}\right)\\ &+\frac{5}{2}nT_ib_z\left(-\frac{D}{T_e+T_i}\frac{b_z}{h_x}\left(\frac{\partial p}{n\partial x}-\frac{3}{2}\frac{\partial T_e}{\partial x}\right)-D_n^{(AN)}\frac{1}{h_xn}\frac{\partial n}{\partial x}-D_p^{(AN)}\frac{1}{h_xn}\frac{\partial p_i}{\partial x}+V_{i\perp}^{(AN)}\right)\\ &+\frac{5}{2}nT_ib_z\tilde{V}_{\perp}^{(dia)}-\kappa_{i\parallel}\frac{b_x^2}{h_x}\frac{\partial T_i}{\partial x}-\kappa_{i\perp}b_z^2\frac{1}{h_x}\frac{\partial T_i}{\partial x},\end{aligned}\tag{B.47}$$

$$\begin{aligned}\tilde{q}_{iy}&=\frac{3}{2}nT_i\left(\frac{B_z}{B^2}\frac{1}{h_x}\frac{\partial\Phi}{\partial x}\right)\\ &+\frac{5}{2}nT_i\left(-\frac{D}{T_e+T_i}\frac{1}{h_y}\left(\frac{\partial p}{n\partial y}-\frac{3}{2}\frac{\partial T_e}{\partial y}\right)-D_n^{(AN)}\frac{1}{h_yn}\frac{\partial n}{\partial y}-D_p^{(AN)}\frac{1}{h_yn}\frac{\partial p_i}{\partial y}+V_{iy}^{(AN)}\right)\\ &-\kappa_{e\perp}\frac{1}{h_y}\frac{\partial T_i}{\partial y}+\frac{5}{2}nT_i\tilde{V}_y^{(dia)}.\end{aligned}\tag{B.48}$$

As for the electrons, the parallel heat conductivity is classical with a possible heat flux limit, while the perpendicular contribution can be either classical or anomalous.

There is discussion in the literature on how to take into account anomalous terms in the energy balance equations. One point of view (see e.g. [32]) is to cancel the contribution from anomalous particle fluxes in the terms $p_i\nabla\cdot\vec{V}_i$ (or $\vec{V}_i\cdot\nabla p_i$ if equations are written in a different form). In contrast, according to the derivation [20], where a kinetic equation with fluctuating term has been integrated, the anomalous fluxes should be kept in all terms of the energy balance equations. This variant was used in `UEDGE` [17, 18, 19]. In `B2.5` only the contribution from anomalous radial particle flux to the convective heat flux are left in the expressions for $\tilde{q}_{e,i}$. In the term $p_i\nabla\cdot\vec{V}_i$ the anomalous particle flux is absent.

B.1 Derivation of parallel viscosity term in momentum equation

To evaluate the parallel viscosity contribution we should start from the general expressions, which can be transformed and simplified in arbitrary cartesian coordinate system assuming only parallel velocity. The Braginskii description for parallel viscosity in cartesian coordinates is:

$$\pi_{\alpha\beta}=\frac{3}{2}\eta_0\left(b_\alpha b_\beta-\frac{1}{3}\delta_{\alpha\beta}\right)\left(b_\mu b_\nu-\frac{1}{3}\delta_{\mu\nu}\right)W_{\mu\nu}$$

where

$$W_{\mu\nu}=\frac{\partial V_\mu}{\partial x_\nu}+\frac{\partial V_\nu}{\partial x_\mu}-\frac{2}{3}\delta_{\mu\nu}\frac{\partial V_i}{\partial x_i},\ \vec{b}=\vec{B}/B.$$

The parallel momentum balance component is:

$$\vec{b}\cdot(\nabla\cdot\overleftrightarrow{\pi}) = b_\alpha \frac{\partial \pi_{\alpha\beta}}{\partial x_\beta} = b_\alpha \frac{\partial}{\partial x_\beta}\left[\frac{3}{2}\eta_0\left(b_\alpha b_\beta - \frac{1}{3}\delta_{\alpha\beta}\right)\left(b_\mu b_\nu - \frac{1}{3}\delta_{\mu\nu}\right)W_{\mu\nu}\right]$$

We can evaluate the scalar value separately:

$$\begin{aligned}
\pi_0 &= \frac{3}{2}\eta_0 W_{\parallel\parallel} = \frac{3}{2}\eta_0\left(b_\mu b_\nu - \frac{1}{3}\delta_{\mu\nu}\right)W_{\mu\nu} \\
&= \frac{3}{2}\eta_0\left(b_\mu b_\nu - \frac{1}{3}\delta_{\mu\nu}\right)\left(\frac{\partial V_\mu}{\partial x_\nu} + \frac{\partial V_\nu}{\partial x_\mu} - \frac{2}{3}\delta_{\mu\nu}\frac{\partial V_i}{\partial x_i}\right) = 3\eta_0\left(b_\mu b_\nu - \frac{1}{3}\delta_{\mu\nu}\right)\left(\frac{\partial V_\mu}{\partial x_\nu} - \frac{1}{3}\delta_{\mu\nu}\frac{\partial V_i}{\partial x_i}\right) \\
&= 3\eta_0\left(b_\mu b_\nu \frac{\partial V_\mu}{\partial x_\nu} - \frac{1}{3}\frac{\partial V_i}{\partial x_i}\right) = 3\eta_0\left(b_\mu b_\nu \frac{\partial V_\parallel b_\mu}{\partial x_\nu} - \frac{1}{3}\frac{\partial V_\parallel b_\nu}{\partial x_\nu}\right) \\
&= 3\eta_0\left(\frac{B_\nu}{B}\frac{\partial V_\parallel}{\partial x_\nu} - \frac{1}{3}B_\nu\frac{\partial V_\parallel/B}{\partial x_\nu}\right) = \eta_0\frac{B_\nu}{B}\left(2\frac{\partial V_\parallel}{\partial x_\nu} + \frac{V_\parallel}{B}\frac{\partial B}{\partial x_\nu}\right) \\
&= 2\eta_0\frac{B_\nu}{B}\frac{1}{\sqrt{B}}\frac{\partial V_\parallel\sqrt{B}}{\partial x_\nu} = 2\eta_0\frac{1}{\sqrt{B}}\vec{b}\cdot\nabla\left(V_\parallel\sqrt{B}\right)
\end{aligned}$$

The projection of the viscosity divergence on the parallel direction is then:

$$\begin{aligned}
\vec{b}\cdot(\nabla\cdot\overleftrightarrow{\pi}) &= b_\alpha\frac{\partial}{\partial x_\beta}\left[\pi_0\left(b_\alpha b_\beta - \frac{1}{3}\delta_{\alpha\beta}\right)\right] = \frac{\partial \pi_0 b_\alpha}{\partial x_\alpha} - \frac{1}{3}b_\alpha\frac{\partial \pi_0}{\partial x_\alpha} = B_\alpha\frac{\partial \pi_0/B}{\partial x_\alpha} - \frac{1}{3}b_\alpha\frac{\partial \pi_0}{\partial x_\alpha} \\
&= \frac{2}{3}b_\alpha\frac{\partial \pi_0}{\partial x_\alpha} - \pi_0\frac{b_\alpha}{B}\frac{\partial B}{\partial x_\alpha} = \frac{2}{3}B^{3/2}b_\alpha\frac{\partial \pi_0 B^{-3/2}}{\partial x_\alpha} = \frac{2}{3}B^{3/2}\vec{b}\cdot\nabla\left(\pi_0 B^{-3/2}\right)
\end{aligned}$$

Then the parallel viscosity in the parallel momentum balance is:

$$\vec{b}\cdot(\nabla\cdot\overleftrightarrow{\pi}) = \frac{4}{3}B^{3/2}\vec{b}\cdot\nabla\left(\frac{\eta_0}{B^2}\vec{b}\cdot\nabla\left(V_\parallel\sqrt{B}\right)\right),\ \text{where } V_\parallel = \vec{b}\cdot\vec{V}$$

For **SOLPS-ITER**, the orthogonal coordinates (x, r, z) are more convenient for further evaluations than the non-orthogonal coordinates (x, y, z). In these coordinates, it is readily seen that $\vec{b}\cdot\nabla f = b_x\frac{\partial f}{h_x\partial x}$ for any scalar function f. The viscosity can be finally written as:

$$\vec{b}\cdot(\nabla\cdot\overleftrightarrow{\pi}) = \frac{4}{3}b_x B^{\frac{3}{2}}\frac{\partial}{h_x\partial x}\left(\frac{\eta_0 b_x}{B}\frac{\partial\left(B^{\frac{1}{2}}V_\parallel\right)}{h_x\partial x}\right)$$

B.2 Boundary conditions with drifts and currents

Whether one deals with the diamagnetic or $\mathbf{E}\times\mathbf{B}$ perpendicular drift velocity, special care must be taken to avoid unphysical flows at the plates. The standard boundary condition is to require that the flow be at least sonic at the sheath entrance [33], which translates into:

$$V_\parallel \geq C_s - \frac{b_z}{b_x}V_\perp\ , \tag{B.49}$$

where C_s is the local sound speed and $\frac{b_z}{b_x}$ is the inverse of the field line pitch. However, for very small pitch angles, as in stellarators, the $V_\perp$ term may overpower the sound velocity [34] and then the boundary condition would require a plasma flow exiting the plate. Therefore, the pitch angle at the plates is limited to be no less than 1 degree (user parameter, see switch `b2agfs_min_pitch` in Appendix A.4), motivated by the engineering limits met when attempting to align the divertor tiles with the magnetic field. The solution procedure of the code was also modified so as ensure that no unphysical flows were being created, and solves the parallel momentum equation using the updated electric potential.

Boundary conditions for the current continuity equation at the plates correspond to sheath current-voltage characteristics

$$j_x = en\left(b_x c_s - b_x \frac{1}{\sqrt{2\pi}}\sqrt{\frac{T_e}{m_e}}\exp\left(-\frac{e\Phi}{T_e}\right)(1-\gamma_e)\right), \tag{B.50}$$

where γ_e is the secondary electron emission coefficient. At the inner (core) flux surface the currents are set either to the divergent part of the diamagnetic current or to zero. Both variants give the same result and both guarantee that the net current integrated over the inner flux surface should be zero. At the walls, the same conditions on the normal current components were imposed as for the inner core.

The electron and ion heat fluxes to the target plates are

$$\tilde{q}_{ex} = b_x \frac{n}{\sqrt{2\pi}}\sqrt{\frac{T_e}{m_e}}\exp\left(-\frac{e\Phi}{T_e}\right)(1-\gamma_e)\left(T_e\frac{1+\gamma_e}{1-\gamma_e} + e\Phi\right), \tag{B.51}$$

$$\tilde{q}_{ix} = \frac{3}{2} n T_i c_s b_x. \tag{B.52}$$

The question of kinetic boundary conditions including drifts is not yet finally clarified and will need further work from both kinetic and fluid modelling [35, 33].

B.3 Problems for a fluid description of the edge plasma

To apply a fluid model for the description of the edge, a specific ordering of lengths is necessary.

The necessary condition for a fluid model is that all characteristic lengths (characteristic length of the sheath, which is a microscopic length scale of about 10 Debye lengths; electron and ion mean free path ranging from sub-cm to metres; neutral mean free path, which is centimetres to metres; heat conduction length which is typically five times the mean free path length due to the dominant contribution from tail electrons or ions) are smaller than the parallel connection length (order of several tens of metres). This is usually fulfilled in edge plasmas with the following possible exceptions:

1. at low temperatures, e.g. near the target in the divertor, steeper gradients result in non-local heat flux contributions. In general, a non-local description might be necessary (weighted mean of temperature profile over five mean free lengths of electrons or ions). However, a simpler correction is sometimes satisfactory and is used in the code (see below).
2. at high temperatures (midplane, near separatrix). Corrections for flux limit and for periodic systems on closed field lines are necessary due to the fact that the heat conduction mean free path is larger than the connection length.
3. the sheath cannot be resolved by a fluid model and is represented by boundary conditions at the sheath entrance deduced from kinetic models.

Simple heat flux and momentum flux (viscosity) limits are available. The heat flux limit is formulated as

$$\kappa_\parallel = \frac{\kappa_{SH}}{1 + |q_{SH}/q_{fl}|}, \tag{B.53}$$

with the classical Spitzer–Härm heat flux

$$q_{SH} = -\kappa_{\parallel,SH} \cdot \frac{\partial T_e}{\partial x} \tag{B.54}$$

and the flux limit

$$q_{fl} = \alpha n_e T_e^{3/2} \sqrt{m_e} \tag{B.55}$$

($\alpha \approx 0.2$ from comparison with kinetic calculations for electrons; the situation for ions is less clear).

The momentum flux limit is done the same way

$$\eta_{\parallel} = \frac{\eta_{cl,a}}{1 + |q_{cl,a}/q_{fl,a}|}, \tag{B.56}$$

with the classical momentum flux

$$q_{cl,a} = -\eta_{\parallel,a} \cdot \frac{\partial u_{\parallel,a}}{\partial x} \tag{B.57}$$

and the flux limit

$$q_{fl,a} = \alpha p_a \tag{B.58}$$

(with $\alpha = 0.5$). Unfortunately, for this correction even less kinetic work exists than for the heat flux limits.

Additionally, long mean free path corrections of classical heat conduction and viscosity (in the core) are applied. For periodic systems and small perturbations one gets [36]:

$$q_{\parallel,eff} = q_{SH} / \left(h_{\parallel}^2 \lambda_{HC}^2 + 1 \right), \tag{B.59}$$

with wavenumber of periodicity

$$h_{\parallel} = 2\pi / \lambda_{\parallel}, \tag{B.60}$$

$$\lambda_{\parallel} = 2\pi R q \tag{B.61}$$

and mean free path of tail electrons [37]

$$\lambda_{HC}(\mathrm{m}) = 7.5 \cdot 10^{16} \cdot T_e^2(\mathrm{eV}) / n_e(\mathrm{m}^{-3}). \tag{B.62}$$

Fokker-Planck computer simulations of electron heat transport in laser plasmas [38] show that a better description is obtained for a linear correction:

$$q_{\parallel,eff} = q_{SH} / (h_{\parallel} \lambda_{HC} + 1). \tag{B.63}$$

Applying these limits for the core, convergence behaviour of the code improves significantly.
A more correct kinetic limit would imply a non-local heat transport (and viscosity,...) description. The parallel heat flux at a point should be determined by classical fluxes at points up to typically five mean free paths away. As an additional feature, one would get pre-heating effects by fast electrons, maybe important for the ELMs. Work is necessary to generalize this to the SOL [39, 40] and this has to be cross-checked with kinetic calculations [41]. Also, numerical problems due to this non–local description have to be solved [42].

In general, if these kinetic corrections become important, quantitative conclusions are questionable, because the fluid model is no longer strictly valid. Considerations of particle, energy and momentum conservation still guarantee a

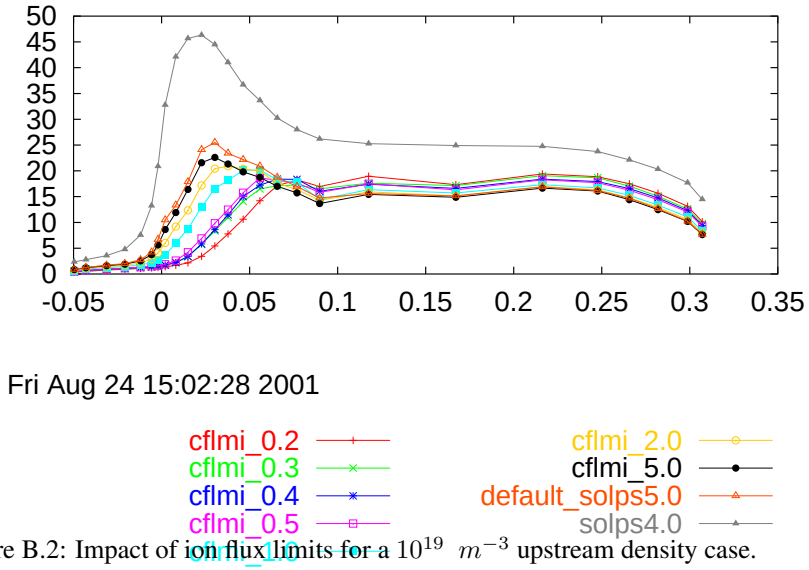


Figure B.2: Impact of ion flux limits for a 10^{19} m^{-3} upstream density case.

qualitatively correct description. The flux limiters are designed are defined to give the correct answer in the fully collisional and in the fully collisionless regime.

Parametric studies varying the various flux limiters should be done, particularly for predictive work. Specific experimental scenarios would be required to calibrate the flux limiters. However, complete experimental data sets including electron and ion temperatures and densities at the target plate and upstream are necessary. Fortunately, at higher densities these corrections are not important. Furthermore, one should note, that mapping of target data to the midplane and the extraction of anomalous radial transport coefficients from such a procedure is questionable in the regime where flux limiters are important [43].

As an example, SOLPS5.0 calculations for JET using deuterium, carbon and helium are shown in Figs. B.2 and B.3. The predicted electron temperatures at the outer target are shown for two different upstream densities. In each graph, the ion energy flux limit is varied and the sensitivity of the target profiles to this variation is obvious.

B.4 Neutral models

B.4.1 Fluid neutrals

As an alternative to the complete kinetic Monte Carlo description offered by `Eirene`, a simpler and faster neutral fluid model is also available as `B2.5` standalone. The surface sources of neutrals are then given by user-defined recycling models, possibly enhanced with local sputtering models when impurities are considered (see e.g. [44]). The volume sources of neutrals are given by plasma recombination rates computed by the `B2.5` atomic physics packages. The neutral species is then treated just as the other plasma species, with its own particle and momentum transport equations in the parallel direction, equivalent to Eqs. (B.5) and (B.10). The neutrals are assumed to have the same common temperature as all other ion species and the neutral energy balance equation becomes simply a component of Eq. (B.46). Of course, terms due to charged particle motions or currents do not appear for neutral species.

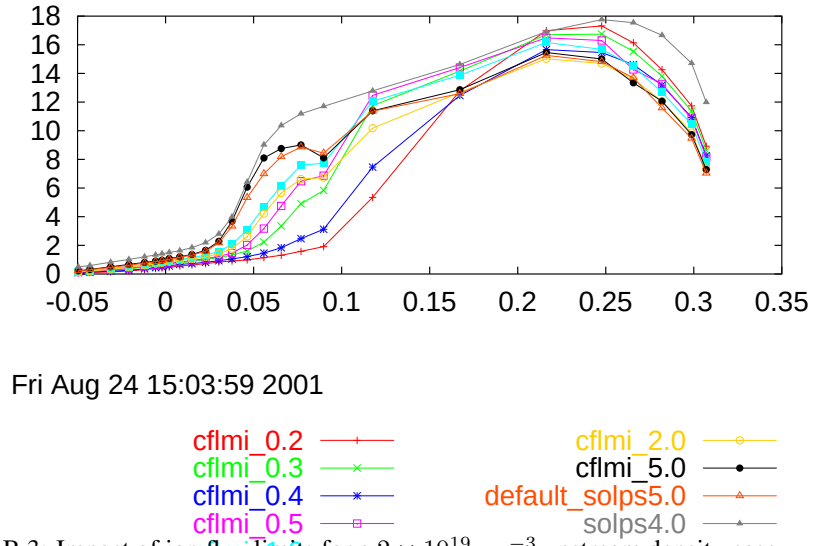


Figure B.3: Impact of ion flux limits for a 2×10^{19} m^{-3} upstream density case.

The neutral species transport coefficients are usually given by the following equations:

$$D_{n,0}^{(AN)} = 0, \tag{B.64}$$

$$D_{p,0}^{(AN)} = C + \sqrt{\frac{T_i}{m_N}} \frac{1}{(\sigma_{CX} n_i + \sigma_{in} n_e)}, \tag{B.65}$$

where m_N is the mass of the neutral species, and C, σ_{CX} and σ_{in} are three constants given by the user, representing hydrogen gas diffusion, and the charge-exchange and ionisation cross-sections, respectively. As an option, one may choose to use cross-sections from atomic databases instead of constants, with their appropriate parameter dependencies. The diffusion coefficients can also be bounded to avoid unphysically large or small values.

Additionally, the heat and particle fluxes carried by neutrals can be flux-limited, following the formalism laid out in [45]. On the one hand, we flux-limit the neutral contribution to the ion heat diffusivity, involved in the conductive heat flux computation. On the other hand, we also flux-limit the convective particle and energy fluxes of neutrals. Because these limits apply for neutral species, both the parallel and perpendicular directions are to be considered. The basic formula used is:

$$X \Rightarrow \frac{X}{\left[1 + \left|\frac{X}{\alpha F}\right|^{\gamma}\right]^{(1/\gamma)}} \tag{B.66}$$

where X is the quantity to limit and F is the limiting value. Typically, the γ coefficient is set to $\gamma = 1$ or 2; if $\gamma = 1$ is used, then the absolute value of the flux ratio is taken. The value of α is of order unity; the precise value is unimportant, but the goal is to limit the flux to a value of approximately $n\bar{v}/4$, where $\bar{v} = \sqrt{(8T/\pi m)} = 2v_T/\sqrt{\pi}$. One could also use v_T rather than $\bar{v}$, but the difference is not very significant; in fact, flux limits of $\sim n\bar{v}/2$ are often chosen, since too severe flux limits sometimes make the solutions more difficult to obtain.

For the thermal diffusivities, the individual species diffusivities are flux-limited, then multiplied by the corresponding density (ion or neutral) and then added up in a term which gets multiplied by the (common) ion temperature gradient. For the neutrals, the heat conductivity is

$$\chi_N = n_N \lambda_{CX} v_T, \tag{B.67}$$

where $\lambda_{CX} = 1/(\sigma_{CX} n_i)$ is the neutral CX mean-free-path, n_N is the neutral density, and v_T is the (neutral, ion) thermal velocity. Thus, this conductivity gets limited as

$$\chi_N \Rightarrow \frac{n_N \lambda_{CX} v_T}{\left[1 + \left|\frac{n_N \lambda_{CX} v_T \nabla T_i}{\alpha_2 v_T n_N T_i}\right|^{\gamma_2}\right]^{(1/\gamma_2)}} \tag{B.68}$$

where $\alpha_2 \sim 1$ again. Some cancellations can be made in the equation, but all terms are included for clarity. Again, either $\gamma_2 = 1$ or $\gamma_2 = 2$ could be used.

In general, both of these terms can be important for limiting the heat flux, as the neutral convective velocity shows up as a convective heat flux. In fact, because neutral gradients can be larger than T_i gradients, the convective velocity corrections might be the most important. The impact of the fluid model and associated flux limits, in comparison with the `Eirene` Monte Carlo model, is covered in detail with examples in [43].

B.4.2 First-flight model

As an intermediate approximation between the fluid and kinetic neutral transport models, a so-called "first-flight" model has been implemented. In this model, the first trajectory flight of neutrals launched from the computational domain boundary (at the plates or the wall boundaries) is followed until the neutral undergoes ionisation or charge-exchange, and then passed on to the neutral fluid model. In practice, as an initial computational step, one defines a number of chords (user-specified, usually 9) radiating from each edge boundary segment into the plasma, and computes the length travelled by each chord within each plasma cell it crosses. Then, at each time step, the locally injected neutral flux (from recycling, sputtering, or the like) is launched along each chord (with a normalized intensity determined by some pre-determined angular dependence) and considered to be attenuated within each intersected cell according to the local CX and ionisation mean free paths, while the balancing plasma sources and sinks in that cell are computed. The effect of such a model is also shown in [43].

B.4.3 Kinetic neutrals: B2.5-Eirene coupling

The coupling procedure between the plasma fluid and Monte Carlo neutral parts of the SOLPS package is done by means of passing arrays in both directions, namely the `braeir` and `eirbra` modules. The plasma fluid code delivers a plasma background on which the neutral trajectories are computed (also available as files `fort.30` for the plasma geometry and `fort.31` for the plasma solution), and the associated particle, parallel momentum and energy sources are deduced (returned *via* the `fort.44` file). Then, in order to translate the total energy sources from the Monte Carlo code into the internal energy sources needed for the plasma fluid solution, we apply the following conversion rules, where the E superscript represents the energy sources in `Eirene` (and old `B2`) and H the heat sources in `B2.5`:

$$\text{total energy} \Rightarrow \; S_a^E = \int \frac{1}{2} m_a v_a^2 S_a \, d^3v, \tag{B.69}$$

$$\text{internal energy} \Rightarrow \; S_a^H = \int \frac{1}{2} m_a \left(v_a - u_a\right)^2 S_a \, d^3v, \tag{B.70}$$

$$\rightarrow \; S_a^H = S_a^E - u_a \cdot S_a^{mom} + \frac{1}{2} m_a u_a^2 S_a^{particle}. \tag{B.71}$$

The derivation is as follows. The energy source rate from `Eirene` is computed as:

$$S_i^E = \sum_a S_a^E = \sum_a \sum_{Ea} \iint \frac{1}{2} m_a v_a^2 \, d^3v \, d^3x$$

where the $\sum_a$ is taken over all species, and $\sum_{Ea}$ over the newly ionised particles of species a coming from the `Eirene` strata, and the spatial integral is taken over the cell volume. We also define the particle and parallel momentum source rates:

$$S_i^{particle} = \sum_a S_{Ea}^{particle} = \sum_a \sum_{Ea} \int 1 \cdot d^3x$$
$$\vec{S}_i^{mom} = \sum_a \vec{S}_{Ea}^{mom} = \sum_a \sum_{Ea} \int m_a \vec{v}_a \, d^3x$$

We need S_i^H, which, neglecting all other terms, is defined by:

$$\frac{\partial}{\partial t}\left(\int \frac{3}{2} n_i T_i \, d^3x\right) = \frac{\partial}{\partial t}\left(\sum_a \int \frac{3}{2} n_a T_a \, d^3x\right) = S_i^H$$

Thus,

$$\frac{3}{2}\sum_a \int \left(n_a^{t+\Delta t} T_a^{t+\Delta t} - n_a^t T_a^t\right) d^3x = S_i^H \Delta t$$

$$\frac{3}{2}\sum_a \left[\left(N_a^t + S_a^{particle}\Delta t\right) T_a^{t+\Delta t} - N_a^t T_a^t\right] = S_i^H \Delta t$$

where $N_a = \int n_a d^3x$ is the cell particle content for species a.

For a drifting Maxwellian distribution with drift velocity $\vec{u}_a$, the temperature (in energy units) is defined as:

$$\frac{3}{2} T_a = \left\langle \frac{1}{2} m_a (\vec{v}_a - \vec{u}_a)^2 \right\rangle$$

In `B2.5`, we assume $T_a = T_i$ and use $n_i = \sum_a n_a$.

So we can write:

$$\sum_a \left(N_a^t + S_a^{particle}\Delta t\right) \left\langle \frac{1}{2} m_a (\vec{v}_a - \vec{u}_a)^2 \right\rangle^{t+\Delta t} - \sum_a N_a^t \left\langle \frac{1}{2} m_a (\vec{v}_a - \vec{u}_a)^2 \right\rangle^t = S_i^H \Delta t \tag{B.72}$$

We define:

$$\vec{u}_a^{t+\Delta t} = \langle \vec{v}_a \rangle^{t+\Delta t} = \frac{N_a^t \vec{u}_a^t + \frac{1}{m_a}\vec{S}_a^{mom}\Delta t}{N_a^t + S_a^{particle}\Delta t} \tag{B.73}$$

$$\delta \vec{u}_a = \vec{u}_a^{t+\Delta t} - \vec{u}_a^t = \frac{\frac{1}{m_a}\vec{S}_a^{mom}\Delta t - \vec{u}_a^t S_a^{particle}\Delta t}{N_a^t + S_a^{particle}\Delta t} \tag{B.74}$$

And make use of:

$$\begin{aligned}
\left\langle \frac{1}{2} m_a (\vec{v}_a - \vec{u}_a)^2 \right\rangle^{t+\Delta t} &= \left\langle \frac{1}{2} m_a \left(\vec{v}_a - \left(\vec{u}_a^t + \delta\vec{u}_a\right)\right)^2 \right\rangle^{t+\Delta t} \\
&= \left\langle \frac{1}{2} m_a \left(v_a^2 - 2\vec{v}_a \cdot \vec{u}_a^t - 2\vec{v}_a \cdot \delta\vec{u}_a + {u_a^t}^2 + 2\delta\vec{u}_a \cdot \vec{u}_a^t + (\delta u_a)^2 \right) \right\rangle^{t+\Delta t} \\
&= \left\langle \frac{1}{2} m_a \left[\left(\vec{v}_a - \vec{u}_a^t\right)^2 + \left(-2\vec{v}_a \cdot \delta\vec{u}_a + 2\delta\vec{u}_a \cdot \vec{u}_a^t + (\delta u_a)^2 \right) \right] \right\rangle^{t+\Delta t} \\
&= \left\langle \frac{1}{2} m_a \left(\vec{v}_a - \vec{u}_a^t\right)^2 \right\rangle^{t+\Delta t} - m_a \langle \vec{v}_a \cdot \delta\vec{u}_a \rangle^{t+\Delta t} + m_a \langle \vec{u}_a^t \cdot \delta\vec{u}_a \rangle^{t+\Delta t} + \frac{1}{2} m_a (\delta u_a)^2 \\
&= \left\langle \frac{1}{2} m_a \left(\vec{v}_a - \vec{u}_a^t\right)^2 \right\rangle^{t+\Delta t} - m_a \delta\vec{u}_a \cdot \left(\vec{u}_a^{t+\Delta t} - \vec{u}_a^t\right) + \frac{1}{2} m_a (\delta u_a)^2 \\
&= \left\langle \frac{1}{2} m_a \left(\vec{v}_a - \vec{u}_a^t\right)^2 \right\rangle^{t+\Delta t} - \frac{1}{2} m_a (\delta u_a)^2
\end{aligned}$$

Writing out the first average explicitly:

$$\begin{aligned}
&\left\langle \frac{1}{2} m_a \left(\vec{v}_a - \vec{u}_a^t\right)^2 \right\rangle^{t+\Delta t} \\
&= \frac{\iint n_a^t \left(\frac{m_a}{2\pi T_a^t}\right)^{\frac{3}{2}} \frac{1}{2} m_a (\vec{v}_a - \vec{u}_a^t)^2 \exp\left(-\frac{1}{2} m_a \frac{\left(\vec{v}_a - \vec{u}_a^t\right)^2}{T_a^t} \right) d^3\vec{v}_a \, d^3x + \sum_a \frac{1}{2} m_a (\vec{v}_a - \vec{u}_a^t)^2 \Delta t}{N_a^t + S_a^{particle} \Delta t} \\
&= \frac{1}{N_a^t + S_a^{particle} \Delta t} \left[\frac{3}{2} N_a^t T_a^t + \sum_a \int \frac{1}{2} m_a v_a^2 \Delta t \, d^3x + \frac{1}{2} m_a {u_a^t}^2 S_a^{particle} \Delta t - \vec{u}_a^t \cdot \vec{S}_a^{mom} \Delta t \right] \\
&= \frac{1}{N_a^t + S_a^{particle} \Delta t} \left[\frac{3}{2} N_a^t T_a^t + S_a^E \Delta t + \frac{1}{2} m_a {u_a^t}^2 S_a^{particle} \Delta t - \vec{u}_a^t \cdot \vec{S}_a^{mom} \Delta t \right]
\end{aligned}$$

Putting all this back into Eq. (B.72), we obtain:

$$\begin{aligned}
&\sum_a \left\{ \left[\frac{3}{2} N_a^t T_a^t + S_a^E \Delta t + \frac{1}{2} m_a {u_a^t}^2 S_a^{particle} \Delta t - \vec{u}_a^t \cdot \vec{S}_a^{mom} \Delta t \right] - \left(N_a^t + S_a^{particle} \Delta t\right) \frac{1}{2} m_a (\delta u_a)^2 \right\} \\
&- \sum_a \frac{3}{2} N_a^t T_a^t = S_i^H \Delta t
\end{aligned}$$

Thus:

$$S_i^H = \sum_a S_a^E - \sum_a \vec{u}_a^t \cdot \vec{S}_a^{mom} + \sum_a \frac{1}{2} m_a {u_a^t}^2 S_a^{particle} - \sum_a \frac{N_a^t + S_a^{particle} \Delta t}{\Delta t} \frac{1}{2} m_a (\delta u_a)^2 \tag{B.75}$$

In the code, we previously had only the first three terms in the R.H.S. of the final equation above. But for very strong sources, dominating the background plasma, the higher order correction is necessary.

In some cases, it may also be essential to ensure excellent particle conservation. During a coupled run timestep, as the `B2` internal iterations make the plasma solution evolve away from the plasma background originally used for the computation of the `Eirene` sources above, it may become necessary to rescale these. A scheme to do so was developed by A. Kukushkin for ITER applications [46] and can be activated using the `eirene_ank_mods` switch. See the listing in Appendix A.4 and the additional description in the `${SOLPSTOP}/doc/Source_Scaling_in_B2.pdf` file.

B.5 Classical transport: Braginskii, Balescu and Zhdanov formulations

In **SOLPS 4.x** the classical transport coefficients were derived from Braginskii [47]. In **SOLPS 5.x** these have been replaced with expressions from Balescu [48], completed in **SOLPS-ITER** with expressions derived from the work of Zhdanov [49], sometimes fitted as a function of the local Z_{eff}. Table B.1 summarizes the differences between the Braginskii and Balescu models. It is possible in the code to use either model, or even the full Balescu 21-moment expressions, as well as a Zhdanov fit to the table published in the Braginskii paper as a function of Z_{eff}. See the use of the switches `'b2tqce_model'`, `'b2tqca_model'` and `'b2sigp_style'` as described in Appendix A.4.

quantity	functional dependence	numerical prefactor		
		Braginskii	fits to Balescu	fits to Zhdanov
default in		**SOLPS 4.x**	**SOLPS 5.x**	**SOLPS-ITER**
$\kappa_{e\|\|}$	$\frac{T_e^{5/2} n_e}{\ln\Lambda \sum_a Z_a^2 n_a}$	3.16	$\frac{5}{2}\frac{2.16 Z_{eff}}{1+0.27 Z_{eff}}$	$\frac{\left(3.9+\frac{2.3}{Z_{eff}}\right)}{0.31+\frac{1.2}{Z_{eff}}+\frac{0.41}{Z_{eff}^2}}$
$\kappa_{i\|\|}$	$\frac{T_i^{5/2}}{\ln\Lambda}\sum_a \frac{n_a}{Z_a^2}$	$\frac{3.9}{\sum_b Z_b^2 n_b \sqrt{\frac{2 m_a m_b}{m_a+m_b}}}$	$\frac{3.95}{\sqrt{m_a}\sum_b Z_b^2 n_b}$	
$\eta_{a\|\|}$	$\frac{T_i^{5/2}}{\ln\Lambda}\frac{n_a}{Z_a^2}$	$\frac{1.37}{\sum_b \frac{Z_b^2 n_b}{\sqrt{\frac{m_a+m_b}{2}}}}$	$\frac{\sqrt{2}\, 0.96\sqrt{m_a}}{\sum_b Z_b^2 n_b}$	
$\sigma_{\|\|}$	$\frac{n_e T_e^{3/2}}{\ln\Lambda \sum_a Z_a^2 n_a}$	$\frac{4.95 Z_{eff}}{1+1.54 Z_{eff}}$	$\frac{(1+0.24 Z_{eff})(1+0.93 Z_{eff})}{(1+2.56 Z_{eff})(1+0.29 Z_{eff})}$	
$\alpha_{e\|\|}$	$\frac{n_e T_e^{3/2}}{\ln\Lambda \sum_a Z_a^2 n_a}$	$-\sqrt{\frac{5}{2}}\frac{1.25 Z_{eff}}{1+0.41 Z_{eff}}$	$-\frac{1.56 Z_{eff}\frac{\left(1+1.4 Z_{eff}\right)\left(1+0.52 Z_{eff}\right)}{\left(1+2.56 Z_{eff}\right)\left(1+0.29 Z_{eff}\right)}\frac{1}{Z_{eff}+\frac{\sqrt{2}}{2}}}{\frac{\left(1+0.24 Z_{eff}\right)\left(1+0.93 Z_{eff}\right)}{\left(1+2.56 Z_{eff}\right)\left(1+0.29 Z_{eff}\right)}}$	

Table B.1: Comparison of classical transport coefficients according to the Braginskii, Balescu, and Zhdanov formulations

Note that this table does not contain the full prefactors, but only the numerically changing pieces between the models.

In figures B.4 to B.6, we illustrate the dependencies of the electronic classical transport coefficients as a function of Z_{eff} , comparing the **SOLPS 5.x** and **SOLPS-ITER** fit used by default (to the 29-moment Balescu result), the exact 21-moment Balescu expressions, the Zhdanov fit to Braginskii and the **SOLPS 4.x** values.

B.6 Classical ion transport: Zhdanov-Grad module

This code feature is still experimental.

Fully multi-ion collisional closure for the components along the magnetic field of the vector and tensor moments of the distribution function is applied for all ions (without specifying main ions and impurities) based on the works of Zhdanov [49] and Makarov *et al.* [50]. It is used if `'b2tral_Zhdanov_closure'` is set to `'1'`. It is not applied by default. The Zhdanov-Grad module operates in two regimes: 1) Improved analytical method (IAM), if `'b2tral_zh_imp_analyt'` is set to `'1'`; 2) Explicit matrix inversion method (EMIM), if `'b2tral_zh_imp_analyt'` is set to `'0'`. The Zhdanov-Grad closure provides transport coefficients for the parallel components of the ion heat conductivity, the ion velocity-dependent heat flux, the ion thermal force, the ion friction force, the ion flow velocity viscous stress and the ion heat viscous stress for all ions. It allows aplication of **SOLPS-ITER** to arbitrary plasma mixtures such as: $D+T+He$+impurities, $H+D$+impurities, $He+H$+impurities, etc... A detailed description of this closure method can be found in section C.11.

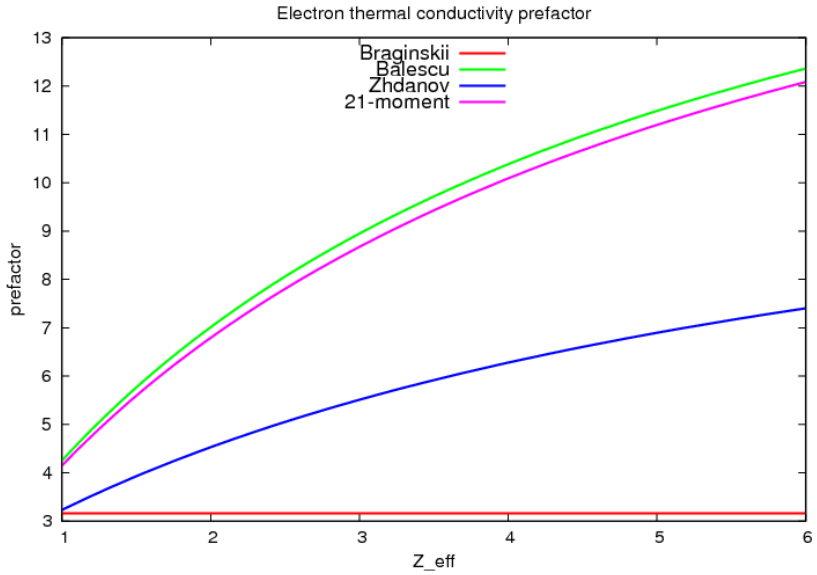


Figure B.4: Dependence of the various models for the $\kappa_{e\|}$ prefactor as a function of Z_{eff}.

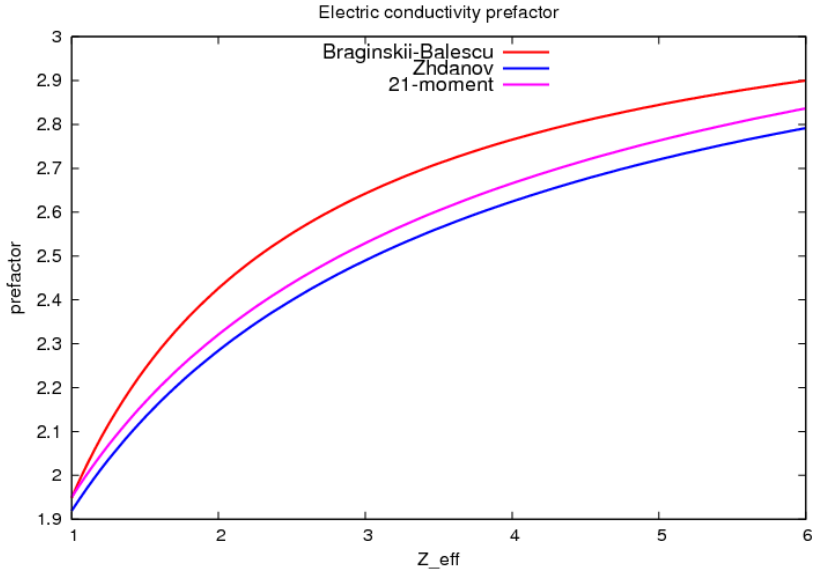


Figure B.5: Dependence of the various models for the parallel electron conductivity $\sigma_{\|}$ prefactor as a function of Z_{eff}.

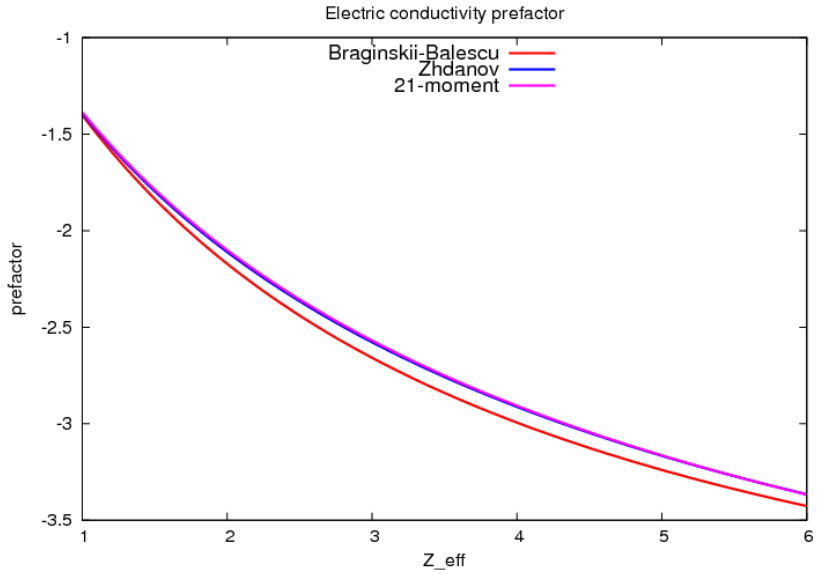


Figure B.6: Dependence of the various models for the thermo-electric coefficient $\alpha_{e\|}$ prefactor as a function of Z_{eff}.

Appendix C

Model equations for B2.5

The version of the equations shown in this Appendix represents the best available description of the standard recommended settings for the **SOLPS-ITER** physics model, built upon the original work on **SOLPS 5.2** from the St. Petersburg group [51]. Use of that physics model is governed by the switch `b2mndt_style`, with a default value of `1`. If set to `0`, then the **SOLPS5.0** physics model, as described in Appendix B and in the review paper [16], will be applied instead. The present Appendix is not meant as an exhaustive registry of the capabilities of the code, for which the reader is encouraged to peruse Chapters 3 and 5 as well as the list of switches in Appendix A.4 and of namelist variables in Appendix A.5.

Some equations include terms which are multiplied by zero. This is meant to indicate that such terms are not included in the standard **SOLPS-ITER** physics model, but that switches exist that allow for their inclusion.

C.1 Sign conventions

The radial coordinate y increases as one crosses the separatrix from the core (or private flux) region into the main SOL region (and so is minimum at the magnetic axis). The poloidal coordinate x increases in the *clockwise* direction when viewing the poloidal plane with the central axis line on its left. The toroidal magnetic field component B_z is taken as positive if the toroidal magnetic field is directed *clockwise* when viewing the tokamak from above. The poloidal magnetic field component B_x follows the same convention as the x component and is positive when directed in the direction of increasing poloidal index, i.e. *clockwise*. This forms a right-handed poloidal coordinate system (x, y, z) equivalent to (θ, ρ, ϕ). In terms of the conventions laid out in [52], the magnetic field is calculated from the provided magnetic equilibrium solution according to a *COCOS* index of **4**. To ensure the correct orientation of B_x and B_z, either provide the magnetic equilibrium solution converted to *COCOS* index of **4**, or manually check and invert the orientations as suggested by the hints provided in `b2ag.log`. Equilibrium file format conversion tools such as `e2d` do not apply any *COCOS* convention conversion unless explicitly specified.

Is it important to know, that, in the equations below, the parallel velocity $V_\parallel$ is a true parallel velocity, i.e. it changes sign if the poloidal magnetic field component does, namely: $V_\parallel = \vec{V} \cdot \vec{B} / |B|$. This applies to all species, including electrons, meaning $V_{\parallel a}$ and $V_{\parallel e}$. However, the poloidal projections of the fluxes and currents, such as Γ_{ax}, are positive when they are colinear with $\vec{e}_x$, i.e. point in the direction of increasing x. The exception to this is the parallel current, $j_\parallel$. In the equations below, it has the same sign convention as the parallel velocity. In the actual code, because it is treated as a flux, it follows the sign convention of the fluxes. This therefore means that the code does not follow $j_\parallel$, but rather $j_\parallel \cdot |B_x| / B_x$.

C.2 Continuity equations for particle conservation

$$\frac{\partial n_a}{\partial t}+\frac{1}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x}\widetilde{\Gamma}_{ax}\right)+\frac{1}{\sqrt{g}}\frac{\partial}{\partial y}\left(\frac{\sqrt{g}}{h_y}\widetilde{\Gamma}_{ay}\right)=S_a^n, a=0,1,\ldots,ns-1 \tag{C.1}$$

$$\widetilde{\Gamma}_{ax}=\left(b_xV_{\parallel a}+V_{ax}^{(E)}+V_{ax}^{(AN)}+V_a^{corr_dpc}\right)n_a+0\cdot\frac{1}{z_ae}\left(j_{ax}^{(AN)}+j_{ax}^{(in)}+\widetilde{j}_{ax}^{(vis\parallel)}+j_{ax}^{(vis\perp)}+\widetilde{j}_{ax}^{(visq)}\right)$$
$$-D_{n,a}\frac{1}{h_x}\frac{\partial n_a}{\partial x}-\widetilde{D}_{p,ax}\frac{1}{h_x}\frac{\partial p_a}{\partial x}-\widetilde{\Gamma}_{ax}^{(PSch)} \tag{C.2}$$

$$\widetilde{\Gamma}_{ay}=\left(V_{ay}^{(E)}+V_{ay}^{(AN)}\right)n_a+\frac{1}{z_ae}\left(j_{ay}^{(AN)}+j_{ay}^{(in)}+\widetilde{j}_{ay}^{(vis\parallel)}+j_{ay}^{(vis\perp)}+\widetilde{j}_{ay}^{(visq)}\right)$$
$$-D_{n,a}\frac{1}{h_y}\frac{\partial n_a}{\partial y}-\widetilde{D}_{p,ay}\frac{1}{h_y}\frac{\partial p_a}{\partial y}-\widetilde{\Gamma}_{ay}^{(PSch)} \tag{C.3}$$

$$\Gamma_{ax}^{(PSch)}=D_{axy}\frac{1}{h_y}\frac{\partial n_aT_i}{\partial y}\qquad\qquad\Gamma_{ay}^{(PSch)}=-D_{ayx}\frac{1}{h_x}\frac{\partial n_aT_i}{\partial x} \tag{C.4}$$

$$\widetilde{\Gamma}_{ax}^{(PSch)}=\begin{cases}\Gamma_{ax}^{(PSch)},\ z_a=0\\ \dfrac{\Gamma_{ax}^{(PSch)}}{1+\frac{\left|\Gamma_{ax}^{(PSch)}\right|}{\frac{\sqrt{g}}{h_x}n_aV_{Ta}}},\ z_a\neq 0\end{cases}\quad\widetilde{\Gamma}_{ay}^{(PSch)}=\begin{cases}\Gamma_{ay}^{(PSch)},\ z_a=0\\ \dfrac{\Gamma_{ay}^{(PSch)}}{1+\frac{\left|\Gamma_{ay}^{(PSch)}\right|}{\frac{\sqrt{g}}{h_y}n_aV_{Ta}}},\ z_a\neq 0\end{cases} \tag{C.5}$$

$$V_{Ta}=\sqrt{\frac{8T_i}{\pi m_a}} \tag{C.6}$$

$$\Gamma_{ax}=\left(b_xV_{\parallel a}+\widetilde{U}_{ax}^{(dia)}+V_{ax}^{(AN)}+V_a^{corr_dpc}\right)n_a+$$
$$0\cdot\frac{1}{z_ae}\left(j_{ax}^{(AN)}+j_{ax}^{(in)}+\widetilde{j}_{ax}^{(vis\parallel)}+j_{ax}^{(vis\perp)}+\widetilde{j}_{ax}^{(visq)}\right)-D_{n,a}\frac{1}{h_x}\frac{\partial n_a}{\partial x}-\widetilde{D}_{p,ax}\frac{1}{h_x}\frac{\partial p_a}{\partial x} \tag{C.7}$$

$$\Gamma_{ay}=\left(\widetilde{U}_{ay}^{(dia)}+V_{ay}^{(AN)}\right)n_a+\frac{1}{z_ae}\left(j_{ay}^{(AN)}+j_{ay}^{(in)}+\widetilde{j}_{ay}^{(vis\parallel)}+j_{ay}^{(vis\perp)}+\widetilde{j}_{ay}^{(visq)}\right)$$
$$-D_{n,a}\frac{1}{h_y}\frac{\partial n_a}{\partial y}-\widetilde{D}_{p,ay}\frac{1}{h_y}\frac{\partial p_a}{\partial y} \tag{C.8}$$

$$\Gamma_{ax}^{Cor}=\left(b_xV_{\parallel a}+\widetilde{U}_{ax}^{(2dia)}+V_{ax}^{(AN)}+V_a^{corr_dpc}\right)n_a+$$
$$0\cdot\frac{1}{z_ae}\left(j_{ax}^{(AN)}+j_{ax}^{(in)}+\widetilde{j}_{ax}^{(vis\parallel)}+j_{ax}^{(vis\perp)}+\widetilde{j}_{ax}^{(visq)}\right)-D_{n,a}\frac{1}{h_x}\frac{\partial n_a}{\partial x}-\widetilde{D}_{p,ax}\frac{1}{h_x}\frac{\partial p_a}{\partial x} \tag{C.9}$$

$$\Gamma_{ay}^{Cor}=\left(\widetilde{U}_{ay}^{(2dia)}+V_{ay}^{(AN)}\right)n_a+$$
$$\frac{1}{z_ae}\left(j_{ay}^{(AN)}+j_{ay}^{(in)}+\widetilde{j}_{ay}^{(vis\parallel)}+j_{ay}^{(vis\perp)}+\widetilde{j}_{ay}^{(visq)}\right)-D_{n,a}\frac{1}{h_y}\frac{\partial n_a}{\partial y}-\widetilde{D}_{p,ay}\frac{1}{h_y}\frac{\partial p_a}{\partial y} \tag{C.10}$$

$$\Gamma_{ax}^{(\text{no drift})} = \left(b_x V_{\|a} + V_{ax}^{(AN)} + 0 \cdot V_a^{corr_dpc}\right) n_a \tag{C.11}$$

$$\Gamma_{ay}^{(\text{no drift})} = \left(V_{ay}^{(AN)}\right) n_a, \text{really 0 as } V_{ay}^{(AN)} = 0 \tag{C.12}$$

where

$$V_{ax}^{(E)} = \begin{cases} -\dfrac{B_z}{B^2}\dfrac{1}{h_y}\dfrac{\partial \Phi}{\partial y}, z_a \neq 0 \\ 0, z_a = 0 \end{cases} \qquad V_{ay}^{(E)} = \begin{cases} \dfrac{B_z}{B^2}\dfrac{1}{h_x}\dfrac{\partial \Phi}{\partial x}, z_a \neq 0 \\ 0, z_a = 0 \end{cases} \tag{C.13}$$

$$\widetilde{V}_{ax}^{(dia)} = \begin{cases} \dfrac{T_i B_z}{z_a e}\dfrac{\partial}{h_y \partial y}\left(\dfrac{1}{B^2}\right), z_a \neq 0 \\ 0, z_a = 0 \end{cases} \qquad \widetilde{V}_{ay}^{(dia)} = \begin{cases} -\dfrac{T_i B_z}{z_a e}\dfrac{\partial}{h_x \partial x}\left(\dfrac{1}{B^2}\right), z_a \neq 0 \\ 0, z_a = 0 \end{cases} \tag{C.14}$$

$$\widetilde{\widetilde{V}}_{ax}^{(dia)} = \begin{cases} \widetilde{V}_{ax}^{(dia)}, z_a = 0 \\ \dfrac{\widetilde{V}_{ax}^{(dia)}}{1 + \frac{\left|\widetilde{V}_{ax}^{(dia)}\right|}{V_{T_a}}}, z_a \neq 0 \end{cases} \qquad \widetilde{\widetilde{V}}_{ay}^{(dia)} = \begin{cases} \widetilde{V}_{ay}^{(dia)}, z_a = 0 \\ \dfrac{\widetilde{V}_{ay}^{(dia)}}{1 + \frac{\left|\widetilde{V}_{ay}^{(dia)}\right|}{V_{T_a}}}, z_a \neq 0 \end{cases} \tag{C.15}$$

$$\Gamma_{ax}^{(Eir)} = \left(b_x V_{\|a} + V_{ax}^{(E)} + \widetilde{W}_{ax}^{(dia)} + V_{ax}^{(AN)} + V_a^{corr_dpc}\right) n_a + 0 \cdot \frac{1}{z_a e}\left(j_{ax}^{(AN)} + j_{ax}^{(in)} + \widetilde{j}_{ax}^{(vis\|)} + j_{ax}^{(vis\perp)} + \widetilde{j}_{ax}^{(visq)}\right) - D_{n,a}\frac{1}{h_x}\frac{\partial n_a}{\partial x} \tag{C.16}$$

$$\Gamma_{ay}^{(Eir)} = \left(V_{ay}^{(E)} + \widetilde{W}_{ya}^{(dia)} + V_{ay}^{(AN)}\right) n_a + \frac{1}{z_a e}\left(j_{ay}^{(AN)} + j_{ay}^{(in)} + \widetilde{j}_{ay}^{(vis\|)} + j_{ay}^{(vis\perp)} + \widetilde{j}_{ay}^{(visq)}\right) - D_{n,a}\frac{1}{h_y}\frac{\partial n_a}{\partial y} \tag{C.17}$$

where

$$W_{ax}^{(dia)} = \begin{cases} -\dfrac{B_z}{z_a e B^2 n_a}\dfrac{1}{h_y}\dfrac{\partial n_a T_i}{\partial y} \; z_a \neq 0 \\ 0, z_a = 0 \end{cases} \qquad W_{ay}^{(dia)} = \begin{cases} \dfrac{B_z}{z_a e B^2 n_a}\dfrac{1}{h_x}\dfrac{\partial n_a T_i}{\partial x} \; z_a \neq 0 \\ 0, z_a = 0 \end{cases} \tag{C.18}$$

$$\widetilde{W}_{ax}^{(dia)} = \begin{cases} W_{ax}^{(dia)}, z_a = 0 \\ \dfrac{W_{ax}^{(dia)}}{1 + \frac{\left|W_{ax}^{(dia)}\right|}{V_{T_a}}}, z_a \neq 0 \end{cases} \qquad \widetilde{W}_{ay}^{(dia)} = \begin{cases} W_{ay}^{(dia)}, z_a = 0 \\ \dfrac{W_{ay}^{(dia)}}{1 + \frac{\left|W_{ay}^{(dia)}\right|}{V_{T_a}}}, z_a \neq 0 \end{cases} \tag{C.19}$$

$$U_{ax}^{(dia)} = \widetilde{V}_{ax}^{(dia)} + V_{ax}^{(E)} \qquad U_{ay}^{(dia)} = \widetilde{V}_{ay}^{(dia)} + V_{ay}^{(E)} \tag{C.20}$$

$$U_{ax}^{(2dia)} = 2\widetilde{V}_{ax}^{(dia)} + V_{ax}^{(E)} \qquad U_{ay}^{(2dia)} = 2\widetilde{V}_{ay}^{(dia)} + V_{ay}^{(E)} \tag{C.21}$$

$$\widetilde{U}_{ax}^{(dia)} = \begin{cases} U_{ax}^{(dia)}, z_a = 0 \\ \dfrac{U_{ax}^{(dia)}}{1 + \frac{\left|U_{ax}^{(dia)}\right|}{V_{T_a}}}, z_a \neq 0 \end{cases} \qquad \widetilde{U}_{ay}^{(dia)} = \begin{cases} U_{ay}^{(dia)}, z_a = 0 \\ \dfrac{U_{ay}^{(dia)}}{1 + \frac{\left|U_{ay}^{(dia)}\right|}{V_{T_a}}}, z_a \neq 0 \end{cases} \tag{C.22}$$

$$\widetilde{U}_{ax}^{(2dia)} = \begin{cases} U_{ax}^{(2dia)}, z_a = 0 \\ \dfrac{U_{ax}^{(2dia)}}{1 + \frac{\left|U_{ax}^{(2dia)}\right|}{V_{T_a}}}, z_a \neq 0 \end{cases} \qquad \widetilde{U}_{ay}^{(2dia)} = \begin{cases} U_{ay}^{(2dia)}, z_a = 0 \\ \dfrac{U_{ay}^{(2dia)}}{1 + \frac{\left|U_{ay}^{(2dia)}\right|}{V_{T_a}}}, z_a \neq 0 \end{cases} \tag{C.23}$$

$$E_{k,a} = \frac{m_a}{2} V_{\|a}^2 \tag{C.24}$$

$$W_{a\perp}^{(dia)} = \begin{cases} -\dfrac{1}{z_a e B n_a} \dfrac{1}{h_y} \dfrac{\partial n_a T_i}{\partial y}, z_a \neq 0 \\ 0, z_a = 0 \end{cases} \qquad W_{ay}^{(dia)} = \begin{cases} \dfrac{B_z}{z_a e B^2 n_a} \dfrac{1}{h_x} \dfrac{\partial n_a T_i}{\partial x}, z_a \neq 0 \\ 0, z_a = 0 \end{cases} \tag{C.25}$$

$$V_{a\perp}^{(\mathbf{E}\times\mathbf{B})} = \begin{cases} -\dfrac{1}{B} \dfrac{1}{h_y} \dfrac{\partial \Phi}{\partial y}, z_a \neq 0 \\ 0, z_a = 0 \end{cases} \qquad V_{ay}^{(\mathbf{E}\times\mathbf{B})} = \begin{cases} \dfrac{B_z}{B^2} \dfrac{1}{h_x} \dfrac{\partial \Phi}{\partial x}, z_a \neq 0 \\ 0, z_a = 0 \end{cases} \tag{C.26}$$

$$n_e = \sum_{a=0}^{ns-1} z_a n_a, \qquad n_i = \sum_{a=0}^{ns-1} n_a, \qquad \widetilde{n}_i = \sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} n_a \tag{C.27}$$

$$p_a = n_a \left(z_a T_e + T_i\right), \qquad p_e = n_e T_e \tag{C.28}$$

$$V_a^{corr_dpc} = \frac{c^{corr_dpc}}{16} |b_x| \sqrt{\frac{p_a}{m_a n_a}} \frac{\partial^2}{\partial x^2} \left(\frac{1}{p_a} \frac{\partial p_a}{\partial x}\right) \tag{C.29}$$

The ion densities n_i and $\widetilde{n}_i$ are stored in the `ni(:,:,0)` and `ni(:,:,1)` array components respectively.

Which speciated j_a current contributions are non-zero depends on the physics model switches `b2tanml_anml_all_ions` for $j_a^{(AN)}$, `b2tiner_inert_all_ions` for $j_a^{(in)}$, and `b2tvspr_visper_all_ions` for $j_a^{(vis\perp)}$, respectively. If the Zhdanov-Grad closure is applied, then all ion species contribute to the $\widetilde{j}_a^{(vis\|)}$ and $\widetilde{j}_a^{(visq)}$ currents. Otherwise, only the one hydrogenic species with $z_a = 1$ and $z_{n,a} = 1$ will. If the code contains multiple hydrogenic species and the Zhdanov closure scheme is not activated, an error will be reported to avoid double-counting. Overall, the assignment of these current contributions to the ion instead of to the electron fluxes is governed by the switches `b2tfnb_xcur` and `b2tfnb_ycur` for the poloidal and radial directions, respectively.

C.2.1 Source terms in density equations

$$S_a^n = S_{I_a}^n + S_{I_{a+1}}^n + S_{R_a}^n + S_{R_{a-1}}^n + S_{CX_a}^n + S_{CX_{a-1}}^n + S_a^{\text{Eirene}} \tag{C.30}$$

z_a — the atomic charge of species a, $z_{n,a}$ — the nuclear charge of species a

$$S_{I_a}^n \quad = \quad \left\{ -r_{I_a} n_e n_a, \quad a = 0, \ldots, ns-2, z_a < z_{a+1} \text{ and } z_{n,a} = z_{n,a+1} \right. \tag{C.31}$$

$$S^n_{I_{a+1}} = \begin{cases} r_{I_a} n_e n_a, & a = 0, \ldots, ns-2, z_a < z_{a+1} \text{ and } z_{n,a} = z_{n,a+1} \\ 0 \end{cases} \tag{C.32}$$

$$r_{I_a} = \exp\left(r_{0,I_a} + r_{1,I_a} \ln\left(\frac{T_e}{e}\right)\right) \tag{C.33}$$

$$S^n_{R_a} = \begin{cases} -r_{R_a} n_e n_a, & a = 1, \ldots, ns-1, z_{a-1} < z_a \text{ and } z_{n-1,a} = z_{n,a} \\ 0 \end{cases} \tag{C.34}$$

$$S^n_{R_{a-1}} = \begin{cases} r_{R_a} n_e n_a, & a = 1, \ldots, ns-1, z_{a-1} < z_a \text{ and } z_{n-1,a} = z_{n,a} \\ 0 \end{cases} \tag{C.35}$$

$$r_{R_a} = \exp\left(r_{0,R_a} + r_{1,R_a} \ln\left(\frac{T_e}{e}\right)\right) \tag{C.36}$$

$$S^n_{CX_a} = -\sum_{\substack{b=0 \\ b \neq a-1}}^{ns_{CX}} r_{CX_{ab}} n_a n_b \tag{C.37}$$

$$S^n_{CX_{a-1}} = \sum_{\substack{b=0 \\ b \neq a-1}}^{ns_{CX}} r_{CX_{ab}} n_a n_b \tag{C.38}$$

(C.39)

$$S^n_{CX_b} = -\sum_{\substack{a=0 \\ b \neq a-1}}^{ns-1} r_{CX_{ab}} n_a n_b \tag{C.40}$$

$$S^n_{CX_{b+1}} = \sum_{\substack{a=0 \\ b \neq a-1}}^{ns-1} r_{CX_{ab}} n_a n_b \tag{C.41}$$

$$r_{CX_{ab}} = \exp\left(r_{0,CX_{ab}} + r_{1,CX_{ab}} \ln\left(\frac{m_p T_i}{m_b e}\right)\right) \tag{C.42}$$

C.2.2 Velocities

$$V_a^{corr_dpc} = \frac{c^{corr_dpc}}{16} |b_x| \sqrt{\frac{p_a}{m_a n_a}} \frac{\partial^2}{\partial x^2} \left(\frac{1}{p_a} \frac{\partial p_a}{\partial x}\right) \tag{C.43}$$

is a (physically meaningless) damping term, often called the "Rhie and Chow" correction [53], meant to avoid the checkerboard instability when solving the momentum equation. The default value of c^{corr_dpc} is 1 and provided by the `'b2tfnb_xfrhie'` switch.

$$V_{ax}^{(E)} = \begin{cases} -c_{\mathbf{E}\times\mathbf{B}} \frac{B_z}{B^2} \frac{1}{h_y} \frac{\partial \Phi}{\partial y}, & z_a \neq 0 \\ 0, & z_a = 0 \end{cases} \tag{C.44}$$

$$V_{ay}^{(E)} = \begin{cases} c_{\mathbf{E}\times\mathbf{B}} \frac{B_z}{B^2} \frac{1}{h_x} \frac{\partial \Phi}{\partial x}, & z_a \neq 0 \\ 0, & z_a = 0 \end{cases} \tag{C.45}$$

$$\widetilde{V}_{ax}^{(dia)} = \begin{cases} c_{dia} \frac{T_i B_z}{z_a e} \frac{\partial}{h_y \partial y} \left(\frac{1}{B^2}\right), & z_a \neq 0 \\ 0, & z_a = 0 \end{cases} \tag{C.46}$$

$$\widetilde{V}_{ay}^{(dia)} = \begin{cases} -c_{dia}\frac{T_i B_z}{z_a e}\frac{\partial}{h_x \partial x}\left(\frac{1}{B^2}\right), & z_a \neq 0 \\ 0, & z_a = 0 \end{cases} \tag{C.47}$$

$V_{ax}^{(AN)} = 0$

$$V_{ay}^{(AN)} = \left(c_{V_a^{(AN)},0} + c_{V_a^{(AN)},1}\frac{10^{20}}{n_e} + c_{V_a^{(AN)},2} D_{f,a} + c_{V_a^{(AN)},3}\frac{T_e}{16e|B|} \right) f_{V_a^{(AN)}}(x,y) \tag{C.48}$$

The $c_{V_a^{(AN)},x}$ coefficients correspond to the `cfvla` array elements provided in the `b2ah.dat` file, or, if present, in the `b2cmpt` block of the `b2mn.dat` file. For charged species, they can be overwritten by means of the `b2.transport.parameters` file, see Section 3.5.3. Typically, no pinch velocity is applied:

$a = 0, c_{V_a^{(AN)},0} = 0, c_{V_a^{(AN)},1} = 0, c_{V_a^{(AN)},2} = 0, c_{V_a^{(AN)},3} = 0$

$a = 1, c_{V_a^{(AN)},0} = 0, c_{V_a^{(AN)},1} = 0, c_{V_a^{(AN)},2} = 0, c_{V_a^{(AN)},3} = 0$

The $f_{V_a^{(AN)}}$ function is an optional modifier, governed by the value of `b2tqna_user_transport`. If $c_{V_a^{(AN)},7} \neq 0$, then $f_{V_a^{(AN)}}$ contains a flux-scaling factor of $\left(\frac{h_y}{h_{y,ref}}\right)^2$.

If `b2tqna_user_transport` is `2`, an ELM model is applied, with three pinch velocities scalings according to radial location and alternating between inter-ELM and intra-ELM values. See code for details.

If `b2tqna_user_transport` is `4`, the pinch velocity in the core is given according to [54].

If `b2tqna_user_transport` is `5`, a pinch velocity for ions is applied between the top of the machine and the X-point on the low-field side. See code for details.

If `b2tqna_user_transport` is `6`, a transport barrier can be applied in a parametric fashion, according to the value of `b2tqna_itype`, and the coefficients `b2tqna_vl_alfa[1-4]`. Blobby transport in the SOL can also be added. See code for details.

If `b2tqna_user_transport` is `9`, a blobby transport model is applied in the SOL. See code for details.

C.2.3 Particle transport coefficients

$$\begin{cases} D_{axy} = \left(\frac{1}{B^2} - \frac{1}{\langle B^2\rangle}\right)\frac{B_z}{z_a e}, & z_a \neq 0 \\ D_{axy} = 0, & z_a = 0 \end{cases} \text{, where } \left\langle B^2 \right\rangle = \frac{\iint_{core} B^2 \sqrt{g} dx dy}{\iint_{core} \sqrt{g} dx dy}$$

$$\begin{cases} D_{ayx} = \left(\frac{1}{B^2} - \frac{1}{\langle B^2\rangle}\right)\frac{B_z}{z_a e}, & z_a \neq 0 \\ D_{ayx} = 0, & z_a = 0 \end{cases} \tag{C.49}$$

$$D_{n,a} = D_{n,a}^{(AN)} \tag{C.50}$$

$$D_{n,a}^{(AN)} = \left(c_{D_{n,a},0} + c_{D_{n,a},1}\frac{10^{20}}{n_e} + c_{D_{n,a},2} D_{f,a} + c_{D_{n,a},3}\frac{T_e}{16e|B|} \right) f_{D_{n,a}}(x,y) \tag{C.51}$$

The $c_{D_{n,a},x}$ coefficients correspond to the `cfdna` array elements provided in the `b2ah.dat` file, or, if present, in the `b2cmpt` block of the `b2mn.dat` file. For charged species, they can be overwritten by means of the `b2.transport.parameters` file, see Section 3.5.3. A typical choice may be:

$a = 0, \quad c_{D_{n,a},0} = 0, \quad c_{D_{n,a},1} = 0, \quad c_{D_{n,a},2} = 0, \quad c_{D_{n,a},3} = 0$

$a = 1, \quad c_{D_{n,a},0} = 0, \quad c_{D_{n,a},1} = 0, \quad c_{D_{n,a},2} = 1, \quad c_{D_{n,a},3} = 0$

$D_{f,a}$ is a diffusivity used to scale the various transport coefficients together and its coefficients $c_{D_{f,a},x}$ correspond to the `cfdf0` array provided in the `b2ah.dat` file, or, if present, in the `b2cmpt` block of the `b2mn.dat` file. Typical values are:

$$D_{f,a} = c_{D_{f,a},0} + c_{D_{f,a},1}\frac{10^{20}}{n_i} + \frac{c_{D_{f,a},2}\sqrt{\frac{T_i}{m_a}}}{c_{D_{f,a},3}n_i + c_{D_{f,a},4}n_e},$$

a	$c_{D_{f,a},0}$	$c_{D_{f,a},1}$	$c_{D_{f,a},2}$	$c_{D_{f,a},3}$	$c_{D_{f,a},4}$	$c_{D_{f,a},5}$	$c_{D_{f,a},6}$	$c_{D_{f,a},7}$
0	10^2	0	1	$5\cdot10^{-20}$	$1.5\cdot10^{-18}$	0	0	0
1	0.3	0	0	–	–	0	0	0

If `'b2tqna_new_df0'` is set to `'1'` (default is `'0'`), then, the diffusivity for neutral species is computed as:

$$D_{f,a} = \frac{\sqrt{\frac{T_i}{m_a}}}{\langle\sigma v\rangle_{ion} n_e + \sum_b \langle\sigma v\rangle_{CX} n_b} + \frac{1}{\frac{n_a}{2\times10^{19}}}, \quad z_a = 0$$

The $f_{D_n,a}$ function is an optional modifier, governed by the value of `b2tqna_user_transport`. If $c_{D_{n,a},7} \neq 0$, then $f_{D_n,a}$ contains a flux-scaling factor of $\left(\frac{h_y}{h_{y,ref}}\right)^2$.

If `b2tqna_user_transport` is `1`, the $D_{n,a}^{(AN)}$ coefficient is scaled by the same amount as $\chi_e^{(AN)}$ necessary to maintain the local T_e gradient at the reference transport location to `set_transport_required_te_gradient`.

If `b2tqna_user_transport` is `2`, an ELM model is applied, with three levels of transport scaling according to radial location and alternating between inter-ELM and intra-ELM values. See code for details.

If `b2tqna_user_transport` is `4`, the diffusion coefficient is modified in the main SOL to reproduce the model from [54].

If `b2tqna_user_transport` is `6`, a transport barrier can be applied in a parametric fashion, according to the value of `b2tqna_itype`, and the coefficients `b2tqna_alfa[1-4]` are used to modify $f_{D_n,a}$. See code for details.

If `b2tqna_user_transport` is `7`, the disruption model is applied, adding $1000 f_{quench}\sqrt{\frac{m_e}{m_a}}\left(\frac{T_i}{1000\text{eV}}\right)^{\frac{5}{2}}\left(\frac{h_y}{h_{y,ref}}\right)^2$ to $D_{n,a}$.

If `b2tqna_user_transport` is `8`, the neoclassical transport model is applied. See code for details.

$$D_{p,a} = D_{p,a}^{(AN)} \tag{C.52}$$

$$D_{p,a}^{(AN)} = \frac{1}{z_a T_e + T_i}\left(c_{D_{p,a},0} + c_{D_{p,a},1}\frac{10^{20}}{n_e} + c_{D_{p,a},2}D_{f,a} + c_{D_{p,a},3}\frac{T_e}{16e|B|}\right) f_{D_p,a}(x,y)$$

Typical values are:

$$a = 0, c_{D_{p,a},0} = 0, c_{D_{p,a},1} = 0, c_{D_{p,a},2} = 1, c_{D_{p,a},3} = 0$$
$$a = 1, c_{D_{p,a},0} = 0, c_{D_{p,a},1} = 0, c_{D_{p,a},2} = 0, c_{D_{p,a},3} = 0$$

If $c_{D_{p,a},7} \neq 0$, then $f_{D_p,a}$ contains a flux-scaling factor of $\left(\frac{h_y}{h_{y,ref}}\right)^2$, otherwise it is 1 everywhere.

The flux limits for $D_{p,a}$ are applied as:

$$\widetilde{D}_{p,ax} = \begin{cases} \frac{D_{p,a}}{\left(1+\left(\frac{\frac{\sqrt{g}}{h_x}D_{p,a}\left|\frac{1}{h_x}\frac{\partial p_a}{\partial x}\right|}{\frac{\sqrt{g}}{h_x}\alpha n_a \frac{V_{T_i}}{4}}\right)^{\gamma_1}\right)^{\frac{1}{\gamma_1}}}, & V_{T_i} = \sqrt{\frac{8T_i}{\pi m_a}}, \quad z_a = 0 \\ D_{p,a}, & z_a \neq 0 \end{cases} \tag{C.53}$$

$$\widetilde{D}_{p,ay} = \begin{cases} \frac{D_{p,a}}{\left(1+\left(\frac{\frac{\sqrt{g}}{h_y}D_{p,a}\left|\frac{1}{h_y}\frac{\partial p_a}{\partial y}\right|}{\frac{\sqrt{g}}{h_y}\alpha n_a\frac{V_{T_i}}{4}}\right)^{\gamma_1}\right)^{\frac{1}{\gamma_1}}}, & V_{T_i}=\sqrt{\frac{8T_i}{\pi m_a}}, \quad z_a = 0 \\ D_{p,a}, & z_a \neq 0 \end{cases} \tag{C.54}$$

C.3 Parallel momentum conservation equations

The standard form of the parallel momentum conservation equation, Eq. (C.55), as implemented in the code for the default choice of `b2sigp_style` set to `'2'`, is:

$$m_a\frac{\partial n_a V_{\|a}}{\partial t}+\frac{1}{h_z\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{h_z\sqrt{g}}{h_x}\Gamma^m_{ax}\right)+\frac{1}{h_z\sqrt{g}}\frac{\partial}{\partial y}\left(\frac{h_z\sqrt{g}}{h_y}\Gamma^m_{ay}\right)+\frac{b_x}{h_x}\frac{\partial n_a T_i}{\partial x}+Z_a e n_a\frac{b_x}{h_x}\frac{\partial\Phi}{\partial x}= \\ = S^m_{a\|}+S^m_{CF_a}+S^m_{fr_a}+S^m_{Therm_a}+S^m_{I_a}+S^m_{R_a}+S^m_{CX_a}+S^m_{AN_a}+S^m_{EIRENE_{a\|}} \tag{C.55}$$

However, one can also use the formulation given as Eq. (C.56), by setting `b2sigp_style` set to `'1'`:

$$m_a\frac{\partial n_a V_{\|a}}{\partial t}+\frac{1}{h_z\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{h_z\sqrt{g}}{h_x}\Gamma^m_{ax}\right)+\frac{1}{h_z\sqrt{g}}\frac{\partial}{\partial y}\left(\frac{h_z\sqrt{g}}{h_y}\Gamma^m_{ay}\right)+\frac{b_x}{h_x}\frac{\partial n_a T_i}{\partial x}+\frac{b_x}{h_x}\frac{Z_a n_a\partial n_e T_e}{n_e\partial x}= \\ = S^m_{a\|}+S^m_{CF_a}+S^m_{fr_a}+S^m_{Therm_a}+S^m_{I_a}+S^m_{R_a}+S^m_{CX_a}+S^m_{AN_a}+S^m_{EIRENE_{a\|}} \tag{C.56}$$

The treatment using Eq. (C.55) is recommended for cases with a single main hydrogenic ion (or at least where the main species is the lightest one) and an arbitrary amount of impurities. It also requires the current continuity equation (C.191) to be solved simulatenously for Φ (i.e. `'b2news_poteq'` set to `'1'`). For cases with hydrogenic ion mixtures or non-hydrogenic plasmas, the use of Eq. (C.56) is preferred.

C.3.1 Sources for parallel momentum equations

$$S^m_a = S^m_{a\|}+S^m_{CF_a}+S^m_{fr_a}+S^m_{Therm_a}+S^m_{I_a}+S^m_{R_a}+S^m_{CX_a}+S^m_{AN_a} \tag{C.57}$$

$$S^m_{a\|} = -\left(\nabla\cdot\overleftrightarrow{\pi}^{\|}_a\right)_{\|} = \begin{cases} -\left(\nabla\cdot\overleftrightarrow{\pi}^{V_\|}_a\right)^{res}_{\|}-\left(\nabla\cdot\overleftrightarrow{\pi}^{q_\|}_a\right)_{\|} & a=a_{main} \\ 0, & \text{otherwise} \end{cases} \tag{C.58}$$

If `'b2tral_Zhdanov_closure'` is set to `'1'` , $S^m_{a\|}$ is applied according to section C.11.

$$\left(\nabla\cdot\overleftrightarrow{\pi}^{V_\|}_a\right)^{res}_{\|} = \begin{cases} -\frac{1}{h_z\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{h_z\sqrt{g}}{h_x}\frac{4}{3}\underset{\sim}{\overset{(CL)}{\eta}}_{ax}\frac{\partial\ln\left(h_z B^{\frac{1}{2}}\right)}{B^{\frac{1}{2}}h_x\partial x}\right)B^{\frac{1}{2}}V_{\|a}, & a=a_{main} \\ 0, & \text{otherwise} \end{cases} \tag{C.59}$$

$$\left(\nabla\cdot\overleftrightarrow{\pi}^{q_\|}_a\right)_{\|} = \begin{cases} -B^{\frac{3}{2}}b_x\frac{\partial}{h_x\partial x}\left(\alpha^{NEO}\frac{b_x\tilde{\tau}_a}{B^2}\frac{\partial\left(B^{\frac{1}{2}}\tilde{q}^{(0)}_a\right)}{h_x\partial x}\right), & a=a_{main} \\ 0, & \text{otherwise} \end{cases} \tag{C.60}$$

$$S^m_{CF_a} = m_a b_x n_a V^2_{\|a} \frac{1}{h_z h_x} \frac{\partial h_z}{\partial x} \tag{C.61}$$

$S^m_{AN_a}$ is a correcting source due to the radial anomalous current:

$$S^m_{AN_a} = \begin{cases} 0, & z_a = 0 \\ -j^{(AN)}_{ay} \frac{BB_x}{B_z}, & z_a \neq 0 \end{cases} \tag{C.62}$$

Parallel momentum source due to ionisation

$$S^m_{I_a} = S^q_{I_a} + S^q_{I_{a+1}} \tag{C.63}$$

$$S^q_{I_a} = \begin{cases} -r_{I_a} n_e n_a \left(\frac{m_a + m_{a+1}}{2}\right) V_{\|a}, & a = 0, \ldots, ns-2, \quad z_a < z_{a+1} \text{ and } z_{n,a} = z_{n,a+1} \\ 0 \end{cases} \tag{C.64}$$

$$S^q_{I_{a+1}} = \begin{cases} r_{I_a} n_e n_a \left(\frac{m_a + m_{a+1}}{2}\right) V_{\|a}, & a = 0, \ldots, ns-2, \quad z_a < z_{a+1} \text{ and } z_{n,a} = z_{n,a+1} \\ 0 \end{cases} \tag{C.65}$$

Parallel momentum source due to recombination

$$S^m_{R_a} = S^q_{R_a} + S^q_{R_{a-1}} \tag{C.66}$$

$$S^q_{R_a} = \begin{cases} -r_{R_a} n_e n_a \left(\frac{m_a + m_{a-1}}{2}\right) V_{\|a}, & a = 1, \ldots, ns-1, \quad z_{a-1} < z_a \text{ and } z_{n,a-1} = z_{n,a} \\ 0 \end{cases} \tag{C.67}$$

$$S^q_{R_{a-1}} = \begin{cases} r_{R_a} n_e n_a \left(\frac{m_a + m_{a+1}}{2}\right) V_{\|a}, & a = 1, \ldots, ns-1, \quad z_{a-1} < z_a \text{ and } z_{n,a-1} = z_{n,a} \\ 0 \end{cases} \tag{C.68}$$

$$S^m_{CX_a} = S^q_{CX_a} + S^q_{CX_{a-1}} + S^q_{CX_b} + S^q_{CX_{b+1}} \tag{C.69}$$

$$S^q_{CX_a} = -\sum_{\substack{b=0 \\ b \neq a-1}}^{ns_{CX}} r_{CX_{ab}} n_a n_b \left(\frac{m_a + m_{a-1}}{2}\right) V_{a\|} \tag{C.70}$$

$$S^q_{CX_{a-1}} = \sum_{\substack{b=0 \\ b \neq a-1}}^{ns_{CX}} r_{CX_{ab}} n_a n_b \left(\frac{m_a + m_{a-1}}{2}\right) V_{a\|} \tag{C.71}$$

$$S^q_{CX_b} = -\sum_{\substack{a=0 \\ b \neq a-1}}^{ns-1} r_{CX_{ab}} n_a n_b \left(\frac{m_b + m_{b+1}}{2}\right) V_{b\|} \tag{C.72}$$

$$S^q_{CX_{b+1}} = \sum_{\substack{a=0 \\ b \neq a-1}}^{ns-1} r_{CX_{ab}} n_a n_b \left(\frac{m_b + m_{b+1}}{2}\right) V_{b\|} \tag{C.73}$$

If using Eq. (C.55), the thermal and friction forces have two parts:

$$S^m_{Therm,a} = S^m_{Therm,ea} + S^m_{Therm,ia} \tag{C.74}$$

$$S^m_{fr,a} = S^m_{fr,ea} + S^m_{fr,ia} \tag{C.75}$$

The electron contributions are:

$$S^m_{Therm,ea} = -\alpha_{ex} c_e^{(1)} \frac{n_a z_a^2 m_e}{b_x e \zeta_e} \frac{1}{c^{(Luciani)}_{\sigma_\parallel^{(CL)}}} \frac{1}{h_x} \frac{\partial T_e}{\partial x},\ z_a \neq 0 \tag{C.76}$$

$$S^m_{fr,ea} = \frac{b_x^2 e^2 z_a^2}{\sigma_\parallel z_{eff}} n_e n_a \left(V_{e\parallel} - V_{a\parallel}\right),\ z_a \neq 0 \tag{C.77}$$

and the ion contributions are:

$$S^m_{Therm,ia} = \sum_{\substack{b=0 \\ b\neq a}}^{ns-1} c^{(F\lim)}_{\alpha_{ab}} z_a^2 z_b^2 \frac{m_p}{\zeta_p e} \frac{n_a n_b}{n_a + n_b} \alpha_{ab} \frac{1}{h_x} \frac{\partial T_i}{\partial x},\ z_a \neq 0 \tag{C.78}$$

If `'b2tral_Zhdanov_closure'` is set to `'1'` , $S^m_{Therm,ia}$ is applied according to section C.11.

$$S^m_{fr,ia} = \sum_{\substack{b=0 \\ b\neq a}}^{ns-1} R^{(V_\parallel)}_{ab} = \sum_{\substack{b=0 \\ b\neq a}}^{ns-1} \frac{1}{\zeta_p} m_p \sqrt{\frac{\frac{m_a}{m_p}\frac{m_b}{m_p}}{\frac{m_a}{m_p}+\frac{m_b}{m_p}}} z_a^2 z_b^2 n_a n_b K^{(V_\parallel)}_{ab} \left(V_{b\parallel} - V_{a\parallel}\right) \tag{C.79}$$

If `'b2tral_Zhdanov_closure'` is set to `'1'` , $S^m_{fr,ia}$ is applied according to section C.11.

where

$\alpha_{ab} = \frac{\zeta_p e (n_a+n_b)}{\sqrt{m_p}} \sqrt{\frac{\frac{m_a}{m_p}\frac{m_b}{m_p}}{\frac{m_a}{m_p}+\frac{m_b}{m_p}}} \left(\frac{K^{(Therm)}_{ab}}{\kappa_b} - \frac{K^{(Therm)}_{ba}}{\kappa_a} \right) b_x,\ z_a \neq 0$

$\zeta_e = \frac{3}{4\sqrt{2\pi}} \frac{\sqrt{m_e} T_e^{\frac{3}{2}}}{\ln\Lambda} \left(\frac{4\pi\epsilon_0}{e^2}\right)^2,\ \ \tau_e = \frac{\zeta_e}{z_{eff} n_e}$

$\zeta_p = \frac{3}{4\sqrt{2\pi}} \frac{\sqrt{m_p} T_i^{\frac{3}{2}}}{\ln\Lambda} \left(\frac{4\pi\epsilon_0}{e^2}\right)^2,\ \ \tau_a = \frac{\zeta_p}{z_a^2 z_{eff} n_e} \sqrt{\frac{m_a}{m_p}}$

$c_e^{(1)} = \frac{(1+0.24 z_{eff})(1+0.93 z_{eff})}{(1+2.56 z_{eff})(1+0.29 z_{eff})},\ \ c_e^{(2)} = 1.56 \frac{(1+1.4 z_{eff})(1+0.52 z_{eff})}{(1+2.56 z_{eff})(1+0.29 z_{eff})}$

$\kappa_a = \sum_{r=0}^{ns-1} z_a^2 z_r^2 n_r \sqrt{m_p} \sqrt{\frac{\frac{m_a}{m_p}\frac{m_r}{m_p}}{\frac{m_a}{m_p}+\frac{m_r}{m_p}}}$

$$K^{(Therm)}_{rs} = \begin{cases} c^{(2)}_{imp}, & s \neq r, \quad s = s_{main}, \quad z_r \neq 0, \quad z_s \neq 0 \\ 0, & r \neq r_{main}, \quad s \neq s_{main}, \quad z_r \neq 0, \quad z_s \neq 0 \\ 0, & r = r_{main}, \quad \forall s \end{cases}$$

If `'b2npmo_impr_form_tf'` is set to `'1'` , $K^{(Therm)}_{rs}$ is applied according to section C.11.

$$K_{rs}^{(V_\parallel)} = \begin{cases} c_{imp}^{(1)}, & r = r_{main}, \quad s \neq r, \quad z_r \neq 0, \quad z_s \neq 0 \\ c_{imp}^{(1)}, & s = s_{main}, \quad r \neq s, \quad z_r \neq 0, \quad z_s \neq 0 \\ 1, & r \neq r_{main}, \quad s \neq s_{main}, \quad z_r \neq 0, \quad z_s \neq 0 \\ 0, & r = s, \quad \text{or} \quad z_r = 0, \quad z_s = 0 \end{cases}$$

If `'b2npmo_impr_form_fr'` is set to `'1'` , $K_{rs}^{(V_\parallel)}$ is applied according to section C.11.

$c_{imp}^{(1)} = \frac{(1+0.24z_{eff_imp})(1+0.93z_{eff_imp})}{(1+2.56z_{eff_imp})(1+0.29z_{eff_imp})}$, $c_{imp}^{(2)} = 1.56\frac{(1+1.4z_{eff_imp})(1+0.52z_{eff_imp})}{(1+2.56z_{eff_imp})(1+0.29z_{eff_imp})}$

$z_{eff_imp} = \frac{\sum_{\substack{a=0 \\ a\neq a_{main}}}^{ns-1} n_a z_a^2}{n_{a_{main}} z_{a_{main}}^2}$, $c_{\sigma_\parallel^{(CL)}}^{(Luciani)} = c_{\kappa_{ex}^{(CL)}}^{(Luciani)}$

$$c_{\alpha_{ab}}^{(F\,\mathrm{lim})} = \begin{cases} 1, & \text{inside the separatrix} \\ \frac{1}{1+\frac{\left|\alpha_{ab}\frac{1}{h_x}\frac{\partial T_i}{\partial x}\right|}{c_{flab} j_{ab,\max}}}, & \text{outside the separatrix} \end{cases}, \quad j_{ab,\max} = eK_{ab}^{(V_\parallel)}\sqrt{\tfrac{T_i}{m_p}}\,(n_a + n_b), \quad c_{flab} = c_{flim_4} = 0.5$$

Otherwise, when using Eq. (C.56), the thermal and friction force term expressions are:

$S_{Therm,a}^m = n_a\left(\frac{z_a}{n_e} - \frac{z_a^2}{n_{e^2}}\right) b_x e\left[\hat{E}_x - \frac{n_e}{e}\left(c_{th_e,a}\Delta_{T_e} + c_{th_i,a}\Delta_{T_i}\right)\right]$,

$$S_{fr,a}^m = \sum_{\substack{b=0 \\ b\neq a, z_a\neq 0, z_b\neq 0}}^{ns-1} \frac{1}{\tau_p} m_p \sqrt{\frac{\frac{m_a}{m_p}\frac{m_b}{m_p}}{\frac{m_a}{m_p}+\frac{m_b}{m_p}}} z_a^2 z_b^2 n_b n_a \left(V_{b\parallel} - V_{a\parallel}\right) \tag{C.80}$$

where

$\tau_p = \frac{3}{4\sqrt{2\pi}}\frac{\sqrt{m_p}T_i^{\frac{3}{2}}}{\ln\Lambda}\left(\frac{4\pi\varepsilon_0}{e^2}\right)^2$, where the Coulomb logarithm $\ln\Lambda$ can be chosen as

$$\ln\Lambda = \begin{cases} 12 & \\ \max\{5; 23.4 - 1.15\cdot\log_{10}\left(\frac{n_e}{10^6}\right) + 3.45\cdot\log_{10}\left(\frac{T_e}{e}\right)\}, & \frac{T_e}{e} \leq 50 \\ \max\{5; 25.3 - 1.15\cdot\log_{10}\left(\frac{n_e}{10^6}\right) + 2.30\cdot\log_{10}\left(\frac{T_e}{e}\right)\}, & \frac{T_e}{e} > 50 \\ \max\{5; 17.3 - 0.5\cdot\ln\left(\frac{n_e}{10^{20}}\right) + 1.5\cdot\ln\left(\frac{T_i}{e\cdot 10^3}\right)\} & \end{cases} \begin{array}{l} \text{Braginskii formula} \\ \\ \text{Wesson formula} \\ \text{for ion-ion collisions} \end{array} \tag{C.81}$$

$$n_{e^2} = \sum_{a=0}^{ns-1} z_a^2 n_a, \quad l_{t_e} = 0.3; \quad l_{t_i} = 0.3; \quad z_{\lambda_e} = \frac{1.5\cdot 10^{16}}{n_e}\left(\frac{T_e}{e}\right)^2, \quad z_{\lambda_i} = \frac{1.5\cdot 10^{16}}{n_{e^2}}\left(\frac{T_i}{e}\right)^2$$

$$\Delta_{T_e} = \begin{cases} \frac{\partial T_e}{h_x \partial x}, & l_{t_e} \leq 0 \\ \left|\min\left\{\left|\frac{\partial T_e}{h_x\partial x}\right|, \frac{l_{t_e}T_e}{z_{\lambda_e}|b_x|}\right\}\right|, & \frac{\partial T_e}{h_x\partial x} \geq 0, \quad l_{t_e} > 0 \\ -\left|\min\left\{\left|\frac{\partial T_e}{h_x\partial x}\right|, \frac{l_{t_e}T_e}{z_{\lambda_e}|b_x|}\right\}\right|, & \frac{\partial T_e}{h_x\partial x} < 0, \quad l_{t_e} > 0 \end{cases} \tag{C.82}$$

$$\Delta_{T_i} = \begin{cases} \frac{\partial T_i}{h_x \partial x}, & l_{t_i} \leq 0 \\ \left|\min\left\{\left|\frac{\partial T_i}{h_x\partial x}\right|, \frac{l_{t_i}T_i}{z_{\lambda_i}|b_x|}\right\}\right|, & \frac{\partial T_i}{h_x\partial x} \geq 0, \quad l_{t_i} > 0 \\ -\left|\min\left\{\left|\frac{\partial T_i}{h_x\partial x}\right|, \frac{l_{t_i}T_i}{z_{\lambda_i}|b_x|}\right\}\right|, & \frac{\partial T_i}{h_x\partial x} < 0, \quad l_{t_i} > 0 \end{cases} \tag{C.83}$$

$c_{th_e,a} = 0, a = 0, \ldots, ns-1$ and $c_{th_i,a} = 2.65, a = 0, \ldots, ns-1$ if the 21-moment approximation is not used.

Neoclassical corrections for high Z impurity transport

These corrections (see [55] for a full derivation) are based on the Zhdanov-Grad module (see section C.11), so `'b2tral_Zhdanov_closure'` should be set to `'1'`.

If `'b2tral_Zhdanov_nc'` is set to `'1'`, neoclassical corrections for impurity transport are applied for a chosen impurity isonuclear sequence, `'b2tral_Zhdanov_nc_spec'`, e.g. for Ne in case of a `D+Ne` plasma `'b2tral_Zhdanov_nc_spec'` should be set to `'2'`. The corrections are more accurate for impurity species with at least Z=10.

`'b2tral_Zhdanov_nc_min'` sets Z_a^{min} ('6' by default). Lower values are not recommended. The corrections are applied for the ion thermal force (see Eq. (C.78)) and friction terms (see Eq. (C.79)) as follows:

$$S^m_{Therm,ia} = S^m_{Therm,ia} \cdot c^{tf}_a \tag{C.84}$$

$$S^m_{fr,ia} = S^m_{fr,ia} \cdot c^{fr}_a \tag{C.85}$$

$$c^{tf}_a = \begin{cases} \frac{c_{0,a} * c_{1,a} + c_{2,a}}{K_2}, & \text{if } Z_a \geq Z_a^{min} \\ 1, & \text{for all other non-main ion species} \\ \frac{1}{Z^2_{\beta_I} n_{\beta_I}} \sum_{b, b \neq b_{main}} Z_b^2 n_b c_b^{tf}, & \text{for main ions} \end{cases} \tag{C.86}$$

$$c^{fr}_a = \begin{cases} 1 + \frac{0.01 - 0.36 \cdot \alpha_{H,a} + 0.44 \cdot \alpha^{0,77}_{H,a}}{1+g^2}, & \text{if } Z_a \geq Z_a^{min} \\ 1, & \text{for all other non-main ion species} \\ \frac{1}{Z^2_{\alpha_I} n_{\alpha_I}} \sum_{b, b \neq b_{main}} Z_b^2 n_b c_b^{fr}, & \text{for main ions} \end{cases} \tag{C.87}$$

$$c_{0,a} = \frac{\sqrt{2} c^{hTA,an}_{\beta_{main}}}{2.5 c^{RTA,an}_{\beta_{main},\alpha Z}} \tag{C.88}$$

$$c_{1,a} = \frac{2.4 \varepsilon^{1.5}}{1 + (0.36 + 0.4 \alpha_{H,a}) g^2} \tag{C.89}$$

$$c_{2,a} = \frac{1.6 \varepsilon^3 g^2}{1 + \varepsilon^{1.5} g^2} \tag{C.90}$$

$$\alpha_{H,a} = \frac{Z_a^2 n_a}{Z^2_{a_{main}} n_{a_{main}}} \tag{C.91}$$

$$g = \frac{\nu_* \varepsilon^{1.5}}{Z_{eff}} \tag{C.92}$$

Quantities (C.86)-(C.92) are printed, if `'b2sifrtf_iout'` is set to `'1'`. Variables such as ν_* and ε are described in Section C.3.3. Overline averaged values and $c^{RTA,an}_{\beta,\alpha}$, $c^{hTA,an}_{\beta}$ are described in section C.11.

If `'b2sifr_Zhdanov_test'` is set to `'1'`, the Zhdanov-Grad variables are calculated but used only for the friction and thermal force. Additionally, the velocity-dependent ion heat flux, see Eq. (C.317), is included. Other parts of the Zhdanov-Grad module are not used.

C.3.2 Parallel momentum fluxes

$$\Gamma^m_{ax} = \begin{cases} m_a V_{\|a} \Gamma^{Cor}_{ax} + \frac{4}{3} \widetilde{\widetilde{\eta}}^{(CL)}_{ax} \frac{\partial \ln h_z}{h_x \partial x} V_{\|a} - \eta_{ax} \frac{\partial V_{\|a}}{h_x \partial x}, & a = a_{main} \\ m_a V_{\|a} \Gamma^{Cor}_{ax} - \eta_{ax} \frac{\partial V_{\|a}}{h_x \partial x}, & \text{otherwise} \end{cases} \tag{C.93}$$

If `'b2tral_Zhdanov_closure'` is set to `'1'` , Γ^m_{ax} is applied according to section C.11.

$$\Gamma^m_{ay} = m_a V_{\|a} \Gamma^{Cor}_{ay} - \eta_{ay} \frac{\partial V_{\|a}}{h_y \partial y} \tag{C.94}$$

C.3.3 Parallel momentum transport coefficients

$$\eta_{ax} = \eta^{(AN)}_a + \frac{4}{3} \widetilde{\widetilde{\eta}}^{(CL)}_{ax} \tag{C.95}$$

$$\eta^{(AN)}_a = m_a n_a \nu^{visc}_a = \left(c_{\eta_a,0} + c_{\eta_a,1} \frac{10^{20}}{n_e} + c_{\eta_a,2} D_{f,a} + c_{\eta_a,3} \frac{T_e}{16e|B|} \right) m_a n_a f_{\eta_a}(x,y) \tag{C.96}$$

The $c_{\eta_a,x}$ coefficients correspond to the `cfvsa` array elements provided in the `b2ah.dat` file, or, if present, in the `b2cmpt` block of the `b2mn.dat` file. For charged species, they can be overwritten by means of the `b2.transport.parameters` file, see Section 3.5.3. Typical values are:

$a = 0, \quad c_{\eta_a,0} = 0, \quad c_{\eta_a,1} = 0, \quad c_{\eta_a,2} = 1, \quad c_{\eta_a,3} = 0$

$a = 1, \quad c_{\eta_a,0} = 0, \quad c_{\eta_a,1} = 0, \quad c_{\eta_a,2} = 1, \quad c_{\eta_a,3} = 0$

The f_{η_a} function is an optional modifier, governed by the value of `b2tqna_user_transport`. If $c_{\eta_a,7} \neq 0$, then f_{η_a} contains a flux-scaling factor of $\left(\frac{h_y}{h_{y,ref}} \right)^2$.

If `b2tqna_user_transport` is `6`, a transport barrier can be applied in a parametric fashion, according to the value of `b2tqna_itype`, and the coefficients `b2tqna_vs_alfa[1-4]` are used to modify f_{η_a}. See code for details.

$$\widetilde{\widetilde{\eta}}^{(CL)}_{ax} = c_{\eta^{(CL)}_{ax}} \widetilde{\eta}^{(CL)}_{ax} \tag{C.97}$$

$$c_{\eta^{(CL)}_{ax}} = \begin{cases} 1, & z_a \neq 0 \text{ and only for inner flux surfaces} \\ \frac{1}{1 + \frac{\frac{\sqrt{g}}{h_x} \widetilde{\eta}^{(CL)}_{ax} \left| \frac{\partial V_{\|a}}{h_x \partial x} \right|}{V^{(\max)}_{m_a}}}, & \text{for all other cases} \end{cases} \tag{C.98}$$

$$V^{(\max)}_{m_a} = c_{flim_2} \frac{\sqrt{g}}{h_x} |b_x| n_a T_i \, , \, c_{flim_2} = \frac{3}{8} \tag{C.99}$$

$$\widetilde{\eta}^{(CL)}_{ax} = c^{Luciani}_{\eta^{(CL)}} \eta^{(CL)}_{ax} \tag{C.100}$$

If `'b2trcl_lluciani'` is set to `'1'`, the Luciani flux limiter is given by:

$$c^{Luciani}_{\eta^{(CL)}} = \begin{cases} \frac{1}{1 + \frac{7.5 \cdot 10^{16} \left(\frac{T_i}{e} \right)^2}{c(y) \sum_{a=0}^{ns-1} z_a^2 n_a}}, & \text{for ions on inner flux surfaces} \\ 1, & \text{for all other cases} \end{cases}$$

For the plateau and banana transport regimes, the correction, with `'b2trcl_lluciani'` set to `'3'` (default), is given by:

if `'b2trcl_luc_et_form'` set to `'1'` (default):

$$c_{\eta^{(CL)}}^{Luciani} = \begin{cases} \frac{1}{1+\frac{1}{\varepsilon^{3/2}\nu_*}} \cdot \frac{1}{1+\frac{1}{\nu_*}}, & \text{for ions on inner flux surfaces} \\ 1, & \text{for all other cases} \end{cases}$$

if `'b2trcl_luc_et_form'` set to `'0'` (only valid for hydrogen-majority plasmas):

$$c_{\eta^{(CL)}}^{Luciani} = \begin{cases} \frac{1}{1+\frac{15.12\cdot 10^{16}\left(\frac{T_i}{e}\right)^2}{c(y)\sum_{a=0}^{ns-1} z_a^2 n_a}} \cdot \frac{1}{1+\frac{15.12\cdot 10^{16}\cdot\varepsilon^{3/2}\left(\frac{T_i}{e}\right)^2}{c(y)\sum_{a=0}^{ns-1} z_a^2 n_a}}, & \text{for ions on inner flux surfaces} \\ 1, & \text{for all other cases} \end{cases}$$

$$\nu_* = \frac{c(y)}{2\pi\varepsilon^{3/2}\tau_{main}^{(Br)}\sqrt{T_i/m_{main}}}, \quad c(y) = \oint h_x |b_x|^{-1} dx, \quad \varepsilon = \frac{\left(\oint h_x dx\right)^2}{\oint \sqrt{g}/h_y dx}, \quad \tau_a^{(Br)} = \frac{3}{4\sqrt{\pi}} \frac{\sqrt{m_a} T_i^{\frac{3}{2}}}{z_a^2 z_{eff} n_e \ln\Lambda} \left(\frac{4\pi\epsilon_0}{e^2}\right)^2$$

If `'b2tral_Zhdanov_closure'` is set to `'1'` , ν_* is applied according to (C.295).

$$\eta_{ax}^{(CL)} = \begin{cases} 0, z_a = 0 \\ \eta\sqrt{2} T_i n_a \tau_a b_x^2, z_a \neq 0 \end{cases}, \eta = 0.96 \tag{C.101}$$

$$\tau_a = \frac{3}{4\sqrt{2\pi}} \frac{\sqrt{m_a} T_i^{\frac{3}{2}}}{z_a^2 z_{eff} n_e \ln\Lambda} \left(\frac{4\pi\varepsilon_0}{e^2}\right)^2, \tag{C.102}$$

where the Coulomb logarithm $\ln\Lambda$ can be chosen as

$$\ln\Lambda = \begin{cases} 12 & \\ \max\{5; 23.4 - 1.15\cdot\log_{10}\left(\frac{n_e}{10^6}\right) + 3.45\cdot\log_{10}\left(\frac{T_e}{e}\right)\}, & \frac{T_e}{e} \leq 50 \\ \max\{5; 25.3 - 1.15\cdot\log_{10}\left(\frac{n_e}{10^6}\right) + 2.30\cdot\log_{10}\left(\frac{T_e}{e}\right)\}, & \frac{T_e}{e} > 50 \\ \max\{5; 17.3 - 0.5\cdot\ln\left(\frac{n_e}{10^{20}}\right) + 1.5\cdot\ln\left(\frac{T_i}{e\cdot 10^3}\right)\} & \end{cases} \begin{matrix} \text{Braginskii formula} \\ \\ \text{Wesson formula} \\ \text{for ion-ion collisions} \end{matrix}$$

$\eta_{ay} = \eta_a^{(AN)} + \eta_{ay}^{(CL)}, \eta_{ay}^{(CL)} = 0$

$\widetilde{q}_a^{(0)} = q^{(1)} \arctan\left(\frac{q_a^{(0)}}{q^{(1)}}\right), \; q^{(1)} = c_{q1}\widetilde{n}_i T_i \sqrt{\frac{T_i+T_e}{m_i}}$

$c_{q1} = 0.5$ or `'b2siav_cqip1'` set to `'0.1'` in `b2mn.dat`

The three-point smoothing procedure is imposed twice on $\widetilde{q}^{(0)}$ along the x direction.

In case `'b2siav_style_qip'` is set to `'0'` in `b2mn.dat`:

$$q_a^{(0)} = \begin{cases} 0 & \text{outside the separatrix} \\ \frac{B}{B_x}\widetilde{q}_{ax}^{(dia)} & \text{inside the separatrix} \end{cases} \tag{C.103}$$

In case `'b2siav_style_qip'` is set to `'1'` in `b2mn.dat`:

$$q_a^{(0)} = \begin{cases} 0 & \text{outside the separatrix} \\ q_{a\|}^{(0)} + \frac{B}{B_x} q_{ax}^{(dia)} & \text{inside the separatrix} \end{cases} \tag{C.104}$$

$q_{a\|}^{(0)} = -\frac{\widetilde{\widetilde{k}}_{ix}^{(CL)}}{b_x} \frac{\partial T_i}{h_x \partial x}$

Note that $\widetilde{\widetilde{k}}_{ix}^{(CL)}$ already contains a factor of b_x^2, see its definition below Eq. (C.181).

$q_{ax}^{(dia)} = -\frac{5}{2}\frac{B_z}{B^2}\frac{\widetilde{n}_i T_i}{e}\frac{\partial T_i}{h_y \partial y}$

$\widetilde{q}_{ax}^{(dia)} = -\frac{5}{2}\frac{B_z}{\langle B^2\rangle}\frac{\widetilde{n}_i T_i}{e}\frac{\partial T_i}{h_y \partial y}$

$\widetilde{\tau}_a = \frac{\tau_a^{(Br)}}{1+\frac{15.12\cdot 10^{16}\left(\frac{T_i}{e}\right)^2}{\overset{c(y)}{\underset{}{\int}}\sum_{a=0}^{ns-1} z_a^2 n_a}} \cdot \frac{1}{1+\frac{15.12\cdot 10^{16}\cdot\varepsilon^{3/2}\left(\frac{T_i}{e}\right)^2}{\overset{c(y)}{\underset{}{\int}}\sum_{a=0}^{ns-1} z_a^2 n_a}},$

$\tau_a^{(Br)} = \frac{3}{4\sqrt{\pi}}\frac{\sqrt{m_a}T_i^{\frac{3}{2}}}{z_a^2 z_{eff} n_e \ln\Lambda}\left(\frac{4\pi\epsilon_0}{e^2}\right)^2$

This correction is compatible **only** with plateau and banana correction ('b2trcl_lluciani' set to '3').

$\alpha^{NEO} = \frac{8}{15}\left(\frac{-0.17+1.05\nu_*^{1/2}+2.7\nu_*^2\varepsilon^3}{1+0.7\nu_*^{1/2}+\nu_*^2\varepsilon^3} - 1\right)$ gives the correct coefficient before the temperature gradient in the neo-classical electric field in the plateau and banana regime

$\nu_* = \frac{c(y)}{2\pi\varepsilon^{3/2}\tau_{main}^{(Br)}\sqrt{T_i/m_{main}}}, \quad c\left(y\right) = \oint h_x |b_x|^{-1} dx, \quad \varepsilon = \frac{\left(\oint h_x dx\right)^2}{\oint \sqrt{g}/h_y dx}$

C.4 Electron energy equations

$$\frac{3}{2}\frac{\partial n_e T_e}{\partial t} + \frac{1}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x}\widetilde{q}_{ex}\right) + \frac{1}{\sqrt{g}}\frac{\partial}{\partial y}\left(\frac{\sqrt{g}}{h_y}\widetilde{q}_{ey}\right) + \frac{n_e T_e}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x}b_x V_{e\|}\right) =$$
$$= Q_e + c_{\mathbf{E}\times\mathbf{B}} n_e T_e B_z \frac{1}{h_x h_y}\left(\frac{\partial\Phi}{\partial y}\frac{\partial}{\partial x}\left(\frac{1}{B^2}\right) - \frac{\partial\Phi}{\partial x}\frac{\partial}{\partial y}\left(\frac{1}{B^2}\right)\right) - \frac{j_y^{(ST)}}{e n_e}\frac{\partial n_e T_e}{h_y \partial y} + Q_{Fei} + Q_e^{\text{EIRENE}} \quad \text{(C.105)}$$

C.4.1 Electron energy sources

$$S_e^{heat} = Q_e + c_{\mathbf{E}\times\mathbf{B}} n_e T_e B_z \frac{1}{h_x h_y}\left(\frac{\partial\Phi}{\partial y}\frac{\partial}{\partial x}\left(\frac{1}{B^2}\right) - \frac{\partial\Phi}{\partial x}\frac{\partial}{\partial y}\left(\frac{1}{B^2}\right)\right) - \frac{j_y^{(ST)}}{e n_e}\frac{\partial n_e T_e}{h_y \partial y} + Q_{Fei} -$$
$$- \frac{n_e T_e}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x}b_x V_{e\|}\right) + Q_e^{\text{EIRENE}} \quad \text{(C.106)}$$

$Q_e = -Q_\Delta - Q_R$

$$Q_\Delta = c_{eqp}\ln\Lambda e^{\frac{3}{2}}\frac{\sum_{a=0}^{ns-1}\frac{z_a^2 m_p}{m_a} n_a}{n_i}\frac{n_e n_i}{T_e^{\frac{3}{2}}}(T_e - T_i), \quad c_{eqp} = 4.8\cdot 10^{-15}, \quad \text{(C.107)}$$

where $\ln\Lambda$ can be chosen as

$$\ln\Lambda = \begin{cases} 12 & \text{Braginskii formula} \\ \max\{5; 23.4 - 1.15\cdot\log_{10}\left(\frac{n_e}{10^6}\right) + 3.45\cdot\log_{10}\left(\frac{T_e}{e}\right)\}, & \frac{T_e}{e} \le 50 \\ \max\{5; 25.3 - 1.15\cdot\log_{10}\left(\frac{n_e}{10^6}\right) + 2.30\cdot\log_{10}\left(\frac{T_e}{e}\right)\}, & \frac{T_e}{e} > 50 \\ \max\{5; 15.2 - 0.5\cdot\ln\left(\frac{n_e}{10^{20}}\right) + \ln\left(\frac{T_e}{e\cdot 10^3}\right)\} & \text{Wesson formula for electron-ion collisions} \end{cases} \quad \text{(C.108)}$$

$$Q_R = \sum_{a=0}^{ns-1} r_{Q_a} n_e n_a \text{ from b2stel} \tag{C.109}$$

$$r_{Q_a} = r_{Q_a}^{\text{ion}} + r_{Q_a}^{\text{rec}} + r_{Q_a}^{\text{brm}} + r_{Q_a}^{\text{line}} \tag{C.110}$$

contains contributions from ionisation, recombination, bremsstrahlung and line radiation.

$$Q_{Fei} = \begin{cases} V_{e\|} \sum\limits_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} \left(S_{fr,ea}^m + S_{Therm,ea}^m \right), \text{ when Eq. (C.55) is activated (`b2sigp_style` set to `2`)} \\ \\ j_x \hat{E}_x + 0 \cdot j_y \hat{E}_y, \text{ when Eq. (C.56) is activated} \end{cases} \tag{C.111}$$

If `b2tral_Zhdanov_closure` is set to `1` , Q_{Fei} is applied according to section C.11.

C.4.2 Poloidal electron energy fluxes

$$\widetilde{q}_{ex} = \widetilde{f}_{ex} T_e - \kappa_{ex} \frac{1}{h_x} \frac{\partial T_e}{\partial x} + 0 \cdot \alpha_{ex} T_e \hat{E}_x - q_{ex}^{P.Sch} \tag{C.112}$$

$$\widetilde{f}_{ex} = \frac{3}{2} \Gamma_{ex} + V_{ex}^{(str)} n_e - c_{071} \frac{j_x^{(\|)}}{e} \tag{C.113}$$

$c_{071} = z_{eff} \widetilde{c_e}^{(2)}$

$$q_{ex}^{P.Sch} = -\frac{5}{2} D_{exy} \frac{1}{h_y} \frac{\partial n_e T_e^2}{\partial y} \tag{C.114}$$

$D_{exy} = \left(\frac{1}{B^2} - \frac{1}{\langle B^2 \rangle} \right) \frac{B_z}{e}$

$D_{eyx} = \left(\frac{1}{B^2} - \frac{1}{\langle B^2 \rangle} \right) \frac{B_z}{e}$, where $\langle B^2 \rangle = \frac{\iint_{core} B^2 \sqrt{g} dx dy}{\iint_{core} \sqrt{g} dx dy}$

$$q_{ex} = \left[\frac{3}{2} \Gamma_{ex} - c_{071} \frac{j_x^{(\|)}}{e} - \frac{5}{2} n_e T_e \frac{B_z}{e} \frac{\partial}{h_y \partial y} \left(\frac{1}{B^2} \right) + V_{ex}^{(str)} n_e \right] T_e - \kappa_{ex} \frac{1}{h_x} \frac{\partial T_e}{\partial x} + 0 \cdot \alpha_{ex} T_e \hat{E}_x \tag{C.115}$$

$$\Gamma_{ex} = \sum_{a=0}^{ns-1} z_a \Gamma_{ax}^{(he)} - \frac{j_x^{(\|)}}{e} \tag{C.116}$$

$$\Gamma_{ax}^{(he)} = \left(b_x V_{a\|} + V_{ax}^{(E)} + \frac{5}{3} V_a^{(AN)} + V_a^{corr_dpc} \right) n_a - \frac{5}{3} \left[D_{n,a} \frac{1}{h_x} \frac{\partial n_a}{\partial x} + \widetilde{D}_{p,ax} \frac{1}{h_x} \frac{\partial p_a}{\partial x} \right] \tag{C.117}$$

$$q_{ex}^{(Eir)} = \left[\frac{5}{2} \Gamma_{ex}^{(Eir)} + V_{ex}^{(str)} n_e - c_{071} \frac{j_\|}{e} \right] T_e - \kappa_{ex} \frac{1}{h_x} \frac{\partial T_e}{\partial x} \tag{C.118}$$

$$q_{ey}^{(Eir)} = \left[\frac{5}{2} \Gamma_{ex}^{(Eir)} + V_{ey}^{(str)} n_e \right] T_e - \kappa_{ey} \frac{1}{h_y} \frac{\partial T_e}{\partial y} \tag{C.119}$$

where

$$\Gamma_{ex}^{(Eir)} = \left(b_x V_{\|e} + V_{ex}^{(E)} + V_{ex}^{(dia)} + V_{ex}^{(AN)}\right) n_e - \sum_{a=0}^{ns-1} D_{n,a} \frac{1}{h_x} \frac{\partial z_a n_a}{\partial x} \tag{C.120}$$

$$\Gamma_{ey}^{(Eir)} = \left(V_{ey}^{(E)} + V_{ey}^{(dia)} + V_{ey}^{(AN)}\right) n_e - \sum_{a=0}^{ns-1} D_{n,a} \frac{1}{h_y} \frac{\partial z_a n_a}{\partial y}$$

$$V_{ex}^{(dia)} = \frac{B_z}{eB^2 n_e} \frac{\partial n_e T_e}{h_y \partial y} \tag{C.121}$$

$$V_{ey}^{(dia)} = -\frac{B_z}{eB^2 n_e} \frac{\partial n_e T_e}{h_x \partial x} \tag{C.122}$$

$V_{ex}^{(str)} = 0$

The parameter $\widetilde{c_e}^{(2)}$ is given by $\widetilde{c_e}^{(2)} = \frac{c_e^{(2)}}{\left(z_{eff}+\frac{\sqrt{2}}{2}\right)} = 1.56 \frac{(1+1.4z_{eff})(1+0.52z_{eff})}{(1+2.56z_{eff})(1+0.29z_{eff})} \frac{1}{\left(z_{eff}+\frac{\sqrt{2}}{2}\right)}$ when `'b2sigp_style'` is set to `'2'` (default) or by its limiting value at $z_{eff} = 1$ of 0.71 otherwise.

C.4.3 Electron energy transport

$$\kappa_{ex} = \kappa_e^{(AN)} + \widetilde{\widetilde{\kappa}}_{ex}^{(CL)} \tag{C.123}$$

$$\kappa_e^{(AN)} = \chi_e^{(AN)} n_e \tag{C.124}$$

$$\chi_e^{(AN)} = \left(c_{h_e,0} + c_{h_e,1} \frac{10^{20}}{n_e} + c_{h_e,3} \frac{T_e}{16e|B|}\right) f_{h_e}(x,y) \tag{C.125}$$

The $c_{h_e,x}$ coefficients correspond to the `cfhce` array elements provided in the `b2ah.dat` file, or, if present, in the `b2cmpt` block of the `b2mn.dat` file. They can be overwritten by means of the `b2.transport.parameters` file, see Section 3.5.3. A typical choice may be:

$c_{h_e,0} = 1.2, \quad c_{h_e,1} = 0, \quad c_{h_e,3} = 0$

The f_{h_e} function is an optional modifier, governed by the value of `b2tqna_user_transport`. If $c_{h_e,7} \neq 0$, then f_{h_e} contains a flux-scaling factor of $\left(\frac{h_y}{h_{y,ref}}\right)^2$.

If `b2tqna_user_transport` is `1`, the $\chi_e^{(AN)}$ coefficient is scaled so as to maintain the local T_e gradient at the reference transport location to `set_transport_required_te_gradient`.

If `b2tqna_user_transport` is `2`, an ELM model is applied, with three levels of transport scaling according to radial location and alternating between inter-ELM and intra-ELM values. See code for details.

If `b2tqna_user_transport` is `3`, use parameters from the `b2.transport_models_save.parameters` namelist to maintain a constant $\frac{\nabla T_e}{\nabla n_e}$ ratio at the reference location via feedback.

If `b2tqna_user_transport` is `6`, a transport barrier can be applied in a parametric fashion, according to the value of `b2tqna_itype`, and the coefficients `b2tqna_te_alfa[1-4]` are used to modify f_{h_e}. See code for details.

If `b2tqna_user_transport` is `7`, the disruption model is applied, adding $1000 f_{quench} \frac{n_{e,ref}}{n_e} \left(\frac{T_e}{1000\text{eV}}\right)^{\frac{5}{2}} \left(\frac{h_y}{h_{y,ref}}\right)^2$ to $\chi_e^{(AN)}$.

If `b2tqna_user_transport` is `8`, the neoclassical transport model is applied. See code for details.

$$\widetilde{\widetilde{\kappa}}_{ex}^{(CL)} = c_{\kappa_{ex}}^{(F\lim)} \widetilde{\kappa}_{ex}^{(CL)} \tag{C.126}$$

$$c_{\kappa_{ex}}^{F\lim} = \begin{cases} 1, & \text{only for inner flux surfaces} \\ \frac{1}{1+\frac{\frac{\sqrt{g}}{h_x}\widetilde{\kappa}_{ex}^{(CL)}\left|\frac{\partial T_e}{h_x \partial x}\right|}{V_{h_e}^{(\max)} n_e T_e}} & \text{for other flux surfaces} \end{cases} \tag{C.127}$$

$$V_{h_e}^{(\max)} = c_{flim_0} \frac{\sqrt{g}}{h_x} |b_x| \sqrt{\frac{T_e}{m_e}} \tag{C.128}$$

$$\widetilde{\kappa}_{ex}^{(CL)} = c_{\kappa_{ex}^{(CL)}}^{(Luciani)} \kappa_{ex}^{(CL)} b_x^2 \tag{C.129}$$

$c_{flim_0} = 0.15$ is the electron heat flux limiter coefficient.

The classical electron thermal diffusivity is expressed as: $\kappa_{ex}^{(CL)} = \frac{T_e}{m_e} f_{k_e}(z_{eff}) n_e \tau_e$ (Zhdanov formulation, default, `'b2tqce_fke_Zhdanov'` set to `'1'`), with $f_{k_e}(z_{eff}) = \frac{3.9+2.3/z_{eff}}{0.31+1.2/z_{eff}+0.41/z_{eff}^2}$, or $\kappa_{ex}^{(CL)} = \frac{5}{2}\frac{T_e}{m_e} f_{k_e}(z_{eff}) n_e \tau_e$ (Balescu formulation, `'b2tqce_fke_Zhdanov'` set to `'0'`), with $f_{k_e}(z_{eff}) = \frac{2.16 z_{eff}}{1+0.27 z_{eff}}$ and $z_{eff} = \frac{1}{n_e}\sum_{a=0}^{ns-1} z_a^2 n_a$.

If the plateau and banana transport regimes correction is applied (default, `'b2trcl_lluciani'` set to `'3'`), one has:

$$c_{\kappa_{ex}^{(CL)}}^{(Luciani)} = \begin{cases} 0.58\varepsilon^{\frac{3}{2}}/K_1, & \text{only for inner flux surfaces} \\ 1, & \text{for other flux surfaces} \end{cases}$$

If `'b2trcl_luciani'` is set to `'1'`, one gets instead:

$$c_{\kappa_{ex}^{(CL)}}^{(Luciani)} = \begin{cases} \frac{1}{1+\frac{7.5\cdot 10^{16}\left(\frac{T_e}{e}\right)^2}{c(y)\sum_{a=0}^{ns-1} z_a^2 n_a}}, & \text{only for inner flux surfaces} \\ 1, & \text{for all other cases} \end{cases}$$

If `'b2trcl_luciani'` is set to `'2'`, the **SOLPS 4.x** formulation is used:

$$c_{\kappa_{ex}^{(CL)}}^{(Luciani)} = \begin{cases} \frac{1}{1+\frac{0.3\cdot 10^{16}\left(\frac{T_e}{e}\right)^2}{\sum_{a=0}^{ns-1} z_a^2 n_a}}, & \text{only for inner flux surfaces} \\ 1, & \text{for all other cases} \end{cases}$$

$$K_1 = \frac{1}{1+2.01\cdot\nu_{*e}^{1/2}+1.53\cdot\nu_{*e}} + \frac{0.52\cdot\varepsilon^3\cdot\nu_{*e}}{1+0.89\cdot\varepsilon^{3/2}\cdot\nu_{*e}}$$

$$\nu_{*e} = \frac{c(y)}{2\pi\varepsilon^{3/2}\tau_e\sqrt{T_e/m_e}} \tag{C.130}$$

$$\tau_e = \frac{3}{4\sqrt{2\pi}} \frac{\sqrt{m_e} T_e^{\frac{3}{2}}}{\ln\Lambda \sum_{a=0}^{ns-1} z_a^2 n_a} \left(\frac{4\pi\varepsilon_0}{e^2}\right)^2 \tag{C.131}$$

where the Coulomb logarithm $\ln\Lambda$ can be chosen as

$$\ln\Lambda = \begin{cases} 12 & \\ \max\{5; 23.4 - 1.15\cdot\log_{10}\left(\frac{n_e}{10^6}\right) + 3.45\cdot\log_{10}\left(\frac{T_e}{e}\right)\}, & \frac{T_e}{e} \leq 50 \\ \max\{5; 25.3 - 1.15\cdot\log_{10}\left(\frac{n_e}{10^6}\right) + 2.30\cdot\log_{10}\left(\frac{T_e}{e}\right)\}, & \frac{T_e}{e} > 50 \\ \max\{5; 14.9 - 0.5\cdot\ln\left(\frac{n_e}{10^{20}}\right) + \ln\left(\frac{T_e}{e\cdot 10^3}\right)\} & \end{cases} \begin{array}{c} \\ \text{Braginskii formula} \\ \\ \text{Wesson formula for} \\ \text{electron-electron collisions} \end{array} \tag{C.132}$$

$$c\left(y\right) = \begin{cases} \int \frac{h_x(x,y)}{|b_x|} dx, & \text{for inner flux surfaces} \\ 0, & \text{for other} \end{cases}$$

Electron parallel velocity

$$V_{e\|} = \frac{1}{en_e}\left[\sum_{a=0}^{ns-1} z_a e n_a V_{a\|} - \frac{j_x^{(\|)}}{b_x}\right] \tag{C.133}$$

$$j_x^{(\|)} = \sigma_\| \left(\frac{1}{en_e h_x}\frac{\partial n_e T_e}{\partial x} - \frac{1}{h_x}\frac{\partial \Phi}{\partial x}\right) - \alpha_{ex}\frac{1}{h_x}\frac{\partial T_e}{\partial x} - \frac{eb_x}{z_{eff}}\sum_{a=0}^{ns-1} n_a V_{a\|}\left(z_a^2 - z_a z_{eff}\right) \tag{C.134}$$

The last term in Eq. (C.134) is activated with the switch `b2sigp_style` set to `'2'`.

$$\hat{E}_x = \frac{1}{en_e h_x}\frac{\partial n_e T_e}{\partial x} - \frac{1}{h_x}\frac{\partial \Phi}{\partial x} \tag{C.135}$$

$$\sigma_\| = \sigma_\|^{(AN)} + \widetilde{\sigma}_\|^{(CL)} \tag{C.136}$$

$\sigma_\|^{(AN)} = 0$

$$\widetilde{\sigma}_\|^{(CL)} = c_{\kappa_{ex}^{(CL)}}^{(Luciani)} \sigma_\|^{(CL)} \tag{C.137}$$

The classical parallel conductivity is given as:

$$\sigma_\|^{(CL)} = b_x^2 \frac{e^2}{m_e} f_{\sigma_e}\left(z_{eff}\right) n_e \tau_e \tag{C.138}$$

where the electron form factor is $f_{\sigma_e}\left(z_{eff}\right) = \frac{1}{c_e^{(1)}}$ when `'b2sigp_style'` is set to `'2'` (Zhdanov formulation), and $f_{\sigma_e}\left(z_{eff}\right) = \frac{4.95 z_{eff}}{1+1.54 z_{eff}}$ otherwise (Balescu formulation). The parameter $c_e^{(1)}$ is given by:
$c_e^{(1)} = \frac{(1+0.24 z_{eff})(1+0.93 z_{eff})}{(1+2.56 z_{eff})(1+0.29 z_{eff})}$. See also Section B.5.

$$\alpha_{ex} = \alpha_x^{(AN)} + \widetilde{\alpha}_x^{(CL)} \tag{C.139}$$

$\alpha_x^{(AN)} = 0$,
The classical thermo-electric coefficient $\alpha_x^{(CL)}$ is given as:

$$\alpha_x^{(CL)} = -b_x^2 \frac{e}{m_e}\frac{\widetilde{c}_e^{(2)}}{c_e^{(1)}} n_e \tau_e z_{eff} \tag{C.140}$$

when `'b2sigp_style'` is set to `'2'` (Zhdanov formulation, default), and $\alpha_x^{(CL)} = b_x^2 \frac{e}{m_e} \sqrt{\frac{5}{2}} f_\alpha \left(z_{eff}\right) n_e \tau_e$ otherwise (Balescu formulation), and $f_\alpha \left(z_{eff}\right) = -\frac{1.25 z_{eff}}{1+0.41 z_{eff}}$. See also Section B.5.

A conductive flux limit is applied to the thermo-electric coefficient, as follows. If `'b2sigp_style'` is set to `'2'` (default):

$$\widetilde{\widetilde{\alpha}}_x^{(CL)} = c_{\alpha_{ex}}^{(F\,\mathrm{lim})} \widetilde{\alpha}_x^{(CL)} \tag{C.141}$$

$$\widetilde{\alpha}_x^{(CL)} = c_{\kappa_{ex}^{(CL)}}^{(Luciani)} \alpha_x^{(CL)} \tag{C.142}$$

$$c_{\alpha_{ex}}^{(F\,\mathrm{lim})} = \begin{cases} 1, & \text{inside the separatrix} \\ \frac{1}{1+\left|\frac{\alpha_x^{(CL)} \frac{1}{h_x} \frac{\partial T_e}{\partial x}}{c_{flal} j_{e\,\max}}\right|}, & \text{outside the separatrix} \end{cases}$$

$$j_{e\,\max} = |b_x|\, e n_e \sqrt{\frac{T_e}{m_e}},\ c_{flal} = c_{flim_3} = 0.5$$

If `'b2sigp_style'` is set to `'1'`:

$$\widetilde{\widetilde{\alpha}}_x^{(CL)} = c_{\kappa_{ex}}^{(F\mathrm{lim})} \widetilde{\alpha}_x^{(CL)} \tag{C.143}$$

$$\widetilde{\alpha}_x^{(CL)} = c_{\kappa_{ex}^{(CL)}}^{(Luciani)} \alpha_x^{(CL)} \tag{C.144}$$

C.4.4 Radial electron energy fluxes

$$\widetilde{q}_{ey} = \widetilde{f}_{ey} T_e - \kappa_{ey} \frac{1}{h_y} \frac{\partial T_e}{\partial y} + 0 \cdot \alpha_{ey} T_e \hat{E}_y - q_{ey}^{P.Sch} \tag{C.145}$$

$$\widetilde{f}_{ey} = \frac{3}{2} \left(\Gamma_{ey} - \frac{5}{3} \frac{j_y^{(ST)}}{e} \right) + V_{ey}^{(str)} n_e$$

$$q_{ey}^{P.Sch} = \frac{5}{2} D_{eyx} \frac{1}{h_x} \frac{\partial n_e T_e^2}{\partial x} \tag{C.146}$$

$$q_{ey} = \left[\frac{3}{2} \Gamma_{ey} - \frac{5}{2} \frac{j_y^{(ST)}}{e} + \frac{5}{2} n_e T_e \frac{B_z}{e} \frac{\partial}{h_x \partial x} \left(\frac{1}{B^2} \right) + V_{ey}^{(str)} n_e \right] T_e - \kappa_{ey} \frac{1}{h_y} \frac{\partial T_e}{\partial y} + 0 \cdot \alpha_{ey} T_e \hat{E}_y \tag{C.147}$$

$$\Gamma_{ey} = \sum_{a=0}^{ns-1} z_a \Gamma_{ay}^{(he)} \tag{C.148}$$

$$\Gamma_{ay}^{(he)} = \left(\frac{5}{3} V_a^{(AN)} + V_{ay}^{(E)} \right) n_a - \frac{5}{3} \left[D_{n,a} \frac{1}{h_y} \frac{\partial n_a}{\partial y} + \widetilde{D}_{p,ay} \frac{1}{h_y} \frac{\partial p_a}{\partial y} \right] \tag{C.149}$$

$V_{ey}^{(str)} = 0$

$$\hat{E}_y = \frac{1}{e n_e h_y} \frac{\partial n_e T_e}{\partial y} - \frac{1}{h_y} \frac{\partial \Phi}{\partial y} \tag{C.150}$$

$$\kappa_{ey} = \kappa_e^{(AN)} + \kappa_{ey}^{(CL)} + \kappa_e^{(ST)} \tag{C.151}$$

$\kappa_{ey}^{(CL)} = 0$

$$\kappa_e^{(ST)} = \begin{cases} 1.6\alpha^{(ST)}\sigma^{(ST)}T_e/e^2, & y \in \left[-\Delta^{(ST)}, 0\right], \quad \alpha^{(ST)} \approx 2 \\ 0, & y \notin \left[-\Delta^{(ST)}, 0\right] \end{cases} \tag{C.152}$$

$\alpha_{ey} = \alpha_y^{(AN)} + \alpha_y^{(CL)}$

$$\alpha_y^{(AN)} = \left(c_{\alpha,0} + c_{\alpha,1}\frac{10^{20}}{n_e} + c_{\alpha,3}\frac{T_e}{16e|B|}\right) n_e\sqrt{\frac{e}{T_e}} f_\alpha(x,y) \tag{C.153}$$

The $c_{\alpha,x}$ coefficients correspond to the `cfalf` array elements provided in the `b2ah.dat` file, or, if present, in the `b2cmpt` block of the `b2mn.dat` file. They can be overwritten by means of the `b2.transport.parameters` file, see Section 3.5.3. A typical choice may be:

$c_{\alpha,0} = 10^{-3}, \quad c_{\alpha,1} = 0, \quad c_{\alpha,3} = 0$

If $c_{\alpha,7} \neq 0$, then f_α contains a flux-scaling factor of $\left(\frac{h_y}{h_{y,ref}}\right)^2$, otherwise it is 1 everywhere.

$\alpha_y^{(CL)} = 0$

C.5 Ion energy equations

$$\begin{aligned} \frac{3}{2}\frac{\partial n_i T_i}{\partial t} + \frac{1}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x}\widetilde{q}_{ix}\right) + \frac{1}{\sqrt{g}}\frac{\partial}{\partial y}\left(\frac{\sqrt{g}}{h_y}\widetilde{q}_{iy}\right) + \sum_{a=0}^{ns-1}\frac{n_a T_i}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x}V_{a\parallel}b_x\right) = \\ = Q_\Delta + Q_{F_{ab}} + c_{\mathbf{E}\times\mathbf{B}}T_i B_z \frac{1}{h_x h_y}\left(\frac{\partial\Phi}{\partial y}\frac{\partial}{\partial x}\left(\frac{1}{B^2}\right) - \frac{\partial\Phi}{\partial x}\frac{\partial}{\partial y}\left(\frac{1}{B^2}\right)\right)\sum_{\substack{a=0 \\ z_a\neq 0}}^{ns-1} n_a + \\ + \sum_{fluid\ species}\left(\eta_{ax}\left(\frac{\partial V_{a\parallel}}{h_x\partial x}\right)^2 + \eta_a^{(AN)}\left(\frac{\partial V_{a\parallel}}{h_y\partial y}\right)^2\right) + Q_I^{(i)} + Q_R^{(i)} + Q_i^{\mathrm{EIRENE}}, \end{aligned} \tag{C.154}$$

The "sum over fluid species" above means the species index a ranges from 0 to $ns-1$ for **B2.5** standalone runs and only over charged species for coupled **SOLPS-ITER** runs.

C.5.1 Ion energy sources

$$\begin{aligned} S_i^{heat} = Q_\Delta + Q_{F_{ab}} + c_{\mathbf{E}\times\mathbf{B}}T_i B_z \frac{1}{h_x h_y}\left(\frac{\partial\Phi}{\partial y}\frac{\partial}{\partial x}\left(\frac{1}{B^2}\right) - \frac{\partial\Phi}{\partial x}\frac{\partial}{\partial y}\left(\frac{1}{B^2}\right)\right)\sum_{\substack{a=0 \\ z_a\neq 0}}^{ns-1} n_a + \\ + Q_{vis} + Q_I^{(i)} + Q_R^{(i)} - \sum_{a=0}^{ns-1}\frac{n_a T_i}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x}V_{a\parallel}b_x\right) \end{aligned} \tag{C.155}$$

$$Q_{vis} = \sum_{fluid\ species} \left(\eta_{ax} \left(\frac{\partial V_{a\parallel}}{h_x \partial x} \right)^2 + \eta_a^{(AN)} \left(\frac{\partial V_{a\parallel}}{h_y \partial y} \right)^2 \right) \tag{C.156}$$

If `'b2tral_Zhdanov_closure'` is set to `'1'` , Q_{vis} is applied according to section C.11.

When Eq. (C.55) is used (`'b2sigp_style'` set to `'2'`), the contribution from friction between charged species is given as:

$$Q_{F_{ab}} = - \sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} V_{a\parallel} \left(S^m_{fr,a} + S^m_{Therm,a} \right) \tag{C.157}$$

Otherwise, it is:

$$Q_{F_{ab}} = \sum_{\substack{b=0 \\ z_b \neq 0}}^{ns-1} \sum_{\substack{a=b+1 \\ z_a \neq 0}}^{ns-1} \frac{1}{4\tau_p} m_p \sqrt{\frac{\frac{m_a}{m_p}\frac{m_b}{m_p}}{\frac{m_a}{m_p} + \frac{m_b}{m_p}}} z_a^2 z_b^2 n_a n_b \left(V_{a\parallel} - V_{b\parallel} \right)^2 \tag{C.158}$$

If `'b2tral_Zhdanov_closure'` is set to `'1'` , $Q_{F_{ab}}$ is applied according to section C.11.

Atom heat source due to ionisation

$$Q_I^{(i)} = \sum_{a=0}^{ns-2} Q_{I_a}^{(i)} \tag{C.159}$$

$$Q_{I_a}^{(i)} = \begin{cases} r_{I_a} n_e n_a \left(\frac{m_a + m_{a+1}}{4} \right) \left(V_{\parallel a} - V_{\parallel a+1} \right)^2, & z_a < z_{a+1}\ \&\ z_n = z_{n+1} \\ 0 \end{cases} \tag{C.160}$$

Atom heat source due to recombination

$$Q_R^{(i)} = \sum_{a=1}^{ns-1} Q_{R_a}^{(i)} \tag{C.161}$$

$$Q_{R_a}^{(i)} = \begin{cases} r_{R_a} n_e n_a \left(\frac{m_a + m_{a-1}}{4} \right) \left(V_{\parallel a} - V_{\parallel a-1} \right)^2, & z_{a-1} < z_a\ \&\ z_{n-1} = z_n \\ 0 \end{cases} \tag{C.162}$$

$\tau_p = \frac{3}{4\sqrt{2\pi}} \frac{\sqrt{m_p} T_i^{\frac{3}{2}}}{\ln \Lambda} \left(\frac{4\pi\varepsilon_0}{e^2} \right)^2$, where the Coulomb logarithm $\ln \Lambda$ can be chosen as

$$\ln \Lambda = \begin{cases} 12 & \\ \max\{5; 23.4 - 1.15 \cdot \log_{10}\left(\frac{n_e}{10^6}\right) + 3.45 \cdot \log_{10}\left(\frac{T_e}{e}\right)\}, & \frac{T_e}{e} \leq 50 \\ \max\{5; 25.3 - 1.15 \cdot \log_{10}\left(\frac{n_e}{10^6}\right) + 2.30 \cdot \log_{10}\left(\frac{T_e}{e}\right)\}, & \frac{T_e}{e} > 50 \\ \max\{5; 17.3 - 0.5 \cdot \ln\left(\frac{n_e}{10^{20}}\right) + 1.5 \cdot \ln\left(\frac{T_i}{e \cdot 10^3}\right)\} & \end{cases} \begin{array}{l} \\ \text{Braginskii formula} \\ \\ \text{Wesson formula for} \\ \text{ion-ion collisions} \end{array}$$

C.5.2 Poloidal ion energy fluxes

$$\widetilde{q}_{ix} = \widetilde{f}_{ix} T_i - \kappa_{ix} \frac{1}{h_x} \frac{\partial T_i}{\partial x} - q_{ix}^{P.Sch} = \\ = \frac{3}{2} \Gamma_{ix} T_i + V_{ix}^{(str)} n_i T_i + \frac{3}{2} \left(\frac{j_x^{(AN)} + j_x^{(in)} + \widetilde{j}_x^{(vis\parallel)} + j_x^{(vis\perp)} + \widetilde{j}_x^{(visq)}}{e} \right) T_i - \\ - \kappa_{ix} \frac{1}{h_x} \frac{\partial T_i}{\partial x} - q_{ix}^{P.Sch} \quad \text{(C.163)}$$

$$\widetilde{f}_{ix} = \frac{3}{2} \Gamma_{ix} + V_{ix}^{(str)} n_i + \frac{3}{2} \left(\frac{j_x^{(AN)} + j_x^{(in)} + \widetilde{j}_x^{(vis\parallel)} + j_x^{(vis\perp)} + \widetilde{j}_x^{(visq)}}{e} \right)$$

If `'b2tral_Zhdanov_closure'` is set to `'1'` , $\widetilde{f}_{ix}$ is applied according to section C.11.

$$q_{ix}^{P.Sch} = \frac{5}{2} \sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} D_{axy} \frac{1}{h_y} \frac{\partial n_a T_i^2}{\partial y} \quad \text{(C.164)}$$

$$q_{ix} = \left[\frac{3}{2} \Gamma_{ix} + \frac{5}{2} \sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} n_a \widetilde{V}_{ax}^{(dia)} + V_{ix}^{(str)} n_i \right. \\ \left. + \frac{3}{2} \left(\frac{j_x^{(AN)} + j_x^{(in)} + \widetilde{j}_x^{(vis\parallel)} + j_x^{(vis\perp)} + \widetilde{j}_x^{(visq)}}{e} \right) \right] T_i - \kappa_{ix} \frac{1}{h_x} \frac{\partial T_i}{\partial x} \quad \text{(C.165)}$$

If `'b2tral_Zhdanov_closure'` is set to `'1'` , the convective part of q_{ix} is applied according to section C.11.

$$\Gamma_{ix} = \sum_{a=0}^{ns-1} \Gamma_{ax}^{(he)} \quad \text{(C.166)}$$

$$q_{ix}^{(Eir)} = \frac{5}{2} \sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} \Gamma_{ax}^{(Eir)} T_i - \kappa_{ix} \frac{1}{h_x} \frac{\partial T_i}{\partial x} \quad \text{(C.167)}$$

If `'b2tral_Zhdanov_closure'` is set to `'1'` , the convective part of $q_{ix}^{(Eir)}$ is applied according to section C.11.

C.5.3 Poloidal ion energy transport coefficients

$$\kappa_{ix} = \kappa_{ix}^{(AN)} + \widetilde{\widetilde{\kappa}}_{ix}^{(CL)} \quad \text{(C.168)}$$

$$\kappa_{ix}^{(AN)} = \sum_{a=0}^{ns-1} \widetilde{\kappa}_{ax}^{(AN)} \quad \text{(C.169)}$$

$$\widetilde{\kappa}_{ax}^{(AN)} = \begin{cases} \kappa_a^{(AN)}, & z_a \neq 0 \\ \frac{\kappa_a^{(AN)}}{\left(1+\left(\frac{\frac{\sqrt{g}}{h_x}\kappa_a^{(AN)}\left|\frac{\partial T_i}{h_x \partial x}\right|}{\frac{\sqrt{g}}{h_x}\alpha V_{h_a} n_a T_i}\right)^{\gamma_2}\right)^{\frac{1}{\gamma_2}}}, & z_a = 0 \end{cases} \tag{C.170}$$

$$V_{h_a} = \sqrt{\frac{2T_i}{m_a}} \tag{C.171}$$

$$\kappa_a^{(AN)} = \chi_a^{(AN)} n_a \tag{C.172}$$

$$\chi_a^{(AN)} = \left(c_{h_a,0} + c_{h_a,1}\frac{10^{20}}{n_e} + c_{h_a,2}D_{f,a} + c_{h_a,3}\frac{T_e}{16e|B|}\right) f_{h_a}(x,y) \tag{C.173}$$

The $c_{h_a,x}$ coefficients correspond to the `cfhci` array elements provided in the `b2ah.dat` file, or, if present, in the `b2cmpt` block of the `b2mn.dat` file. They can be overwritten by means of the `b2.transport.parameters` file, see Section 3.5.3. A typical choice may be:

$a = 0, \quad c_{h_a,0} = 0, \quad c_{h_a,1} = 0, \quad c_{h_a,2} = 1.4, \quad c_{h_a,3} = 0$

$a = 1, \quad c_{h_a,0} = 0, \quad c_{h_a,1} = 0, \quad c_{h_a,2} = 1.2, \quad c_{h_a,3} = 0$

The f_{h_a} function is an optional modifier, governed by the value of `b2tqna_user_transport`. If $c_{h_a,7} \neq 0$, then f_{h_a} contains a flux-scaling factor of $\left(\frac{h_y}{h_{y,ref}}\right)^2$.

If `b2tqna_user_transport` is `2`, an ELM model is applied, with three levels of transport scaling according to radial location and alternating between inter-ELM and intra-ELM values. See code for details.

If `b2tqna_user_transport` is `6`, a transport barrier can be applied in a parametric fashion, according to the value of `b2tqna_itype`, and the coefficients `b2tqna_ti_alfa[1-4]` are used to modify f_{h_a}. See code for details.

If `b2tqna_user_transport` is `7`, the disruption model is applied, adding $1000 f_{quench}\sqrt{\frac{m_e}{m_a}}\frac{n_{a,ref}}{n_a}\left(\frac{T_i}{1000\text{eV}}\right)^{\frac{5}{2}}\left(\frac{h_y}{h_{y,ref}}\right)^2$ to $\chi_a^{(AN)}$.

If `b2tqna_user_transport` is `8`, the neoclassical transport model is applied. See code for details.

$$\widetilde{\widetilde{\kappa}}_{ix}^{(CL)} = c_{\kappa_{ix}}^{(F\,\lim)} \widetilde{\kappa}_{ix}^{(CL)} \tag{C.174}$$

If `'b2tral_Zhdanov_closure'` is set to `'1'`, $\widetilde{\widetilde{\kappa}}_{ix}^{(CL)}$ is applied according to section C.11.

$$c_{\kappa_{ix}}^{F\,\lim} = \begin{cases} 1, & \text{for inner flux surfaces} \\ \frac{1}{1+\frac{\frac{\sqrt{g}}{h_x}\widetilde{\kappa}_{ix}^{(CL)}\left|\frac{\partial T_i}{h_x \partial x}\right|}{V_{h_i}^{(\max)} n_i T_i}}, & \text{for other flux surfaces} \end{cases} \tag{C.175}$$

$$V_{h_i}^{(\max)} = c_{flim_1}\frac{\sqrt{g}}{h_x}|b_x|\sqrt{\frac{T_i}{m_p}}\frac{\sum_{a=0}^{ns-1} n_a\sqrt{\frac{m_p}{m_a}}}{n_i} \tag{C.176}$$

$c_{flim_1} = 0.15$ is the ion heat flux limiter coefficient.

$$\widetilde{\kappa}_{ix}^{(CL)} = c_{\kappa_{ix}^{(CL)}}^{(Luciani)} \kappa_{ix}^{(CL)} \tag{C.177}$$

If 'b2trcl_luciani' is set to '1', one gets:

$$c^{(Luciani)}_{\kappa^{(CL)}_{ix}} = \begin{cases} \frac{1}{1+\frac{7.5\cdot 10^{16}\left(\frac{T_i}{e}\right)^2}{c(y)\sum_{a=0}^{ns-1} z_a^2 n_a}}, & \text{only for inner flux surfaces} \\ 1, & \text{for other flux surfaces} \end{cases} \tag{C.178}$$

If 'b2trcl_luciani' is set to '2', the **SOLPS 4.x** formulation is used:

$$c^{(Luciani)}_{\kappa^{(CL)}_{ix}} = \begin{cases} \frac{1}{1+\frac{0.3\cdot 10^{16}\left(\frac{T_i}{e}\right)^2}{\sum_{a=0}^{ns-1} z_a^2 n_a}}, & \text{only for inner flux surfaces} \\ 1, & \text{for other flux surfaces} \end{cases} \tag{C.179}$$

If the plateau and banana transport regimes correction is applied (default, 'b2trcl_lluciani' set to '3'), one has:

$$c^{(Luciani)}_{\kappa^{(CL)}_{ix}} = \begin{cases} 1.6\varepsilon^{\frac{3}{2}}/K_2, & \text{only for inner flux surfaces} \\ 1, & \text{for other flux surfaces} \end{cases} \tag{C.180}$$

$$K_2 = \frac{0.66 + 1.88\cdot\sqrt{\varepsilon} - 1.54\cdot\varepsilon}{1 + 1.03\cdot\nu_*^{1/2} + 0.31\cdot\nu_*} + \frac{1.17\cdot\varepsilon^3\cdot\nu_*}{1 + 0.74\cdot\varepsilon^{3/2}\cdot\nu_*}$$

$$\kappa^{(CL)}_{ix} = \frac{5}{2} b_x^2 \sum_{\substack{a=0 \\ z_a\neq 0}}^{ns-1} \frac{T_i}{m_a} n_a f_{k_i} \tau_a,\ f_{k_i} = 2.253 \tag{C.181}$$

For integrated modeling workflows ('b2trcl_integral_modeling' set to '1'), in the region of the overlap with ASTRA in the inner part of the core:

$$\widetilde{\kappa}^{(CL)}_{ix} = \frac{25}{4}\left(n_{ismain} T_i \frac{B_z}{B_x h_y}\right)^2 \frac{h_{y(\text{outer midplane})}}{e}\left[\oint \frac{\sqrt{g}}{B^2}\left(1 - \frac{B^2}{\langle B^2\rangle}\right)^2 dx\right] \times K_{\text{ASTRA}} \tag{C.182}$$

Here $\langle B^2\rangle = \frac{\oint B^2\sqrt{g}dx}{\oint \sqrt{g}dx}$ and ismain is the index of the main plasma ion species.

From ASTRA we get T_i/e and $\oint q_i^{NEO} ds$ as a functions of $R_{(\text{outer midplane})}$ and calculate

$$K_{ASTRA} = \frac{-1}{\oint q_i^{NEO} ds}\frac{d(T_i/e)}{dR_{(\text{outer midplane})}} \tag{C.183}$$

as a flux surface function in B2.5.

$V^{(str)}_{ix} = 0$

C.5.4 Radial ion energy fluxes

$$\begin{aligned} \widetilde{q}_{iy} &= \widetilde{f}_{iy} T_i - \kappa_{iy}\frac{1}{h_y}\frac{\partial T_i}{\partial y} - q^{P.Sch}_{iy} \\ &= \left[\frac{3}{2}\Gamma_{iy} + V^{(str)}_{iy} n_i + \frac{3}{2}\frac{\left(j_y^{(AN)} + \widetilde{j}_y^{(vis\|)} + j_y^{(vis\perp)} + \widetilde{j}_y^{(visq)} + j_y^{(in)}\right)}{e}\right] T_i - \kappa_{iy}\frac{1}{h_y}\frac{\partial T_i}{\partial y} - q^{P.Sch}_{iy} \end{aligned} \tag{C.184}$$

$$\widetilde{f}_{iy} = \frac{3}{2}\Gamma_{iy} + V_{iy}^{(str)} n_i + \frac{3}{2}\frac{\left(j_y^{(AN)} + \widetilde{j}_y^{(vis\parallel)} + j_y^{(vis\perp)} + \widetilde{j}_y^{(visq)} + j_y^{(in)}\right)}{e}$$

$$q_{iy}^{P.Sch} = -\frac{5}{2}\sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} D_{ayx}\frac{1}{h_x}\frac{\partial n_a T_i^2}{\partial x} \tag{C.185}$$

$$q_{iy} = \left[\frac{3}{2}\Gamma_{iy} + \frac{5}{2}\sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} n_a \widetilde{V}_{ay}^{(dia)} + V_{iy}^{(str)} n_i + \frac{3}{2}\frac{\left(j_y^{(AN)} + \widetilde{j}_y^{(vis\parallel)} + j_y^{(vis\perp)} + \widetilde{j}_y^{(visq)} + j_y^{(in)}\right)}{e}\right] T_i - \kappa_{iy}\frac{1}{h_y}\frac{\partial T_i}{\partial y} \tag{C.186}$$

$\Gamma_{iy} = \sum_{a=0}^{ns-1} \Gamma_{ay}^{(he)}$

$$q_{iy}^{(Eir)} = \frac{5}{2}\sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} \Gamma_{ay}^{(Eir)} T_i - \kappa_{iy}\frac{1}{h_y}\frac{\partial T_i}{\partial y} \tag{C.187}$$

$V_{iy}^{(str)} = 0$

C.5.5 Radial ion energy transport coefficients

$$\kappa_{iy} = \kappa_{iy}^{(AN)} + \kappa_{iy}^{(CL)} \tag{C.188}$$

$$\kappa_{iy}^{(AN)} = \sum_{a=0}^{ns-1} \widetilde{\kappa}_{ay}^{(AN)} \tag{C.189}$$

$$\widetilde{\kappa}_{ay}^{(AN)} = \begin{cases} \kappa_a^{(AN)}, & z_a \neq 0 \\ \frac{\kappa_a^{(AN)}}{\left(1+\left(\frac{\frac{\sqrt{g}}{h_y}\kappa_a^{(AN)}\left|\frac{\partial T_i}{h_y \partial y}\right|}{\frac{\sqrt{g}}{h_y}\alpha V_{h_a} n_a T_i}\right)^{\gamma_2}\right)^{\frac{1}{\gamma_2}}}, & z_a = 0 \end{cases} \tag{C.190}$$

$\kappa_{iy}^{(CL)} = 0$

C.6 Current continuity equation

$$\frac{1}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x} j_x\right) + \frac{1}{\sqrt{g}}\frac{\partial}{\partial y}\left(\frac{\sqrt{g}}{h_y} j_y\right) = 0 \tag{C.191}$$

C.6.1 Poloidal currents

$$j_x = j_x^{(AN)} + \widetilde{j}_x^{(dia)} + j_x^{(in)} + \widetilde{j}_x^{(vis\parallel)} + j_x^{(vis\perp)} + \widetilde{j}_x^{(visq)} + \widetilde{j}_x^{(s)} + j_x^{(\parallel)} \tag{C.192}$$

$$j_x^{(AN)} = \sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} j_{ax}^{(AN)} \tag{C.193}$$

$$j_{ax}^{(AN)} = -\frac{Z_a n_a}{n_e} \sigma^{(AN)} \frac{\partial \Phi}{h_x \partial x} \tag{C.194}$$

$j_{ax}^{(AN)}$ is the component of the poloidal anomalous current due to species a.

$$\widetilde{j}_x^{(dia)} = B_z \frac{\sum\limits_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} n_a \left(z_a T_e + T_i \right)}{h_y} \frac{\partial}{\partial y} \left(\frac{1}{B^2} \right) \tag{C.195}$$

$$j_x^{(in)} = \sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} j_{ax}^{(in)} \tag{C.196}$$

$$j_{ax}^{(in)} = \begin{cases} \dfrac{B_z}{2} \dfrac{\partial}{h_y \partial y} \left(\dfrac{1}{B^2} \right) m_a n_a V_{a\parallel}^2, z_a \neq 0 \\ 0, z_a = 0 \end{cases} \tag{C.197}$$

$j_{ax}^{(in)}$ is the component of the poloidal inertial current due to species a.

$$\widetilde{j}_x^{(vis\parallel)} = \sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} \widetilde{j}_{ax}^{(vis\parallel)} \tag{C.198}$$

$$\widetilde{j}_{ax}^{(vis\parallel)} = -\delta_{1,a} \frac{B_z \sqrt{B}}{3 B_x} \frac{\partial}{h_y \partial y} \left(\frac{1}{B^2} \right) \widetilde{\widetilde{\eta}}_{ax}^{(CL)} \frac{\partial \left(\sqrt{B} V_{a\parallel} \right)}{h_x \partial x}, \left(\widetilde{\widetilde{\eta}}_{ax}^{(CL)} \text{ already contains a factor of } b_x^2 \right) \tag{C.199}$$

$\widetilde{j}_{ax}^{(vis\parallel)}$ is the component of the poloidal current due to parallel viscosity from species a. It is non-zero only for the main hydrogenic species in the problem.

If `'b2tral_Zhdanov_closure'` is set to `'1'`, $\widetilde{j}_{ax}^{(vis\parallel)}$ is applied according to section C.11.

$$j_x^{(vis\perp)} = \sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} j_{ax}^{(vis\perp)} \tag{C.200}$$

$$j_{ax}^{(vis\perp)} = 0 \tag{C.201}$$

$j_{ax}^{(vis\perp)}$ is the poloidal component of the perpendicular viscosity current due to species a.

$$\widetilde{j}_x^{(visq)} = \sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} \widetilde{j}_{ax}^{(visq)} \tag{C.202}$$

$$\widetilde{j}_{ax}^{(visq)} = -\delta_{1,a} 0.24 \alpha^{NEO} \widetilde{\tau}_a b_z \frac{B_x}{B^{\frac{1}{2}}} \frac{\partial}{h_y \partial y} \left(\frac{1}{B^2} \right) \frac{\partial \left(\widetilde{q}_a^{(0)} B^{\frac{1}{2}} \right)}{h_x \partial x} \tag{C.203}$$

$\widetilde{j}_{ax}^{(visq)}$ is the component of the poloidal heat viscosity current due to species a. It is non-zero only for the main hydrogenic species in the problem.

If `'b2tral_Zhdanov_closure'` is set to `'1'`, $\widetilde{j}_{ax}^{(visq)}$ is applied according to section C.11.

For the definition of $\widetilde{q}_a^{(0)}$, see Eqs. (C.103) and (C.104).

The poloidal component of the ion-neutral current is

$$\widetilde{j}_x^{(s)} = \sum_{\substack{a_0=0 \\ z_{a_0}=0 \ \& \ z_{n,a_0}=1}}^{ns-1} \sum_{\substack{a=0 \\ z_a=1 \ \& \ z_{n,a}=1}}^{ns-1} \left(-\sigma_{aa_0} b_z^2 \frac{\partial \Phi}{h_x \partial x} - \sigma_{aa_0} b_z^2 \frac{1}{z_a e n_a} \frac{\partial n_a T_i}{h_x \partial x} + \sigma_{aa_0} B_z V_{a_0 y} \right) \tag{C.204}$$

If `'b2tinnt_style'` is set to `'1'` (default), the gradient of $n_a T_i$ is calculated directly. If `'b2tinnt_style'` is set to `'0'`, the implementation follows from rewriting the equation as

$$\widetilde{j}_x^{(s)} = \sum_{\substack{a_0=0 \\ z_{a_0}=0 \ \& \ z_{n,a_0}=1}}^{ns-1} \sum_{\substack{a=0 \\ z_a=1 \ \& \ z_{n,a}=1}}^{ns-1} \left(-\sigma_{aa_0} b_z^2 \frac{\partial \Phi}{h_x \partial x} - \sigma_{aa_0} b_z^2 \frac{1}{z_a e} \left(\frac{\partial T_i}{h_x \partial x} + T_i \frac{\partial log(n_a)}{h_x \partial x} \right) + \sigma_{aa_0} B_z V_{a_0 y} \right) \tag{C.205}$$

The ion-neutral conductivity is given by:

$$\sigma_{aa_0} = \begin{cases} \frac{\tilde{\mu}_{aa_0} m_p n_a \langle V_{aa_0} \sigma_{ex} \rangle n_{a_0}}{B^2}, & z_{a_0} = 0 \ \& \ z_{n,a_0} = 1, z_a \neq 0 \ \& \ z_{n_a} = 1 \\ 0, & \text{otherwise} \end{cases} \tag{C.206}$$

If `'b2tqin_csigin_style'` is `'0'` the diffusivity of room-temperature hydrogen is used:

$$\langle V_{aa_0} \sigma_{ex} \rangle = 3.2 \cdot 10^{-15} \sqrt{\frac{T_i}{0.026e}} \sqrt{\frac{1}{\tilde{\mu}_{aa_0}}} \tag{C.207}$$

$$\tilde{\mu}_{aa_0} = \frac{\frac{m_a}{m_p} \frac{m_{a_0}}{m_p}}{\frac{m_a}{m_p} + \frac{m_{a_0}}{m_p}} \tag{C.208}$$

If `'b2tqin_csigin_style'` is `'1'` the local B2.5 CX rate is used.

$V_{a_0 y} = \frac{\Gamma_{a_0 y}}{n_{a_0}}$

If `'b2tinnt_kinetic'` is set to `'1'`, the ion-neutral current is instead computed directly from the perpendicular momentum sources provided by `Eirene`, in which case:

$$\widetilde{j}_x^{(s)} = \sum_{a=0}^{ns-1} \widetilde{j}_{ax}^{(s)} \tag{C.209}$$

$$\widetilde{j}_{ax}^{(s)} = \frac{b_z S_{EIRENE_{ay}}^m - S_{EIRENE_{a\perp}}^m}{B} \tag{C.210}$$

$j_x^{(\|)}$ - see Eq. (C.134) above.

C.6.2 Radial currents

$$j_y = j_y^{(AN)} + \widetilde{j}_y^{(dia)} + j_y^{(in)} + \widetilde{j}_y^{(vis\|)} + j_y^{(vis\perp)} + \widetilde{j}_y^{(visq)} + \widetilde{j}_y^{(s)} + j_y^{(ST)} \tag{C.211}$$

$$j_y^{(AN)} = \sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} j_{ay}^{(AN)} \tag{C.212}$$

$$j_{ay}^{(AN)} = -\frac{Z_a n_a}{n_e} \sigma^{(AN)} \frac{\partial \Phi}{h_y \partial y} \tag{C.213}$$

$j_{ay}^{(AN)}$ is the component of the radial anomalous current due to species a.

$$\widetilde{j}_y^{(dia)} = -B_z \frac{\sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} n_a \left(z_a T_e + T_i\right)}{h_x} \frac{\partial}{\partial x}\left(\frac{1}{B^2}\right) \tag{C.214}$$

$$j_y^{(in)} = \sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} j_{ay}^{(in)} \tag{C.215}$$

$$j_{ay}^{(in)} = \begin{cases} -\dfrac{B_z}{2} \dfrac{\partial}{h_x \partial x}\left(\dfrac{1}{B^2}\right) m_a n_a V_{a\|}^2, z_a \neq 0 \\ 0, z_a = 0 \end{cases} \tag{C.216}$$

$j_{ay}^{(in)}$ is the component of the radial inertial current due to species a.

$$\widetilde{j}_y^{(vis\|)} = \sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} \widetilde{j}_{ay}^{(vis\|)} \tag{C.217}$$

$$\widetilde{j}_{ay}^{(vis\|)} = \delta_{1,a} \frac{B_z \sqrt{B}}{3 B_x} \frac{\partial}{h_x \partial x}\left(\frac{1}{B^2}\right) \widetilde{\widetilde{\eta}}_{ax}^{(CL)} \frac{\partial\left(\sqrt{B} V_{a\|}\right)}{h_x \partial x}, \left(\widetilde{\widetilde{\eta}}_{ax}^{(CL)} \text{already contains a factor of } b_x^2\right) \tag{C.218}$$

$\widetilde{j}_{ay}^{(vis\|)}$ is the radial component of the current due to parallel viscosity of species a. It is non-zero only for the main hydrogenic species in the problem.

If `'b2tral_Zhdanov_closure'` is set to `'1'`, $\widetilde{j}_{ay}^{(vis\|)}$ is applied according to section C.11.

$$\widetilde{j}_y^{(vis\perp)} = \sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} \widetilde{j}_{ay}^{(vis\perp)} \tag{C.219}$$

$$\widetilde{j}_{ay}^{(vis\perp)} = \begin{cases} -\frac{1}{B\sqrt{g}}\frac{\partial}{\partial y}\left(\frac{\sqrt{g}}{h_y}\frac{\eta_a^{(AN)}}{4}\frac{\partial \widetilde{V}_{a\perp}}{h_y \partial y}\right), z_a \neq 0 \\ 0, z_a = 0 \end{cases} \tag{C.220}$$

$\widetilde{j}_{ay}^{(vis\perp)}$ is the radial component of the perpendicular viscosity current due to species a.

$$\widetilde{V}_{a\perp} = \frac{V_{a\perp}}{1 + \frac{|V_{a\perp}|}{V_{T_a}}} \tag{C.221}$$

$$V_{T_a} = \sqrt{\frac{8T_i}{\pi m_a}} \tag{C.222}$$

$$V_{a\perp} = -\frac{1}{Bh_y}\frac{\partial \Phi}{\partial y} - \frac{1}{z_a e B h_y n_a}\frac{\partial n_a T_i}{\partial y} \tag{C.223}$$

$$\widetilde{j}_y^{(visq)} = \sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} \widetilde{j}_{ay}^{(visq)} \tag{C.224}$$

$$\widetilde{j}_{ay}^{(visq)} = \delta_{1,a} 0.24 \alpha^{NEO} \widetilde{\tau}_a b_z \frac{B_x}{B^{\frac{1}{2}}}\frac{\partial}{h_x \partial x}\left(\frac{1}{B^2}\right)\frac{\partial\left(\widetilde{q}_a^{(0)} B^{\frac{1}{2}}\right)}{h_x \partial x} \tag{C.225}$$

$\widetilde{j}_{ay}^{(visq)}$ is the radial component of the heat viscosity current due to species a. It is non-zero only for the main hydrogenic species in the problem.

If `'b2tral_Zhdanov_closure'` is set to `'1'`, $\widetilde{j}_{ay}^{(visq)}$ is applied according to section C.11.

For definition of $\widetilde{q}_a^{(0)}$, see Eqs. (C.103) and (C.104).

The radial component of the ion-neutral current is

$$\widetilde{j}_y^{(s)} = \sum_{\substack{a_0=0 \\ z_{a_0}=0 \ \& \ z_{n,a_0}=1}}^{ns-1} \sum_{\substack{a=0 \\ z_a=1 \ \& \ z_{n,a}=1}}^{ns-1} \left(-\sigma_{aa_0}\frac{\partial \Phi}{h_y \partial y} - \sigma_{aa_0}\frac{1}{z_a e n_a}\frac{\partial n_a T_i}{h_y \partial y} - \sigma_{aa_0} B V_{a_0\perp}\right) \tag{C.226}$$

The `'b2tinnt_style'` switch determines the numerical implementation, similarly as for Eq. (C.204) and Eq. (C.205).

If `'b2tinnt_kinetic'` is set to `'1'`, we use the `Eirene` momentum sources instead:

$$\widetilde{j}_y^{(s)} = \sum_{a=0}^{ns-1} \widetilde{j}_{ay}^{(s)} \tag{C.227}$$

$$\widetilde{j}_{ay}^{(s)} = \frac{S^m_{EIRENE_{a\perp}}}{B} \tag{C.228}$$

$$j_y^{(ST)} = \begin{cases} -\sigma^{(ST)}\left(\frac{1}{h_y}\frac{\partial \Phi}{\partial y} - \frac{T_e}{e n_e}\frac{1}{h_y}\frac{\partial n_e}{\partial y} - \frac{0.5}{e}\frac{1}{h_y}\frac{\partial T_e}{\partial y}\right), & y \in \left[-\Delta^{(ST)}, 0\right] \\ 0, & y \notin \left[-\Delta^{(ST)}, 0\right] \end{cases} \tag{C.229}$$

$V_{a_0\perp} = \frac{\Gamma_{a_0\perp}}{n_{a_0}}$

σ_{aa_0} see Eq. (C.206) above.

C.6.3 Conductivities

$$\sigma^{(ST)} = \begin{cases} c_{\sigma^{(ST)}} e n_e, & y \in \left[-\Delta^{(ST)}, 0\right], \quad c_{\sigma^{(ST)}} > c_{\sigma_y^{(AN)}} \\ 0, & y \notin \left[-\Delta^{(ST)}, 0\right] \end{cases} \tag{C.230}$$

is the radial conductivity associated with the presence of a magnetic stochastic layer, of thickness $\Delta^{(ST)}$ (in m, given by the `'b2tstch_delta'` switch, default value `0.0`), just inside the nominal separatrix. The multiplier $c_{\sigma^{(ST)}}$ is given by the value of the `'b2tstch_sig'` switch (default value `1.0`). Overall, the presence of the stochastic current term is governed by the `'b2tfhe_stochastic'` switch (default value `'0.0'`, i.e. off).

$\sigma_y = \sigma^{(AN)} + \sigma_y^{(CL)}$

$\sigma^{(AN)}$ depends on `'b2tqna_model_sig'`.

If `'b2tqna_model_sig'` is `'0'`, then:

$$\sigma^{(AN)} = \left(c_{\sigma^{(AN)},0} + c_{\sigma^{(AN)},1} \frac{10^{20}}{n_e} + c_{\sigma^{(AN)},3} \frac{T_e}{16e|B|} + c_{\sigma^{(AN)},4} e^{-c_{\sigma^{(AN)},5}\left(\frac{y}{L(x)}\right)^2} \right) e n_e f_\sigma(x,y), \tag{C.231}$$

otherwise:

$$\sigma^{(AN)} = \left(c_{\sigma^{(AN)},0} + c_{\sigma^{(AN)},1} \frac{10^{20}}{n_e} + c_{\sigma^{(AN)},3} \frac{T_e}{16e|B|} + c_{\sigma^{(AN)},4} e^{-c_{\sigma^{(AN)},5}\left(\frac{y}{L(x)}\right)^2} \right) e \widetilde{n}_e f_\sigma(x,y) \tag{C.232}$$

where $\widetilde{n}_e$ is the electron density at the location where the innermost flux surface crosses the outer midplane.

$$L(x) = \int_0^{y_{out}} h_y(x,y)\ dy$$

The $c_{\sigma^{(AN)},x}$ coefficients correspond to the `cfsig` array elements provided in the `b2ah.dat` file, or, if present, in the `b2cmpt` block of the `b2mn.dat` file. They can be overwritten by means of the `b2.transport.parameters` file, see Section 3.5.3. A typical choice may be:

$c_{\sigma^{(AN)},0} = 10^{-6} \div 10^{-3}$, $c_{\sigma^{(AN)},1} = 0$, $c_{\sigma^{(AN)},3} = 0$

If $c_{\sigma^{(AN)},7} \neq 0$, then f_σ contains a flux-scaling factor of $\left(\frac{h_y}{h_{y,ref}}\right)^2$, otherwise it is 1 everywhere.

$\sigma_y^{(CL)} = 0$

C.7 Typical boundary conditions

Most boundary conditions are either of flux or Dirichlet type.

Flux

To impose a given flux, the net source term in the guard cell must be equal to that flux. Considering that fluxes between guard cells are negligibly small, the flux across the guard cell face must equal the source term.

Dirichlet

To impose a fixed value in a guard cell, the code makes use of the different source contributions and very large numbers. For example, to set a fixed density n_{BC}, the code sets `sna(ix,iy,0,is) = VBIG` × `fcS` × n_{BC}, and `sna(ix,iy,1,is) = - VBIG` × `fcS`, with `sna(ix,iy,is) = sna(ix,iy,0,is) +`

`sna(ix,iy,1,is) × na(ix,iy,is)` and where `VBIG` is a very large number and `fcS` the contact area between the guard cell and the domain cell. Because `VBIG` is so large, the only way to get a physical source is to have `na(ix,iy,is) =` n_{BC}.

In this Section, the boundaries are denoted as S_0 for the PFR boundaries, S_1 for the core boundaries, E for the Eastern boundaries (right targets), W for the Western boundaries (left targets), and N for the outer main wall boundary.

C.7.1 Particle balance equations

$S_{n_a} = S_{n_a,0} + S_{n_a,1} n_a$

South. Private region

If $z_a = 0$

For leakage boundary condition: `BCCON = 10`

$$\left.\frac{\sqrt{g}}{h_y}\Gamma_{ay}\right|_{S_0} = \left.\left[p_{n_a,a,1}\frac{\sqrt{g}}{h_y}\sqrt{\frac{z_aT_e+T_i}{m_a}}n_a + \sum_{a=0}^{ns-1} r_{rec,a,0}c_{nsr}\max\left\{0, -\frac{\sqrt{g}}{h_y}\Gamma_{ay}\right\}\right]\right|_{S_0} \tag{C.233}$$

$p_{n_a,a,1} = -10^{-2}$, $r_{rec,0,0} = 0$, $r_{rec,1,0} = 1$

$\widetilde{\Gamma}_{ay}$ was replaced by Γ_{ay}

c_{nsr} — neutral sources rescale coefficient, 1 — when working with B2.5 standalone and 10^{-10} when coupling to Eirene

$p_{n_a,a,1}$ is the radial leakage velocity, in units of the local species sound speed. Negative velocities are directed out of the plasma.

If $z_a \neq 0$

For decay length boundary condition: `BCCON = 9`

$$\left.\frac{\sqrt{g}}{h_y}\widetilde{\Gamma}_{ay}\right|_{S_0} = \begin{cases} \left.\left[-\frac{1}{p_{n_a,a,1}}\frac{\sqrt{g}}{h_y}\left(D_{n,a}^{(AN)} + D_{p,a}^{(AN)}\left(z_aT_e+T_i\right)\right)n_a\right]\right|_{S_0}, & \text{mdf_fnb} = 0 \\ \left.\begin{bmatrix} -\frac{1}{p_{n_a,a,1}}\frac{\sqrt{g}}{h_y} & \left(D_{n,a}^{(AN)} + D_{p,a}^{(AN)}\left(z_aT_e+T_i\right)\right)n_a + \\ \frac{\sqrt{g}}{h_y} & D_{ayx}\frac{1}{h_x}\frac{\partial n_aT_i}{\partial x} + \frac{\sqrt{g}}{h_y}V_{ay}^{(E)}n_a \end{bmatrix}\right|_{S_0}, & \text{mdf_fnb} = 1 \end{cases} \tag{C.234}$$

$$S_{n_a,0}^{no_mdf} = 0 \tag{C.235}$$

$$S_{n_a,1}^{no_mdf} = -\frac{1}{p_{n_a,a,1}}\frac{\sqrt{g}}{h_y}\left(D_{n,a}^{(AN)} + D_{p,a}^{(AN)}\left(z_aT_e+T_i\right)\right) \tag{C.236}$$

$p_{n_a,a,1}$ is the density radial decay length of species a, in m.

For leakage boundary conditions: `BCCON = 10`

$$\left.\frac{\sqrt{g}}{h_y}\widetilde{\Gamma}_{ay}\right|_{S_0} = \begin{cases} \left.\left[p_{n_a,a,1}\frac{\sqrt{g}}{h_y}\sqrt{\frac{z_aT_e+T_i}{m_a}}n_a + \min\left\{0, \frac{\sqrt{g}}{h_y}n_aV_{ay}^{(E)}\right\}\right]\right|_{S_0}, & \text{mdf_fnb} = 0 \\ \left.\left[p_{n_a,a,1}\frac{\sqrt{g}}{h_y}\sqrt{\frac{z_aT_e+T_i}{m_a}}n_a + \frac{\sqrt{g}}{h_y}D_{ayx}\frac{1}{h_x}\frac{\partial n_aT_i}{\partial x} + \min\left\{0, \frac{\sqrt{g}}{h_y}n_aV_{ay}^{(E)}\right\}\right]\right|_{S_0}, & \text{mdf_fnb} = 1 \end{cases} \tag{C.237}$$

$p_{n_a,a,1} = -10^{-3}$ is the radial leakage velocity, in units of the species local sound speed. Negative velocities are directed out of the plasma.

$$S^{no_mdf}_{n_a,0} = 0 \tag{C.238}$$

$$S^{no_mdf}_{n_a,1} = p_{n_a,a,1}\frac{\sqrt{g}}{h_y}\sqrt{\frac{z_aT_e + T_i}{m_a}}n_a \tag{C.239}$$

Imposing radial leakage: `BCCON = 15`

When using drifts, one should use `BCCON = 10` instead.

$$-\frac{\sqrt{g}}{h_y^2}D_{n,ay}\frac{\partial n_a}{\partial y}\bigg|_{S_0} = \left[c_{n_a,a,2}\frac{\sqrt{g}}{h_y}\sqrt{\frac{z_aT_e + T_i}{m_a}}n_a\right]\bigg|_{S_0} \tag{C.240}$$

$c_{n_a,a,2} = -10^{-3}$ is the radial leakage velocity, in units of the species local thermal velocity. Negative velocities are directed out of the plasma.

South 1. Core plasma

If $z_a = 0$

Imposing total particle flux: `BCCON = 8`

$\frac{\sqrt{g}}{h_y}\widetilde{\Gamma}_{ay}\Big|_{S_1} = 0$

$\frac{\sqrt{g}}{h_y}\Gamma_{ay}\Big|_{S_1} = 0$

If $z_a \neq 0$

Imposing a constant density : `BCCON = 1`

$$\frac{\sqrt{g}}{h_y}\widetilde{\Gamma}_{ay}\bigg|_{S_1} = \frac{\sqrt{g}}{h_y}\left(v_{big}n_a|_{S_1} - v_{big}n_a\right)\Big|_{S_1}, \quad v_{big} = 10^8\sqrt{\frac{100\text{eV}}{m_p}}, \quad n_a|_{S_1} \text{ is the desired density in } m^{-3} \tag{C.241}$$

Imposing the total particle flux with constant flux density: `BCCON = 8`

$$\frac{\sqrt{g}}{h_y}\widetilde{\Gamma}_{ay}\bigg|_{S_1} = p_{n,a,0}\frac{\frac{\sqrt{g}}{h_y}}{\sum\limits_i\left(\frac{\sqrt{g}}{h_y}\right)_{i,-1}}\bigg|_{S_1} + \frac{\sqrt{g}}{h_y}D_{ayx}\frac{1}{h_x}\frac{\partial n_aT_i}{\partial x}\bigg|_{S_1} + \frac{\sqrt{g}}{h_y}V^{(E)}_{ya}n_a\bigg|_{S_1}$$

$p_{n,a,0}$ is the desired total particle flux, per unit time.

North

If $z_a = 0$

For leakage boundary condition: `BCCON = 10`

$$-\frac{\sqrt{g}}{h_y}\Gamma_{ay}\Big|_N = \left[p_{n_a,a,1}\frac{\sqrt{g}}{h_y}\sqrt{\frac{z_aT_e+T_i}{m_a}}n_a + \sum_{a=0}^{ns-1} r_{rec,a,0}c_{nsr}\max\left\{0, \frac{\sqrt{g}}{h_y}\Gamma_{ay}\right\}\right]\Big|_N \tag{C.242}$$

$p_{n_a,a,1} = -10^{-2}$, $r_{rec,0,0} = 0$, $r_{rec,1,0} = 1$

c_{nsr} — neutral sources rescale coefficient, 1 — when working with B2.5 standalone and 10^{-10} when coupling to Eirene

$p_{n_a,a,1}$ is the radial leakage velocity, in units of the local species sound speed. Negative velocities are directed out of the plasma.

$\widetilde{\Gamma}_{ay}$ was replaced by Γ_{ay}

If $z_a \neq 0$

For decay length boundary condition: `BCCON = 9`

$$-\frac{\sqrt{g}}{h_y}\widetilde{\Gamma}_{ay}\Big|_N = \begin{cases} \left[\frac{1}{p_{n_a,a,1}}\frac{\sqrt{g}}{h_y}\left(D_{n,a}^{(AN)} + D_{p,a}^{(AN)}\left(z_aT_e+T_i\right)\right)n_a\right]\Big|_N, & \text{mdf_fnb} = 0 \\ \begin{bmatrix} \frac{1}{p_{n_a,a,1}} & \frac{\sqrt{g}}{h_y}\left(D_{n,a}^{(AN)} + D_{p,a}^{(AN)}\left(z_aT_e+T_i\right)\right)n_a \\ -\frac{\sqrt{g}}{h_y} & D_{ayx}\frac{1}{h_x}\frac{\partial n_aT_i}{\partial x} - \frac{\sqrt{g}}{h_y}V_{ay}^{(E)}n_a \end{bmatrix}\Big|_N, & \text{mdf_fnb} = 1 \end{cases} \tag{C.243}$$

$$S_{n_a,0}^{no_mdf} = 0 \tag{C.244}$$

$$S_{n_a,1}^{no_mdf} = \frac{1}{p_{n_a,a,1}}\frac{\sqrt{g}}{h_y}\left(D_{n,a}^{(AN)} + D_{p,a}^{(AN)}\left(z_aT_e+T_i\right)\right) \tag{C.245}$$

$p_{n_a,a,1}$ is the density radial decay length of species a, in m.

For leakage boundary conditions: `BCCON = 10`

$$-\frac{\sqrt{g}}{h_y}\widetilde{\Gamma}_{ay}\Big|_N = \begin{cases} \left[p_{n_a,a,1}\frac{\sqrt{g}}{h_y}\sqrt{\frac{z_aT_e+T_i}{m_a}}n_a - \max\left\{0, \frac{\sqrt{g}}{h_y}n_aV_{ay}^{(E)}\right\}\right]\Big|_N, & \text{mdf_fnb} = 0 \\ \left[p_{n_a,a,1}\frac{\sqrt{g}}{h_y}\sqrt{\frac{z_aT_e+T_i}{m_a}}n_a - \frac{\sqrt{g}}{h_y}D_{ayx}\frac{1}{h_x}\frac{\partial n_aT_i}{\partial x} - \max\left\{0, \frac{\sqrt{g}}{h_y}n_aV_{ay}^{(E)}\right\}\right]\Big|_N, & \text{mdf_fnb} = 1 \end{cases} \tag{C.246}$$

$p_{n_a,a,1} = -10^{-3}$ is the radial leakage velocity, in units of the species local sound speed. Negative velocities are directed out of the plasma.

$$S_{n_a,0}^{no_mdf} = 0 \tag{C.247}$$

$$S_{n_a,1}^{no_mdf} = p_{n_a,a,1}\frac{\sqrt{g}}{h_y}\sqrt{\frac{z_aT_e+T_i}{m_a}}n_a \tag{C.248}$$

Imposing radial leakage: `BCCON = 15`

When using drifts, one should use `BCCON = 10` instead.

$$\frac{\sqrt{g}}{h_y^2}D_{n,ay}\frac{\partial n_a}{\partial y}\Big|_N = \left[c_{n_a,a,2}\frac{\sqrt{g}}{h_y}\sqrt{\frac{z_aT_e+T_i}{m_a}}n_a\right]\Big|_N \tag{C.249}$$

$c_{n_a,a,2} = -10^{-3}$ is the radial leakage velocity, in units of the species local thermal velocity. Negative velocities are directed out of the plasma.

North with puffing

If $z_a = 0$

Imposing a particle flux per unit area: `BCCON = 5`

$$-\frac{\sqrt{g}}{h_y}\widetilde{\Gamma}_{ay}\bigg|_N = p_{n,a,0}\frac{\sqrt{g}}{h_y}\bigg|_N + \sum_{a=0}^{ns-1}\left[r_{rec,a,0}c_{nsr}\max\left\{0,-\frac{\sqrt{g}}{h_y}\Gamma_{ay}\right\}\right]\bigg|_N \tag{C.250}$$

c_{nsr} — neutral sources rescale coefficient, 1 — when working with B2.5 standalone and 10^{-10} when coupling to Eirene

$r_{rec,0,0} = 0, \;\; r_{rec,1,0} = 1$

$p_{n,a,0}$ is the desired total particle flux density, in $m^{-2}.s^{-1}$.

$\widetilde{\Gamma}_{ay}$ was replaced by Γ_{ay}

or

Imposing a total particle flux with constant flux density: `BCCON = 8`

$$-\frac{\sqrt{g}}{h_y}\widetilde{\Gamma}_{ay}\bigg|_N = \frac{p_{n,a,0}}{\sum_{i=0}^{Nx-1}\left(\frac{\sqrt{g}}{h_y}\right)_{i,Ny}}\frac{\sqrt{g}}{h_y}\bigg|_N + \sum_{a=0}^{ns-1}\left[r_{rec,a,0}c_{nsr}\max\left\{0,\frac{\sqrt{g}}{h_y}\Gamma_{ay}\right\}\right]\bigg|_N$$

c_{nsr} — neutral sources rescale coefficient, 1 — when working with B2.5 standalone and 10^{-10} when coupling to Eirene

$r_{rec,0,0} = 0, \;\; r_{rec,1,0} = 1$

$p_{n,a,0}$ is the desired total particle flux, per unit time.

$\widetilde{\Gamma}_{ay}$ was replaced by Γ_{ay}

West

If $z_a = 0$

Imposing a given particle flux per unit area: `BCCON = 5`

$$\frac{\sqrt{g}}{h_x}\Gamma_{ax}\bigg|_W = \left[p_{n,a,1}\frac{\sqrt{g}}{h_y} + c_{nsr}\sum_{a=0}^{ns-1} r_{rec,a,0}\max\left\{0,-\frac{\sqrt{g}}{h_x}\Gamma_{ax}\right\}\right]\bigg|_W \tag{C.251}$$

$r_{rec,0,0} = 0, \;\; r_{rec,1,0} = 1, \;\; p_{n_a,a,1} = 0$

$$\frac{\sqrt{g}}{h_x}\widetilde{\Gamma}_{ax}\bigg|_W = \left[p_{n,a,1}\frac{\sqrt{g}}{h_x} + c_{nsr}\sum_{a=0}^{ns-1} r_{rec,a,0}\max\left\{0,-\frac{\sqrt{g}}{h_x}\Gamma_{ax}\right\}\right]\bigg|_W \tag{C.252}$$

$\widetilde{\Gamma}_{ax}$ was replaced by Γ_{ax}

$r_{rec,0,0} = 0, \;\; r_{rec,1,0} = 1, \;\; p_{n_a,a,1} = 0$

If $z_a \neq 0$

Sheath density boundary condition: `BCCON = 3`

$$\left.\frac{\sqrt{g}}{h_x}\Gamma_{ax}\right|_W = 10^{20}\left.\frac{\sqrt{g}}{h_x}\frac{\partial n_a}{\partial x}\right|_W$$

Drift-compatible, Bohm-Chodura sheath entrance condition `BCCON = 14`

- Marginal imposition, with collective sound speed: `MOMPAR(,,2)` = $p_{m,a,2} < 0.5$

To be used in conjunction with `BCENE = 15`, `BCENI = 15`, `BCMOM = 13`, and `BCPOT = 11`.

$$\left.\frac{\sqrt{g}}{h_x}\widetilde{\Gamma}_{ax}\right|_W = \left[-p_{n,a,1}\frac{\sqrt{g}}{h_x}|b_x|\sqrt{\frac{\sum_{\substack{b=0\\ z_b\neq 0}}^{ns-1} p_b}{\sum_{\substack{b=0\\ z_b\neq 0}}^{ns-1} m_b n_b}}n_a - \frac{\sqrt{g}}{h_x}D_{axy}\frac{1}{h_y}\frac{\partial n_a T_i}{\partial y}\right]\Bigg|_W$$

$p_{n,a,1} = +1$

- Non-marginal imposition, with individual species sound speed: `MOMPAR(,,2)` = $p_{m,a,2} \geq 0.5$

to be used in conjunction with `BCMOM = 13`, `BCENE = 15` with `bcene_15_style = 1`, `BCENI = 15`, `BCPOT = 3` or `BCPOT = 11` with `istyle_fchi=0`.

This condition imposes that the sum of the poloidal projection of the parallel velocity and the poloidal $\mathbf{E}\times\mathbf{B}$ velocity (restricted to not exceed $\pm 2c_{s,a}|b_x|$), equals *at least* the specific species sound speed $c_{s,a}$, reprojected in the the poloidal direction.

The parameter $p_{m,a,1}$ is set by `MOMPAR(,,1)`. The parameter χ is set by `GAMMAI`.

$$\left.\frac{\sqrt{g}}{h_x}\widetilde{\Gamma}_{ax}\right|_W = \left[-\frac{\sqrt{g}}{h_x}|b_x|U_{out}n_a - \frac{\sqrt{g}}{h_x}D_{axy}\frac{1}{h_y}\frac{\partial n_a T_i}{\partial y}\right]\Bigg|_W$$

$$U_{out} = \max\left(-u_{\parallel,a}^{extrap} - \frac{V_{\mathbf{E}\times\mathbf{B}}^{restr}}{|b_x|}, p_{m,a,1}\cdot\sqrt{\frac{\chi T_i + z_a T_e}{m_a}}\right)$$

where $u_{\parallel,a}^{extrap}$ is the extrapolated parallel velocity at the target, and $V_{\mathbf{E}\times\mathbf{B}}^{restr}$ is the poloidal $\mathbf{E}\times\mathbf{B}$ velocity restricted to not exceed $\pm 2c_{s,a}|b_x|$.

East

<u>If $z_a = 0$</u>

Imposing a given particle flux per unit area: `BCCON = 5`

$$\left.-\frac{\sqrt{g}}{h_x}\Gamma_{ax}\right|_E = \left[p_{n,a,1}\frac{\sqrt{g}}{h_y} + c_{nsr}\sum_{a=0}^{ns-1} r_{rec,a,0}\max\left\{0, \frac{\sqrt{g}}{h_x}\Gamma_{ax}\right\}\right]\Bigg|_E \tag{C.253}$$

$r_{rec,0,0} = 0,\ \ r_{rec,1,0} = 1,\ \ p_{n_a,a,1} = 0$

$$\left.-\frac{\sqrt{g}}{h_x}\widetilde{\Gamma}_{ax}\right|_E = \left[c_{nsr}\sum_{a=0}^{ns-1} r_{rec,a,0}\max\left\{0, \frac{\sqrt{g}}{h_x}\Gamma_{ax}\right\}\right]\Bigg|_E \tag{C.254}$$

$\widetilde{\Gamma}_{ax}$ was replaced by Γ_{ax}

$r_{rec,0,0} = 0, \;\; r_{rec,1,0} = 1$

If $z_a \neq 0$

Sheath density boundary condition: `BCCON = 3`

$$-\frac{\sqrt{g}}{h_x}\Gamma_{ax}\bigg|_E = -10^{20}\;\frac{\sqrt{g}}{h_x}\frac{\partial n_a}{\partial x}\bigg|_E$$

Drift-compatible, Bohm-Chodura sheath entrance condition `BCCON = 14`

- Marginal imposition, with collective sound speed: `MOMPAR(,,2)` = $p_{m,a,2} < 0.5$

To be used in conjunction with `BCENE = 15`, `BCENI = 15`, `BCMOM = 13`, and `BCPOT = 11`.

$$-\frac{\sqrt{g}}{h_x}\widetilde{\Gamma}_{ax}\bigg|_E = \left[-p_{n,a,1}\frac{\sqrt{g}}{h_x}|b_x|\sqrt{\frac{\sum_{\substack{b=0\\ z_b\neq 0}}^{ns-1} p_b}{\sum_{\substack{b=0\\ z_b\neq 0}}^{ns-1} m_b n_b}}n_a + \frac{\sqrt{g}}{h_x}D_{axy}\frac{1}{h_y}\frac{\partial n_a T_i}{\partial y}\right]\bigg|_E$$

$p_{n,a,1} = +1$

- Non-marginal imposition, with individual species sound speed: `MOMPAR(,,2)` = $p_{m,a,2} \geq 0.5$

to be used in conjunction with `BCMOM = 13`, `BCENE = 15` with `bcene_15_style = 1`, `BCENI = 15`, `BCPOT = 3` or `BCPOT = 11` with `istyle_fchi=0`.

This condition imposes that the sum of the poloidal projection of the parallel velocity and the poloidal $\mathbf{E}\times\mathbf{B}$ velocity (restricted to not exceed $\pm 2c_{s,a}|b_x|$), equals *at least* the specific species sound speed $c_{s,a}$, reprojected in the the poloidal direction.

The parameter $p_{m,a,1}$ is set by `MOMPAR(,,1)`. The parameter χ is set by `GAMMAI`.

$$-\frac{\sqrt{g}}{h_x}\widetilde{\Gamma}_{ax}\bigg|_E = \left[-\frac{\sqrt{g}}{h_x}|b_x|U_{out}n_a + \frac{\sqrt{g}}{h_x}D_{axy}\frac{1}{h_y}\frac{\partial n_a T_i}{\partial y}\right]\bigg|_E$$

$$U_{out} = \max\left(u_{\parallel,a}^{extrap} + \frac{V_{\mathbf{E}\times\mathbf{B}}^{restr}}{|b_x|}, p_{m,a,1}\cdot\sqrt{\frac{\chi T_i + z_a T_e}{m_a}}\right)$$

where $u_{\parallel,a}^{extrap}$ is the extrapolated parallel velocity at the target, and $V_{\mathbf{E}\times\mathbf{B}}^{restr}$ is the poloidal $\mathbf{E}\times\mathbf{B}$ velocity restricted to not exceed $\pm 2c_{s,a}|b_x|$.

C.7.2 Parallel momentum balance for ions

$S_{m_a} = S_{m_a,0} + S_{m_a,1}V_{\parallel a} + S_{m_a,2}m_a n_a + S_{m_a,3}m_a n_a V_{\parallel a}$

South. Private region

If $z_a = 0$

Weakly imposing a velocity: `BCMOM = 4`

$$h_z \frac{\sqrt{g}}{h_y} \Gamma^m_{ay}\Big|_{S_0} = \left[p_{m,a,1} p_{m,a,2} h_z \frac{\sqrt{g}}{h_y} - p_{m,a,2} h_z \frac{\sqrt{g}}{h_y} V_{\|a} \right]\Big|_{S_0}$$

$p_{m,a,1} = 0$ is the imposed velocity, in $m.s^{-1}$, and $p_{m,a,2} = 10^5$ is the strength with which this velocity is imposed.

We had this b.c.

$h_z \frac{\sqrt{g}}{h_y} \Gamma^m_{ay}\Big|_{S_0} = \left[c_{m,a,3} h_z \frac{\sqrt{g}}{h_y} m_a n_a V_{\|a} + c_{m,a,6} \max\left(0, -h_z \frac{\sqrt{g}}{h_y} \frac{\Gamma^{Cor}_{ay}}{n_a}\right) m_a n_a V_{\|a} \right]\Big|_{S_0}$, delete term with `max` !

$\Gamma^{Cor}_{ay} = \Gamma_{ay} + \Gamma^{(dia)}_{ay}$

$c_{m,a,3} = -10^{10}$, we have $c_{m,a,6} = -1$, but not $c_{m,a,6} = 0$

Inserted into `b2stbc_phys` for `BCMOM=12` as $c_{m,a,3} \to p_{m,a,2} = -10^{10}$, and $c_{m,a,6} \to p_{m,a,1} = -1$

If $z_a \neq 0$

Weakly imposing a velocity: `BCMOM = 4`

$$h_z \frac{\sqrt{g}}{h_y} \Gamma^m_{ay}\Big|_{S_0} = \left[p_{m,a,1} p_{m,a,2} h_z \frac{\sqrt{g}}{h_y} - p_{m,a,2} h_z \frac{\sqrt{g}}{h_y} V_{\|a} \right]\Big|_{S_0} \tag{C.255}$$

$p_{m,a,1} = 0$ is the imposed velocity, in $m.s^{-1}$, and $p_{m,a,2} = 10^5$ is the strength with which this velocity is imposed.

$$h_z \frac{\sqrt{g}}{h_y} \Gamma^m_{ay}\Big|_{S_0} = \left[c_{m,a,3} h_z \frac{\sqrt{g}}{h_y} m_a n_a V_{\|a} + c_{m,a,6} \max\left(0, -h_z \frac{\sqrt{g}}{h_y} \frac{\Gamma^{Cor}_{ay}}{n_a}\right) m_a n_a V_{\|a} \right]\Big|_{S_0} \tag{C.256}$$

delete term with `max`!

$\Gamma^{Cor}_{ay} = \Gamma_{ay} + \Gamma^{(dia)}_{ay}$

$c_{m,a,3} = 0$, $c_{m,a,6} = -1$

South 1. Core

If $z_a = 0$

Weakly imposing a velocity: `BCMOM = 4`

$h_z \frac{\sqrt{g}}{h_y} \Gamma^m_{ay}\Big|_{S_1} = \left[p_{m,a,1} p_{m,a,2} h_z \frac{\sqrt{g}}{h_y} - p_{m,a,2} h_z \frac{\sqrt{g}}{h_y} V_{\|a} \right]\Big|_{S_1}$

$p_{m,a,1} = 0$ is the imposed velocity, in $m.s^{-1}$, and $p_{m,a,2} = 10^5$ is the strength with which this velocity is imposed.

$h_z \frac{\sqrt{g}}{h_y} \Gamma^m_{ay}\Big|_{S_1} = c_{m,a,3} h_z \frac{\sqrt{g}}{h_y} m_a n_a V_{\|a}\Big|_{S_1}$

$c_{m,a,3} = -10^{10}$

Inserted into `b2stbc_phys` for `BCMOM=12` as $c_{m,a,3} \to p_{m,a,2} = -10^{10}$, $c_{m,a,6} \to p_{m,a,1} = 0$

If $z_a \neq 0$

Weakly imposing a velocity: `BCMOM = 4`

$$h_z \frac{\sqrt{g}}{h_y} \Gamma^m_{ay}\Big|_{S_1} = \left[p_{m,a,1} p_{m,a,2} h_z \frac{\sqrt{g}}{h_y} - p_{m,a,2} \frac{\sqrt{g}}{h_y} V_{\|a} \right]\Big|_{S_1}$$

$p_{m,a,1} = 0$ is the imposed velocity, in $m.s^{-1}$, and $p_{m,a,2} = 10^5$ is the strength with which this velocity is imposed.

`b2stbc_spb` *style:* `BCMOM = 14`

$$\begin{aligned} h_z \frac{\sqrt{g}}{h_y} \Gamma^m_{ay}\Big|_{S_1} =& c_{m,a,3} h_z \left. \frac{\sqrt{g}}{h_y} m_a n_a V_{\|a} \right|_{S_1} + \\ & \left[h_z \frac{\sqrt{g}}{h_y} m_a \langle n_a \rangle (-2) \frac{\langle T_i \rangle B_z}{e z_a} \frac{\partial}{h_x \partial x} \left(\frac{1}{B^2} \right) \langle V_{\|a} \rangle c_{drift} + c_{corflux} h_z \frac{\sqrt{g}}{h_y} \right]\Big|_{S_1} \\ & + c_{m,a,1} \max \left(0, -h_z \frac{\sqrt{g}}{h_y} \frac{\Gamma^{Cor}_{ay}}{n_a} \right) m_a n_a V_{\|a} \Big|_{S_1} \end{aligned}$$

$c_{m,a,1} = 0$, $c_{m,a,3} = -10^{10}$

Prescribe the value of the parallel velocity as a function of the poloidal coordinate: `BCMOM = 15`

$$h_z \frac{\sqrt{g}}{h_y} \Gamma^m_{ay}\Big|_{S_1} = v_{big} n_{e0} m_a h_z \frac{\sqrt{g}}{h_y} \left[\frac{\langle B \rangle}{B} p_{m,a,1} - V_{\|a} \right]\Big|_{S_1}$$

where $\langle B \rangle = \frac{\oint_{S_1} B \sqrt{g}\, dx}{\oint_{S_1} \sqrt{g}\, dx}$, the integration is done over the innermost magnetic surface, $v_{big} = 10^8 \sqrt{\frac{100\text{eV}}{m_p}}$, $n_{e0} = 10^{20}$, $p_{m,a,1}$ is the given value of the average parallel velocity, in $m.s^{-1}$.

North

If $z_a = 0$

Weakly imposing a velocity: `BCMOM = 4`

$$-h_z \frac{\sqrt{g}}{h_y} \Gamma^m_{ay}\Big|_{N} = \left[p_{m,a,1} p_{m,a,2} h_z \frac{\sqrt{g}}{h_y} - p_{m,a,2} h_z \frac{\sqrt{g}}{h_y} V_{\|a} \right]\Big|_{N}$$

$p_{m,a,1} = 0$ is the imposed velocity, in $m.s^{-1}$, and $p_{m,a,2} = 10^5$ is the strength with which this velocity is imposed.

$$-h_z \frac{\sqrt{g}}{h_y} \Gamma^m_{ay}\Big|_{N} = \left[c_{m,a,3} h_z \frac{\sqrt{g}}{h_y} m_a n_a V_{\|a} + c_{m,a,6} \max \left(0, h_z \frac{\sqrt{g}}{h_y} \frac{\Gamma^{Cor}_{ay}}{n_a} \right) m_a n_a V_{\|a} \right]\Big|_{N}$$

delete term with `max` !

$c_{m,a,3} = -10^{10}$, we have $c_{m,a,6} = -1$, but not $c_{m,a,6} = 0$

Inserted into `b2stbc_phys` for `BCMOM=12` as $c_{m,a,3} \to p_{m,a,2} = -10^{10},\ c_{m,a,6} \to p_{m,a,1} = -1$

If $z_a \neq 0$

Weakly imposing a velocity: `BCMOM = 4`

$$-h_z \frac{\sqrt{g}}{h_y} \Gamma^m_{ay}\Big|_{N} = \left[p_{m,a,1} p_{m,a,2} h_z \frac{\sqrt{g}}{h_y} - p_{m,a,2} h_z \frac{\sqrt{g}}{h_y} V_{\|a} \right] |_N$$

$p_{m,a,1} = 0$ is the imposed velocity, in $m.s^{-1}$, and $p_{m,a,2} = 10^5$ is the strength with which this velocity is imposed.

$$-h_z\frac{\sqrt{g}}{h_y}\Gamma^m_{ay}\Big|_N = \left[c_{m,a,3}h_z\frac{\sqrt{g}}{h_y}m_a n_a V_{\|a} + c_{m,a,6}\max\left(0, h_z\frac{\sqrt{g}}{h_y}\frac{\Gamma^{Cor}_{ay}}{n_a}\right)m_a n_a V_{\|a}\right]\Big|_N$$

$c_{m,a,3} = 0$, $c_{m,a,6} = -1$

West

If $z_a = 0$, $p_{m,a,1} = 0$

If $z_a \neq 0$, $p_{m,a,1} = 1$

Drift-compatible, Bohm-Chodura sheath entrance condition `BCMOM = 13`

- Marginal imposition, with collective sound speed: `MOMPAR(,,2)` = $p_{m,a,2} < 0.5$

To be used in conjunction with `BCENE = 15`, `BCENI = 15`, `BCCON = 14`, and `BCPOT = 11`.

$$h_z\frac{\sqrt{g}}{h_x}\Gamma^m_{ax}\Big|_W = \left[p_{m,a,1}\frac{u_{big}}{\epsilon^2}V_{West}m_a n_a - \frac{u_{big}}{\epsilon^2}h_z\frac{\sqrt{g}}{h_x}|b_x|m_a n_a V_{\|a}\right]\Big|_W$$

with $u_{big} = 10^{10}\sqrt{\frac{100\text{eV}}{m_p}}$ and $\epsilon = 10^{-10}$.

$$V_{West} = -h_z\frac{\sqrt{g}}{h_x}\frac{|b_x|}{b_x}V^{(\mathbf{E}\times\mathbf{B})}_{ax} - h_z\frac{\sqrt{g}}{h_x}b_x\sqrt{\frac{\sum\limits_{\substack{a=0\\z_a\neq 0}}^{ns-1}p_a}{\sum\limits_{\substack{a=0\\z_a\neq 0}}^{ns-1}m_a n_a}}$$

- Non-marginal imposition, with individual species sound speed: `MOMPAR(,,2)` = $p_{m,a,2} \geq 0.5$

to be used in conjunction with `BCCON = 14`, `BCENE = 15` with `bcene_15_style = 1`, `BCENI = 15`, `BCPOT = 3` or `BCPOT = 11` with `istyle_fchi = 0`.

This condition imposes that the sum of the poloidal projection of the parallel velocity and the poloidal $\mathbf{E}\times\mathbf{B}$ velocity (restricted to not exceed $\pm 2c_{s,a}|b_x|$), equals *at least* the specific species sound speed $c_{s,a}$, reprojected in the the poloidal direction.

The parameter $p_{m,a,1}$ is set by `MOMPAR(,,1)`. The parameter χ is set by `GAMMAI`.

$$\frac{\sqrt{g}}{h_x}\Gamma^m_{ax}\Big|_W = \left[-\frac{u_{big}}{\epsilon^2}n_a m_a\frac{\sqrt{g}}{h_x}U_{out}\frac{B_x}{|B_x|} - \frac{u_{big}}{\epsilon^2}n_a m_a\frac{\sqrt{g}}{h_x}V_{\|,a}\right]\Big|_W$$

$$U_{out} = \max\left(-u^{extrap}_{\|,a} - \frac{V^{restr}_{\mathbf{E}\times\mathbf{B}}}{|b_x|}, p_{m,a,1}\cdot\sqrt{\frac{\chi T_i + z_a T_e}{m_a}}\right)$$

where $u^{extrap}_{\|,a}$ is the extrapolated parallel velocity at the target, and $V^{restr}_{\mathbf{E}\times\mathbf{B}}$ is the poloidal $\mathbf{E}\times\mathbf{B}$ velocity restricted to not exceed $\pm 2c_{s,a}|b_x|$.

If $z_a = 0$

Imposing a specified parallel velocity: `BCMOM = 1`

$\frac{\sqrt{g}}{h_x}\Gamma^m_{ax}\Big|_W = \left[u_{big}n_{e00}\frac{m_a}{m_p}\frac{\sqrt{g}}{h_x}p_{m,a,1} - u_{big}n_{e00}\frac{m_a}{m_p}\frac{\sqrt{g}}{h_x}V_{\|a}\right]\Big|_W$

$p_{m,a,1} = 0$ is the imposed velocity in $m.s^{-1}$, $n_{e00} = 10^{20}$.

If $z_a \neq 0$

Imposing a sheath velocity boundary condition: `BCMOM = 3`

$$\frac{\sqrt{g}}{h_x}\Gamma^m_{ax}\Big|_W = \left[-\frac{u_{big}}{\epsilon^2}n_a m_a \frac{\sqrt{g}}{h_x}u_{p_{out}} - \frac{u_{big}}{\epsilon^2}n_a m_a \frac{\sqrt{g}}{h_x}V_{\|a}\right]\Big|_W$$

If $p_{m,a,2} < 0.5$:

$$c_s = \sqrt{\frac{\sum\limits_{\substack{a=0\\ z_a\neq 0}}^{ns-1} p_a}{\sum\limits_{\substack{a=0\\ z_a\neq 0}}^{ns-1} m_a n_a}}$$

$u_{u_{out}} = p_{m,a,1} c_s |b_x|$

$u_{p_{out}} = \max\left\{0, \frac{u_{u_{out}}}{|b_x|} + u_a^{(dia)} b_z\right\}$

If $p_{m,a,2} > 0.5$:

$c_{s,a} = \sqrt{\frac{\gamma_i T_i + z_a^2 \frac{n_a}{n_e} T_e}{m_a}}$

where γ_i is the adiabatic ratio of specific heats (`GAMMAI` in `b2.boundary.parameters`),

$u_{u_{out}} = \max\{p_{m,a,1} c_{s,a}, u_{extrap}\}$

where u_{extrap} is the extrapolated value to the sheath edge of the poloidal velocity u_a,

$u_{p_{out}} = \max\left\{0, u_{u_{out}} + u_a^{(dia)} \frac{b_z}{b_x}\right\}$.

East

If $z_a = 0$, $p_{m,a,1} = 0$

If $z_a \neq 0$, $p_{m,a,1} = 1$

Drift-compatible, Bohm-Chodura sheath entrance condition `BCMOM = 13`

- Marginal imposition, with collective sound speed: `MOMPAR(,,2)` = $p_{m,a,2} < 0.5$

To be used in conjunction with `BCENE = 15`, `BCENI = 15`, `BCCON = 14`, and `BCPOT = 11`.

$-h_z \frac{\sqrt{g}}{h_x}\Gamma^m_{ax}\Big|_E = \left[p_{m,a,1}\frac{u_{big}}{\epsilon^2}V_{East} m_a n_a - \frac{u_{big}}{\epsilon^2}h_z \frac{\sqrt{g}}{h_x}|b_x| m_a n_a V_{\|a}\right]\Big|_E$

$$V_{East} = -h_z \frac{\sqrt{g}}{h_x}\frac{|b_x|}{b_x}V_{ax}^{(\mathbf{E}\times\mathbf{B})} + h_z \frac{\sqrt{g}}{h_x} b_x \sqrt{\frac{\sum\limits_{\substack{a=0\\ z_a\neq 0}}^{ns-1} p_a}{\sum\limits_{\substack{a=0\\ z_a\neq 0}}^{ns-1} m_a n_a}}$$

- Non-marginal imposition, with individual species sound speed: `MOMPAR(,,2)` = $p_{m,a,2} \geq 0.5$

to be used in conjunction with BCCON=14, BCENE=15 with `bcene_15_style=1`, BCENI=15, BCPOT=3 or BCPOT=11 with `istyle_fchi=0`.

This condition imposes that the sum of the poloidal projection of the parallel velocity and the poloidal $\mathbf{E}\times\mathbf{B}$ velocity (restricted to not exceed $\pm 2c_{s,a}|b_x|$), equals *at least* the specific species sound speed $c_{s,a}$, reprojected in the the poloidal direction.

The parameter $p_{m,a,1}$ is set by `MOMPAR(,,1)`. The parameter χ is set by `GAMMAI`.

$$-\left.\frac{\sqrt{g}}{h_x}\Gamma^m_{ax}\right|_E = \left.\left[\frac{u_{big}}{\epsilon^2}n_a m_a \frac{\sqrt{g}}{h_x}U_{out}\frac{B_x}{|B_x|} - \frac{u_{big}}{\epsilon^2}n_a m_a \frac{\sqrt{g}}{h_x}V_{\|,a}\right]\right|_E$$

$$U_{out} = \max\left(u^{extrap}_{\|,a} + \frac{V^{restr}_{\mathbf{E}\times\mathbf{B}}}{|b_x|}, p_{m,a,1}\cdot\sqrt{\frac{\chi T_i + z_a T_e}{m_a}}\right)$$

where $u^{extrap}_{\|,a}$ is the extrapolated parallel velocity at the target, and $V^{restr}_{\mathbf{E}\times\mathbf{B}}$ is the poloidal $\mathbf{E}\times\mathbf{B}$ velocity restricted to not exceed $\pm 2c_{s,a}|b_x|$.

If $z_a = 0$

Imposing a specified parallel velocity: `BCMOM = 1`

$$-\left.\frac{\sqrt{g}}{h_x}\Gamma^m_{ax}\right|_E = \left.\left[u_{big}n_{e00}\frac{m_a}{m_p}\frac{\sqrt{g}}{h_x}p_{m,a,1} - u_{big}n_{e00}\frac{m_a}{m_p}\frac{\sqrt{g}}{h_x}V_{\|a}\right]\right|_E$$

$p_{m,a,1} = 0$ is the imposed parallel velocity in $m.s^{-1}$.

If $z_a \neq 0$

Imposing a sheath velocity boundary condition: `BCMOM = 3`

$$-\left.\frac{\sqrt{g}}{h_x}\Gamma^m_{ax}\right|_E = \left.\left[-\frac{u_{big}}{\epsilon^2}n_a m_a \frac{\sqrt{g}}{h_x}u_{p_{out}} - \frac{u_{big}}{\epsilon^2}n_a m_a \frac{\sqrt{g}}{h_x}V_{\|a}\right]\right|_E$$

If $p_{m,a,2} < 0.5$:

$$c_s = \sqrt{\frac{\sum\limits_{\substack{a=0\\ z_a\neq 0}}^{ns-1} p_a}{\sum\limits_{\substack{a=0\\ z_a\neq 0}}^{ns-1} m_a n_a}}$$

$u_{u_{out}} = p_{m,a,1}c_s|b_x|,$

$u_{p_{out}} = \max\left\{0, \frac{u_{u_{out}}}{|b_x|} - u^{(dia)}_a b_z\right\}$

If $p_{m,a,2} > 0.5$:

$c_{s,a} = \sqrt{\frac{\gamma_i T_i + z_a^2 \frac{n_a}{n_e} T_e}{m_a}}$

$u_{u_{out}} = \max\left\{p_{m,a,1}c_{s,a}, u_{extrap}\right\}$

$u_{p_{out}} = \max\left\{0, u_{u_{out}} - u^{(dia)}_a \frac{b_z}{b_x}\right\}.$

C.7.3 Current continuity equation

$S_{ch} = S_{ch,0} + S_{ch,1}\Phi + S_{ch,2} n_e + S_{ch,3} n_e \Phi$

South. Private region

$$\left.\frac{\sqrt{g}}{h_y} j_y\right|_{S_0} = 0$$

$$\left.\frac{\sqrt{g}}{h_y} j_y\right|_{S_0} = \begin{cases} 0, & \text{facdrift} = 0 \\ \left.\frac{\sqrt{g}}{h_y}\left(\widetilde{j}_y^{(dia)} + j_y^{(in)} + \widetilde{j}_y^{(vis\parallel)} + j_y^{(vis\perp)} + \widetilde{j}_y^{(visq)} + \widetilde{j}_y^{(s)}\right)\right|_{S_0}, & \text{facdrift} \neq 0 \end{cases}$$

South 1. Core

$$\left.\frac{\sqrt{g}}{h_y} j_y\right|_{S_1} = \begin{cases} 0, & \text{facdrift} = 0 \\ \left.\frac{\sqrt{g}}{h_y}\left(\left\langle \widetilde{j}_y^{(dia)} \right\rangle\right)\right|_{S_1}, & \text{facdrift} \neq 0 \end{cases}$$

$$\left.\frac{\sqrt{g}}{h_y} j_y\right|_{S_1} = \begin{cases} 0, & \text{facdrift} = 0 \\ \left.\frac{\sqrt{g}}{h_y}\left(\left\langle \widetilde{j}_y^{(dia)} \right\rangle + \left\langle j_y^{(in)} \right\rangle\right)\right|_{S_1}, & \text{facdrift} \neq 0 \end{cases}$$

$$\left\langle \widetilde{j}_y^{(dia)} \right\rangle = -\frac{B_z \left\langle \sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} n_a \left(z_a T_e + T_i\right) \right\rangle}{h_x} \frac{\partial}{\partial x}\left(\frac{1}{B^2}\right)$$

$$\left\langle j_y^{(in)} \right\rangle = -\frac{B_z}{2} \frac{\partial}{h_x \partial x}\left(\frac{1}{B^2}\right) \sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} m_a n_a \left\langle V_{a\parallel}^2 \right\rangle$$

North

$$\left.\frac{\sqrt{g}}{h_y} j_y\right|_{N} = \begin{cases} 0, & \text{facdrift} = 0 \\ \left.\frac{\sqrt{g}}{h_y}\left(\left\langle \widetilde{j}_y^{(dia)} \right\rangle\right)\right|_{N}, & \text{facdrift} \neq 0 \end{cases}$$

$$-\left.\frac{\sqrt{g}}{h_y} j_y\right|_{N} = \begin{cases} 0, & \text{facdrift} = 0 \\ \left.-\frac{\sqrt{g}}{h_y}\left(\widetilde{j}_y^{(dia)} + j_y^{(in)} + \widetilde{j}_y^{(vis\parallel)} + j_y^{(vis\perp)} + \widetilde{j}_y^{(visq)} + \widetilde{j}_y^{(s)}\right)\right|_{N}, & \text{facdrift} \neq 0 \end{cases}$$

West

Imposing a sound speed velocity flux: `BCPOT = 11`

To be used in conjunction with `BCENE = 15`, `BCENI = 15`, `BCMOM = 13`, and `BCCON = 14`.

$$\left.\frac{\sqrt{g}}{h_x} j_x\right|_{W} = \left.\frac{\sqrt{g}}{h_x} e \left[-\sum_{a=0}^{ns-1} z_a \left(p_{n,a,1} |b_x| \sqrt{\frac{\sum_{\substack{b=0 \\ z_b \neq 0}}^{ns-1} p_b}{\sum_{\substack{b=0 \\ z_b \neq 0}}^{ns-1} m_b n_b}} n_a \right) + (1-\gamma_e) |b_x| n_e \frac{1}{\sqrt{2\pi}} \sqrt{\frac{T_e}{m_e}} \exp\left(-\frac{e\left(\Phi - p_{\Phi,2}\right)}{T_e}\right) \right] \right|_{W}$$

if 'b2stbc_istyle_fchi' is '1', or

$$\left.\frac{\sqrt{g}}{h_x}j_x\right|_W = \left.\frac{\sqrt{g}}{h_x}e\left[\sum_{a=0}^{ns-1} z_a\Gamma_{ax} + (1-\gamma_e)\,|b_x|n_e\frac{1}{\sqrt{2\pi}}\sqrt{\frac{T_e}{m_e}}\exp\left(-\frac{e\,(\Phi-p_{\Phi,2})}{T_e}\right)\right]\right|_W$$

if 'b2stbc_istyle_fchi' is '0'.

Imposing a sheath potential boundary condition: BCPOT = 3

$\left.\frac{\sqrt{g}}{h_x}j_x\right|_W = \left.\frac{\sqrt{g}}{h_x}e\left[\sum_{b=0}^{ns-1} z_a\Gamma_{ax} + (1-\gamma_e)\,|b_x|n_e\frac{1}{\sqrt{2\pi}}\sqrt{\frac{T_e}{m_e}}\exp\left(-\frac{e(\Phi-p_{\Phi,2})}{T_e}\right)\right]\right|_W$

East

Imposing a sound speed velocity flux: BCPOT = 11

To be used in conjunction with BCENE = 15, BCENI = 15, BCMOM = 13, and BCCON = 11.

$$-\left.\frac{\sqrt{g}}{h_x}j_x\right|_E = \left.\frac{\sqrt{g}}{h_x}e\left[-\sum_{a=0}^{ns-1} z_a\left(p_{n,a,1}|b_x|\sqrt{\frac{\sum_{\substack{b=0\\ z_b\neq 0}}^{ns-1} p_b}{\sum_{\substack{b=0\\ z_b\neq 0}}^{ns-1} m_b n_b}}n_a\right) + (1-\gamma_e)\,|b_x|n_e\frac{1}{\sqrt{2\pi}}\sqrt{\frac{T_e}{m_e}}\exp\left(-\frac{e\,(\Phi-p_{\Phi,2})}{T_e}\right)\right]\right|_E$$

if 'b2stbc_istyle_fchi' is '1', or

$$-\left.\frac{\sqrt{g}}{h_x}j_x\right|_E = \left.\frac{\sqrt{g}}{h_x}e\left[-\sum_{a=0}^{ns-1} z_a\Gamma_{ax} + (1-\gamma_e)\,|b_x|n_e\frac{1}{\sqrt{2\pi}}\sqrt{\frac{T_e}{m_e}}\exp\left(-\frac{e\,(\Phi-p_{\Phi,2})}{T_e}\right)\right]\right|_E$$

if 'b2stbc_istyle_fchi' is '0'.

Imposing a sheath potential boundary condition: BCPOT = 3

$-\left.\frac{\sqrt{g}}{h_x}j_x\right|_E = \left.\frac{\sqrt{g}}{h_x}e\left[-\sum_{a=0}^{ns-1} z_a\Gamma_{ax} + (1-\gamma_e)\,|b_x|n_e\frac{1}{\sqrt{2\pi}}\sqrt{\frac{T_e}{m_e}}\exp\left(-\frac{e(\Phi-p_{\Phi,2})}{T_e}\right)\right]\right|_E$

C.7.4 Energy balance equation for electrons

$S_{h_e} = S_{h_e,0} + S_{h_e,1}T_e + S_{h_e,2}n_e + S_{h_e,3}n_eT_e$

South. Private region

Imposing the b2stbc_spb *style boundary condition:* BCENE = 11

$$\left.\frac{\sqrt{g}}{h_y}\widetilde{q}_{ey}\right|_{S_0} = \left[\left(c_{h_e,2}\frac{\sqrt{g}}{h_y}\sqrt{\frac{T_e}{m_e}}\right)T_en_e + c_{h_e,7}\max\left(0,-\frac{\sqrt{g}}{h_y}\widetilde{f}_{ey}\right)T_e - q_{ex}^{P.Sch}\right]\Bigg|_{S_0}$$

$c_{h_e,2} = -10^{-4}$, $c_{h_e,7} = -1$, $c_{h_e,2} \to p_{h_e,2} = -10^{-4}$, $c_{h_e,7} \to p_{h_e,1} = -1$

Imposing a decay length for the electron temperature: BCENE = 22

$$-\frac{\sqrt{g}}{h_y}\kappa_{ey}\frac{\partial T_e}{h_y\partial y}\Big|_{S_0} = \left[\left(p_{h_e,1}\frac{\sqrt{g}}{h_y}\sqrt{\frac{T_e}{m_e}}\right)T_e n_e\right]\Big|_{S_0}$$

Imposing a decay length for the electron temperature: `BCENE=9`

$$\frac{\sqrt{g}}{h_y}\widetilde{q}_{ey}\Big|_{S_0} = \begin{cases} \left[\frac{5}{2}\sum_{a=0}^{ns-1} z_a S_{n_a,0}^{no_mdf} T_e - \frac{\sqrt{g}}{h_y}\frac{\kappa_e^{(AN)}}{p_{h_e,1}} T_e + \frac{5}{2}\sum_{a=0}^{ns-1} z_a S_{n_a,1}^{no_mdf} n_a T_e\right]\Big|_{S_0}, & \text{mdf_fhe} = 0 \\ \left[\begin{array}{c} \frac{5}{2}\sum_{a=0}^{ns-1} z_a S_{n_a,0}^{no_mdf} T_e - \frac{\sqrt{g}}{h_y}\frac{\kappa_e^{(AN)}}{p_{h_e,1}} T_e \\ + \quad \frac{5}{2}\sum_{a=0}^{ns-1} z_a S_{n_a,1}^{no_mdf} n_a T_e - q_{ey}^{P.Sch} + \frac{\sqrt{g}}{h_y}\frac{3}{2} V_{ey}^{(E)} n_e T_e \end{array}\right]\Big|_{S_0}, & \text{mdf_fhe} = 1 \end{cases}$$

South 1. Core

In `b2stbc_spb`*:*

$$\frac{\sqrt{g}}{h_y}\widetilde{q}_{ey}\Big|_{S_1} = \left[\left(c_{h_e,0}\frac{\sqrt{g}}{h_y} + c_{h_e,1}\frac{\sqrt{g}}{h_y}T_e\right)\right]\Big|_{S_1}$$

$c_{h_e,0} = 4\cdot 10^{13}$, $c_{h_e,a,1} = -10^{30}$

Imposing a total electron energy flux: `BCENE = 8`

$$\frac{\sqrt{g}}{h_y}q_{ey}\Big|_{S_1} = p_{h_e,1}\left.\frac{\frac{\sqrt{g}}{h_y}}{\sum_{core}\left(\frac{\sqrt{g}}{h_y}\right)}\right|_{S_1}$$

$p_{h_e,1}$ is the imposed total electron energy flux, in W.

North

Imposing the `b2stbc_spb` *style boundary condition:* `BCENE = 11`

$$-\frac{\sqrt{g}}{h_y}\widetilde{q}_{ey}\Big|_{N} = \left[\left(p_{h_e,1}\frac{\sqrt{g}}{h_y}\sqrt{\frac{T_e}{m_e}}\right)T_e n_e + p_{h_e,2}\max\left(0, \frac{\sqrt{g}}{h_y}\widetilde{f}_{ey}\right)T_e - q_{ex}^{P.Sch}\right]\Big|_{N}$$

$p_{h_e,1} = -10^{-4}$, $p_{h_e,2} = -1$

Imposing a decay length for electron temperature: `BCENE = 22`

$$\frac{\sqrt{g}}{h_y}\kappa_{ey}\frac{\partial T_e}{h_y\partial y}\Big|_{N} = \left[\left(p_{h_e,1}\frac{\sqrt{g}}{h_y}\sqrt{\frac{T_e}{m_e}}\right)T_e n_e\right]\Big|_{N}$$

Imposing a decay length for electron temperature: `BCENE = 9`

$p_{h_e,1}$ is the radial electron temperature decay length, in m.

$$-\frac{\sqrt{g}}{h_y}\widetilde{q}_{ey}\Big|_{N} = \begin{cases} \left[\frac{5}{2}\sum_{a=0}^{ns-1} z_a S_{n_a,0}^{no_mdf} T_e - \frac{\sqrt{g}}{h_y}\frac{\kappa_e^{(AN)}}{p_{h_e,1}} T_e + \frac{5}{2}\sum_{a=0}^{ns-1} z_a S_{n_a,1}^{no_mdf} n_a T_e\right]\Big|_{N}, & \text{mdf_fhe} = 0 \\ \left[\begin{array}{c} \frac{5}{2}\sum_{a=0}^{ns-1} z_a S_{n_a,0}^{no_mdf} T_e - \frac{\sqrt{g}}{h_y}\frac{\kappa_e^{(AN)}}{p_{h_e,1}} T_e \\ + \quad \frac{5}{2}\sum_{a=0}^{ns-1} z_a S_{n_a,1}^{no_mdf} n_a T_e + q_{ey}^{P.Sch} - \frac{\sqrt{g}}{h_y}\frac{3}{2} V_{ey}^{(E)} n_e T_e \end{array}\right]\Big|_{N}, & \text{mdf_fhe} = 1 \end{cases}$$

West

Imposing the `b2stbc_spb` *style boundary condition for electron temperature:* `BCENE = 15`

To be used in conjunction with `BCENI = 15`, `BCCON = 14`, `BCMOM = 13`, and `BCPOT = 11`.

If `'bcene_15_style'` is set to `'0'`

$$\left.\frac{\sqrt{g}}{h_x}\widetilde{q}_{ex}\right|_W = \left[-\left(\frac{1+\gamma_e}{1-\gamma_e}T_e + e\Phi\right) \max\left(0, \frac{\sqrt{g}}{h_x}\sum_{a=0}^{ns-1} z_a p_{n,a,1}|b_x| \sqrt{\frac{\sum\limits_{\substack{b=0\\ z_b\neq 0}}^{ns-1} p_b}{\sum\limits_{\substack{b=0\\ z_b\neq 0}}^{ns-1} m_b n_b}} n_a + \frac{\sqrt{g}}{h_x}\frac{j_\parallel}{e} \right) - \frac{\sqrt{g}}{h_x} q_{ex}^{P.Sch} \right]\Bigg|_W$$

If `'bcene_15_style'` is set to `'1'` (recommended for stability, default):

$$\left.\frac{\sqrt{g}}{h_x}\widetilde{q}_{ex}\right|_W = \left[-\frac{\sqrt{g}}{h_x}|b_x| \left((1+\gamma_e+p_{e,1})T_e + (1-\gamma_e)\, e\,(\Phi - p_{\Phi,2}) \right) n_e \frac{1}{\sqrt{2\pi}} \sqrt{\frac{T_e}{m_e}} \exp\left(-\frac{e\,(\Phi - p_{\Phi,2})}{T_e} \right) - \frac{\sqrt{g}}{h_x} q_{ex}^{P.Sch} \right]\Bigg|_W$$

with $p_{e,1}$ set by `ENEPAR(:,1)` and $p_{\Phi,2}$ set by `POTPAR(:,2)`.

In case of electron temperature or potential oscillations, one can make use of the `b2stbc_stab_coeff_sheath_te` switch, assigning it a positive value (recommended range between 1 and 100), to include an additional zero-sum stabilizing term.

Imposing the sheath boundary condition for electron temperature: `BCENE = 3`

$\left.\frac{\sqrt{g}}{h_x} q_{ex}\right|_W = \left[\left(p_{h_e,1} + \frac{e(\Phi - p_{\Phi,2})}{T_e} \right) \frac{\sqrt{g}}{h_x} \nu_{e_{out}} T_e - \frac{\sqrt{g}}{h_x} q_{ex}^{P.Sch} \right]\Big|_W$

$p_{h_e,1} = 0.9$, $p_{\Phi,2}$ is the applied electric potential at the target, in V.

If $p_{m,a_{main},2} < 0.5$:

$\nu_{e_{out}} = \min\left\{0, -u_{u_{out}} n_e - \frac{j_x}{e}\right\}$, $u_{u_{out}} = p_{m,a_{main},1} c_s |b_x|$

$$c_s = \sqrt{\frac{\sum\limits_{\substack{a=0\\ z_a\neq 0}}^{ns-1} p_a}{\sum\limits_{\substack{a=0\\ z_a\neq 0}}^{ns-1} m_a n_a}}$$

If $p_{m,a_{main},2} > 0.5$:

$$\nu_{e_{out}} = \min\left\{ 0, -\sum_{\substack{a=0\\ z_a\neq 0}}^{ns-1} u_{a_{out}} z_a n_a - \frac{j_x}{e} \right\}$$

$u_{a_{out}} = \max\left\{ p_{m,a,1} c_{s,a} |b_x|, u_{extrap} \right\}$

where u_{extrap} is the extrapolated value of the plasma poloidal velocity defined as $u = \sum\limits_{a=0}^{ns-1} \frac{\Gamma_{a,x}}{n_a}$.

$c_{s,a} = \sqrt{\frac{\gamma_i T_i + z_a^2 \frac{n_a}{n_e} T_e}{m_a}}$

where γ_i is the adiabatic ratio of specific heats (`GAMMAI` in `b2.boundary.parameters`).

East

Imposing the `b2stbc_spb` *style boundary condition for electron temperature:* `BCENE = 15`

To be used in conjunction with `BCENI = 15`, `BCCON = 14`, `BCMOM = 13`, and `BCPOT = 11`.

If `'bcene_15_style` is set to `'0'`:

$$-\frac{\sqrt{g}}{h_x}\widetilde{q}_{ex}\bigg|_E = \left[-\left(\frac{1+\gamma_e}{1-\gamma_e}T_e + e\Phi\right)\max\left(0, \frac{\sqrt{g}}{h_x}\sum_{a=0}^{ns-1} z_a p_{n,a,1}|b_x| \sqrt{\frac{\sum\limits_{\substack{b=0\\ z_b\neq 0}}^{ns-1} p_b}{\sum\limits_{\substack{b=0\\ z_b\neq 0}}^{ns-1} m_b n_b}} n_a - \frac{\sqrt{g}}{h_x} b_x \frac{j_\parallel}{e}\right) + \frac{\sqrt{g}}{h_x} q_{ex}^{P.Sch}\right]\bigg|_E$$

If `'bcene_15_style'` is set to `'1'` (recommended for stability, default):

$$-\frac{\sqrt{g}}{h_x}\widetilde{q}_{ex}\bigg|_E = \left[-\frac{\sqrt{g}}{h_x}|b_x|\left(\left(1+\gamma_e+p_{e,1}\right)T_e + \left(1-\gamma_e\right)e\left(\Phi - p_{\phi,2}\right)\right) n_e \frac{1}{\sqrt{2\pi}}\sqrt{\frac{T_e}{m_e}}\exp\left(-\frac{e\left(\Phi - p_{\Phi,2}\right)}{T_e}\right) + \frac{\sqrt{g}}{h_x} q_{ex}^{P.Sch}\right]\bigg|_E$$

with $p_{e,1}$ set by `ENEPAR(:,1)` and $p_{\Phi,2}$ set by `POTPAR(:,2)`.

In case of electron temperature or potential oscillations, one can make use of the `b2stbc_stab_coeff_sheath_te` switch, assigning it a positive value (recommended range between 1 and 100), to include an additional zero-sum stabilizing term.

Imposing the sheath boundary condition for electron temperature: `BCENE = 3`

$$-\frac{\sqrt{g}}{h_x}\widetilde{q}_{ex}\bigg|_E = -\left[\left(p_{h_e,1} + \frac{e\left(\Phi - p_{\Phi,2}\right)}{T_e}\right)\frac{\sqrt{g}}{h_x}\nu_{e_{out}} T_e + \frac{\sqrt{g}}{h_x} q_{ex}^{P.Sch}\right]\bigg|_E$$

$p_{h_e,1} = 0.9$, $p_{\Phi,2}$ is the applied electric potential at the target, in V.

If $p_{m,a_{main},2} < 0.5$:

$\nu_{e_{out}} = \max\left\{0, u_{u_{out}} n_e - \frac{j_x}{e}\right\}$, $u_{u_{out}} = p_{m,a_{main},1} c_s |b_x|$

$$c_s = \sqrt{\frac{\sum\limits_{\substack{a=0\\ z_a\neq 0}}^{ns-1} p_a}{\sum\limits_{\substack{a=0\\ z_a\neq 0}}^{ns-1} m_a n_a}}$$

If $p_{m,a_{main},2} > 0.5$:

$$\nu_{e_{out}} = \min\left\{0, \sum_{\substack{a=0\\ z_a\neq 0}}^{ns-1} u_{a_{out}} z_a n_a - \frac{j_x}{e}\right\}$$

$u_{a_{out}} = \max\left\{p_{m,a,1} c_{s,a} |b_x|, u_{extrap}\right\}$

$c_{s,a} = \sqrt{\frac{\gamma_i T_i + z_a^2 \frac{n_a}{n_e} T_e}{m_a}}$

C.7.5 Energy balance equation for ions

$S_{h_i} = S_{h_i,0} + S_{h_i,1} T_i + S_{h_i,2} n_i + S_{h_i,3} n_i T_i$

South. Private region

Imposing a decay length for the ion temperature: `BCENI = 9`

$p_{h_i,1}$ is the radial ion temperature decay length, in m.

$e_{rec,0} = 0$ and $e_{rec,1} = 0.3$ are energy recycling coefficients.

$r_{rec,0,0} = 0$ and $r_{rec,1,0} = 1$ are particle recycling coefficients.

$$\left.\frac{\sqrt{g}}{h_y}\widetilde{q}_{iy}\right|_{S_0} = \begin{cases} \left.\begin{bmatrix} \frac{5}{2}\sum_{a=0}^{ns-1} S_{n_a,0}^{no_mdf} T_i - \frac{\sqrt{g}}{h_y}\frac{\kappa_i^{(AN)}}{p_{h_i,1}} T_i + \frac{5}{2}\sum_{a=0}^{ns-1} S_{n_a,1}^{no_mdf} n_a T_i + \\ + c_{nsr}\left(\frac{3}{2} + BoRiS\right)\sum_{a=0}^{ns-1} \max\left\{e_{rec,a}\left(T_i + z_a\left(\Phi - p_{\Phi,2}\right)\right), \right. \\ \left. k_B T_{surf}\right\} r_{rec,a,0} \max\left\{0, -\frac{\sqrt{g}}{h_y}\Gamma_{ay}\right\} \end{bmatrix}\right|_{S_0} & \text{mdf_fhi} = 0 \\ \left.\begin{bmatrix} \frac{5}{2}\sum_{a=0}^{ns-1} S_{n_a,0}^{no_mdf} T_i - \frac{\sqrt{g}}{h_y}\frac{\kappa_i^{(AN)}}{p_{h_i,1}} T_i + \frac{5}{2}\sum_{a=0}^{ns-1} S_{n_a,1}^{no_mdf} n_a T_i \\ - q_{iy}^{P.Sch} + \frac{\sqrt{g}}{h_y}\frac{3}{2}\sum_{a=0}^{ns-1} V_{ay}^{(E)} n_a T_i + \\ + c_{nsr}\left(\frac{3}{2} + BoRiS\right)\sum_{a=0}^{ns-1} \max\left\{e_{rec,a}\left(T_i + z_a\left(\Phi - p_{\Phi,2}\right)\right), \right. \\ \left. k_B T_{surf}\right\} r_{rec,a,0} \max\left\{0, -\frac{\sqrt{g}}{h_y}\Gamma_{ay}\right\} \end{bmatrix}\right|_{S_0} & \text{mdf_fhi} = 1 \end{cases}$$

Imposing a constant ion temperature: `BCENI = 21`

$$\left.\frac{\sqrt{g}}{h_y}\widetilde{q}_{iy}\right|_{S_0} = p_{h_i,2}\sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1}\left(\frac{\sqrt{g}}{h_y} n_a \sqrt{\frac{z_a T_e + T_i}{m_a}}\right) T_i\Big|_{S_0} + p_{h_i,1}\max\left\{0, -\frac{\sqrt{g}}{h_y}\widetilde{f}_{iy}\right\} T_i\Big|_{S_0}$$

$p_{h_i,1} = -1$, $p_{h_i,2} = -10^{-2}$

Imposing a decay length for the ion temperature: `BCENI = 22`

$$-\left.\frac{\sqrt{g}}{h_y}\kappa_{iy}\frac{\partial T_i}{h_y \partial y}\right|_{S_0} = p_{h_i,1}\sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1}\left(\frac{\sqrt{g}}{h_y} n_a \sqrt{\frac{z_a T_e + T_i}{m_a}}\right) T_i\Big|_{S_0}$$

$p_{h_i,1}$ is the imposed ion temperature decay length, in m.

`b2stbc_spb` *boundary condition:*

$$\left.\frac{\sqrt{g}}{h_y}\widetilde{q}_{iy}\right|_{S_0} = \sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1}\left(c_{h_i,a,2}\frac{\sqrt{g}}{h_y}\frac{n_a}{n_i}\sqrt{\frac{z_a T_e + T_i}{m_a}}\right) n_i T_i\Big|_{S_0} + c_{h_i,a,7}\max\left\{0, -\frac{\sqrt{g}}{h_y}\widetilde{f}_{iy}\right\} T_i\Big|_{S_0}$$

$c_{h_i,a,2} = -10^{-2}$

South 1. Core

Imposing a total ion energy flux: `BCENI = 8`

$$\frac{\sqrt{g}}{h_y}\widetilde{q}_{iy}\Big|_{S_1} = \left[p_{h_i,1} \frac{\frac{\sqrt{g}}{h_y}}{\sum\limits_{core}\left(\frac{\sqrt{g}}{h_y}\right)} - q_{iy}^{P.Sch} + \frac{3}{2}\frac{\left(j_y^{(AN)} + j_y^{(in)} + \widetilde{j}_y^{(vis\parallel)} + \widetilde{j}_y^{(visq)} + j_y^{(vis\perp)}\right)}{e} T_i \right]\Bigg|_{S_1}$$

$p_{h_i,1}$ is the imposed total ion energy flux, in W.

`b2stbc_spb` *boundary condition:*

$$\frac{\sqrt{g}}{h_y}\widetilde{q}_{iy}\Big|_{S_1} = \sum_{a=0}^{ns-1}\left[\left(c_{h_i,a,0}\frac{\sqrt{g}}{h_y}\frac{n_a}{n_i} + c_{h_i,a,1}\frac{\sqrt{g}}{h_y}\frac{n_a}{n_i}T_i\right)\right]\Big|_{S_1}$$

$c_{h_i,a,0} = 4\cdot 10^{13}$, $c_{h_i,a,1} = -10^{30}$, $a = 0.1$.

North

Imposing an ion temperature decay length: `BCENE = 9`

$p_{h_i,1}$ is the radial ion temperature decay length, in m.

$e_{rec,0} = 0$ and $e_{rec,1} = 0.3$ are energy recycling coefficients.

$r_{rec,0,0} = 0$ and $r_{rec,1,0} = 1$ are particle recycling coefficients.

$$-\frac{\sqrt{g}}{h_y}\widetilde{q}_{iy}\Big|_N = \begin{cases} \left[\begin{array}{c} \frac{5}{2}\sum\limits_{a=0}^{ns-1} S_{n_a,0}^{no_mdf} T_i - \frac{\sqrt{g}}{h_y}\frac{\kappa_i^{(AN)}}{p_{h_i,1}} T_i + \frac{5}{2}\sum\limits_{a=0}^{ns-1} S_{n_a,1}^{no_mdf} n_a T_i \\ +c_{nsr}\left(\frac{3}{2} + BoRiS\right)\sum\limits_{a=0}^{ns-1}\max\left\{e_{rec,a}\left(T_i + z_a\left(\Phi - p_{\Phi,2}\right)\right), \right. \\ \left. k_B T_{surf}\right\} r_{rec,a,0}\max\left\{0, -\frac{\sqrt{g}}{h_y}\Gamma_{ay}\right\} \end{array}\right]\Big|_N & \text{mdf_fhi} = 0 \\ \left[\begin{array}{c} \frac{5}{2}\sum\limits_{a=0}^{ns-1} S_{n_a,0}^{no_mdf} T_i - \frac{\sqrt{g}}{h_y}\frac{\kappa_i^{(AN)}}{p_{h_i,1}} T_i + \frac{5}{2}\sum\limits_{a=0}^{ns-1} S_{n_a,1}^{no_mdf} n_a T_i \\ -q_{iy}^{P.Sch} + \frac{\sqrt{g}}{h_y}\frac{3}{2}\sum\limits_{a=0}^{ns-1} V_{ay}^{(E)} n_a T_i \\ +c_{nsr}\left(\frac{3}{2} + BoRiS\right)\sum\limits_{a=0}^{ns-1}\max\left\{e_{rec,a}\left(T_i + z_a\left(\Phi - p_{\Phi,2}\right)\right), \right. \\ \left. k_B T_{surf}\right\} r_{rec,a,0}\max\left\{0, -\frac{\sqrt{g}}{h_y}\Gamma_{ay}\right\} \end{array}\right]\Big|_N & \text{mdf_fhi} = 1 \end{cases}$$

Boundary condition from `b2stbc_spb`*:* `BCENI = 21`

$$-\frac{\sqrt{g}}{h_y}\widetilde{q}_{iy}\Big|_N = p_{h_i,2}\sum_{\substack{a=0\\ z_a\neq 0}}^{ns-1}\left(\frac{\sqrt{g}}{h_y} n_a \sqrt{\frac{z_a T_e + T_i}{m_a}}\right) T_i\Big|_N + p_{h_i,1}\max\left\{0, -\frac{\sqrt{g}}{h_y}\widetilde{f}_{iy}\right\} T_i\Big|_N$$

$p_{h_i,1} = -1$, $p_{h_i,2} = 10^{-2}$

Prescribing a decay length for the ion temperature: `BCENI = 22`

$$\frac{\sqrt{g}}{h_y} k_{iy}\frac{\partial T_i}{h_y \partial y}\Big|_N = p_{h_i,1}\sum_{\substack{a=0\\ z_a\neq 0}}^{ns-1}\left(\frac{\sqrt{g}}{h_y} n_a \sqrt{\frac{z_a T_e + T_i}{m_a}}\right) T_i\Big|_N$$

$p_{h_i,1}$ is the imposed ion temperature decay length, in m.

Boundary condition in `b2stbc_spb`*:*

$$-\frac{\sqrt{g}}{h_y}\widetilde{q}_{iy}\bigg|_N = \sum_{\substack{a=0\\ z_a\neq 0}}^{ns-1}\left(c_{h_i,a,2}\frac{\sqrt{g}}{h_y}\frac{n_a}{n_i}\sqrt{\frac{z_aT_e+T_i}{m_a}}\right)n_iT_i\bigg|_N + c_{h_i,a,7}\max\left\{0,-\frac{\sqrt{g}}{h_y}\widetilde{f}_{iy}\right\}T_i\bigg|_N$$

$c_{h_i,a,2} = -10^{-2}$

West

Drift-compatible, Bohm-Chodura sheath entrance condition `BCENI = 15`

- Marginal imposition, with collective sound speed: `MOMPAR(ismain,,2)` = $p_{m,ismain,2} < 0.5$

To be used in conjunction with `BCENE = 15`, `BCCON = 14`, `BCMOM = 13`, and `BCPOT = 11`.

$$\frac{\sqrt{g}}{h_x}\widetilde{q}_{ix}\bigg|_W = -p_{h_i,1}\frac{\sqrt{g}}{h_x}|b_x|\sqrt{\frac{\sum_{\substack{b=0\\ z_b\neq 0}}^{ns-1}p_b}{\sum_{\substack{b=0\\ z_b\neq 0}}^{ns-1}m_bn_b}}T_i\sum_{\substack{a=0\\ z_a\neq 0}}^{ns-1}p_{n,a,1}n_a - \frac{\sqrt{g}}{h_x}q_{ix}^{P.Sch}\bigg|_W$$

$p_{h_i,1} = +1.5$

In case of ion temperature oscillations, one can make use of the `b2stbc_stab_coeff_sheath_ti` switch, assigning it a positive value (recommended range between 1 and 100), to include an additional zero-sum stabilizing term.

- Non-marginal imposition, with individual species sound speed: `MOMPAR(ismain,,2)` = $p_{m,ismain,2} \geq 0.5$

to be used in conjunction with `BCMOM = 13`, `BCENE = 15` with `bcene_15_style = 1`, `BCCON = 14`, `BCPOT = 3` or `BCPOT = 11` with `istyle_fchi = 0`.

This condition imposes that the ion heat flux, minus its Pfirsch-Schlueter component, is equal to the ion convective particle flux (excluding its Pfirsch-Schlueter component) times the ion sheath transmission coefficient.

The parameter $p_{i,1}$ is set by `ENIPAR(:,1)`.

$$\frac{\sqrt{g}}{h_x}\widetilde{q}_{ix}\bigg|_W = -p_{i,1}\frac{\sqrt{g}}{h_x}\sum_a\left(-\widetilde{\Gamma}_{ax} - D_{axy}\frac{1}{h_y}\frac{\partial n_aT_i}{\partial y}\right)T_i - \frac{\sqrt{g}}{h_x}q_{ix}^{P.Sch}\bigg|_W$$

In case of ion temperature or potential oscillations, one can make use of the `b2stbc_stab_coeff_sheath_ti` switch, assigning it a positive value (recommended range between 1 and 100), to include an additional zero-sum stabilizing term.

Boundary condition from `b2stbc_spb`*:*

$$\frac{\sqrt{g}}{h_x}\widetilde{q}_{ix}\bigg|_W = \left[\sum_{\substack{a=0\\ z_a\neq 0}}^{ns-1}\left(c_{h_i,a,7}\frac{\sqrt{g}}{h_x}|b_x|\sqrt{\frac{\sum_{\substack{b=0\\ z_b\neq 0}}^{ns-1}p_b}{\sum_{\substack{b=0\\ z_b\neq 0}}^{ns-1}m_bn_b}}\frac{n_a}{n_i}\right)n_iT_i - \frac{\sqrt{g}}{h_x}q_{ix}^{P.Sch}\right]\bigg|_W$$

Imposing a sheath ion boundary condition: `BCENI = 3`

$$\left.\frac{\sqrt{g}}{h_x}\widetilde{q}_{ix}\right|_W = \left.\begin{bmatrix} -\frac{\sqrt{g}}{h_x}p_{h_i,1}\sum\limits_{z_a\neq 0}\left(u_{u_{out}}n_a\right)T_i - \frac{\sqrt{g}}{h_x}q_{ix}^{P.Sch} \\ +c_{nsr}\left(\frac{3}{2}+BoRiS\right)\sum\limits_{a=0}^{ns-1}\max\left\{r_{rec,a,1}\left(T_i+z_a\left(\Phi-p_{\Phi,2}\right)\right),\right. \\ \left. k_BT_{surf}\right\}r_{rec,a,0}\max\left\{0,-\frac{\sqrt{g}}{h_x}\Gamma_{ax}\right\} \end{bmatrix}\right|_W$$

$p_{m,a_{main},1} = 1.0$

If $p_{m,a_{main},2} < 0.5$:

$u_{u_{out}} = p_{m,a_{main},1}c_s|b_x|$,

$$c_s = \sqrt{\frac{\sum\limits_{\substack{a=0\\z_a\neq 0}}^{ns-1} p_a}{\sum\limits_{\substack{a=0\\z_a\neq 0}}^{ns-1} m_a n_a}} \tag{C.257}$$

If $p_{m,a_{main},2} > 0.5$:

$c_{s,a} = \sqrt{\frac{\gamma_i T_i + z_a^2\frac{n_a}{n_e}T_e}{m_a}}$

where γ_i is the adiabatic ratio of specific heats (`GAMMAI` in `b2.boundary.parameters`),

$u_{u_{out}} = \max\left\{p_{m,a_{main},1}c_{s,a}, u_{extrap}\right\}$

If `neutrals_namelist.eq.1.or.use_eirene.ne.0` then $e_{rec,0} = 0$, $e_{rec,1} = 0.3$, $r_{rec,0,0} = 0$, $r_{rec,1,0} = 1$

else $r_{rec,0,0} = 0$, $r_{rec,1,0} = 1$, $r_{rec,0,1} = 0$, $r_{rec,1,1} = 0.3$

$p_{h_i,1} = 2.5$ should be $p_{h_i,1} = 1.5$

$p_{h_i,2} = 0$ without drifts

$p_{h_i,2} = 10^{-3}$ with drifts, should be $p_{h_i,2} = 0$

East

Drift-compatible, Bohm-Chodura sheath entrance condition `BCENI = 15`

- Marginal imposition, with collective sound speed: MOMPAR(ismain,,2) = $p_{m,ismain,2} < 0.5$

To be used in conjunction with `BCENE = 15`, `BCCON = 14`, `BCMOM = 13`, and `BCPOT = 11`.

$$-\left.\frac{\sqrt{g}}{h_x}\widetilde{q}_{ix}\right|_E = -p_{h_i,1}\frac{\sqrt{g}}{h_x}|b_x|\left.\sqrt{\frac{\sum\limits_{\substack{b=0\\z_b\neq 0}}^{ns-1} p_b}{\sum\limits_{\substack{b=0\\z_b\neq 0}}^{ns-1} m_b n_b}T_i\sum_{\substack{a=0\\z_a\neq 0}}^{ns-1} p_{n,a,1}n_a + \frac{\sqrt{g}}{h_x}q_{ix}^{P.Sch}}\right|_E$$

$p_{h_i,1} = +1.5$

In case of ion temperature oscillations, one can make use of the `b2stbc_stab_coeff_sheath_ti` switch, assigning it a positive value (recommended range between 1 and 100), to include an additional zero-sum stabilizing term.

- Non-marginal imposition, with individual species sound speed: `MOMPAR(ismain,,2)` = $p_{m,ismain,2} \geq 0.5$

to be used in conjunction with `BCMOM = 13`, `BCENE=15` with `bcene_15_style = 1`, `BCCON = 14`, `BCPOT = 3` or `BCPOT = 11` with `istyle_fchi = 0`.

This condition imposes that the ion heat flux, minus its Pfirsch-Schlueter component, is equal to the ion convective particle flux (excluding its Pfirsch-Schlueter component) times the ion sheath transmission coefficient.

The parameter $p_{i,1}$ is set by `ENIPAR(:,1)`.

$$-\left.\frac{\sqrt{g}}{h_x}\widetilde{q}_{ix}\right|_E = -p_{i,1}\left.\frac{\sqrt{g}}{h_x}\sum_a\left(\widetilde{\Gamma}_{ax} + D_{axy}\frac{1}{h_y}\frac{\partial n_a T_i}{\partial y}\right)T_i + \frac{\sqrt{g}}{h_x}q_{ix}^{P.Sch}\right|_E$$

In case of ion temperature or potential oscillations, one can make use of the `b2stbc_stab_coeff_sheath_ti` switch, assigning it a positive value (recommended range between 1 and 100), to include an additional zero-sum stabilizing term.

Boundary condition from `b2stbc_spb`*:*

$$-\left.\frac{\sqrt{g}}{h_x}\widetilde{q}_{ix}\right|_E = \left.\left[-\sum_{\substack{a=0\\z_a\neq 0}}^{ns-1}\left(c_{h_i,a,7}\frac{\sqrt{g}}{h_x}|b_x|\sqrt{\frac{\sum_{\substack{b=0\\z_b\neq 0}}^{ns-1}p_b}{\sum_{\substack{b=0\\z_b\neq 0}}^{ns-1}m_b n_b}}\frac{n_a}{n_i}\right)n_i T_i + \frac{\sqrt{g}}{h_x}q_{ix}^{P.Sch}\right]\right|_E$$

Imposing a sheath ion boundary condition: `BCENI = 3`

$$-\left.\frac{\sqrt{g}}{h_x}\widetilde{q}_{ix}\right|_E = \left.\begin{bmatrix} -\sum_{z_a\neq 0}\left(\frac{\sqrt{g}}{h_x}p_{h_i,1}u_{u_{out}}n_a T_i\right) + \frac{\sqrt{g}}{h_x}q_{ix}^{P.Sch} \\ +c_{nsr}\left(\frac{3}{2} + BoRiS\right)\sum_{a=0}^{ns-1}\max\left\{r_{rec,a,1}\left(T_i + z_a\left(\Phi - p_{\Phi,2}\right)\right), \right. \\ \left. k_B T_{surf}\right\} r_{rec,a,0}\max\left\{0, \frac{\sqrt{g}}{h_x}\Gamma_{ax}\right\} \end{bmatrix}\right|_E,$$

$p_{m,a_{main},1} = 1.0$

If $p_{m,a_{main},2} < 0.5$:

$u_{u_{out}} = p_{m,a_{main},1}c_s|b_x|$,

$$c_s = \sqrt{\frac{\sum_{\substack{a=0\\z_a\neq 0}}^{ns-1}p_a}{\sum_{\substack{a=0\\z_a\neq 0}}^{ns-1}m_a n_a}}$$

If $p_{m,a_{main},2} > 0.5$:

$c_{s,a} = \sqrt{\frac{\gamma_i T_i + z_a^2\frac{n_a}{n_e}T_e}{m_a}}$

where γ_i is the adiabatic ratio of specific heats (`GAMMAI` in `b2.boundary.parameters`),

$u_{u_{out}} = \max\{p_{m,a,1} c_{s,a}, u_{extrap}\}$

If `neutrals_namelist.eq.1.or.use_eirene.ne.0` then $e_{rec,0} = 0$, $e_{rec,1} = 0.3$, $r_{rec,0,0} = 0$, $r_{rec,1,0} = 1$

else $r_{rec,0,0} = 0$, $r_{rec,1,0} = 1$, $r_{rec,0,1} = 0$, $r_{rec,1,1} = 0.3$

$p_{h_i,1} = 2.5$, should be $p_{h_i,1} = 1.5$

$p_{h_i,2} = 0$, without drifts

$p_{h_i,2} = 10^{-3}$ with drifts, should be $p_{h_i,2} = 0$,

C.8 Total energy balance

The aim of this section is to derive an equation which resembles the total energy balance equation, i.e. has the form

$$\frac{\partial \varepsilon}{\partial t} + \left(\nabla \cdot \vec{\mathbf{q}}^{\mathrm{ENERGY}}\right) = \left(\vec{\mathbf{j}} \cdot \vec{\mathbf{E}}\right) + V_{e\|} R_{e\|}^{\mathrm{INELASTIC}} + \sum_a V_{a\|} R_{a\|}^{\mathrm{INELASTIC}} + Q_e^{\mathrm{ATOMIC}} + Q_i^{\mathrm{ATOMIC}} + Q^{\mathrm{res}} \quad \text{(C.258)}$$

from the code equations written in the sections above. Then it is explained how to compute each term of the equation (C.258), including the residual term Q^{res}, making use of the quantites available from the code output activated by the switch `b2wdat_iout = 4` (see Appendix H).

The equation (C.258) may be derived from the code equations, following [47], as

$$\text{Eq. (C.105)} + \text{Eq. (C.154)} + \sum_a V_{a\|} \cdot (\text{Eq. (C.55)}) - \sum_a \frac{m_a V_{a\|}^2}{2} \cdot (\text{Eq. (C.1)}) \quad \text{(C.259)}$$

The term $\left\{V_{e\|} \cdot (\text{ electron parallel force balance equation })\right\}$ does not appear in the sum (C.259), because **SOLPS-ITER** neglects electron inertia, so the corresponding equation does not appear explicitly in the B2.5 equation system and there is no term involving the electron kinetic energy $\frac{m_e u_e^2}{2}$. One should make use of equations (C.133) and (C.134) instead. Similarly, the electron continuity equation (or its equivalent (C.191)) does not enter the sum (C.259) either.

The kinetic energy of perpendicular motion is neglected as well for the same reason — the inertia terms are neglected in the perpendicular force balance equations (even if these equations are implicitly taken into account).

Thus the total energy density is

$$\varepsilon = \sum_a \frac{m_a V_{a\|}^2}{2} + \frac{3}{2} \sum_a n_a T_i + \frac{3}{2} n_e T_e. \quad \text{(C.260)}$$

The energy flux

$$\vec{\mathbf{q}}^{\mathrm{ENERGY}} = \vec{\mathbf{q}}_e^{\mathrm{ENERGY}} + \vec{\mathbf{q}}_i^{\mathrm{ENERGY}} \quad \text{(C.261)}$$

should be associated with the motion of guiding centers in its convective part, BUT it should NOT contain $\vec{\tilde{\mathbf{\Gamma}}}_{\mathbf{a}}$, $\vec{\tilde{\mathbf{q}}}_{\mathbf{e}}$ and $\vec{\tilde{\mathbf{q}}}_{\mathbf{i}}$, since these are modified fluxes where divergence-free components have been analytically removed for numerical stability reasons.

The explicit expressions may be rewritten in components:

$$q_{ex}^{\mathrm{ENERGY}} = \frac{5}{2}\left[\sum_a z_a \Gamma_{ax}^{\mathrm{ENERGY}} - \frac{j_x^{(\|)}}{e} - n_e T_e \frac{B_z}{e}\frac{\partial}{h_y \partial y}\left(\frac{1}{B^2}\right)\right] T_e - \kappa_{ex}\frac{1}{h_x}\frac{\partial T_e}{\partial x} - \tilde{c}_e^{(2)}\frac{j_x^{(\|)}}{e}T_e \tag{C.262}$$

$$q_{ey}^{\mathrm{ENERGY}} = \frac{5}{2}\left[\sum_a z_a \Gamma_{ay}^{\mathrm{ENERGY}} + n_e T_e \frac{B_z}{e}\frac{\partial}{h_x \partial x}\left(\frac{1}{B^2}\right)\right] T_e - \kappa_{ey}\frac{1}{h_y}\frac{\partial T_e}{\partial y} \tag{C.263}$$

$$q_{ix}^{\mathrm{ENERGY}} = \sum_a \left(\Gamma_{ax}^{\mathrm{ENERGY}} + n_a \tilde{V}_{ax}^{(dia)} + \frac{j_x^{(AN)} + j_x^{(in)} + \widetilde{j}_x^{(vis\|)} + \widetilde{j}_x^{(vis\perp)} + \widetilde{j}_x^{(visq)}}{e}\right) \times \left(\frac{m_a n_a V_{a\|}^2}{2} + \frac{5}{2} n_a T_i\right) - \kappa_{ix}\frac{1}{h_x}\frac{\partial T_i}{\partial x} \tag{C.264}$$

$$q_{iy}^{\mathrm{ENERGY}} = \sum_a \left(\Gamma_{ax}^{\mathrm{ENERGY}} + n_a \tilde{V}_{ax}^{(dia)} + \frac{j_y^{(AN)} + j_y^{(in)} + \widetilde{j}_y^{(vis\|)} + \widetilde{j}_y^{(vis\perp)} + \widetilde{j}_y^{(visq)}}{e}\right) \times \left(\frac{m_a V_{a\|}^2}{2} + \frac{5}{2} T_i\right) - \kappa_{iy}\frac{1}{h_y}\frac{\partial T_i}{\partial y} \tag{C.265}$$

$$\Gamma_{ax}^{\mathrm{ENERGY}} = \left(b_x V_{\|a} + V_x^{(E)} + V_{ax}^{(AN)} + V_a^{corr_dpc}\right) n_a - D_{n,a}\frac{1}{h_x}\frac{\partial n_a}{\partial x} - \widetilde{D}_{p,ax}\frac{1}{h_x}\frac{\partial p_a}{\partial x} \tag{C.266}$$

$$\Gamma_{ay}^{\mathrm{ENERGY}} = \left(V_y^{(E)} + V_{ay}^{(AN)}\right) n_a - D_{n,a}\frac{1}{h_y}\frac{\partial n_a}{\partial y} - \widetilde{D}_{p,ay}\frac{1}{h_y}\frac{\partial p_a}{\partial y} \tag{C.267}$$

$R_{e\|}$ and $R_{a\|}$ are contributions from inelastic collisions to the parallel projection of friction forces acting on electrons and ions correspondingly. They are

$$R_{e\|} = 0; \tag{C.268}$$

$$R_{a\|} = S_{I_a}^m + S_{R_a}^m + S_{CX_a}^m + S_{AN_a}^m + S_{\mathrm{EIRENE}_{a\|}}^m \tag{C.269}$$

Q_e^{ATOMIC} and Q_i^{ATOMIC} are electron and ion energy sources and sinks due to inelastic collisions.

$$Q_e^{\mathrm{ATOMIC}} = Q_R + Q_e^{\mathrm{EIRENE}} \tag{C.270}$$

$$Q_i^{\mathrm{ATOMIC}} = Q_I^{(i)} + Q_R^{(i)} + Q_i^{\mathrm{EIRENE}} \tag{C.271}$$

The residual term Q^{res} consists of residuals coming from the inaccuracies of code equations and from numerical approximation. They can be grouped according to their origin as follows:

$$Q^{\mathrm{res}} = Q_{\mathrm{visc}} + Q_{\mathrm{inert}}^{\mathrm{res}} + \left(\left[\vec{\mathbf{j}} - \left(\vec{\mathbf{j}}\cdot\vec{\mathbf{b}}\right)\vec{\mathbf{b}} - \vec{\mathbf{j}}_{\mathrm{dia}}\right]\cdot\vec{\mathbf{E}}\right) + Q_{\mathrm{j}_\perp}^{\mathrm{res}} + Q_{\mathrm{corr_dpc}}^{\mathrm{res}} + Q_{i,nTu_\|}^{\mathrm{res}} + Q_{e,dia}^{\mathrm{res}} + Q_{i,dia}^{\mathrm{res}} \tag{C.272}$$

The viscosity residual $Q_{\rm visc}$ comes from the fact that the term $\left(\overset{\leftrightarrow}{\pi}_a \cdot \vec{\mathbf{V}}_{\mathbf{a}}\right)$ does not appear in the code equations and cannot be extracted from the code output, therefore it is excluded from the total energy flux $\vec{\mathbf{q}}^{\rm ENERGY}$. Instead $Q_{\rm visc}$ appears in the right hand side in the form:

$$Q_{\rm visc} = \sum_a \left[\frac{4}{3} \overset{\sim}{\eta}{}^{(CL)}_{ax} \left(\frac{\partial V_{a\parallel}}{h_x \partial x} \right)^2 + \eta_a^{(AN)} \left[\left(\frac{\partial V_{a\parallel}}{h_x \partial x} \right)^2 + \left(\frac{\partial V_{a\parallel}}{h_y \partial y} \right)^2 \right] \right] - \sum_a \left[V_{a\parallel} \left(\left(\nabla \cdot \overset{\leftrightarrow}{\pi}{}^{V_\parallel}_a \right)_\parallel + \left(\nabla \cdot \overset{\leftrightarrow}{\pi}{}^{\perp}_a \right)_\parallel + \left(\nabla \cdot \overset{\leftrightarrow}{\pi}{}^{q}_a \right)_\parallel \right) \right] \quad \text{(C.273)}$$

which is in fact only the approximation to

$$\sum_a \left(\nabla \cdot \left(\overset{\leftrightarrow}{\pi}_a \cdot \vec{\mathbf{V}}_{\mathbf{a}} \right) \right) - \sum_a \left(\vec{\mathbf{b}} \cdot V_{a\parallel} - \left(\nabla \cdot \overset{\leftrightarrow}{\pi}_a \right) \right)$$

Here

$$\left(\nabla \cdot \overset{\leftrightarrow}{\pi}{}^{V_\parallel}_a \right)_\parallel = \frac{4}{3} \frac{1}{h_z \sqrt{g}} \frac{\partial}{\partial x} \left(\frac{h_z \sqrt{g}}{h_x} \overset{\sim}{\eta}{}^{(CL)}_{ax} \frac{\partial \ln h_z}{h_x \partial x} V_{\parallel a} - \overset{\sim}{\eta}{}^{(CL)}_{ax} \frac{\partial V_{\parallel a}}{h_x \partial x} \right) - \frac{4}{3} \frac{1}{h_z \sqrt{g}} \frac{\partial}{\partial x} \left(\frac{h_z \sqrt{g}}{h_x} \overset{\sim}{\eta}{}^{(CL)}_{ax} \frac{\partial \ln \left(h_z B^{\frac{1}{2}} \right)}{B^{\frac{1}{2}} h_x \partial x} \right) B^{\frac{1}{2}} V_{\parallel a}, \qquad a = a_{main} \quad \text{(C.274)}$$

$$\left(\nabla \cdot \overset{\leftrightarrow}{\pi}{}^{\perp}_a \right)_\parallel = - \frac{1}{h_z \sqrt{g}} \frac{\partial}{\partial x} \left(\frac{h_z \sqrt{g}}{h_x} \eta_a^{(AN)} \frac{\partial V_{\parallel a}}{h_x \partial x} \right) - \frac{1}{h_z \sqrt{g}} \frac{\partial}{\partial y} \left(\frac{h_z \sqrt{g}}{h_y} \eta_a^{(AN)} \frac{\partial V_{\parallel a}}{h_y \partial y} \right) \quad \text{(C.275)}$$

The inertia residual $Q^{\rm res}_{\rm inert}$ is the difference between what we want and what we have regarding the convective flux of the ion parallel kinetic energy:

$$\begin{aligned} Q^{\rm res}_{\rm inert} = & \\ & \frac{1}{\sqrt{g}} \frac{\partial}{\partial x} \left[\frac{\sqrt{g}}{h_x} \sum_a \left(\Gamma^{\rm ENERGY}_{ax} + n_a \tilde{V}^{(dia)}_{ax} + \frac{j^{(AN)}_{ax} + j^{(in)}_{ax} + \widetilde{j}^{(vis\parallel)}_{ax} + \widetilde{j}^{(vis\perp)}_{ax} + \widetilde{j}^{(visq)}_{ax}}{z_a e} \right) \frac{m_a V^2_{a\parallel}}{2} \right] + \\ & \frac{1}{\sqrt{g}} \frac{\partial}{\partial y} \left[\frac{\sqrt{g}}{h_y} \sum_a \left(\Gamma^{\rm ENERGY}_{ay} + n_a \tilde{V}^{(dia)}_{ay} + \frac{j^{(AN)}_{ay} + j^{(in)}_{ay} + \widetilde{j}^{(vis\parallel)}_{ay} + \widetilde{j}^{(vis\perp)}_{ay} + \widetilde{j}^{(visq)}_{ay}}{z_a e} \right) \frac{m_a V^2_{a\parallel}}{2} \right] - \\ & \sum_a V_{a\parallel} \left[\frac{1}{h_z \sqrt{g}} \frac{\partial}{\partial x} \left(\frac{h_z \sqrt{g}}{h_x} \Gamma^{Cor}_{ax} \right) + \frac{1}{h_z \sqrt{g}} \frac{\partial}{\partial y} \left(\frac{h_z \sqrt{g}}{h_y} \Gamma^{Cor}_{ay} \right) - S^m_{CF_a} \right] + \\ & \sum_a \frac{m_a V^2_{a\parallel}}{2} \left[\frac{1}{\sqrt{g}} \frac{\partial}{\partial x} \left(\frac{\sqrt{g}}{h_x} \widetilde{\Gamma}_{ax} \right) + \frac{1}{\sqrt{g}} \frac{\partial}{\partial y} \left(\frac{\sqrt{g}}{h_y} \widetilde{\Gamma}_{ay} \right) \right] \end{aligned} \quad \text{(C.276)}$$

This residual comes not only from numerical approximations, but also from the inaccuracies in the definition of $\left(\vec{\mathbf{b}} \cdot \left(\nabla \cdot m_a n_a \vec{V}_a \vec{V}_a \right) \right)$.

The Joule heating term $\left(\vec{\mathbf{j}} \cdot \vec{\mathbf{E}} \right)$ requires a separate discussion.

It may be noted that, analytically:

$$j_\parallel E_\parallel = \frac{1}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x}n_e T_e b_x V_{e\parallel}\right) - \frac{n_e T_e}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x}b_x V_{e\parallel}\right) + V_{e\parallel}\sum_a \left(S^m_{fr,ea} + S^m_{Therm,ea}\right) - \frac{b_x}{h_x}\frac{\partial \Phi}{\partial x}\sum_a Z_a e n_a V_{a\parallel} \quad \text{(C.277)}$$

and

$$\left(\vec{\mathbf{j}}_{\text{dia}}\cdot\vec{\mathbf{E}}\right) = \left(\nabla(n_e T_e + n_i T_i)\cdot\vec{\mathbf{V}}^{(E)}\right) \quad \text{(C.278)}$$

No other currents contribute to the total energy balance Eq. (C.258), which is why the term $\left(\left[\vec{\mathbf{j}} - \left(\vec{\mathbf{j}}\cdot\vec{\mathbf{b}}\right)\vec{\mathbf{b}} - \vec{\mathbf{j}}_{\text{dia}}\right]\cdot\vec{\mathbf{E}}\right)$ appears in the expression for the residual. Practically, it is only possible to compute the right hand sides of Eqs. (C.277) and (C.278), since terms like $j_\parallel E_\parallel$, $\left(\vec{\mathbf{j}}_{\text{dia}}\cdot\vec{\mathbf{E}}\right)$ or $\left(\left[\vec{\mathbf{j}} - \left(\vec{\mathbf{j}}\cdot\vec{\mathbf{b}}\right)\vec{\mathbf{b}} - \vec{\mathbf{j}}_{\text{dia}}\right]\cdot\vec{\mathbf{E}}\right)$ are not available in the code.

The term

$$Q^{\text{res}}_{\text{j}_\perp} = \frac{1}{\sqrt{g}}\frac{\partial}{\partial x}\left[\frac{\sqrt{g}}{h_x}\frac{T_i}{e}\left(j^{(AN)}_x + j^{(in)}_x + \widetilde{j}^{(vis\parallel)}_x + \widetilde{j}^{(vis\perp)}_x + \widetilde{j}^{(visq)}_x\right)\right] + \frac{1}{\sqrt{g}}\frac{\partial}{\partial y}\left[\frac{\sqrt{g}}{h_y}\frac{T_i}{e}\left(j^{(AN)}_y + j^{(in)}_y + \widetilde{j}^{(vis\parallel)}_y + \widetilde{j}^{(vis\perp)}_y + \widetilde{j}^{(visq)}_y\right)\right] \quad \text{(C.279)}$$

appears due to the approximate treatment of the ion perpendicular force balance equations.

The term

$$Q^{\text{res}}_{\text{corr_dpc}} = \frac{1}{\sqrt{g}}\frac{\partial}{\partial x}\left[\frac{\sqrt{g}}{h_x}\sum_a \left(n_e T_e + n_a T_i\right) V^{corr_dpc}_a\right] \quad \text{(C.280)}$$

is physically meaningless, but it may give a notable contribution to the total energy balance residual in cases, for example, where it compensates numerical oscillations in the parallel velocity.

The rest of the residual terms are analytically zero, but may be non-zero due to numerical discretization errors:

$$Q^{\text{res}}_{i,nTu_\parallel} = \sum_a \left[\frac{1}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x}n_a T_i b_x V_{a\parallel}\right) - \frac{n_a T_i}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x}b_x V_{a\parallel}\right) - \frac{b_x V_{a\parallel}}{h_x}\frac{\partial n_a T_i}{\partial x}\right] \quad \text{(C.281)}$$

$$Q^{\text{res}}_{e,dia} = \frac{5}{2}\frac{1}{\sqrt{g}}\frac{\partial}{\partial x}\left[-\frac{\sqrt{g}}{h_x}n_e T_e^2\frac{B_z}{e h_y}\frac{\partial}{\partial y}\left(\frac{1}{B^2}\right)\right] + \frac{1}{\sqrt{g}}\frac{\partial}{\partial y}\left[\frac{\sqrt{g}}{h_y}n_e T_e^2\frac{B_z}{e h_x}\frac{\partial}{\partial x}\left(\frac{1}{B^2}\right)\right] + \frac{1}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x}q^{P.Sch}_{ex}\right) + \frac{1}{\sqrt{g}}\frac{\partial}{\partial y}\left(\frac{\sqrt{g}}{h_y}q^{P.Sch}_{ey}\right) \quad \text{(C.282)}$$

$$Q^{\text{res}}_{i,dia} = \frac{5}{2}\sum_a \left[\frac{1}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x}n_a T_i \tilde{V}^{(dia)}_{ax}\right) + \frac{5}{2}\frac{1}{\sqrt{g}}\frac{\partial}{\partial y}\left(\frac{\sqrt{g}}{h_y}n_a T_i \tilde{V}^{(dia)}_{ay}\right) + \frac{1}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x}q^{P.Sch}_{ix}\right) + \frac{1}{\sqrt{g}}\frac{\partial}{\partial y}\left(\frac{\sqrt{g}}{h_y}q^{P.Sch}_{iy}\right)\right] \quad \text{(C.283)}$$

C.9 Parallel viscosity

Old form of $\left(\nabla\pi_a^{\parallel}\right)_{\parallel}$ was

$$\left(\nabla\pi_a^{\parallel}\right)_{\parallel} = \begin{bmatrix} 0, & a\neq 1 \\ -\frac{4}{3}b_x B^{\frac{3}{2}}\frac{\partial}{h_x\partial x}\left(\frac{c_{\eta_{ax}^{(CL)}}c_{\eta^{(CL)}}^{(Luciani)}\eta_{ax}^{(CL)}b_x}{B^2}\frac{\partial\left(B^{\frac{1}{2}}\left(V_{a\parallel}+0\cdot\frac{B}{B_x}V_{ax}^{(dia)}+0\cdot\frac{B}{B_x}V_x^{(E)}\right)\right)}{h_x\partial x}\right), & a=1 \\ -0\cdot B^{\frac{3}{2}}b_x\frac{\partial}{h_x\partial x}\left(\frac{b_x}{\nu_{aa}B^2}\frac{\partial\left(B^{\frac{1}{2}}\left(q_{a\parallel}^{(0)}+\frac{B}{B_x}q_{ax}^{(dia)}\right)\right)}{h_x\partial x}\right) & \end{bmatrix}$$

Contribution from anomalous viscosity was contained in l.h.s.

Because of $\eta_a^{(AN)} \ll \widetilde{\widetilde{\eta}}_{ax}^{(CL)}$, $\eta_{ax} \approx \widetilde{\widetilde{\eta}}_{ax}^{(CL)}$ and taking into account $B_x h_x h_y = \text{const}$

$$\begin{aligned} &-\frac{4}{3}b_x B^{\frac{3}{2}} && \frac{\partial}{h_x\partial x}\left(\frac{c_{\eta_{ax}^{(CL)}}c_{\eta^{(CL)}}^{(Luciani)}\eta_{ax}^{(CL)}b_x}{B^2}\frac{\partial\left(B^{\frac{1}{2}}V_{a\parallel}\right)}{h_x\partial x}\right) = \\ &\frac{\partial}{h_z\sqrt{g}\partial x}\left(\frac{h_z\sqrt{g}}{h_x}\left(\frac{4}{3}\eta_{ax}\frac{\partial\ln h_z}{h_x\partial x}V_{\parallel a}-\frac{4}{3}\eta_{ax}\frac{\partial V_{\parallel a}}{h_x\partial x}\right)\right) && \\ &-\frac{\partial}{h_z\sqrt{g}\partial x}\left(\frac{h_z\sqrt{g}}{h_x}\frac{4}{3}\eta_{ax}\frac{\partial\ln\left(h_z B^{\frac{1}{2}}\right)}{B^{\frac{1}{2}}h_x\partial x}\right)B^{\frac{1}{2}}V_{\parallel a}, && a=1 \end{aligned} \tag{C.284}$$

New form of $\left(\nabla\pi_a^{\parallel}\right)_{\parallel}$

$$\left(\nabla\pi_a^{\parallel}\right)_{\parallel} = \begin{bmatrix} 0, & a\neq 1 \\ \frac{\partial}{h_z\sqrt{g}\partial x}\left(\frac{h_z\sqrt{g}}{h_x}\left(\frac{4}{3}\eta_{ax}\frac{\partial\ln h_z}{h_x\partial x}V_{\parallel a}-\frac{4}{3}\eta_{ax}\frac{\partial V_{\parallel a}}{h_x\partial x}\right)\right)-\frac{\partial}{h_z\sqrt{g}\partial x}\left(\frac{h_z\sqrt{g}}{h_x}\frac{4}{3}\eta_{ax}\frac{\partial\ln\left(h_z B^{\frac{1}{2}}\right)}{B^{\frac{1}{2}}h_x\partial x}\right)B^{\frac{1}{2}}V_{\parallel a}, & a=1 \end{bmatrix}$$

C.10 Scaling of residuals

$$\text{Continuity equation residual for species } a: \frac{\sum_{i=0}^{n_x-1}\sum_{j=0}^{n_y-1}|r_{n_a,i,j}|}{\sum_{i=0}^{n_x-1}\sum_{j=0}^{n_y-1}\sqrt{g_{i,j}}n_{a,i,j}} \tag{C.285}$$

$$\text{Parallel momentum equation residual for species } a: \frac{\sum_{i=0}^{n_x-1}\sum_{j=0}^{n_y-1}|r_{u_a,i,j}|}{\sum_{i=0}^{n_x-1}\sum_{j=0}^{n_y-1}\sqrt{g_{i,j}}n_{a,i,j}\sqrt{m_a T_i}} \tag{C.286}$$

$$\text{Electron energy equation residual: } \frac{\sum_{i=0}^{n_x-1}\sum_{j=0}^{n_y-1}|r_{h_e,i,j}|}{\left(\frac{3}{2}+BoRiS\right)\sum_{a=0}^{ns-1}\sum_{i=0}^{n_x-1}\sum_{j=0}^{n_y-1}\sqrt{g_{i,j}}n_{a,i,j}\left(z_a T_e+T_i\right)} \tag{C.287}$$

$$\text{Ion energy equation residual:} \frac{\sum_{i=0}^{n_x-1}\sum_{j=0}^{n_y-1}|r_{h_i,i,j}|}{\left(\frac{3}{2}+BoRiS\right)\sum_{a=0}^{ns-1}\sum_{i=0}^{n_x-1}\sum_{j=0}^{n_y-1}\sqrt{g_{i,j}}n_{a,i,j}\left(z_aT_e+T_i\right)} \tag{C.288}$$

$$\text{Electric potential equation residual:} \frac{\sum_{i=0}^{n_x-1}\sum_{j=0}^{n_y-1}|r_{po,i,j}|}{\sum_{i=0}^{n_x-1}\sum_{j=0}^{n_y-1}\sqrt{g_{i,j}}en_{e,i,j}} \tag{C.289}$$

C.11 Zhdanov-Grad closure

C.11.1 Introduction

The closure scheme described in this Section is still experimental.

This section describes the equations and switches that are used when the switch `b2tral_Zhdanov_closure` is equal to `1` in `b2mn.dat`. It enables a more detailed calculation of the ion transport coefficients instead of the standard ion transport coefficients described above. By default `b2tral_Zhdanov_closure` is equal to `0`.

The relevant switches for the Zhdanov closure are:

b2tral_Zhdanov_closure The main switch that turns on the Zhdanov closure calculation, if set to `'1'`. Default is `'0'`.

b2sifr_Zhdanov_test If `'1'`, the Zhdanov closure calculation takes place, but the results are not used and the standard calculation is applied. Default is `'0'`. Meant for testing purposes.

b2sifr_Zhdanov_tf_fr If `'1'` (default), the Zhdanov thermal and friction forces formulations are applied.

b2tral_zhd_corr Multiplier to the corrections due to the difference in definition for temperature and Joule heating between Zhdanov and Braginskii. Default is `'1'`.

b2tral_zhd_corrT Multiplier to the corrections due to the difference in definition for temperature between Zhdanov and Braginskii in the thermal force calculation. Default is `'1'`.

b2tral_zhcscorr If `'1'` (default), the corrections for each individual ion charge state from the isonuclear sequence value are applied.

b2tral_zhcscorr_vq If `'0'`, the corrections for each individual ion charge state from the isonuclear sequence value is not applied for the heat stress viscosity. Default is `'1'`.

b2tral_zh_imp_analyt If `'1'`, the matrix inversion method is not used. Instead, the improved analytical expressions from [50] are applied. Default is `'0'`.

b2tral_zh_analyt_mdf If `'1'`, the MDF friction force calculation is applied (see [50]). Default is `'0'`.

b2tral_zhflcorr Flux-limiting coefficient multiplier. Default is `'3.0'`.

b2tral_Zhdanov_cond If `'1'` (default), the Zhdanov ion conductivity formulation is applied instead of the Braginskii expression.

b2tral_zhflcorrh Additional flux-limiting coefficient multiplier for the heat flux. Default is `'0.4'`.

b2tfhi_Zhdanov_vel_heat If `'1'` (default), the Zhdanov velocity-dependent ion heat flux is added.

b2tral_Zhdanov_visu If `'1'` (default), the Zhdanov velocity-dependent viscous stress formulation is applied instead of the Braginskii expression.

b2tral_Zhdanov_vish If `'1'` (default), the Zhdanov heat flux dependent viscous stress formulation is applied instead of the Braginskii expression.

b2tral_zh_tf_toff String of integers with (nspecies*(nspecies+1))/2-nspecies digits. If a digit is `1`, turn off the thermal force between corresponding species starting from 1 and 2; 1 and 3;....; up to nspecies-1 and nspecies. Default is `'0'`.

b2tral_zh_vis_type If `'0'` (default), the Braginskii-like velocity-dependent stress-viscosity is used. If `'1'`, the Zhdanov-like velocity-dependent stress-viscosity is used. If `'2'`, the Makarov-like velocity-dependent stress-viscosity is used [56].

b2npmo_smbvi_factor This switch defines the fraction of the cross-ion part of the stress-viscosity term (Eq. (C.380)) which is applied. The cross-ion part of the stress-viscosity term can lead to numerical problems. If so, it is recommended to start from `'0.0'` and increase it step by step. Default is `'1.0'`.

b2tral_amfact_on If `'1'`, the mass correction for the thermal and friction force calculation is applied (for details see [57]). Default is `'0'`.

b2tral_amfact_* If `b2tral_amfact_on.eq.1`, this multiplier is applied for the mass of the isonuclear sequence with number `'***'`. Recall the numbering of the isonuclear sequences starts at zero (e.g. `b2tral_amfact_000`, `b2tral_amfact_001`, ...). Default is `'1.0'`.

Legacy switches (using the reduced improved analytical expressions that were included in earlier 3.0.7 code versions; they can be applied, if `b2tral_Zhdanov_closure.eq.0` or `b2sifr_Zhdanov_test.eq.1` or `b2sifr_Zhdanov_tf_fr.eq.0`; details in subsection C.11.11):

b2npmo_impr_form_tf If `'1'`, the thermal force is calculated using reduced improved analytical expressions (corresponding to `b2sigp_style.eq.2`). Default is `'0'`.

b2npmo_impr_form_fr If `'1'`, the friction force is calculated using reduced improved analytical expressions (corresponding to `b2sigp_style.eq.2`). Default is `'0'`.

b2npmo_mass_multiplicator Using this number the impurity mass can be artificialy increased. Default is `'1.0'`.

C.11.2 Notation

Latin letters correspond to the indexing of the ion charge states:

$$\underbrace{D^0}_{a=0},\underbrace{D^{+1}}_{a=1},\underbrace{T^0}_{a=2},\underbrace{T^{+1}}_{a=3},\underbrace{He^0}_{a=4},\underbrace{He^{+1}}_{a=5},\underbrace{He^{+2}}_{a=6}...$$

Greek letters correspond to the indexing of the isonuclear sequences:

$$\underbrace{D}_{\alpha=0},\underbrace{T}_{\alpha=1},\underbrace{He}_{\alpha=2}...$$

For isonuclear sequence α and charge state Z, there is a one-to-one match with the Latin notation: $a(\alpha, Z)$.

C.11.3 Ion heat conductivity

If `Zhdanov_cond.eq.1`, then:

In the transport part of B2.5:

$$\tilde{\tilde{\kappa}}_{ix}^{CL} = c_{\kappa_{ix}}^{(Luciani)} \tilde{\kappa}_{ix}^{CL} \tag{C.290}$$

$$\overline{\tilde{\tilde{\kappa}}_{\alpha x}^{CL}} = c_{\kappa_{ix}}^{(Luciani)} \overline{\tilde{\kappa}_{\alpha x}^{CL}}, \quad \left(\frac{Z^2}{\overline{Z_\alpha^2}} \text{weighted} \right) \tag{C.291}$$

$$\tilde{\tilde{\kappa}}_{ax}^{CL} = c_{\kappa_{ix}}^{(Luciani)} \tilde{\kappa}_{ax}^{CL} \tag{C.292}$$

For plateau and banana correction, (`'b2trcl_lluciani'.eq.'3'`):

$$c_{\kappa_{ix}}^{(Luciani)} = 1.6 \cdot \frac{\varepsilon^{3/2}}{K_2} \tag{C.293}$$

$$K_2 = \frac{0.66 + 1.88 \cdot \sqrt{\varepsilon} - 1.54 \cdot \varepsilon}{1 + 1.03 \cdot \nu_*^{1/2} + 0.31 \nu_*} + \frac{1.17 \cdot \varepsilon^3 \cdot \nu_*}{1 + 0.74 \cdot \varepsilon^{3/2} \cdot \nu_*} \tag{C.294}$$

$$\nu_* = \frac{c(y)}{2\pi \left(\tau_{ii} V_{th}\right)^{eff}}, \quad \left(\tau_{ii} V_{th}\right)^{eff} = \sum_a \frac{n_a}{n_i} \tau_a^{(Zh)} \sqrt{\frac{T_i}{m_a}} \tag{C.295}$$

$$\tau_a^{(Zh)} = \tau_a^{(Br)} \cdot \frac{Z_{eff} n_e}{\sum_b \sqrt{\frac{2\mu_{ab}}{m_a}} Z_b^2 n_b}, \quad \tau_a^{(Br)} = \frac{3}{4\sqrt{\pi}} \frac{1}{Z_a^2 Z_{eff} n_e} \frac{\sqrt{m_a} T_i^{\frac{3}{2}}}{\ln \Lambda} \left(\frac{4\pi\epsilon_0}{e^2} \right)^2 \tag{C.296}$$

Where the Zhdanov ion heat conductivity is:

$$\tilde{\kappa}_{ix}^{CL} = \sum_\alpha \left(\overline{\tilde{\kappa}_{\alpha x}^{CL}} + zhcscorr \cdot \overline{\tilde{\kappa}_{\alpha x}^{B}} \left(\tilde{I}_\alpha - 1 \right) \right) \tag{C.297}$$

$$\kappa_{ix}^{CL} = \sum_\alpha \left(\overline{\kappa_{\alpha x}^{CL}} + zhcscorr \cdot \overline{\kappa_{\alpha x}^{B}} \left(\tilde{I}_\alpha - 1 \right) \right) \tag{C.298}$$

$$\tilde{I}_\alpha = \sum_Z \frac{1}{I_{a(\alpha,Z)}} \left(\frac{n_a}{\overline{n}_\alpha} \right)^2 \tag{C.299}$$

$$\tilde{\kappa}_{ax}^{CL} = \frac{n_a}{\overline{n}_\alpha} \left(\overline{\tilde{\kappa}_{\alpha x}^{CL}} + zhcscorr \cdot \overline{\tilde{\kappa}_{\alpha x}^{B}} \left(\frac{\overline{Z_\alpha^2}}{Z^2} - 1 \right) \right) \tag{C.300}$$

$$\kappa_{ax}^{CL} = \frac{n_a}{\overline{n}_\alpha} \left(\overline{\kappa_{\alpha x}^{CL}} + zhcscorr \cdot \overline{\kappa_{\alpha x}^{B}} \left(\frac{\overline{Z_\alpha^2}}{Z^2} - 1 \right) \right) \tag{C.301}$$

If `zh_imp_analyt.eq.0` ***then***

$$\overline{\tilde{\kappa}_{\alpha x}^{CL}} = -\frac{5}{2} b_x^2 \frac{\zeta_p}{\left(Z_{eff} n_e\right)^2} \frac{T_i}{m_p} \sum_\beta \tilde{\tilde{a}}_{\alpha\beta}^{fl\ h} \overline{n}_\alpha \overline{n}_\beta, \quad \left(\frac{\overline{Z^2}}{\overline{Z_\alpha^2}} \text{weighted} \right) \tag{C.302}$$

$$\tilde{\tilde{a}}_{\alpha\beta}^{fl\ h} = c_\alpha^{(F\ lim\ h)} \tilde{\tilde{a}}_{\alpha\beta} \tag{C.303}$$

$$\tilde{c}_\alpha^{(F\ lim\ h)} = \begin{cases} 1, & \text{inside the separatrix} \\ \dfrac{1}{\left(1 + \left(\dfrac{\frac{5}{2} \frac{\zeta_p}{\left(Z_{eff} n_e\right)^2 m_p} \left| \sum_\gamma \tilde{\tilde{a}}_{\alpha\gamma} \overline{n}_\gamma b_x \frac{\widetilde{\partial T_\gamma}}{h_x \partial x} \right|}{zhflcorrh \cdot zhflcorr \cdot c_{fl\ lim} \sqrt{\frac{T}{m_\alpha}}} \right)^{\gamma_1} \right)^{\frac{1}{\gamma_1}}}, & \text{outside the separatrix} \end{cases} \tag{C.304}$$

$$\overline{\kappa_{\alpha x}^{CL}} = -\frac{5}{2} b_x^2 \frac{\zeta_p}{\left(Z_{eff} n_e\right)^2} \frac{T_i}{m_p} \sum_\beta \tilde{\tilde{a}}_{\alpha\beta} \overline{n}_\alpha \overline{n}_\beta, \quad \left(\frac{\overline{Z^2}}{\overline{Z_\alpha^2}} \text{weighted} \right) \tag{C.305}$$

otherwise

$$\overline{\tilde{\kappa}_{\alpha x}^{CL}} = b_x^2 \frac{\zeta_p}{\left(Z_{eff} n_e\right)^2} \frac{T_i}{m_p} c_\alpha^{\left(h_T^A\ fl\right)} \sqrt{\frac{2m_p}{m_\alpha}} \overline{n}_\alpha^2 \left(\frac{Z_{eff} n_e}{\overline{Z_\alpha^2 n_\alpha}} \right)^2, \quad \left(\frac{\overline{Z^2}}{\overline{Z_\alpha^2}} \text{weighted} \right) \tag{C.306}$$

$$c_\alpha^{(h_T^A\ fl)} = \tilde{c}_\alpha^{(F\ lim\ h)} c_\alpha^{(h_T^A)} \tag{C.307}$$

$$\tilde{c}_\alpha^{(F\ lim\ h)} = \begin{cases} 1, & \text{inside the separatrix} \\ \dfrac{1}{\left(1 + \left(\dfrac{\frac{\zeta_p}{(Z_\alpha^2 n_\alpha)^2 m_p} \overline{n}_\alpha \sqrt{\frac{2m_p}{m_\alpha}} \left| c_\alpha^{(h_T^A)} b_x \frac{\widetilde{\partial T_\alpha}}{h_x \partial x} \right|}{zhflcorrh \cdot zhflcorr \cdot c_{fl\ lim} \sqrt{\frac{T}{m_\alpha}}} \right)^{\gamma_1} \right)^{\frac{1}{\gamma_1}}}, & \text{outside the separatrix} \end{cases} \tag{C.308}$$

$$\overline{\kappa_{\alpha x}^{CL}} = b_x^2 \frac{\zeta_p}{\left(Z_{eff} n_e\right)^2} \frac{T_i}{m_p} c_\alpha^{(h_T^A)} \sqrt{\frac{2m_p}{m_\alpha}} \overline{n}_\alpha^2 \left(\frac{Z_{eff} n_e}{\overline{Z_\alpha^2 n_\alpha}} \right)^2, \quad \left(\frac{\overline{Z^2}}{\overline{Z_\alpha^2}} \text{weighted} \right) \tag{C.309}$$

$$c_\alpha^{(h_T^A)} = \frac{125}{32} \frac{1}{\tilde{\Delta}_\alpha} \left(1 + \frac{433\sqrt{2}}{360} Z_{11\alpha}^s \right) \tag{C.310}$$

end

$$\overline{\tilde{\kappa}_{\alpha x}^{B}} = b_x^2 \frac{\zeta_p}{\left(Z_{eff} n_e\right)^2} \frac{T_i}{m_p} c_\alpha^{(F\ limB\ h)} c_\alpha^{(h_T^B)} \sqrt{\frac{2m_p}{m_\alpha}} \overline{n}_\alpha^2 \left(\frac{Z_{eff} n_e}{\overline{Z_\alpha^2 n_\alpha}} \right)^2 \tag{C.311}$$

$$\overline{\kappa_{\alpha x}^{B}} = b_x^2 \frac{\zeta_p}{\left(Z_{eff} n_e\right)^2} \frac{T_i}{m_p} c_\alpha^{(h_T^B)} \sqrt{\frac{2m_p}{m_\alpha}} \overline{n}_\alpha^2 \left(\frac{Z_{eff} n_e}{\overline{Z_\alpha^2 n_\alpha}} \right)^2 \tag{C.312}$$

$$c_\alpha^{(h_T^B)} = \frac{139750}{53471} \frac{1 + \frac{6928\sqrt{2}}{8385} Z_{11\alpha}^s}{D_\alpha^{part}} \tag{C.313}$$

where:

$$D_\alpha^{part} = 1 + \frac{204376\sqrt{2}}{160413} Z_{11\alpha}^s + \frac{72670\sqrt{2}}{53471} Z_{5\alpha}^s - \frac{76728\sqrt{2}}{53471} Z_{9\alpha}^s + \frac{360256}{160413} Z_{5\alpha}^s Z_{11\alpha}^s - \frac{101568}{53471} \left(Z_{9\alpha}^s\right)^2 \tag{C.314}$$

If `zh_imp_analyt.eq.0` ***then***

$$c_\alpha^{(F\ limB\ h)} = \begin{cases} 1, & \text{inside the separatrix} \\ 0.5 + 0.5 \cdot \tanh \left[10 \cdot \left(\frac{zhflcorrh \cdot zhflcorr \cdot c_{fl\ lim} \sqrt{\frac{T}{m_\alpha}}}{\frac{5}{2} \frac{\zeta_p}{\left(Z_{eff} n_e\right)^2 m_p} \left| \sum_\gamma \tilde{a}_{\alpha\gamma} \overline{n}_\gamma b_x \frac{\widetilde{\partial T_\gamma}}{h_x \partial x} \right|} - 1 \right) \right], & \text{outside the separatrix} \end{cases} \tag{C.315}$$

else

$$c_\alpha^{(F\ limB\ h)} = \begin{cases} 1, & \text{inside the separatrix} \\ 0.5 + 0.5 \cdot \tanh \left[10 \cdot \left(\frac{zhflcorrh \cdot zhflcorr \cdot c_{fl\ lim} \sqrt{\frac{T}{m_\alpha}}}{\frac{\zeta_p}{\left(\overline{Z_\alpha^2 n_\alpha}\right)^2 m_p} \overline{n}_\alpha \sqrt{\frac{2m_p}{m_\alpha}} \left| c_\alpha^{(h_T^A)} b_x \frac{\widetilde{\partial T_\alpha}}{h_x \partial x} \right|} - 1 \right) \right], & \text{outside the separatrix} \end{cases} \tag{C.316}$$

C.11.4 Velocity-dependent ion heat flux

If `Zhdanov_vel_heat.eq.1`, then:

$$\frac{\sqrt{g}}{h_x} h_{ix}^V = \frac{\sqrt{g}}{h_x} b_x T_i \sum_{\substack{a \\ z_a \neq 0}} n_a \sum_\beta c_{\beta\alpha(a)}^{(h_w)} \left(V_{a\|} - \overline{V}_{\beta\|} \right) \tag{C.317}$$

Therefore:

$$\begin{aligned} \widetilde{q}_{ix} = \widetilde{f}_{ix} T_i - \kappa_{ix} \frac{1}{h_x} \frac{\partial T_i}{\partial x} - q_{ix}^{P.Sch} = \\ = \frac{3}{2} \Gamma_{ix} T_i + V_{ix}^{(str)} n_i T_i + \frac{3}{2} \left(\frac{j_x^{(AN)} + j_x^{(in)} + \widetilde{j}_x^{(vis\|)} + j_x^{(vis\perp)} + \widetilde{j}_x^{(visq)}}{e} \right) T_i \\ - \kappa_{ix} \frac{1}{h_x} \frac{\partial T_i}{\partial x} - q_{ix}^{P.Sch} + b_x T_i \sum_\alpha \overline{n}_\alpha \sum_\beta c_{\beta\alpha}^{(h_w^A)} \left(\overline{V}_{\alpha\|} - \overline{V}_{\beta\|} \right) \end{aligned} \tag{C.318}$$

$$\widetilde{f}_{ix} = \frac{3}{2}\Gamma_{ix} + V_{ix}^{(str)} n_i + \frac{3}{2}\left(\frac{j_x^{(AN)} + j_x^{(in)} + \widetilde{j}_x^{(vis\|)} + j_x^{(vis\perp)} + \widetilde{j}_x^{(visq)}}{e}\right) + b_x \sum_\alpha \overline{n}_\alpha \sum_\beta c_{\beta\alpha}^{(h_w^A)} \left(\overline{V}_{\alpha\|} - \overline{V}_{\beta\|}\right)$$

$$q_{ix} = \left[\frac{3}{2}\Gamma_{ix} + \frac{5}{2}\sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} n_a \widetilde{V}_{ax}^{(dia)} + V_{ix}^{(str)} n_i + b_x \sum_\alpha \overline{n}_\alpha \sum_\beta c_{\beta\alpha}^{(h_w^A)} \left(\overline{V}_{\alpha\|} - \overline{V}_{\beta\|}\right) + \frac{3}{2}\left(\frac{j_x^{(AN)} + j_x^{(in)} + \widetilde{j}_x^{(vis\|)} + j_x^{(vis\perp)} + \widetilde{j}_x^{(visq)}}{e}\right)\right] T_i - \kappa_{ix}\frac{1}{h_x}\frac{\partial T_i}{\partial x} \quad \text{(C.319)}$$

$$q_{ix}^{(Eir)} = \frac{5}{2}\sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} \Gamma_{ax}^{(Eir)} T_i - \kappa_{ix}\frac{1}{h_x}\frac{\partial T_i}{\partial x} + h_{ix}^V \quad \text{(C.320)}$$

C.11.5 Thermal force

If `Zhdanov_tf_fr.eq.1`, then:

In the friction source part of B2.5:

For isonuclear sequence α and charge state Z:

$$S_{Therm,ia}^m = [a \to \alpha Z, \text{ then } a(\alpha, Z)] = R_{\alpha Z}^T = - I_a \sum_\beta \overline{n}_\beta \tilde{c}_{\alpha\beta}^{(R_T^A)} b_x \frac{\widetilde{\partial T_\beta}}{h_x \partial x} - zhcscorr \cdot \overline{n}_\alpha c_\alpha^{(R_T^B)} \left(\frac{n_a}{\overline{n}_\alpha} b_x \frac{\partial T_a}{h_x \partial x} - I_a b_x \frac{\widetilde{\partial T_\alpha}}{h_x \partial x}\right) \quad \text{(C.321)}$$

Temperature gradients:

$$\frac{\widetilde{\partial T_\alpha}}{h_x \partial x} = \sum_Z \frac{n_{a(\alpha,Z)}}{\overline{n}_\alpha} \frac{\partial T_{a(\alpha,Z)}}{h_x \partial x} \quad \text{(C.322)}$$

$$T_a = T_i + zhd_corr \cdot zhd_corrT \cdot \frac{1}{3} m_a \mathbf{w}_a^2 \quad \text{(C.323)}$$

$$T = T_i, \quad \mathbf{w}_a^2 = w_{a\|}^2 \quad \text{(C.324)}$$

(The drift velocity could be added to the equation above)

$$w_{a\|} = V_{a\|} - V_\|, \quad V_\| = \frac{1}{\rho}\sum_a \rho_a V_{a\|} \quad \text{(C.325)}$$

$$\rho = \sum_a \rho_a, \quad \rho_a = m_a n_a \tag{C.326}$$

$$I_a = \frac{Z^2 n_a}{\overline{Z_\alpha^2 n_\alpha}}, \quad \overline{n}_\alpha = \sum_Z n_{a(\alpha,Z)}, \quad \overline{Z_\alpha^2 n_\alpha} = \sum_Z Z^2 n_{a(\alpha,Z)} \tag{C.327}$$

$$Z_{eff} n_e = \sum_\alpha \overline{Z_\alpha^2 n_\alpha}, \quad \overline{Z_\alpha^2} = \frac{\overline{Z_\alpha^2 n_\alpha}}{\overline{n}_\alpha} \tag{C.328}$$

In the transport part of B2.5:

Thermal force coefficients are:

First coefficient is:

If `zh_imp_analyt.eq.0` ***then***

$$\tilde{c}_{\alpha\beta}^{(R_T^A)} = -\frac{5}{2}\sum_\gamma \left[\frac{\mu_{\alpha\gamma}}{m_\alpha} \check{G}_{\alpha\gamma}^{(2)} \left(\tilde{\check{a}}_{\alpha\beta}^{fl} - \frac{m_\alpha}{m_\gamma} \tilde{\check{a}}_{\gamma\beta}^{fl} \right) + \left(\frac{\mu_{\alpha\gamma}}{m_\alpha} \right)^2 \check{G}_{\alpha\gamma}^{(8)} \left(\tilde{\check{a}}_{\alpha+n_s\beta}^{mdf_fl} - \left(\frac{m_\alpha}{m_\gamma} \right)^2 \tilde{\check{a}}_{\gamma+n_s\beta}^{mdf_fl} \right) \right] \tag{C.329}$$

$$\tilde{\check{a}}_{\alpha\beta}^{fl} = c_\alpha^{(F\ lim)} \tilde{\check{a}}_{\alpha\beta} \tag{C.330}$$

$$\tilde{\check{a}}_{\alpha\beta}^{mdf_fl} = c_\alpha^{(F\ lim)} \tilde{\check{a}}_{\alpha\beta}^{mdf} \tag{C.331}$$

$$c_\alpha^{(F\ lim)} = \begin{cases} 1, & \text{inside the separatrix} \\ \dfrac{1}{\left(1 + \left(\dfrac{\frac{5}{2}\frac{\zeta_p}{\left(Z_{eff} n_e\right)^2 m_p} \left| \sum_\gamma \tilde{\check{a}}_{\alpha\gamma} \overline{n}_\gamma b_x \frac{\partial \widetilde{T_\gamma}}{h_x \partial x} \right|}{zhflcorr \cdot c_{fl\ lim} \sqrt{\frac{T}{m_\alpha}}} \right)^{\gamma_1} \right)^{\frac{1}{\gamma_1}}}, & \text{outside the separatrix} \end{cases} \tag{C.332}$$

$$\tilde{c}_{\alpha\beta}^{(R_T^A)\text{-}nofl} = -\frac{5}{2}\sum_\gamma \left[\frac{\mu_{\alpha\gamma}}{m_\alpha} \check{G}_{\alpha\gamma}^{(2)} \left(\tilde{\check{a}}_{\alpha\beta} - \frac{m_\alpha}{m_\gamma} \tilde{\check{a}}_{\gamma\beta} \right) + \left(\frac{\mu_{\alpha\gamma}}{m_\alpha} \right)^2 \check{G}_{\alpha\gamma}^{(8)} \left(\tilde{\check{a}}_{\alpha+n_s\beta}^{mdf} - \left(\frac{m_\alpha}{m_\gamma} \right)^2 \tilde{\check{a}}_{\gamma+n_s\beta}^{mdf} \right) \right] \tag{C.333}$$

else

$$\tilde{c}_{\alpha\beta}^{(R_T^A)} = \delta_{\alpha\beta} \sum_\gamma c_{\alpha\gamma}^{(R_T^A)} \frac{\overline{Z_\gamma^2 n_\gamma}}{\overline{Z_\alpha^2 n_\alpha}} - c_{\beta\alpha}^{(R_T^A)} \frac{\overline{Z_\alpha^2 n_\alpha}}{\overline{Z_\beta^2 n_\beta}} \tag{C.334}$$

$$c_{\alpha\beta}^{(R_T^A)} = \tilde{c}_\alpha^{(F\ lim)} c_{\alpha\beta}^{(R_T^A)\text{-}nofl} \tag{C.335}$$

$$\tilde{c}_{\alpha}^{(F\ lim)} = \begin{cases} 1, & \text{inside the separatrix} \\ \dfrac{1}{\left(1+\left(\dfrac{\frac{\zeta_p}{\left(Z_{\alpha}^2 n_{\alpha}\right)^2 m_p}\overline{n}_{\alpha}\sqrt{\frac{2m_p}{m_{\alpha}}}\left|c_{\alpha}^{(h_T^A)} b_x \frac{\widetilde{\partial T_{\alpha}}}{h_x \partial x}\right|}{zhflcorr\cdot c_{fl\ lim}\sqrt{\frac{T}{m_{\alpha}}}}\right)^{\gamma_1}\right)^{\frac{1}{\gamma_1}}}, & \text{outside the separatrix} \end{cases} \tag{C.336}$$

$$c_{\alpha}^{(h_T^A)} = \frac{125}{32}\frac{1}{\tilde{\Delta}_{\alpha}}\left(1+\frac{433\sqrt{2}}{360}Z_{11\alpha}^s\right) \tag{C.337}$$

$$\tilde{c}_{\alpha\beta}^{(R_T^A)_nofl} = \delta_{\alpha\beta}\sum_{\gamma} c_{\alpha\gamma}^{(R_T^A)_nofl}\frac{\overline{Z_{\gamma}^2 n_{\gamma}}}{\overline{Z_{\alpha}^2 n_{\alpha}}} - c_{\beta\alpha}^{(R_T^A)_nofl}\frac{\overline{Z_{\alpha}^2 n_{\alpha}}}{\overline{Z_{\beta}^2 n_{\beta}}} \tag{C.338}$$

$$c_{\alpha\beta}^{(R_T^A)_nofl} = \frac{1}{\tilde{\Delta}_{\alpha}}\frac{25\sqrt{2}}{16}\frac{\mu_{\alpha\beta}}{m_{\alpha}}\sqrt{\frac{\mu_{\alpha\beta}}{m_{\alpha}}}\left[\frac{3}{2}+\frac{433\sqrt{2}}{240}Z_{11\alpha}^s - \frac{1}{2}\frac{\mu_{\alpha\beta}}{m_{\alpha}}\left(1+\frac{23\sqrt{2}}{8}Z_{9\alpha}^s\right)\right] \tag{C.339}$$

$$\tilde{\Delta}_{\alpha} = \frac{5629}{1152}Z_{5\alpha}^s Z_{11\alpha}^s - \frac{529}{128}{Z_{9\alpha}^s}^2 + \frac{65\sqrt{2}}{32}Z_{5\alpha}^s + \frac{433\sqrt{2}}{288}Z_{11\alpha}^s - \frac{23\sqrt{2}}{16}Z_{9\alpha}^s + 1 \tag{C.340}$$

end

Second coefficient is:

$$c_{\alpha}^{(R_T^B)} = c_{\alpha}^{(F\ limB)} c_{\alpha}^{(R_T^B)_nofl} \tag{C.341}$$

If `zh_imp_analyt.eq.0` ***then***

$$c_{\alpha}^{(F\ limB)} = \begin{cases} 1, & \text{inside the separatrix} \\ 0.5+0.5\cdot\tanh\left[10\cdot\left(\dfrac{zhflcorr\cdot c_{fl\ lim}\sqrt{\frac{T}{m_{\alpha}}}}{\frac{5}{2}\frac{\zeta_p}{\left(z_{eff}n_e\right)^2 m_p}\left|\sum_{\gamma}\tilde{a}_{\alpha\gamma}\overline{n}_{\gamma} b_x \frac{\widetilde{\partial T_{\gamma}}}{h_x \partial x}\right|}-1\right)\right], & \text{outside the separatrix} \end{cases} \tag{C.342}$$

else

$$c_{\alpha}^{(F\ limB)} = \begin{cases} 1, & \text{inside the separatrix} \\ 0.5+0.5\cdot\tanh\left[10\cdot\left(\dfrac{zhflcorr\cdot c_{fl\ lim}\sqrt{\frac{T}{m_{\alpha}}}}{\frac{\zeta_p}{\left(Z_{\alpha}^2 n_{\alpha}\right)^2 m_p}\overline{n}_{\alpha}\sqrt{\frac{2m_p}{m_{\alpha}}}\left|c_{\alpha}^{(h_T^A)} b_x \frac{\widetilde{\partial T_{\alpha}}}{h_x \partial x}\right|}-1\right)\right], & \text{outside the separatrix} \end{cases} \tag{C.343}$$

end

$$c_{\alpha}^{(R_T^B)_nofl} = \frac{31500}{53471}\frac{\frac{559}{420}\left(1+\frac{6928\sqrt{2}}{8385}Z_{11\alpha}^s\right)\left(1+2\sqrt{2}Z_{2\alpha}^s\right) - \frac{139}{420}\left(1+\frac{184\sqrt{2}}{139}Z_{9\alpha}^s\right)\left(1+4\sqrt{2}Z_{8\alpha}^s\right)}{D_{\alpha}^{part}} \tag{C.344}$$

where:

$$D_{\alpha}^{part} = 1 + \frac{204376\sqrt{2}}{160413} Z_{11\alpha}^{s} + \frac{72670\sqrt{2}}{53471} Z_{5\alpha}^{s} - \frac{76728\sqrt{2}}{53471} Z_{9\alpha}^{s} + \frac{360256}{160413} Z_{5\alpha}^{s} Z_{11\alpha}^{s} - \frac{101568}{53471} \left(Z_{9\alpha}^{s}\right)^{2} \quad \text{(C.345)}$$

The Z-variables can be found in subsection C.11.8.

C.11.6 Friction force

If `Zhdanov_tf_fr.eq.1`, then:

In the friction source part of B2.5:

For isonuclear sequence α and charge state Z:

$$S_{fr,ia}^{m} = [a \rightarrow \alpha Z] = R_{\alpha Z}^{w} = -Z^{2} n_{a} \frac{m_{p}}{\zeta_{p}} \sum_{\beta} \overline{Z_{\beta}^{2} n_{\beta}} \sqrt{\frac{\mu_{\alpha\beta}}{m_{p}}} c_{\beta\alpha}^{(R_w)} \left(V_{\alpha Z\|} - \overline{V}_{\beta\|}\right) \quad \text{(C.346)}$$

where:

$$\overline{V}_{\alpha\|} = \sum_{Z} I_{a(\alpha,Z)} V_{a(\alpha,Z)\|} \quad \text{(C.347)}$$

In the transport part of B2.5:

if `zhcscorr.eq.1` ***then***

$$c_{\beta\alpha}^{(R_w)} = \delta_{\alpha\beta} \sum_{\gamma} \sqrt{\frac{\mu_{\alpha\gamma}}{\mu_{\alpha\beta}}} \frac{\overline{Z_{\gamma}^{2} n_{\gamma}}}{\overline{Z_{\beta}^{2} n_{\beta}}} \left[c_{\gamma\alpha}^{(R_w^B)} - c_{\gamma\alpha}^{(R_w^A)}\right] + c_{\beta\alpha}^{(R_w^A)} \quad \text{(C.348)}$$

else

$$c_{\beta\alpha}^{(R_w)} = \begin{cases} c_{\beta\alpha}^{(R_w^A)}, & \beta \neq \alpha \\ 1, & \beta = \alpha \end{cases} \quad \text{(C.349)}$$

end

First coefficient:

$$c_{\beta\alpha}^{(R_w^A)} = 1 + c_{\beta\alpha}^{(R_w^A 1)} + c_{\beta\alpha}^{(R_w^A 2)} + c_{\beta\alpha}^{(R_w^A 3)} \quad \text{(C.350)}$$

$$c_{\beta\alpha}^{(R_w^A 1)} = -\sum_{\gamma \neq \alpha} \frac{\mu_{\alpha\gamma}}{m_{\alpha}} \sqrt{\frac{\mu_{\alpha\gamma}}{\mu_{\alpha\beta}}} \frac{\overline{Z_{\gamma}^{2} n_{\gamma}}}{\overline{Z_{\beta}^{2} n_{\beta}}} \left[\frac{3}{5} c_{\beta\alpha}^{(h_w^A)} - \frac{3}{14} \frac{\mu_{\alpha\gamma}}{m_{\alpha}} c_{\beta\alpha}^{(r_w^A)}\right] \quad \text{(C.351)}$$

$$c_{\beta\alpha}^{(R_w^A 2)} = \sum_{\gamma \neq \alpha} \frac{\mu_{\alpha\gamma}}{m_{\gamma}} \sqrt{\frac{\mu_{\alpha\gamma}}{\mu_{\alpha\beta}}} \frac{\overline{Z_{\gamma}^{2} n_{\gamma}}}{\overline{Z_{\beta}^{2} n_{\beta}}} \left[\frac{3}{5} c_{\beta\gamma}^{(h_w^A)} - \frac{3}{14} \frac{\mu_{\alpha\gamma}}{m_{\gamma}} c_{\beta\gamma}^{(r_w^A)}\right] \quad \text{(C.352)}$$

$$c_{\beta\alpha}^{(R_w^A 3)} = -\frac{\mu_{\alpha\beta}}{m_\beta} \sum_\gamma \left[\frac{3}{5} c_{\gamma\beta}^{(h_w^A)} - \frac{3}{14} \frac{\mu_{\alpha\beta}}{m_\beta} c_{\gamma\beta}^{(r_w^A)} \right] \tag{C.353}$$

If `zh_imp_analyt.eq.0` ***then***

$$c_{\beta\alpha}^{(h_w^A)} = \sum_\gamma \left(c_{\alpha\gamma\beta}^{(h_w^A *)} - c_{\alpha\beta\gamma}^{(h_w^A *)} \right) \tag{C.354}$$

$$c_{\alpha\gamma\beta}^{(h_w^A *)} = -\frac{5}{2} \frac{\mu_{\gamma\beta}}{m_\gamma} \check{G}_{\gamma\beta}^{(2)} \tilde{\check{a}}_{\alpha\gamma} - \frac{35}{2} \left(\frac{\mu_{\gamma\beta}}{m_\gamma} \right)^2 \check{G}_{\gamma\beta}^{(8)} \tilde{\check{a}}_{\alpha\gamma+n_s} \tag{C.355}$$

$$c_{\beta\alpha}^{(r_w^A)} = \sum_\gamma \left(c_{\alpha\gamma\beta}^{(r_w^A *)} - c_{\alpha\beta\gamma}^{(r_w^A *)} \right) \tag{C.356}$$

$$c_{\alpha\gamma\beta}^{(r_w^A *)} = -\frac{5}{2} \frac{\mu_{\gamma\beta}}{m_\gamma} \check{G}_{\gamma\beta}^{(2)} \tilde{\check{a}}_{\alpha+n_s\gamma}^{mdf} - \frac{35}{2} \left(\frac{\mu_{\gamma\beta}}{m_\gamma} \right)^2 \check{G}_{\gamma\beta}^{(8)} \tilde{\check{a}}_{\alpha+n_s\gamma+n_s}^{mdf} \tag{C.357}$$

else

$$c_{\beta\alpha}^{(h_w^A)} = \frac{1}{\tilde{\Delta}_\alpha} \frac{25\sqrt{2}}{16} \frac{\mu_{\alpha\beta}}{m_\alpha} \sqrt{\frac{\mu_{\alpha\beta}}{m_\alpha}} \left[\frac{3}{2} + \frac{433\sqrt{2}}{240} Z_{11\alpha}^s - \frac{1}{2} \frac{\mu_{\alpha\beta}}{m_\alpha} \left(1 + \frac{23\sqrt{2}}{8} Z_{9\alpha}^s \right) \right] \frac{\overline{Z_\beta^2} n_\beta}{\overline{Z_\alpha^2} n_\alpha} \tag{C.358}$$

$$c_{\beta\alpha}^{(r_w^A)} = -\frac{1}{\tilde{\Delta}_\alpha} \frac{35\sqrt{2}}{24} \frac{\mu_{\alpha\beta}}{m_\alpha} \sqrt{\frac{\mu_{\alpha\beta}}{m_\alpha}} \left[-\frac{3}{2} - \frac{69\sqrt{2}}{16} Z_{9\alpha}^s + \frac{5}{2} \frac{\mu_{\alpha\beta}}{m_\alpha} \left(1 + \frac{13\sqrt{2}}{8} Z_{5\alpha}^s \right) \right] \frac{\overline{Z_\beta^2} n_\beta}{\overline{Z_\alpha^2} n_\alpha} \tag{C.359}$$

If `zh_analyt_mdf.eq.1` ***then***

If $m_\alpha > m_\beta$ ***then***

$$c_{\beta\alpha}^{(h_w^A)} = 0 \tag{C.360}$$

$$c_{\beta\alpha}^{(r_w^A)} = 0 \tag{C.361}$$

end

end

end

A-coefficients and G-objects can be found in subsection C.11.8.

Second coefficient:

$$c_{\beta\alpha}^{(R_w^B)} = 1 - \frac{3}{5}\frac{\mu_{\alpha\beta}}{m_\alpha}c_\alpha^{(h_w^B)} + \frac{3}{14}\left(\frac{\mu_{\alpha\beta}}{m_\alpha}\right)^2 c_\alpha^{(r_w^B)} \tag{C.362}$$

$$c_\alpha^{(h_w^B)} = \frac{31500}{53471}\frac{1 - \frac{139\sqrt{2}}{105}Z_{8\alpha}^s - \frac{46\sqrt{2}}{105}Z_{9\alpha}^s + \frac{1732\sqrt{2}}{1575}Z_{11\alpha}^s + \frac{559\sqrt{2}}{210}Z_{2\alpha}^s - \frac{368}{105}Z_{8\alpha}^s Z_{9\alpha}^s + \frac{6928}{1575}Z_{2\alpha}^s Z_{11\alpha}^s}{D_\alpha^{part}} \tag{C.363}$$

$$c_\alpha^{(r_w^B)} = \frac{17080}{53471}\frac{1 + \frac{276\sqrt{2}}{61}Z_{9\alpha}^s + \frac{417\sqrt{2}}{61}Z_{2\alpha}^s - \frac{590\sqrt{2}}{61}Z_{8\alpha}^s - \frac{130\sqrt{2}}{61}Z_{5\alpha}^s + \frac{1104}{61}Z_{2\alpha}^s Z_{9\alpha}^s - \frac{1040}{61}Z_{5\alpha}^s Z_{8\alpha}^s}{D_\alpha^{part}} \tag{C.364}$$

where:

$$D_\alpha^{part} = 1 + \frac{204376\sqrt{2}}{160413}Z_{11\alpha}^s + \frac{72670\sqrt{2}}{53471}Z_{5\alpha}^s - \frac{76728\sqrt{2}}{53471}Z_{9\alpha}^s + \frac{360256}{160413}Z_{5\alpha}^s Z_{11\alpha}^s - \frac{101568}{53471}\left(Z_{9\alpha}^s\right)^2 \tag{C.365}$$

Z-variables can be found in subsection C.11.8.

C.11.7 Viscous stress

From Zhdanov for ions:

If `zh_vis_type.eq.0` ***then***

$$\left(\nabla \cdot \overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{full} = -B^{\frac{3}{2}} b_x \frac{\partial}{h_x \partial x}\left(\frac{4}{3}\frac{\widetilde{\widetilde{\eta}}_{ax}^{(CL)}}{\rho_a^{rel}}\frac{1}{b_x B^2}\frac{\partial\left(B^{\frac{1}{2}}V_{\parallel a}\right)}{h_x \partial x}\right) \tag{C.366}$$

where:

$$\rho_a^{rel} = \frac{\rho_a}{\rho}, \quad \rho = \sum_a \rho_a, \quad \rho_a = m_a n_a \tag{C.367}$$

$$\left(\nabla \cdot \overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{full} = \left(\nabla \cdot \overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{res} + \left(\nabla \cdot \overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{ai} + \left(\nabla \cdot \overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{div} \tag{C.368}$$

$$\left(\nabla \cdot \overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{res} = -\frac{1}{h_z\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{h_z\sqrt{g}}{h_x}\frac{4}{3}\frac{\widetilde{\widetilde{\eta}}_{ax}^{(CL)}}{\rho_a^{rel}}\frac{\partial \ln\left(h_z B^{\frac{1}{2}}\right)}{B^{\frac{1}{2}}h_x \partial x}\right) B^{\frac{1}{2}} V_{\parallel a} \tag{C.369}$$

$$\left(\nabla \cdot \overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{ai} = 0 \tag{C.370}$$

$$\left(\nabla\cdot\overset{\leftrightarrow}{\pi}{}_a^{V_\parallel}\right)_\parallel^{div}=\frac{1}{h_z\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{h_z\sqrt{g}}{h_x}\frac{4}{3}\frac{\widetilde{\widetilde{\eta}}_{ax}^{(CL)}}{\rho_a^{rel}}\left(\frac{\partial\ln h_z}{h_x\partial x}V_{\parallel a}-\frac{\partial V_{\parallel a}}{h_x\partial x}\right)\right)=$$
$$\frac{1}{h_z\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{h_z\sqrt{g}}{h_x}\left(\frac{4}{3}\frac{\widetilde{\widetilde{\eta}}_{ax}^{(CL)}}{\rho_a^{rel}}\frac{\partial\ln h_z}{h_x\partial x}V_{\parallel a}-\frac{4}{3}\frac{\widetilde{\widetilde{\eta}}_{ax}^{(CL)}}{\rho_a^{rel}}\frac{\partial V_{\parallel a}}{h_x\partial x}\right)\right) \tag{C.371}$$

Thus the momentum flux is:

$$\Gamma_{ax}^m=\begin{cases}m_aV_{\parallel a}\Gamma_{ax}^{Cor}+\frac{4}{3}\frac{\widetilde{\widetilde{\eta}}_{ax}^{(CL)}}{\rho_a^{rel}}\frac{\partial\ln h_z}{h_x\partial x}V_{\parallel a}-\eta_{ax}\frac{\partial V_{\parallel a}}{h_x\partial x}, & z_a\neq 0\\ m_aV_{\parallel a}\Gamma_{ax}^{Cor}-\eta_{ax}\frac{\partial V_{\parallel a}}{h_x\partial x}, & z_a=0\end{cases} \tag{C.372}$$

where:

$$\eta_{ax}=\eta_{ax}^{(AN)}+\frac{4}{3}\frac{\widetilde{\widetilde{\eta}}_{ax}^{(CL)}}{\rho_a^{rel}} \tag{C.373}$$

and the momentum source is:

$$S_{a\parallel}^m=-\left(\nabla\cdot\overset{\leftrightarrow}{\pi}{}_a^{\parallel}\right)_\parallel=\begin{cases}-\left(\nabla\cdot\overset{\leftrightarrow}{\pi}{}_a^{V_\parallel}\right)_\parallel^{res}-\left(\nabla\cdot\overset{\leftrightarrow}{\pi}{}_a^{V_\parallel}\right)_\parallel^{ai}-\left(\nabla\cdot\overset{\leftrightarrow}{\pi}{}_a^{h_i}\right)_\parallel, & z_a\neq 0\\ 0, & z_a=0\end{cases} \tag{C.374}$$

elseif `zh_vis_type.eq.1` ***then***

$$\left(\nabla\cdot\overset{\leftrightarrow}{\pi}{}_a^{V_\parallel}\right)_\parallel^{full}=-B^{\frac{3}{2}}b_x\frac{\partial}{h_x\partial x}\left(\frac{4}{3}\frac{\widetilde{\widetilde{\eta}}_{ax}^{(CL)}}{\rho_a^{rel}}\frac{1}{b_xB^2}\frac{\partial\left(B^{\frac{1}{2}}V_\parallel\right)}{h_x\partial x}\right)=$$
$$-B^{\frac{3}{2}}b_x\frac{\partial}{h_x\partial x}\left(\frac{4}{3}\frac{\widetilde{\widetilde{\eta}}_{ax}^{(CL)}}{\rho_a^{rel}}\frac{1}{b_xB^2}\frac{\partial\left(B^{\frac{1}{2}}\left(\rho_a^{rel}V_{\parallel a}\right)\right)}{h_x\partial x}\right)$$
$$-B^{\frac{3}{2}}b_x\frac{\partial}{h_x\partial x}\left(\frac{4}{3}\frac{\widetilde{\widetilde{\eta}}_{ax}^{(CL)}}{\rho_a^{rel}}\frac{1}{b_xB^2}\frac{\partial\left(B^{\frac{1}{2}}\left(V_\parallel-\rho_a^{rel}V_{\parallel a}\right)\right)}{h_x\partial x}\right) \tag{C.375}$$

where:

$$\rho_a^{rel}=\frac{\rho_a}{\rho},\quad \rho=\sum_a\rho_a,\quad \rho_a=m_an_a \tag{C.376}$$

Note that:

$$V_\parallel-\rho_a^{rel}V_{\parallel a}=\sum_{b\neq a}\rho_b^{rel}V_{\parallel b} \tag{C.377}$$

does not depend on $V_{\parallel a}$.

$$\left(\nabla\cdot\overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{full}=\left(\nabla\cdot\overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{res}+\left(\nabla\cdot\overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{ai}+\left(\nabla\cdot\overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{div} \tag{C.378}$$

$$\left(\nabla\cdot\overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{res}=-\frac{1}{h_z\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{h_z\sqrt{g}}{h_x}\frac{4}{3}\frac{\widetilde{\widetilde{\eta}}_{ax}^{(CL)}}{\rho_a^{rel}}\frac{\partial\ln\left(h_zB^{\frac{1}{2}}\right)}{B^{\frac{1}{2}}h_x\partial x}\right)B^{\frac{1}{2}}\rho_a^{rel}V_{\parallel a} \tag{C.379}$$

$$\left(\nabla\cdot\overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{ai}=-B^{\frac{3}{2}}b_x\frac{\partial}{h_x\partial x}\left(\frac{h_z\sqrt{g}}{h_x}\frac{4}{3}\frac{\widetilde{\widetilde{\eta}}_{ax}^{(CL)}}{\rho_a^{rel}}\frac{1}{b_xB^2}\frac{h_x}{h_z\sqrt{g}}\frac{\partial\left(B^{\frac{1}{2}}\left(V_\parallel-\rho_a^{rel}V_{\parallel a}\right)\right)}{h_x\partial x}\right) \tag{C.380}$$

$$\begin{aligned}\left(\nabla\cdot\overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{div}&=\frac{1}{h_z\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{h_z\sqrt{g}}{h_x}\frac{4}{3}\frac{\widetilde{\widetilde{\eta}}_{ax}^{(CL)}}{\rho_a^{rel}}\left(\frac{\partial\ln h_z}{h_x\partial x}\rho_a^{rel}V_{\parallel a}-\frac{\partial\rho_a^{rel}V_{\parallel a}}{h_x\partial x}\right)\right)=\\&\frac{1}{h_z\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{h_z\sqrt{g}}{h_x}\left(\frac{4}{3}\widetilde{\widetilde{\eta}}_{ax}^{(CL)}\left(\frac{\partial\ln h_z}{h_x\partial x}-\frac{\partial\ln\rho_a^{rel}}{h_x\partial x}\right)V_{\parallel a}-\frac{4}{3}\widetilde{\widetilde{\eta}}_{ax}^{(CL)}\frac{\partial V_{\parallel a}}{h_x\partial x}\right)\right)\end{aligned} \tag{C.381}$$

Thus the momentum flux is:

$$\Gamma_{ax}^m=\begin{cases}m_aV_{\parallel a}\Gamma_{ax}^{Cor}+\frac{4}{3}\widetilde{\widetilde{\eta}}_{ax}^{(CL)}\left(\frac{\partial\ln h_z}{h_x\partial x}-\frac{\partial\ln\rho_a^{rel}}{h_x\partial x}\right)V_{\parallel a}-\eta_{ax}\frac{\partial V_{\parallel a}}{h_x\partial x}, & z_a\neq 0\\ m_aV_{\parallel a}\Gamma_{ax}^{Cor}-\eta_{ax}\frac{\partial V_{\parallel a}}{h_x\partial x}, & z_a=0\end{cases} \tag{C.382}$$

where:

$$\eta_{ax}=\eta_{ax}^{(AN)}+\frac{4}{3}\widetilde{\widetilde{\eta}}_{ax}^{(CL)} \tag{C.383}$$

and the momentum source is:

$$S_{a\parallel}^m=-\left(\nabla\cdot\overset{\leftrightarrow}{\pi}_a^{\parallel}\right)_\parallel\quad=\quad\begin{cases}-\left(\nabla\cdot\overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{res}-\left(\nabla\cdot\overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{ai}-\left(\nabla\cdot\overset{\leftrightarrow}{\pi}_a^{h_i}\right)_\parallel, & z_a\neq 0\\ 0, & z_a=0\end{cases} \tag{C.384}$$

elseif `zh_vis_type.eq.2` ***then***

$$\begin{aligned}\left(\nabla\cdot\overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{full}&=-\frac{4}{3}b_xB^{3/2}\frac{\partial}{h_x\partial x}\left[\widetilde{\widetilde{\eta}}_{ax}^{(CL)}\frac{1}{B^2b_x}\frac{\partial}{h_x\partial x}\left(\sqrt{B}V_{\parallel a}\right)\right]\\&-\frac{4}{3}b_xB^{3/2}\frac{\partial}{h_x\partial x}\left[\frac{1}{B^2b_x}\frac{n_a}{\overline{n}_\alpha}\left[\sum_{\beta\neq\alpha}\frac{\widetilde{\widetilde{\eta}}_{\alpha\beta x}^{(u_iA)}}{p_\alpha}\sum_\zeta n_{(\beta\zeta)}T_i\frac{\partial}{h_x\partial x}\left(\sqrt{B}V_{\parallel(\beta\zeta)}\right)+\right.\right.\\&\left.\left.\left(\frac{\widetilde{\widetilde{\eta}}_{\alpha\alpha x}^{(u_iA)}}{p_\alpha}-zhcscorr\cdot\frac{\widetilde{\widetilde{\eta}}_{\alpha x}^{(u_iB)}}{p_\alpha}\right)\sum_{\zeta\neq Z}n_{(\alpha\zeta)}T_i\frac{\partial}{h_x\partial x}\left(\sqrt{B}V_{\parallel(\alpha\zeta)}\right)\right]\right]\end{aligned} \tag{C.385}$$

$$\left(\nabla\cdot\overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{full}=\left(\nabla\cdot\overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{res}+\left(\nabla\cdot\overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{ai}+\left(\nabla\cdot\overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{div} \tag{C.386}$$

$$\left(\nabla\cdot\overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{res}=-\frac{1}{h_z\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{h_z\sqrt{g}}{h_x}\frac{4}{3}\overset{\sim}{\eta}_{ax}^{(CL)}\frac{\partial\ln\left(h_zB^{\frac{1}{2}}\right)}{B^{\frac{1}{2}}h_x\partial x}\right)B^{\frac{1}{2}}V_{\parallel a} \tag{C.387}$$

$$\left(\nabla\cdot\overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{ai}=-\frac{4}{3}b_xB^{3/2}\frac{\partial}{h_x\partial x}\left[\frac{1}{B^2b_x}\frac{n_a}{\overline{n}_\alpha}\left[\sum_{\beta\neq\alpha}\frac{\overset{\sim}{\eta}_{\alpha\beta x}^{(u_iA)}}{p_\alpha}\sum_\zeta n_{(\beta\zeta)}T_i\frac{\partial}{h_x\partial x}\left(\sqrt{B}V_{\parallel(\beta\zeta)}\right)+\left(\frac{\overset{\sim}{\eta}_{\alpha\alpha x}^{(u_iA)}}{p_\alpha}-zhcscorr\cdot\frac{\overset{\sim}{\eta}_{\alpha x}^{(u_iB)}}{p_\alpha}\right)\sum_{\zeta\neq Z}n_{(\alpha\zeta)}T_i\frac{\partial}{h_x\partial x}\left(\sqrt{B}V_{\parallel(\alpha\zeta)}\right)\right]\right] \tag{C.388}$$

$$\left(\nabla\cdot\overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{div}=\frac{1}{h_z\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{h_z\sqrt{g}}{h_x}\frac{4}{3}\overset{\sim}{\eta}_{ax}^{(CL)}\left(\frac{\partial\ln h_z}{h_x\partial x}V_{\parallel a}-\frac{\partial V_{\parallel a}}{h_x\partial x}\right)\right)=\frac{1}{h_z\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{h_z\sqrt{g}}{h_x}\left(\frac{4}{3}\overset{\sim}{\eta}_{ax}^{(CL)}\frac{\partial\ln h_z}{h_x\partial x}V_{\parallel a}-\frac{4}{3}\overset{\sim}{\eta}_{ax}^{(CL)}\frac{\partial V_{\parallel a}}{h_x\partial x}\right)\right) \tag{C.389}$$

Thus the momentum flux is:

$$\Gamma_{ax}^m=\begin{cases}m_aV_{\parallel a}\Gamma_{ax}^{Cor}+\frac{4}{3}\overset{\sim}{\eta}_{ax}^{(CL)}\frac{\partial\ln h_z}{h_x\partial x}V_{\parallel a}-\eta_{ax}\frac{\partial V_{\parallel a}}{h_x\partial x}, & z_a\neq 0\\ m_aV_{\parallel a}\Gamma_{ax}^{Cor}-\eta_{ax}\frac{\partial V_{\parallel a}}{h_x\partial x}, & z_a=0\end{cases} \tag{C.390}$$

where:

$$\eta_{ax}=\eta_{ax}^{(AN)}+\frac{4}{3}\overset{\sim}{\eta}_{ax}^{(CL)} \tag{C.391}$$

and the momentum source is:

$$S_{a\parallel}^m=-\left(\nabla\cdot\overset{\leftrightarrow}{\pi}_a^{\parallel}\right)_\parallel=\begin{cases}-\left(\nabla\cdot\overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{res}-\left(\nabla\cdot\overset{\leftrightarrow}{\pi}_a^{V_\parallel}\right)_\parallel^{ai}-\left(\nabla\cdot\overset{\leftrightarrow}{\pi}_a^{h_i}\right)_\parallel, & z_a\neq 0\\ 0, & z_a=0\end{cases} \tag{C.392}$$

end

Where the viscous stress coefficient is:

If `zh_vis_type.eq.0` ***or*** `zh_vis_type.eq.1` ***then***

$$\overset{\sim}{\eta}_{ax}^{(CL)}=c_{\eta^{(CL)}}^{Luciani}\overset{\sim}{\eta}_{ax}^{(CL)} \tag{C.393}$$

$$\widetilde{\eta}_{ax}^{(CL)} = c_{\eta_{ax}^{(CL)}} \eta_{ax}^{(CL)} \tag{C.394}$$

The $c_{\eta^{(CL)}}^{Luciani}$ is defined in subsection C.3.3.

$$c_{\eta_{ax}^{(CL)}} = \begin{cases} 1, & z_a \neq 0 \text{ and only for inner flux surfaces} \\ \dfrac{1}{\left(1+\left(\dfrac{\eta_a^{(CL)}\left|b_x \frac{\partial V_{\|}}{h_x \partial x}\right|}{V_{m_a}^{(\max)}}\right)^{\gamma_1}\right)^{\frac{1}{\gamma_1}}}, & \text{for all other cases} \end{cases} \tag{C.395}$$

$$V_{m_a}^{(\max)} = c_{flim_2} n_a T_i \, , c_{flim_2} = \frac{3}{8} \tag{C.396}$$

$$\eta_{ax}^{(CL)} = \begin{cases} 0, z_a = 0 \\ \rho_a^{rel} \eta_a^{(CL)} b_x^2, z_a \neq 0 \end{cases} \tag{C.397}$$

$$\eta_a^{(CL)} = n_a T_i \left[\sum_{\beta} \check{c}_{\alpha\beta}^{(\pi_u^A)} \tau_{\alpha\beta}^{(Zh)} \frac{\overline{n}_{\beta}}{\overline{n}_{\alpha}} + zhcscorr \cdot 2 c_{\alpha}^{(\pi_u^B)} \tau_{\alpha\alpha}^{(Zh)} \left(\frac{\overline{Z_{\alpha}^2}}{Z^2} - 1 \right) \right] \tag{C.398}$$

elseif `zh_vis_type.eq.2` ***then***

$$\widetilde{\widetilde{\eta}}_{ax}^{(CL)} = c_{\eta^{(CL)}}^{Luciani} \widetilde{\eta}_{ax}^{(CL)} \tag{C.399}$$

$$\widetilde{\eta}_{ax}^{(CL)} = c_{\eta_{\alpha x}^{(CL)}} \eta_{ax}^{(CL)} \tag{C.400}$$

The $c_{\eta^{(CL)}}^{Luciani}$ is defined in subsection C.3.3.

$$c_{\eta_{\alpha x}^{(CL)}} = \begin{cases} 1, & z_a \neq 0 \text{ and only for inner flux surfaces} \\ \dfrac{1}{\left(1+\left(\dfrac{\eta_{\alpha}^{(CL)eff}\left|b_x \frac{\partial V_{\|}}{h_x \partial x}\right|}{V_{m_{\alpha}}^{(\max)}}\right)^{\gamma_1}\right)^{\frac{1}{\gamma_1}}}, & \text{for all other cases} \end{cases} \tag{C.401}$$

$$V_{m_{\alpha}}^{(\max)} = c_{flim_2} \overline{n}_{\alpha} T_i \, , c_{flim_2} = \frac{3}{8} \tag{C.402}$$

$$\eta_{\alpha}^{(CL)eff} = \sum_{\beta} \frac{\eta_{\alpha\beta}^{(u_i A)}}{p_{\alpha}} \overline{n}_{\beta} T_i \tag{C.403}$$

$$\eta_{ax}^{(CL)} = \begin{cases} 0, z_a = 0 \\ \eta_a^{(CL)} b_x^2, z_a \neq 0 \end{cases} \tag{C.404}$$

$$\eta_a^{(CL)} = [a \to \alpha Z] = \left[\frac{\eta_{\alpha\alpha x}^{(u_i A)}}{p_\alpha} + zhcscorr \cdot \frac{\eta_{\alpha x}^{(u_i B)}}{p_\alpha} \left(\frac{1}{I_a} - 1 \right) \right] \frac{n_a}{\overline{n}_\alpha} n_a T_i \tag{C.405}$$

$$\frac{\widetilde{\eta}_{\alpha\beta x}^{(u_i A)}}{p_\alpha} = c_{\eta^{(CL)}}^{Luciani} \frac{\widetilde{\eta}_{\alpha\beta x}^{(u_i A)}}{p_\alpha}, \quad \frac{\widetilde{\eta}_{\alpha x}^{(u_i B)}}{p_\alpha} = c_{\eta^{(CL)}}^{Luciani} \frac{\widetilde{\eta}_{\alpha x}^{(u_i B)}}{p_\alpha} \tag{C.406}$$

$$\frac{\widetilde{\eta}_{\alpha\beta x}^{(u_i A)}}{p_\alpha} = c_{\eta_{\alpha x}^{(CL)}} \frac{\eta_{\alpha\beta x}^{(u_i A)}}{p_\alpha}, \quad \frac{\widetilde{\eta}_{\alpha x}^{(u_i B)}}{p_\alpha} = c_{\eta_{\alpha x}^{(CL)}} \frac{\eta_{\alpha x}^{(u_i B)}}{p_\alpha} \tag{C.407}$$

$$\frac{\eta_{\alpha\beta x}^{(u_i A)}}{p_\alpha} = b_x^2 \frac{\eta_{\alpha\beta}^{(u_i A)}}{p_\alpha}; \quad \frac{\eta_{\alpha\beta}^{(u_i A)}}{p_\alpha} = \check{c}_{\alpha\beta}^{(\pi_u^A)} \tau_{\alpha\beta}^{(Zh)}; \quad \frac{\eta_{\alpha x}^{(u_i B)}}{p_\alpha} = b_x^2 \frac{\eta_{\alpha}^{(u_i B)}}{p_\alpha}; \quad \frac{\eta_{\alpha}^{(u_i B)}}{p_\alpha} = 2 c_\alpha^{(\pi_u^B)} \tau_{\alpha\alpha}^{(Zh)}, \tag{C.408}$$

end

where:

$$\tau_{\alpha\beta}^{(Zh)} = \sqrt{\frac{\mu_{\alpha\beta}}{m_p}} \frac{\zeta_p}{\overline{\overline{Z_\alpha^2} Z_\beta^2 n_\beta}} \tag{C.409}$$

Heat viscosity:

$$\left(\nabla \cdot \overset{\leftrightarrow}{\pi}_a^{h_i} \right)_\parallel = -B^{3/2} b_x \frac{\partial}{h_x \partial x} \left[\sum_\beta \left(\frac{n_a}{\overline{n}_\alpha} \frac{h_z \sqrt{g}}{h_x} \frac{8}{15} \frac{\widetilde{\eta}_{\alpha\beta x}^{(h_i A)}}{p_\alpha} \frac{1}{b_x B^2} \frac{h_x}{h_z \sqrt{g}} \frac{\partial}{h_x \partial x} \left(\sqrt{B} \frac{\widetilde{\overline{h}_{\beta x}^{*(0)}}}{b_x} \right) \right) \right] +$$
$$zhcscorr \cdot zhcscorr_vq \cdot$$
$$\left(-B^{3/2} b_x \frac{\partial}{h_x \partial x} \left[\frac{n_a}{\overline{n}_\alpha} \frac{h_z \sqrt{g}}{h_x} \frac{8}{15} \frac{\widetilde{\eta}_{\alpha x}^{(h_i B)}}{p_\alpha} \frac{1}{b_x B^2} \frac{h_x}{h_z \sqrt{g}} \left(\frac{1}{I_a} \frac{\partial}{h_x \partial x} \left(\sqrt{B} \frac{\widetilde{h_{ax}^{(0)}}}{b_x} \right) - \frac{\partial}{h_x \partial x} \left(\sqrt{B} \frac{\widetilde{\overline{h}_{\alpha x}^{*(0)}}}{b_x} \right) \right) \right] \right) \tag{C.410}$$

where the heat fluxes are given by:

$$\frac{\widetilde{\overline{h}_{\beta x}^{*(0)}}}{b_x} = \overline{h}_\beta^{(1)} \arctan \left(\frac{1}{\overline{h}_\beta^{(1)}} \frac{\overline{h}_{\beta x}^{*(0)}}{b_x} \right), \quad \overline{h}_\beta^{(1)} = c_{q1} \overline{n}_\beta T_i \sqrt{\frac{T_i + T_e}{m_\beta}} \tag{C.411}$$

$$\frac{\widetilde{h_{ax}^{(0)}}}{b_x} = h_a^{(1)} \arctan \left(\frac{1}{h_a^{(1)}} \frac{h_{ax}^{(0)}}{b_x} \right), \quad h_a^{(1)} = c_{q1} n_a T_i \sqrt{\frac{T_i + T_e}{m_a}} \tag{C.412}$$

In case 'b2siav_style_qip' is set to '0' in b2mn.dat:

$$\frac{\overline{h}^{*(0)}_{\beta x}}{b_x} = \frac{\overline{\tilde{h}}^{*(dia)}_{\beta x}}{b_x} = \begin{cases} 0 & \text{outside the separatrix} \\ -\frac{5}{2}\frac{B_z B}{B_x \langle B^2 \rangle} \frac{\overline{(Z^{-1})}^*_\beta \overline{n}_\beta T_i}{e} \frac{\partial T_i}{h_y \partial y} & \text{inside the separatrix} \end{cases} \tag{C.413}$$

$$\frac{h^{(0)}_{ax}}{b_x} = \frac{\tilde{h}^{(dia)}_{ax}}{b_x} = \begin{cases} 0 & \text{outside the separatrix} \\ -\frac{5}{2}\frac{B_z B}{B_x \langle B^2 \rangle} \frac{n_a T_i}{Z_a e} \frac{\partial T_i}{h_y \partial y} & \text{inside the separatrix} \end{cases} \tag{C.414}$$

In case 'b2siav_style_qip' is set to '1' in b2mn.dat:

$$\frac{\overline{h}^{*(0)}_{\beta x}}{b_x} = \overline{h}^*_{\beta\|} + \frac{\overline{h}^{*(dia)}_{\beta x}}{b_x} = \begin{cases} 0 & \text{outside the separatrix} \\ -\frac{\overline{\tilde{\kappa}}^{*CL}_{\beta x}}{b_x} \frac{\partial T_i}{h_x \partial x} - \frac{5}{2}\frac{B_z B}{B_x B^2} \frac{\overline{(Z^{-1})}^*_\beta \overline{n}_\beta T_i}{e} \frac{\partial T_i}{h_y \partial y} & \text{inside the separatrix} \end{cases} \tag{C.415}$$

$$\frac{h^{(0)}_{ax}}{b_x} = h_{a\|} + \frac{h^{(dia)}_{ax}}{b_x} = \begin{cases} 0 & \text{outside the separatrix} \\ -\frac{\tilde{\kappa}^{CL}_{ax}}{b_x} \frac{\partial T_i}{h_x \partial x} - \frac{5}{2}\frac{B_z B}{B_x B^2} \frac{n_a T_i}{Z_a e} \frac{\partial T_i}{h_y \partial y} & \text{inside the separatrix} \end{cases} \tag{C.416}$$

$$\overline{(Z^{-1})}^*_\beta = \frac{1}{\overline{n}_\beta} \sum_\zeta \frac{n_{b(\beta,\zeta)}}{\zeta}, \quad \overline{\tilde{\kappa}}^{*CL}_{\beta x} = \sum_\zeta \tilde{\kappa}^{CL}_{\beta\zeta x} \tag{C.417}$$

And the viscosity coefficients are:

$$\frac{\widetilde{\eta}^{(h_i A)}_{\alpha\beta x}}{p_\alpha} = \tilde{\alpha}^{neo} c^{Luciani}_{\eta^{(CL)}} \frac{\eta^{(h_i A)}_{\alpha\beta x}}{p_\alpha}; \quad \frac{\eta^{(h_i A)}_{\alpha\beta x}}{p_\alpha} = b_x^2 \frac{\eta^{(h_i A)}_{\alpha\beta}}{p_\alpha}; \quad \frac{\eta^{(h_i A)}_{\alpha\beta}}{p_\alpha} = \check{c}^{(\pi^A_h)}_{\alpha\beta} \tau^{(Zh)}_{\alpha\beta} \tag{C.418}$$

$$\frac{\widetilde{\eta}^{(h_i B)}_{\alpha x}}{p_\alpha} = \tilde{\alpha}^{neo} c^{Luciani}_{\eta^{(CL)}} \frac{\eta^{(h_i B)}_{\alpha x}}{p_\alpha}; \quad \frac{\eta^{(h_i B)}_{\alpha x}}{p_\alpha} = b_x^2 \frac{\eta^{(h_i B)}_{\alpha}}{p_\alpha}; \quad \frac{\eta^{(h_i B)}_{\alpha}}{p_\alpha} = 2 c^{(\pi^B_h)}_{\alpha} \tau^{(Zh)}_{\alpha\alpha} \tag{C.419}$$

$$\tilde{\alpha}^{neo} = \frac{1}{1.7}\left(\frac{-0.17 + 1.05\nu_*^{1/2} + 2.7\nu_*^2\varepsilon^3}{1 + 0.7\nu_*^{1/2} + \nu_*^2\varepsilon^3} - 1\right) \tag{C.420}$$

The old formulation was:

$$\alpha^{NEO} = \underbrace{\frac{4}{3}}_{tensor} \cdot \underbrace{\frac{2}{5}}_{heat\ flux} \underbrace{\cdot \frac{1}{1.7}\left(\frac{-0.17 + 1.05\nu_*^{1/2} + 2.7\nu_*^2\varepsilon^3}{1 + 0.7\nu_*^{1/2} + \nu_*^2\varepsilon^3} - 1\right)}_{neoclassical\ correction} \cdot \underbrace{1.7}_{kinetic} \tag{C.421}$$

If zh_imp_analyt.eq.0 ***then***

$$\check{c}^{(\pi^A_u)}_{\alpha\beta} = -\frac{\overline{Z^2_\alpha n_\alpha}}{Z_{eff} n_e} \frac{\overline{Z^2_\beta n_\beta}}{Z_{eff} n_e} \frac{\tilde{\tilde{c}}_{\alpha\beta}}{\sqrt{\frac{\mu_{\alpha\beta}}{m_p}}} \tag{C.422}$$

$$\check{c}_{\alpha\beta}^{(\pi_h^A)} = -\frac{\overline{Z_\alpha^2 n_\alpha}}{Z_{eff} n_e} \frac{\overline{Z_\beta^2 n_\beta}}{Z_{eff} n_e} \frac{\left(\tilde{\check{c}}_{\alpha\beta} + \frac{7}{2}\tilde{\check{c}}_{\alpha\beta+n_s}\right)}{\sqrt{\frac{\mu_{\alpha\beta}}{m_p}}} \tag{C.423}$$

else

$$\check{c}_{\alpha\beta}^{(\pi_u^A)} = 2c_\alpha^{(\pi_u^A)}\delta_{\alpha\beta}; \quad c_\alpha^{(\pi_u^A)} = \frac{1025}{1068}\frac{1}{\Delta_\alpha^\pi}\left(1 + \frac{204\sqrt{2}}{205}Z_{22\alpha}^\pi\right) \tag{C.424}$$

$$\check{c}_{\alpha\beta}^{(\pi_h^A)} = 2c_\alpha^{(\pi_h^A)}\delta_{\alpha\beta}; \quad c_\alpha^{(\pi_h^A)} = \frac{1655}{1068}\frac{1}{\Delta_\alpha^\pi}\left(1 + \frac{204\sqrt{2}}{331}Z_{22\alpha}^\pi + \frac{252\sqrt{2}}{331}Z_{12\alpha}^\pi\right) \tag{C.425}$$

$$\Delta_\alpha^\pi = \frac{204}{89}Z_{11\alpha}^\pi Z_{22\alpha}^\pi - \frac{108}{89}{Z_{12\alpha}^\pi}^2 + \frac{205\sqrt{2}}{178}Z_{11\alpha}^\pi + \frac{102\sqrt{2}}{89}Z_{22\alpha}^\pi - \frac{54\sqrt{2}}{89}Z_{12\alpha}^\pi + 1 \tag{C.426}$$

end

$$c_\alpha^{(\pi_u^B)} = \frac{265}{334}\frac{1 + \frac{204\sqrt{2}}{265}Z_{22\alpha}^\pi}{D_\alpha^{\pi part}} \tag{C.427}$$

$$c_\alpha^{(\pi_h^B)} = \frac{475}{334}\frac{1 + \frac{252\sqrt{2}}{475}Z_{12\alpha}^\pi + \frac{204\sqrt{2}}{475}Z_{22\alpha}^\pi}{D_\alpha^{\pi part}} \tag{C.428}$$

$$D_\alpha^{\pi part} = 1 + \frac{159\sqrt{2}}{167}Z_{11\alpha}^\pi + \frac{816\sqrt{2}}{835}Z_{22\alpha}^\pi - \frac{108\sqrt{2}}{167}Z_{12\alpha}^\pi + \frac{1224}{835}Z_{11\alpha}^\pi Z_{22\alpha}^\pi - \frac{648}{835}\left(Z_{12\alpha}^\pi\right)^2 \tag{C.429}$$

C.11.8 A-coefficients, G-objects and Z-variables

Re-normalized part of the inverted matrix is:
$\alpha = 0..n_s - 1$

$\beta = 0..2n_s - 1$

$$\tilde{\check{a}}_{\alpha\beta}^{mdf} = \frac{m_\alpha}{T}\tilde{\check{a}}_{\alpha+n_s\beta} \tag{C.430}$$

and $\tilde{\check{a}}_{\alpha\beta}$ are the elements of the inverted matrix. Elements of the original matrix are:
$\alpha = 0..n_s - 1$

$\beta = 0..n_s - 1$

$$\check{a}_{\alpha\beta} = \check{G}_{\alpha\beta}^{(6)}, \quad \check{a}_{\alpha+n_s\beta} = 7\check{G}_{\alpha\beta}^{(10)} \tag{C.431}$$

$$\check{a}_{\alpha\beta+n_s} = \frac{m_\alpha}{T}\check{G}^{(10)}_{\alpha\beta}, \quad \check{a}_{\alpha+n_s\beta+n_s} = \frac{m_\beta}{T}\check{G}^{(12)}_{\alpha\beta} \tag{C.432}$$

Addition for diagonal elements:
$\beta = 0..n_s - 1$

$$\check{a}_{\beta\beta} = \check{a}_{\beta\beta} + \sum_\gamma \check{G}^{(5)}_{\beta\gamma}, \quad \check{a}_{\beta+n_s\beta} = \check{a}_{\beta+n_s\beta} + 7\sum_\gamma \check{G}^{(9)}_{\beta\gamma} \tag{C.433}$$

$$\check{a}_{\beta\beta+n_s} = \check{a}_{\beta\beta+n_s} + \frac{m_\beta}{T}\sum_\gamma \check{G}^{(9)}_{\beta\gamma}, \quad \check{a}_{\beta+n_s\beta+n_s} = \check{a}_{\beta+n_s\beta+n_s} + \frac{m_\beta}{T}\sum_\gamma \check{G}^{(11)}_{\beta\gamma} \tag{C.434}$$

Elements of the matrix for the viscous stress tensor are:
$\alpha = 0..n_s - 1$

$\beta = 0..n_s - 1$

$$\check{c}_{\alpha\beta} = \check{G}^{(4)}_{\alpha\beta}, \quad \check{c}_{\alpha+n_s\beta} = \frac{7}{2}\frac{m_\alpha}{m_\beta}\check{G}^{(14)}_{\alpha\beta} \tag{C.435}$$

$$\check{c}_{\alpha\beta+n_s} = \frac{m_\alpha}{T}\frac{m_\beta}{m_\alpha}\check{G}^{(14)}_{\alpha\beta}, \quad \check{c}_{\alpha+n_s\beta+n_s} = \frac{m_\alpha}{T}\check{G}^{(16)}_{\alpha\beta} \tag{C.436}$$

Addition for diagonal elements:
$\beta = 0..n_s - 1$

$$\check{c}_{\beta\beta} = \check{c}_{\beta\beta} + \sum_\gamma \check{G}^{(3)}_{\beta\gamma}, \quad \check{c}_{\beta+n_s\beta} = \check{c}_{\beta+n_s\beta} + \frac{7}{2}\sum_\gamma \check{G}^{(13)}_{\beta\gamma} \tag{C.437}$$

$$\check{c}_{\beta\beta+n_s} = \check{c}_{\beta\beta+n_s} + \frac{m_\alpha}{T}\sum_\gamma \check{G}^{(13)}_{\beta\gamma}, \quad \check{c}_{\beta+n_s\beta+n_s} = \check{c}_{\beta+n_s\beta+n_s} + \frac{m_\alpha}{T}\sum_\gamma \check{G}^{(15)}_{\beta\gamma} \tag{C.438}$$

Where the $\check{G}^{(n)}_{\alpha\beta}$ objects are:

$$\check{G}^{(n)}_{\alpha\beta} = \frac{\overline{Z^2_\alpha n_\alpha}}{Z_{eff} n_e}\frac{\overline{Z^2_\beta n_\beta}}{Z_{eff} n_e} N^{(n)}_{\alpha\beta} \tag{C.439}$$

Where $N^{(n)}_{\alpha\beta}$ is a different part of each $\check{G}^{(n)}_{\alpha\beta}$:

$$N_{\alpha\beta}^{(2)} = \frac{3}{5}\sqrt{\frac{\mu_{\alpha\beta}}{m_p}}, \quad N_{\alpha\beta}^{(3)} = -2\left(1+\frac{3}{5}\frac{m_\beta}{m_\alpha}\right)\frac{m_p}{m_\alpha+m_\beta}\sqrt{\frac{\mu_{\alpha\beta}}{m_p}}, \quad N_{\alpha\beta}^{(4)} = \frac{4}{5}\frac{m_p}{m_\alpha+m_\beta}\sqrt{\frac{\mu_{\alpha\beta}}{m_p}} \tag{C.440}$$

$$N_{\alpha\beta}^{(5)} = -\left(\frac{13}{10}\frac{m_\beta}{m_\alpha}+\frac{8}{5}+3\frac{m_\alpha}{m_\beta}\right)\kappa_{\alpha\beta}\sqrt{\frac{\mu_{\alpha\beta}}{m_p}}, \quad N_{\alpha\beta}^{(6)} = \frac{27}{10}\kappa_{\alpha\beta}\sqrt{\frac{\mu_{\alpha\beta}}{m_p}}, \quad N_{\alpha\beta}^{(8)} = -\frac{3}{14}\sqrt{\frac{\mu_{\alpha\beta}}{m_p}} \tag{C.441}$$

$$N_{\alpha\beta}^{(9)} = \frac{3}{5}\left(\frac{23}{28}\frac{m_\beta}{m_\alpha}+\frac{8}{7}+3\frac{m_\alpha}{m_\beta}\right)\kappa_{\alpha\beta}\sqrt{\frac{\mu_{\alpha\beta}}{m_p}}\frac{\mu_{\alpha\beta}}{m_\alpha}, \quad N_{\alpha\beta}^{(10)} = -\frac{45}{28}\kappa_{\alpha\beta}\sqrt{\frac{\mu_{\alpha\beta}}{m_p}}\frac{\mu_{\alpha\beta}}{m_\alpha} \tag{C.442}$$

$$N_{\alpha\beta}^{(11)} = -\left(\frac{433}{280}\frac{m_\beta^2}{m_\alpha^2}+\frac{136}{35}\frac{m_\beta}{m_\alpha}+\frac{459}{35}+\frac{32}{5}\frac{m_\alpha}{m_\beta}+5\frac{m_\alpha^2}{m_\beta^2}\right)\kappa_{\alpha\beta}^2\sqrt{\frac{\mu_{\alpha\beta}}{m_p}}, \quad N_{\alpha\beta}^{(12)} = \frac{75}{8}\kappa_{\alpha\beta}^2\sqrt{\frac{\mu_{\alpha\beta}}{m_p}} \tag{C.443}$$

$$N_{\alpha\beta}^{(13)} = \left(\frac{18}{35}\frac{m_\beta}{m_\alpha}+\frac{6}{5}\right)\frac{m_p}{m_\alpha}\kappa_{\alpha\beta}\sqrt{\frac{\mu_{\alpha\beta}}{m_p}}, \quad N_{\alpha\beta}^{(14)} = -\frac{24}{35}\frac{m_p}{m_\alpha}\kappa_{\alpha\beta}\sqrt{\frac{\mu_{\alpha\beta}}{m_p}} \tag{C.444}$$

$$N_{\alpha\beta}^{(15)} = -\left(\frac{51}{35}\frac{m_\beta^2}{m_\alpha^2}+\frac{37}{7}\frac{m_\beta}{m_\alpha}+\frac{22}{5}+4\frac{m_\alpha}{m_\beta}\right)\frac{m_p}{m_\alpha+m_\beta}\kappa_{\alpha\beta}\sqrt{\frac{\mu_{\alpha\beta}}{m_p}}, \tag{C.445}$$

$$N_{\alpha\beta}^{(16)} = \frac{24}{7}\frac{m_\beta}{m_\alpha}\frac{m_p}{m_\alpha+m_\beta}\kappa_{\alpha\beta}\sqrt{\frac{\mu_{\alpha\beta}}{m_p}} \tag{C.446}$$

Finally Z-variables are defined by:

$$Z_\alpha^* = \sum_{\beta\neq\alpha}\frac{\overline{Z_\beta^2 n_\beta}}{\overline{Z_\alpha^2 n_\alpha}}, \quad Z_{c\alpha}^* = \sum_{\beta\neq\alpha}\sqrt{\frac{\mu_{\alpha\beta}}{m_\alpha}}\frac{\overline{Z_\beta^2 n_\beta}}{\overline{Z_\alpha^2 n_\alpha}} \tag{C.447}$$

$$Z_{2\alpha}^s = \sum_{\beta\neq\alpha}\left(\frac{\mu_{\alpha\beta}}{m_\alpha}\right)^{3/2}\frac{\overline{Z_\beta^2 n_\beta}}{\overline{Z_\alpha^2 n_\alpha}} \tag{C.448}$$

$$Z_{8\alpha}^s = \sum_{\beta\neq\alpha}\left(\frac{\mu_{\alpha\beta}}{m_\alpha}\right)^{5/2}\frac{\overline{Z_\beta^2 n_\beta}}{\overline{Z_\alpha^2 n_\alpha}} \tag{C.449}$$

$$Z_{5\alpha}^s = \sum_{\beta\neq\alpha}\left(1+\frac{16}{13}\frac{m_\alpha}{m_\beta}+\frac{30}{13}\left(\frac{m_\alpha}{m_\beta}\right)^2\right)\left(\frac{\mu_{\alpha\beta}}{m_\alpha}\right)^{5/2}\frac{\overline{Z_\beta^2 n_\beta}}{\overline{Z_\alpha^2 n_\alpha}} \tag{C.450}$$

$$Z_{9\alpha}^s = \sum_{\beta\neq\alpha}\left(1+\frac{32}{23}\frac{m_\alpha}{m_\beta}+\frac{84}{23}\left(\frac{m_\alpha}{m_\beta}\right)^2\right)\left(\frac{\mu_{\alpha\beta}}{m_\alpha}\right)^{7/2}\frac{\overline{Z_\beta^2 n_\beta}}{\overline{Z_\alpha^2 n_\alpha}} \tag{C.451}$$

$$Z_{11\alpha}^s = \sum_{\beta\neq\alpha}\left(1+\frac{1088}{433}\frac{m_\alpha}{m_\beta}+\frac{3672}{433}\left(\frac{m_\alpha}{m_\beta}\right)^2+\frac{1792}{433}\left(\frac{m_\alpha}{m_\beta}\right)^3+\frac{1400}{433}\left(\frac{m_\alpha}{m_\beta}\right)^4\right)\left(\frac{\mu_{\alpha\beta}}{m_\alpha}\right)^{9/2}\frac{\overline{Z_\beta^2 n_\beta}}{\overline{Z_\alpha^2 n_\alpha}} \tag{C.452}$$

$$Z^{\pi}_{11\alpha} = \sum_{\beta\neq\alpha}\left(1+\frac{5}{3}\frac{m_\alpha}{m_\beta}\right)\left(\frac{\mu_{\alpha\beta}}{m_\alpha}\right)^{3/2}\frac{\overline{Z^2_\beta}n_\beta}{\overline{Z^2_\alpha}n_\alpha} \tag{C.453}$$

$$Z^{\pi}_{12\alpha} = \sum_{\beta\neq\alpha}\left(1+\frac{7}{3}\frac{m_\alpha}{m_\beta}\right)\left(\frac{\mu_{\alpha\beta}}{m_\alpha}\right)^{5/2}\frac{\overline{Z^2_\beta}n_\beta}{\overline{Z^2_\alpha}n_\alpha} \tag{C.454}$$

$$Z^{\pi}_{22\alpha} = \sum_{\beta\neq\alpha}\left(1+\frac{185}{51}\frac{m_\alpha}{m_\beta}+\frac{154}{51}\left(\frac{m_\alpha}{m_\beta}\right)^2+\frac{140}{51}\left(\frac{m_\alpha}{m_\beta}\right)^3\right)\left(\frac{\mu_{\alpha\beta}}{m_\alpha}\right)^{7/2}\frac{\overline{Z^2_\beta}n_\beta}{\overline{Z^2_\alpha}n_\alpha} \tag{C.455}$$

C.11.9 Heat source

If `Zhdanov_tf_fr.eq.1`, then:

In the heat source part of B2.5:

$$Q_{F_{ab}} = -\sum_\alpha\sum_Z w_{\alpha Z\|}\left(R_{\alpha Z}+R^e_{\alpha Z}\right) = \\ -\left(\sum_a V_{a\|}\left(S^m_{fr,a}+S^m_{Therm,a}\right) - zhd_corr\cdot V_\|\sum_a\left(S^m_{fr,ea}+S^m_{Therm,ea}\right)\right) \tag{C.456}$$

$$Q_{F_{ei}} = w_{e\|}\sum_\alpha\sum_Z R^e_{\alpha Z} = \\ V_{e\|}\left(\sum_a\left(S^m_{fr,ea}+S^m_{Therm,ea}\right) - zhd_corr\cdot\frac{V_\|}{V_{e\|}}\sum_a\left(S^m_{fr,ea}+S^m_{Therm,ea}\right)\right) \tag{C.457}$$

If `Zhdanov_visu.eq.1`, then:

In the heat source part of B2.5:

If `zh_vis_type.eq.0` ***then***

$$Q_{vis} = \sum_{fluid\ species}\left(\eta_{ax}\left(\frac{\partial V_{a\|}}{h_x\partial x}\right)^2+\eta^{(AN)}_a\left(\frac{\partial V_{a\|}}{h_y\partial y}\right)^2\right) \tag{C.458}$$

elseif `zh_vis_type.eq.1` ***then***

$$Q_{vis} = \sum_{\substack{a=0\\z_a\neq0}}^{ns-1}\left(\eta^{(AN)}_a\left(\frac{\partial V_{a\|}}{h_x\partial x}\right)^2+\eta_{ay}\left(\frac{\partial V_{a\|}}{h_y\partial y}\right)^2+\frac{4}{3}\frac{\widetilde{\eta}^{(CL)}_{ax}}{\rho^{rel}_a}\left(\frac{\partial V_\|}{h_x\partial x}\right)^2\right)+ \\ \sum_{\substack{a=0\\z_a=0\\fluid}}^{ns-1}\left(\eta_{ax}\left(\frac{\partial V_{a\|}}{h_x\partial x}\right)^2+\eta_{ay}\left(\frac{\partial V_{a\|}}{h_y\partial y}\right)^2\right) \tag{C.459}$$

elseif `zh_vis_type.eq.2` ***then***

$$Q_{vis} = \sum_{fluid\ species} \Bigg[\eta_{ax} \left(\frac{\partial V_{a\parallel}}{h_x \partial x} \right)^2 + \eta_a^{(AN)} \left(\frac{\partial V_{a\parallel}}{h_y \partial y} \right)^2 + \frac{n_a}{\overline{n}_\alpha} \Bigg[\sum_{\beta \neq \alpha} \frac{\eta_{\alpha\beta x}^{(u_i A)}}{p_\alpha} \sum_\zeta n_{(\beta\zeta)} T_i \frac{\partial}{h_x \partial x} \left(V_{\parallel(\beta\zeta)} \right) + \left(\frac{\eta_{\alpha\alpha x}^{(u_i A)}}{p_\alpha} - zhcscorr \cdot \frac{\eta_{\alpha x}^{(u_i B)}}{p_\alpha} \right) \sum_{\zeta \neq Z} n_{(\alpha\zeta)} T_i \frac{\partial}{h_x \partial x} \left(V_{\parallel(\alpha\zeta)} \right) \Bigg] \left(\frac{\partial V_{a\parallel}}{h_x \partial x} \right) \Bigg] \quad \text{(C.460)}$$

end

C.11.10 Currents

In the current part of B2.5:

If `Zhdanov_visu.eq.1`, then:

$$\widetilde{j}_x^{(vis\parallel)} = \sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} \widetilde{j}_{ax}^{(vis\parallel)} \quad \text{(C.461)}$$

If `zh_vis_type.eq.0` ***then***

$$\widetilde{j}_{ax}^{(vis\parallel)} = \begin{cases} -\dfrac{B_z \sqrt{B}}{3 B_x} \dfrac{\partial}{h_y \partial y} \left(\dfrac{1}{B^2} \right) \dfrac{\widetilde{\widetilde{\eta}}_{ax}^{(CL)}}{\rho_a^{rel}} \dfrac{\partial \left(\sqrt{B} V_{a\parallel} \right)}{h_x \partial x}, & z_a \neq 0 \\ 0, & z_a = 0 \end{cases} \left(\widetilde{\widetilde{\eta}}_{ax}^{(CL)} \text{already contains a factor of } \rho_a^{rel} b_x^2 \right) \quad \text{(C.462)}$$

elseif `zh_vis_type.eq.1` ***then***

$$\widetilde{j}_{ax}^{(vis\parallel)} = \begin{cases} -\dfrac{B_z \sqrt{B}}{3 B_x} \dfrac{\partial}{h_y \partial y} \left(\dfrac{1}{B^2} \right) \dfrac{\widetilde{\widetilde{\eta}}_{ax}^{(CL)}}{\rho_a^{rel}} \dfrac{\partial \left(\sqrt{B} V_{\parallel} \right)}{h_x \partial x}, & z_a \neq 0 \\ 0, & z_a = 0 \end{cases} \left(\widetilde{\widetilde{\eta}}_{ax}^{(CL)} \text{already contains a factor of } \rho_a^{rel} b_x^2 \right) \quad \text{(C.463)}$$

elseif `zh_vis_type.eq.2` ***then***

$$\widetilde{j}_{ax}^{(vis\|)} = \begin{cases} -\dfrac{B_z\sqrt{B}}{3B_x}\dfrac{\partial}{h_y\partial y}\left(\dfrac{1}{B^2}\right)\left[\widetilde{\widetilde{\eta}}_{ax}^{(CL)}\dfrac{\partial\left(\sqrt{B}V_{a\|}\right)}{h_x\partial x}+\right. \\ \dfrac{n_a}{\overline{n}_\alpha}\left[\sum\limits_{\beta\neq\alpha}\dfrac{\eta_{\alpha\beta x}^{(u_iA)}}{p_\alpha}\sum\limits_{\zeta} n_{(\beta\zeta)}T_i\dfrac{\partial}{h_x\partial x}\left(\sqrt{B}V_{\|(\beta\zeta)}\right)+\right. \\ \left.\left.\left(\dfrac{\eta_{\alpha\alpha x}^{(u_iA)}}{p_\alpha} - zhcscorr\cdot\dfrac{\eta_{\alpha x}^{(u_iB)}}{p_\alpha}\right)\sum\limits_{\zeta\neq Z} n_{(\alpha\zeta)}T_i\dfrac{\partial}{h_x\partial x}\left(\sqrt{B}V_{\|(\alpha\zeta)}\right)\right]\right], z_a\neq 0 \\ 0, z_a = 0 \end{cases}$$

$$\left(\widetilde{\widetilde{\eta}}_{ax}^{(CL)} \text{already contains a factor of } \rho_a^{rel}b_x^2\right) \quad \text{(C.464)}$$

end

$$\widetilde{j}_{y}^{(vis\|)} = \sum_{\substack{a=0\\ z_a\neq 0}}^{ns-1}\widetilde{j}_{ay}^{(vis\|)} \quad \text{(C.465)}$$

If `zh_vis_type.eq.0` ***then***

$$\widetilde{j}_{ay}^{(vis\|)} = \begin{cases} \dfrac{B_z\sqrt{B}}{3B_x}\dfrac{\partial}{h_x\partial x}\left(\dfrac{1}{B^2}\right)\dfrac{\widetilde{\widetilde{\eta}}_{ax}^{(CL)}}{\rho_a^{rel}}\dfrac{\partial\left(\sqrt{B}V_{a\|}\right)}{h_x\partial x}, z_a\neq 0 \\ 0, z_a = 0 \end{cases}$$

$$\left(\widetilde{\widetilde{\eta}}_{ax}^{(CL)} \text{already contains a factor of } \rho_a^{rel}b_x^2\right) \quad \text{(C.466)}$$

elseif `zh_vis_type.eq.1` ***then***

$$\widetilde{j}_{ay}^{(vis\|)} = \begin{cases} \dfrac{B_z\sqrt{B}}{3B_x}\dfrac{\partial}{h_x\partial x}\left(\dfrac{1}{B^2}\right)\dfrac{\widetilde{\widetilde{\eta}}_{ax}^{(CL)}}{\rho_a^{rel}}\dfrac{\partial\left(\sqrt{B}V_{\|}\right)}{h_x\partial x}, z_a\neq 0 \\ 0, z_a = 0 \end{cases}$$

$$\left(\widetilde{\widetilde{\eta}}_{ax}^{(CL)} \text{already contains a factor of } \rho_a^{rel}b_x^2\right) \quad \text{(C.467)}$$

elseif `zh_vis_type.eq.2` ***then***

$$\widetilde{j}_{ay}^{(vis\|)} = \begin{cases} \dfrac{B_z\sqrt{B}}{3B_x}\dfrac{\partial}{h_x\partial x}\left(\dfrac{1}{B^2}\right)\left[\widetilde{\widetilde{\eta}}_{ax}^{(CL)}\dfrac{\partial\left(\sqrt{B}V_{a\|}\right)}{h_x\partial x}+\right. \\ \dfrac{n_a}{\overline{n}_\alpha}\left[\displaystyle\sum_{\beta\neq\alpha}\frac{\eta_{\alpha\beta x}^{(u_iA)}}{p_\alpha}\sum_{\zeta} n_{(\beta\zeta)}T_i\frac{\partial}{h_x\partial x}\left(\sqrt{B}V_{\|(\beta\zeta)}\right)+\right. \\ \left.\left.\left(\dfrac{\eta_{\alpha\alpha x}^{(u_iA)}}{p_\alpha} - zhcscorr\cdot\dfrac{\eta_{\alpha x}^{(u_iB)}}{p_\alpha}\right)\displaystyle\sum_{\zeta\neq Z} n_{(\alpha\zeta)}T_i\frac{\partial}{h_x\partial x}\left(\sqrt{B}V_{\|(\alpha\zeta)}\right)\right]\right], z_a\neq 0 \\ 0,\ z_a = 0 \end{cases}$$

$$\left(\widetilde{\widetilde{\eta}}_{ax}^{(CL)}\text{already contains a factor of } \rho_a^{rel}b_x^2\right) \quad \text{(C.468)}$$

end

If `Zhdanov_vish.eq.1`, then:

$$\widetilde{j}_x^{(visq)} = \sum_{\substack{a=0\\ z_a\neq 0}}^{ns-1}\widetilde{j}_{ax}^{(visq)} \quad \text{(C.469)}$$

$$\widetilde{j}_{ax}^{(visq)} = \begin{cases} -\dfrac{2}{5}\dfrac{B_z\sqrt{B}}{3B_x}\dfrac{\partial}{h_y\partial y}\left(\dfrac{1}{B^2}\right)\left[\displaystyle\sum_\beta\left(\frac{n_a}{\overline{n}_\alpha}\frac{\widetilde{\eta}_{\alpha\beta x}^{(h_iA)}}{p_\alpha}\frac{\partial}{h_x\partial x}\left(\sqrt{B}\widetilde{\frac{\overline{h}_{\beta x}^{*(0)}}{b_x}}\right)\right)+\right. \\ \left. zhcscorr\cdot zhcscorr_vq\cdot\dfrac{n_a}{\overline{n}_\alpha}\dfrac{\widetilde{\eta}_{\alpha x}^{(h_iB)}}{p_\alpha}\left(\dfrac{1}{I_a}\dfrac{\partial}{h_x\partial x}\left(\sqrt{B}\widetilde{\frac{h_{ax}^{(0)}}{b_x}}\right)-\dfrac{\partial}{h_x\partial x}\left(\sqrt{B}\widetilde{\frac{\overline{h}_{\alpha x}^{*(0)}}{b_x}}\right)\right)\right], z_a\neq 0 \\ 0,\ z_a = 0 \end{cases}$$

$$\left(\frac{\widetilde{\eta}_{\alpha\beta x}^{(h_iA)}}{p_\alpha},\ \frac{\widetilde{\eta}_{\alpha x}^{(h_iB)}}{p_\alpha}\text{already contain a factor of } b_x^2\right) \quad \text{(C.470)}$$

$$\widetilde{j}_y^{(visq)} = \sum_{\substack{a=0\\ z_a\neq 0}}^{ns-1}\widetilde{j}_{ay}^{(visq)} \quad \text{(C.471)}$$

$$\widetilde{j}_{ay}^{(visq)} = \begin{cases} \dfrac{2}{5}\dfrac{B_z\sqrt{B}}{3B_x}\dfrac{\partial}{h_x\partial x}\left(\dfrac{1}{B^2}\right)\left[\displaystyle\sum_\beta\left(\frac{n_a}{\overline{n}_\alpha}\frac{\widetilde{\eta}_{\alpha\beta x}^{(h_iA)}}{p_\alpha}\frac{\partial}{h_x\partial x}\left(\sqrt{B}\widetilde{\frac{\overline{h}_{\beta x}^{*(0)}}{b_x}}\right)\right)+\right. \\ \left. zhcscorr\cdot zhcscorr_vq\cdot\dfrac{n_a}{\overline{n}_\alpha}\dfrac{\widetilde{\eta}_{\alpha x}^{(h_iB)}}{p_\alpha}\left(\dfrac{1}{I_a}\dfrac{\partial}{h_x\partial x}\left(\sqrt{B}\widetilde{\frac{h_{ax}^{(0)}}{b_x}}\right)-\dfrac{\partial}{h_x\partial x}\left(\sqrt{B}\widetilde{\frac{\overline{h}_{\alpha x}^{*(0)}}{b_x}}\right)\right)\right], z_a\neq 0 \\ 0,\ z_a = 0 \end{cases}$$

$$\left(\frac{\widetilde{\eta}_{\alpha\beta x}^{(h_iA)}}{p_\alpha},\ \frac{\widetilde{\eta}_{\alpha x}^{(h_iB)}}{p_\alpha}\text{already contain a factor of } b_x^2\right) \quad \text{(C.472)}$$

C.11.11 The thermal and friction forces (legacy)

Reduced improved analytical expressions were included into the standard 3.0.7 **SOLPS-ITER** code structure.

If `Zhdanov_closure.eq.0` ***.or.*** `Zhdanov_test.eq.1` ***.or.*** `Zhdanov_tf_fr.eq.0` ***then***

If `impr_form_tf.eq.1` ***then***

The corresponding term in subsection C.3.1 is replaced by:

$$K_{\beta 0}^{(Therm)} = \frac{1}{\tilde{\Delta}_0}\frac{25}{16}\frac{\mu_{0\beta}}{m_0}\left[\frac{3}{2}\left(1+\frac{433\sqrt{2}}{360}Z_{110}^{s}\right)-\frac{1}{2}\frac{\mu_{0\beta}}{m_0}\left(1+\frac{23\sqrt{2}}{8}Z_{90}^{s}\right)\right]\left(1+\sqrt{2}Z_{c\alpha}^{*}\right) \tag{C.473}$$

Note, a transition from the Greek to Latin letters should be done $[0 \rightarrow s;\ \beta \rightarrow r]$ (details in subsection C.11.2), where $s = s_{main},\ s \neq r$

else

Standard model is used.

end

If `impr_form_fr.eq.1` ***then***

The corresponding term in subsection C.3.1 is replaced by:

$$K_{\beta 0}^{(V_\parallel)} = K_{0\beta}^{(V_\parallel)} = 1-\frac{15\sqrt{2}}{16}Z_{20}^{s}\frac{\mu_{0\beta}}{m_0}\frac{1}{\tilde{\Delta}_0}\left(\frac{3}{2}-\frac{1}{2}\frac{\mu_{0\beta}}{m_0}+\frac{433\sqrt{2}}{240}Z_{110}^{s}-\frac{23\sqrt{2}}{16}\frac{\mu_{0\beta}}{m_0}Z_{90}^{s}\right)$$
$$-\frac{5\sqrt{2}}{16}Z_{80}^{s}\frac{\mu_{0\beta}}{m_0}\frac{1}{\tilde{\Delta}_0}\left(\frac{5}{2}\frac{\mu_{0\beta}}{m_0}-\frac{3}{2}+\frac{65\sqrt{2}}{16}\frac{\mu_{0\beta}}{m_0}Z_{50}^{s}-\frac{69\sqrt{2}}{16}Z_{90}^{s}\right) \tag{C.474}$$

Note, a transition from the Greek to Latin letters should be done $[0 \rightarrow s;\ \beta \rightarrow r]$ (details in subsection C.11.2), where $s = s_{main},\ s \neq r$

else

Standard model is used.

end

end

C.11.12 The distribution function

$$f_a(\mathbf{c}) = n_a\left(\frac{m_a}{2\pi T_i}\right)^{3/2}\exp\left(-\frac{m_a c^2}{2T_i}\right)\cdot\left[1+\gamma_a w_{\parallel a}c_\parallel+\right.$$
$$\left.\frac{1}{5}\gamma_a^2\left(\frac{h_{\parallel a}^{T}}{p_a}+\frac{h_{\parallel a}^{w}}{p_a}\right)c_\parallel\left(c^2-\frac{5}{\gamma_a}\right)+\frac{1}{70}\gamma_a^3\left(\frac{r_{\parallel a}^{T}\gamma_a}{p_a}+\frac{r_{\parallel a}^{w}\gamma_a}{p_a}\right)c_\parallel\left(c^4-\frac{14}{\gamma_a}c^2+\frac{35}{\gamma_a^2}\right)\right] \tag{C.475}$$

$$f_e(\mathbf{c}) = n_e \left(\frac{m_e}{2\pi T_e}\right)^{3/2} \exp\left(-\frac{m_e c^2}{2T_e}\right) \cdot \left[1 + \gamma_e w_{\|e} c_\| + \right.$$
$$\left. \frac{1}{5}\gamma_e^2 \left(\frac{h_{\|e}^T}{p_e} + \frac{h_{\|e}^w}{p_e}\right) c_\| \left(c^2 - \frac{5}{\gamma_e}\right) + \frac{1}{70}\gamma_e^3 \left(\frac{r_{\|e}^T \gamma_e}{p_e} + \frac{r_{\|e}^w \gamma_e}{p_e}\right) c_\| \left(c^4 - \frac{14}{\gamma_e}c^2 + \frac{35}{\gamma_e^2}\right)\right] \quad \text{(C.476)}$$

where:

$$\gamma_e = \frac{m_e}{T_e}, \quad \gamma_a = \frac{m_a}{T_i}, \quad \mathbf{c} = \mathbf{v} - \mathbf{u}, \quad \mathbf{u} = \frac{\sum_a m_a n_a \mathbf{u}_a}{\sum_a m_a n_a} \quad \text{(C.477)}$$

NB: if $\mathbf{u}_e^\perp, \mathbf{u}_a^\perp \neq \mathbf{u}^\perp$ (charge-dependent drifts are significant), this distribution function should be corrected.

$$\int_0^\infty \int_0^{2\pi} f_a\left(c_\|, c_\perp\right) c_\perp d\varphi dc_\perp = n_a \left(\frac{\gamma_a}{2\pi}\right)^{1/2} \exp\left(-\frac{\gamma_a}{2}c_\|^2\right) \cdot \left[1 + \gamma_a w_{\|a} c_\| + \right.$$
$$\left. \frac{1}{5}\gamma_a^2 \left(\frac{h_{\|a}^T}{p_a} + \frac{h_{\|a}^w}{p_a}\right) c_\| \left(c_\|^2 - \frac{3}{\gamma_a}\right) + \frac{1}{70}\gamma_a^3 \left(\frac{r_{\|a}^T \gamma_a}{p_a} + \frac{r_{\|a}^w \gamma_a}{p_a}\right) c_\| \left(c_\|^4 - \frac{10}{\gamma_a}c_\|^2 + \frac{15}{\gamma_a^2}\right)\right] \quad \text{(C.478)}$$

Appendix D

Boundary conditions in B2.5

D.1 Introduction

The boundary conditions are implemented in the `B2.5` code by means of the boundary sources, that is in a similar way as the volume sources due to the atomic processes, but they are defined only in the boundary cells of the computational grid [27]. It means that in a case of the boundary conditions at the material surfaces the boundary cells have a role of combined electrostatic sheath and quasineutral collisionless magnetic presheath, which cannot be modelled with the `B2.5` fluid equations. According to the existing fluid models of the magnetized boundary plasma, the role of the magnetic presheath, which is located between the collisional plasma presheath and electrostatic sheath, is to deflect the charged particles from their average motion along the magnetic field lines to the direction perpendicular to the wall. The average ion velocity along the magnetic field direction at the entrance of the magnetic presheath from the collisional presheath side is at least the local ion sound velocity, that is $c_{sMPSE} = \sqrt{\frac{k_B(ZT_{eMPSE}+T_{iMPSE})}{m_i}}$ in a two-component plasma, or $u_{\|MPSE} \geq c_{sMPSE}$ [58]. The component of the average ion velocity at the entrance of the electrostatic sheath, which is perpendicular to the wall, is also greater than or equal to the local ion sound velocity, i.e. $u_{SE} \geq c_{sSE}$ [59, 58]. Since both the magnetic presheath and electrostatic sheath are collisionless regions and assuming that the dissipative effects due to some turbulent mechanisms are negligible in these regions, particle, momentum and energy fluxes are conserved in these regions. The `B2.5` equations for the bulk plasma are approximately valid in the collisional presheath, assuming that the charged particle velocity distributions do not differ significantly from Maxwellian or drifting-Maxwellian distributions. (However, it should be emphasized that according to the two-scale kinetic analysis of the boundary plasma near the solid surface, pure Maxwellian or drifting-Maxwellian ion velocity distributions cannot satisfy the generalized Bohm criterion [59, 60, 61]). It means that they provide the values for the charged particle densities, drift velocities, ion and electron temperatures at the magnetic presheath edge ($MPSE$), which can be used for calculating the fluxes at the magnetic presheath edge. The value of the poloidal component of the ion drift velocity at the magnetic presheath edge obtained from the bulk transport equations should be corrected, because of the effects related to the stronger local electric field in the magnetic presheath. This electric field causes a strong acceleration of the ions along the magnetic field lines, similarly to a non-magnetic plasma presheath or when the magnetic field is perpendicular to the surface, and an additional poloidal $\mathbf{E} \times \mathbf{B}$ drift due to the radial component of the electric field. In some cases the $\mathbf{E} \times \mathbf{B}$ drift velocity can be comparable with the local ion sound velocity [33, 19, 62]. The correction of the poloidal component of the ion drift velocity due to the $\mathbf{B} \times \nabla p$ drift is usually negligible, since it contributes mostly to the ion flow parallel to the surface [35]. However, more detailed investigations of the importance of this drift should be performed in the future studies, which should include also kinetic models and numerical simulations (e.g. particle and/or kinetic simulations) of the magnetized boundary plasma.

The boundary sources in the `B2.5` code are defined by the boundary cell averages or volume integrals of the negative divergences of the incoming fluxes into the boundary cells. The boundary condition for the continuity equation for

each atomic species is given by:

$$S_p = -\int_{B.V.} \nabla \cdot \mathbf{\Gamma}\, dV = -\int_{B.S.} \mathbf{\Gamma} \cdot d\mathbf{A}, \tag{D.1}$$

which means that in the discretized form of this equation the value of the particle flux, $\Gamma_j = n\langle v_j \rangle$, at the magnetic presheath edge as a function of the macroscopic plasma parameters should be specified. The boundary condition for the momentum equation is:

$$\mathbf{S}_m = -\int_{B.V.} \nabla \cdot \underline{\mathbf{M}}\, dV = -\int_{B.S.} \underline{\mathbf{M}} \cdot d\mathbf{A}. \tag{D.2}$$

In this case the appropriate components of the momentum tensor, $M_{jk} = mn\langle v_j v_k \rangle$, at the magnetic presheath edge should be specified in the discretized equations. Similarly, the energy or heat boundary sources are given by:

$$S_q = -\int_{B.V.} \nabla \cdot \mathbf{Q}\, dV = -\int_{B.S.} \mathbf{Q} \cdot d\mathbf{A}, \tag{D.3}$$

where the appropriate components of the electron and/or ion heat flux, $Q_j = \frac{1}{2} mn \langle v^2 v_j \rangle$, at the magnetic presheath edge as functions of the macroscopic plasma parameters should be specified in the discretized equations. The boundary charge source is given by:

$$S_{ch} = -\int_{B.V.} \nabla \cdot \mathbf{j}\, dV = -\int_{B.S.} \mathbf{j} \cdot d\mathbf{A}, \tag{D.4}$$

where the electric current density, $\mathbf{j} = e\left(\sum_\alpha Z_\alpha \mathbf{\Gamma}_\alpha - \mathbf{\Gamma}_e\right)$, at the magnetic presheath edge should be specified in the discretized equations.

D.2 Analytical Expressions for the Standard ("Old") Boundary Sources in the B2.5 Code

The boundary sources are expressed as linear and/or weakly nonlinear functions of the macroscopic plasma parameters ($n_\alpha, n_e, u_{\alpha x}, T_i, T_e, \Phi$), calculated in the cells neighbouring the boundary cells. Specific dependencies of the boundary sources on these parameters are determined by the choice of the coefficients $C_{pj\alpha}$, $C_{mj\alpha}$, $C_{hij\alpha}$, C_{hej} and C_{chj}. These coefficients are free input parameters, which depend on the theoretical model used for describing the boundary conditions. They must be written by the user in the input file `b2ah.dat` (section 3.2). If the `b2ah.dat` file already exists and the boundary conditions for the new simulation are not significantly different, it is practical to write any minor changes to the boundary conditions in the input file `b2mn.dat` (section 3.5). (If the existing `b2ah.dat` file is corrected, it is not necessary to do that.) When the theoretical model predicts more complicated nonlinear dependence of the boundary fluxes on the plasma parameters, the effects of these nonlinearities can be only approximately taken into consideration in the calculation, e.g. by some appropriate choice of the numerical values of the input parameters. More details about the numerical implementation of the boundary sources can be found in the `b2stbc.F` source code.

D.2.1 Definitions of Physical Parameters Used in the Analytical Expressions for the Standard Boundary Sources

All following quantities are calculated locally at the appropriate boundary – "SOUTH" (S), "NORTH" (N), "WEST" (W), and "EAST" (E) –, i.e. in the boundary cells:

$$S_0 = V/h_{y1}, \quad h_{y1} = h_y \cos(\widehat{xy}) \tag{D.5}$$

for the "SOUTH" and "NORTH" boundaries, where V is the volume of the boundary cell, h_y is the y-diameter (radial length) of the boundary cell and $\cos(\widehat{xy})$ is the cosine of the complementary angle between the x-direction and y-direction on the boundary cell;

$$S_0 = V/h_x \tag{D.6}$$

for the "WEST" and "EAST" boundaries, where V is the volume of the boundary cell and h_x is the x-diameter (poloidal length) of the boundary cell;

$$S_1 = \left|S_1^{\pm}\right| = \left|\frac{B_y}{B}A\right| \tag{D.7}$$

for the "SOUTH" and "NORTH" boundaries, where $S_1^{\pm}$ is the product of the magnetic field pitch and the surface area on the cell face between the boundary cell and its neighbouring (previous) cell ($(ix,-1)$ and $(ix,0)$ for the "SOUTH" and (ix,ny) and $(ix,ny-1)$ for the "NORTH");

$$S_1 = \left|S_1^{\pm}\right| = \left|\frac{B_x}{B}A\right| \tag{D.8}$$

for the "WEST" and "EAST" boundaries, where $S_1^{\pm}$ is the product of the magnetic field pitch and the surface area on the cell face between the boundary cell and its neighbouring (previous) cell ($(-1,iy)$ and $(0,iy)$ for the "WEST" and (nx,iy) and $(nx-1,iy)$ for the "EAST");

$$c_{s\alpha} = \sqrt{\frac{k_B\left(Z_\alpha T_e + T_i\right)}{A_\alpha m_p}} = \sqrt{\frac{k_B\left(Z_\alpha T_e + T_i\right)}{m_\alpha}} \tag{D.9}$$

is the isothermal ion sound velocity for the atomic species α;

$$c_{st} = \sqrt{\frac{\sum_{Z_\alpha \neq 0}^{\alpha} n_\alpha k_B\left(Z_\alpha T_e + T_i\right)}{\sum_{Z_\alpha \neq 0}^{\alpha} n_\alpha A_\alpha m_p}} = \sqrt{\frac{\sum_{Z_\alpha \neq 0}^{\alpha} n_\alpha k_B\left(Z_\alpha T_e + T_i\right)}{\sum_{Z_\alpha \neq 0}^{\alpha} n_\alpha m_\alpha}} \tag{D.10}$$

is the isothermal ion sound velocity for a multi-species plasma.

$$c_{sD\alpha} = \max\left(c_{s\alpha}, 0.1c_{s\alpha} \pm \left(B_z/B_x\right)u_{Dx}\right) \tag{D.11}$$

is the conditionally corrected ion drift velocity of the atomic species α due to the poloidal $\mathbf{E}\times\mathbf{B}$ drift, where '$+$' applies to the "WEST" and '$-$' to the "EAST" boundaries; the approximate correction is used only if $0.1c_{s\alpha} \pm \left(B_z/B_x\right)u_{Dx} > c_{s\alpha}$;

$$c_{stD} = \max\left(c_{st}, 0.1c_{st} \pm \left(B_z/B_x\right)u_{Dx}\right) \tag{D.12}$$

is the conditionally corrected ion drift velocity of the multi-species plasma due to the poloidal $\mathbf{E}\times\mathbf{B}$ drift, where '$+$' applies to the "WEST" and '$-$' to the "EAST" boundaries; the approximate correction is used only if $0.1c_{st} \pm \left(B_z/B_x\right)u_{Dx} > c_{st}$;

$$\rho_\alpha = A_\alpha m_p n_\alpha = m_\alpha n_\alpha \tag{D.13}$$

is the mass density of atomic species α;

$$n_i = \sum_\alpha n_\alpha \tag{D.14}$$

is the total atomic species (ion) particle density;

$$t_\alpha = \frac{n_\alpha}{n_i} \tag{D.15}$$

is the relative concentration (particle density) of the atomic species α;

$$\mathbf{j}_i = \pm e\sum_\alpha Z_\alpha \mathbf{\Gamma}_\alpha \tag{D.16}$$

is the positive ion electric current density in the boundary cells, where '$+$' applies to the "NORTH" and "EAST" boundaries and '$-$' to the "SOUTH" and "WEST" boundaries;

$$v_{th,e} = \sqrt{\frac{k_B T_e}{m_e}} \tag{D.17}$$

is the electron thermal velocity;

$$J_e = \frac{1}{\sqrt{2\pi}} e n_e v_{th,e} S_1 \exp\left(-\frac{e\Phi}{k_B T_e}\right) \tag{D.18}$$

is the current carried by the electrons; the electron and ion temperatures are expressed in energy units and

$$\mathbf{Q}_{h\alpha D} = \kappa_{\times\alpha} \left(\mathbf{b} \times \nabla T_\alpha\right) S_0 \tag{D.19}$$

is the diamagnetic component of the heat current for species α, where $\mathbf{b} = \frac{\mathbf{B}}{B}$.

D.2.2 Standard Boundary Sources

Analytical formulae for the standard boundary sources are presented in this subsection. Their numerical implementation is included in the `b2stbc.F` source code. The `B2.5` code architecture is flexible enough so that it is possible to change the expressions for the boundary sources without changing the other parts of the source code, i.e. only the numerical values of the boundary sources are used in further calculations.

"SOUTH" Boundary (Cells (ix,-1))

Particle Sources

$$S_{p\alpha} = S_{p0\alpha} + S_{p1\alpha} n_\alpha \tag{D.20}$$

$$S_{p0\alpha} = C_{p0\alpha} S_0 \tag{D.21}$$

$$S_{p1\alpha} = C_{p1\alpha} S_0 + C_{p2\alpha} c_{s\alpha} S_0 + C_{p3\alpha} c_{s\alpha} S_1 + C_{p4\alpha} c_{st} S_0 + C_{p5\alpha} c_{st} S_1 \tag{D.22}$$

Parallel momentum Sources

$$S_{m\alpha} = S_{m0\alpha} + S_{m1\alpha} u_{\alpha x} + S_{m2\alpha} \rho_\alpha + S_{m3\alpha} \rho_\alpha u_{\alpha x} \tag{D.23}$$

$$S_{m0\alpha} = C_{m0\alpha} S_0 \tag{D.24}$$

$$S_{m1\alpha} = 0 \tag{D.25}$$

$$S_{m2\alpha} = C_{m1\alpha} \left(-c_{s\alpha}\right) S_1^{\pm} + C_{m2\alpha} \left(-c_{st}\right) S_1^{\pm} + C_{m5\alpha} \frac{\max\left(0, \Gamma_{\alpha y}\right)}{n_\alpha} \tag{D.26}$$

$$S_{m3\alpha} = C_{m3\alpha} S_0 + C_{m4\alpha} S_1 + C_{m6\alpha} \frac{\max\left(0, -\Gamma_{\alpha y}\right)}{n_\alpha} \tag{D.27}$$

All Atom/Ion Heat Source

$$S_{hi} = S_{hi0} + S_{hi1}T_i + S_{hi2}n_i + S_{hi3}n_iT_i \tag{D.28}$$

$$S_{hi0} = \sum_{\alpha} \left[C_{hi0\alpha}t_{\alpha}S_0 + C_{hi6\alpha}\max\left(0, \Gamma_{\alpha y}\right)\right] + Q_{hiDy} \tag{D.29}$$

$$S_{hi1} = \sum_{\alpha} C_{hi1\alpha}t_{\alpha}S_0 + \frac{Q_{hiDy}}{T_i} \tag{D.30}$$

$$S_{hi2} = 0 \tag{D.31}$$

$$S_{hi3} = \sum_{\alpha} \Big[\, C_{hi2\alpha}t_{\alpha}c_{s\alpha}S_0 + C_{hi3\alpha}t_{\alpha}c_{s\alpha}S_1 + C_{hi4\alpha}t_{\alpha}c_{st}S_0 + C_{hi5\alpha}t_{\alpha}c_{st}S_1 + C_{hi7\alpha}\frac{\max\left(0, -\Gamma_{\alpha y}\right)}{n_i}\Big] \tag{D.32}$$

Electron Heat Source

$$S_{he} = S_{he0} + S_{he1}T_e + S_{he2}n_e + S_{he3}n_eT_e \tag{D.33}$$

$$S_{he0} = C_{he0}S_0 + C_{he6}\max\left(0, \Gamma_{ey}\right) + Q_{heDy} \tag{D.34}$$

$$S_{he1} = C_{he1}S_0 + \frac{Q_{heDy}}{T_e} \tag{D.35}$$

$$S_{he2} = 0 \tag{D.36}$$

$$S_{he3} = C_{he2}v_{th,e}S_0 + C_{he3}v_{th,e}S_1 + C_{he4}c_{st}S_0 + C_{he5}c_{st}S_1 + C_{he7}\frac{\max\left(0, -\Gamma_{ey}\right)}{n_e} \tag{D.37}$$

Electric Charge Source

$$S_{ch} = S_{ch0} + S_{ch1}\Phi + S_{ch2}n_e + S_{ch3}n_e\Phi \tag{D.38}$$

$$S_{ch0} = C_{ch7}J_{ey} - C_{ch6}J_{iy} + \max\left(C_{ch6}J_{iy}, C_{ch7}J_{ey}\right)\frac{e\Phi}{T_e} \tag{D.39}$$

$$S_{ch1} = -\frac{\max\left(C_{ch6}J_{iy}, C_{ch7}J_{ey}\right)e}{T_e} \tag{D.40}$$

$$S_{ch2} = C_{ch0}ev_{th,e}S_0 + C_{ch2}ev_{th,e}S_1 + C_{ch4}ec_{st}S_1 \tag{D.41}$$

$$S_{ch3} = C_{ch1}\frac{e^2v_{th,e}S_0}{T_e} + C_{ch3}\frac{e^2v_{th,e}S_1}{T_e} + C_{ch5}\frac{e^2c_{st}S_1}{T_e} \tag{D.42}$$

"NORTH" Boundary (Cells (ix,ny))

Particle Sources

$$S_{p\alpha} = S_{p0\alpha} + S_{p1\alpha} n_\alpha \tag{D.43}$$

$$S_{p0\alpha} = C_{p0\alpha} S_0 \tag{D.44}$$

$$S_{p1\alpha} = C_{p1\alpha} S_0 + C_{p2\alpha} c_{s\alpha} S_0 + C_{p3\alpha} c_{s\alpha} S_1 + C_{p4\alpha} c_{st} S_0 + C_{p5\alpha} c_{st} S_1 \tag{D.45}$$

Parallel momentum Sources

$$S_{m\alpha} = S_{m0\alpha} + S_{m1\alpha} u_{\alpha x} + S_{m2\alpha} \rho_\alpha + S_{m3\alpha} \rho_\alpha u_{\alpha x} \tag{D.46}$$

$$S_{m0\alpha} = C_{m0\alpha} S_0 \tag{D.47}$$

$$S_{m1\alpha} = 0 \tag{D.48}$$

$$S_{m2\alpha} = C_{m1\alpha} c_{s\alpha} S_1^{\pm} + C_{m2\alpha} c_{st} S_1^{\pm} + C_{m5\alpha} \frac{\max(0, -\Gamma_{\alpha y})}{n_\alpha} \tag{D.49}$$

$$S_{m3\alpha} = C_{m3\alpha} S_0 + C_{m4\alpha} S_1 + C_{m6\alpha} \frac{\max(0, \Gamma_{\alpha y})}{n_\alpha} \tag{D.50}$$

All Atom/Ion Heat Source

$$S_{hi} = S_{hi0} + S_{hi1} T_i + S_{hi2} n_i + S_{hi3} n_i T_i \tag{D.51}$$

$$S_{hi0} = \sum_\alpha \left[C_{hi0\alpha} t_\alpha S_0 + C_{hi6\alpha} \max(0, -\Gamma_{\alpha y}) \right] - Q_{hiDy} \tag{D.52}$$

$$S_{hi1} = \sum_\alpha C_{hi1\alpha} t_\alpha S_0 - \frac{Q_{hiDy}}{T_i} \tag{D.53}$$

$$S_{hi2} = 0 \tag{D.54}$$

$$S_{hi3} = \sum_\alpha \Big[C_{hi2\alpha} t_\alpha c_{s\alpha} S_0 + C_{hi3\alpha} t_\alpha c_{s\alpha} S_1 + C_{hi4\alpha} t_\alpha c_{st} S_0 + C_{hi5\alpha} t_\alpha c_{st} S_1 + C_{hi7\alpha} \frac{\max(0, \Gamma_{\alpha y})}{n_i} \Big] \tag{D.55}$$

Electron Heat Source

$$S_{he} = S_{he0} + S_{he1} T_e + S_{he2} n_e + S_{he3} n_e T_e \tag{D.56}$$

$$S_{he0} = C_{he0} S_0 + C_{he6} \max\left(0, -\Gamma_{ey}\right) - Q_{heDy} \tag{D.57}$$

$$S_{he1} = C_{he1} S_0 - \frac{Q_{heDy}}{T_e} \tag{D.58}$$

$$S_{he2} = 0 \tag{D.59}$$

$$\begin{aligned} S_{he3} = \; & C_{he2} v_{th,e} S_0 + C_{he3} v_{th,e} S_1 + C_{he4} c_{st} S_0 + \\ & C_{he5} c_{st} S_1 + C_{he7} \frac{\max\left(0, \Gamma_{ey}\right)}{n_e} \end{aligned} \tag{D.60}$$

Electric Charge Source

$$S_{ch} = S_{ch0} + S_{ch1} \Phi + S_{ch2} n_e + S_{ch3} n_e \Phi \tag{D.61}$$

$$S_{ch0} = C_{ch7} J_{ey} - C_{ch6} J_{iy} + \max\left(C_{ch6} J_{iy}, C_{ch7} J_{ey}\right) \frac{e\Phi}{T_e} \tag{D.62}$$

$$S_{ch1} = -\frac{\max\left(C_{ch6} J_{iy}, C_{ch7} J_{ey}\right) e}{T_e} \tag{D.63}$$

$$S_{ch2} = C_{ch0} e v_{th,e} S_0 + C_{ch2} e v_{th,e} S_1 + C_{ch4} e c_{st} S_1 \tag{D.64}$$

$$S_{ch3} = C_{ch1} \frac{e^2 v_{th,e} S_0}{T_e} + C_{ch3} \frac{e^2 v_{th,e} S_1}{T_e} + C_{ch5} \frac{e^2 c_{st} S_1}{T_e} \tag{D.65}$$

"WEST" Boundary (Cells (-1,iy), Divertor Plate)

Particle Sources

$$S_{p\alpha} = S_{p0\alpha} + S_{p1\alpha} n_\alpha \tag{D.66}$$

$$S_{p0\alpha} = C_{p0\alpha} S_0 + C_{p6\alpha} \max\left(0, u_{\alpha x} S_1^{\pm}\right) \tag{D.67}$$

$$\begin{aligned} S_{p1\alpha} = \; & C_{p1\alpha} S_0 + C_{p2\alpha} c_{sD\alpha} S_0 + C_{p3\alpha} c_{sD\alpha} S_1 + C_{p4\alpha} c_{stD} S_0 + \\ & C_{p5\alpha} c_{stD} S_1 + C_{p7\alpha} \max\left(0, -u_{\alpha x} S_1^{\pm}\right) \end{aligned} \tag{D.68}$$

Parallel momentum Sources

$$S_{m\alpha} = S_{m0\alpha} + S_{m1\alpha} u_{\alpha x} + S_{m2\alpha} \rho_\alpha + S_{m3\alpha} \rho_\alpha u_{\alpha x} \tag{D.69}$$

$$S_{m0\alpha} = C_{m0\alpha} S_0 \tag{D.70}$$

$$S_{m1\alpha} = 0 \tag{D.71}$$

$$S_{m2\alpha} = C_{m1\alpha} \left(-c_{sD\alpha}\right) S_1^{\pm} + C_{m2\alpha} \left(-c_{stD}\right) S_1^{\pm} + C_{m5\alpha} \frac{\max\left(0, \Gamma_{\alpha x}\right)}{n_\alpha} \tag{D.72}$$

$$S_{m3\alpha} = C_{m3\alpha} S_0 + C_{m4\alpha} S_1 + C_{m6\alpha} \frac{\max\left(0, -\Gamma_{\alpha x}\right)}{n_\alpha} \tag{D.73}$$

All Atom/Ion Heat Source

$$S_{hi} = S_{hi0} + S_{hi1} T_i + S_{hi2} n_i + S_{hi3} n_i T_i \tag{D.74}$$

$$S_{hi0} = \sum_\alpha \left[C_{hi0\alpha} t_\alpha S_0 + C_{hi6\alpha} \max\left(0, \Gamma_{\alpha x}\right) \right] + Q_{hiDx} \tag{D.75}$$

$$S_{hi1} = \sum_\alpha C_{hi1\alpha} t_\alpha S_0 + \frac{Q_{hiDx}}{T_i} \tag{D.76}$$

$$S_{hi2} = 0 \tag{D.77}$$

$$\begin{aligned} S_{hi3} = \sum_\alpha \big[& C_{hi2\alpha} t_\alpha c_{sD\alpha} S_0 + C_{hi3\alpha} t_\alpha c_{sD\alpha} S_1 + C_{hi4\alpha} t_\alpha c_{stD} S_0 + \\ & C_{hi5\alpha} t_\alpha c_{stD} S_1 + C_{hi7\alpha} \frac{\max\left(0, -\Gamma_{\alpha x}\right)}{n_i} \Big] \end{aligned} \tag{D.78}$$

Electron Heat Source

$$S_{he} = S_{he0} + S_{he1} T_e + S_{he2} n_e + S_{he3} n_e T_e \tag{D.79}$$

$$S_{he0} = C_{he0} S_0 + C_{he6} \max\left(0, \Gamma_{ex}\right) + Q_{heDx} \tag{D.80}$$

$$S_{he1} = C_{he1} S_0 + \frac{Q_{heDx}}{T_e} \tag{D.81}$$

$$S_{he2} = 0 \tag{D.82}$$

$$\begin{aligned} S_{he3} = \; & C_{he2} v_{th,e} S_0 + C_{he3} v_{th,e} S_1 + C_{he4} c_{stD} S_0 + \\ & C_{he5} c_{stD} S_1 + C_{he7} \frac{\max\left(0, -\Gamma_{ex}\right)}{n_e} \end{aligned} \tag{D.83}$$

Electric Charge Source

$$S_{ch} = S_{ch0} + S_{ch1}\Phi + S_{ch2}n_e + S_{ch3}n_e\Phi \tag{D.84}$$

$$S_{ch0} = C_{ch7}J_{ex} - C_{ch6}J_{ix} + \max\left(C_{ch6}J_{ix}, C_{ch7}J_{ex}\right)\frac{e\Phi}{T_e} \tag{D.85}$$

$$S_{ch1} = -\frac{\max\left(C_{ch6}J_{ix}, C_{ch7}J_{ex}\right)e}{T_e} \tag{D.86}$$

$$S_{ch2} = C_{ch0}ev_{th,e}S_0 + C_{ch2}ev_{th,e}S_1 + C_{ch4}ec_{stD}S_1 \tag{D.87}$$

$$S_{ch3} = C_{ch1}\frac{e^2v_{th,e}S_0}{T_e} + C_{ch3}\frac{e^2v_{th,e}S_1}{T_e} + C_{ch5}\frac{e^2c_{stD}S_1}{T_e} \tag{D.88}$$

"EAST" Boundary (Cells (nx,iy), Divertor Plate)

Particle Sources

$$S_{p\alpha} = S_{p0\alpha} + S_{p1\alpha}n_\alpha \tag{D.89}$$

$$S_{p0\alpha} = C_{p0\alpha}S_0 + C_{p6\alpha}\max\left(0, -u_{\alpha x}S_1^{\pm}\right) \tag{D.90}$$

$$\begin{aligned} S_{p1\alpha} = \; & C_{p1\alpha}S_0 + C_{p2\alpha}c_{sD\alpha}S_0 + C_{p3\alpha}c_{sD\alpha}S_1 + C_{p4\alpha}c_{stD}S_0 + \\ & C_{p5\alpha}c_{stD}S_1 + C_{p7\alpha}\max\left(0, u_{\alpha x}S_1^{\pm}\right) \end{aligned} \tag{D.91}$$

Parallel momentum Sources

$$S_{m\alpha} = S_{m0\alpha} + S_{m1\alpha}u_{\alpha x} + S_{m2\alpha}\rho_\alpha + S_{m3\alpha}\rho_\alpha u_{\alpha x} \tag{D.92}$$

$$S_{m0\alpha} = C_{m0\alpha}S_0 \tag{D.93}$$

$$S_{m1\alpha} = 0 \tag{D.94}$$

$$S_{m2\alpha} = C_{m1\alpha}c_{sD\alpha}S_1^{\pm} + C_{m2\alpha}c_{stD}S_1^{\pm} + C_{m5\alpha}\frac{\max\left(0, -\Gamma_{\alpha x}\right)}{n_\alpha} \tag{D.95}$$

$$S_{m3\alpha} = C_{m3\alpha}S_0 + C_{m4\alpha}S_1 + C_{m6\alpha}\frac{\max\left(0, \Gamma_{\alpha x}\right)}{n_\alpha} \tag{D.96}$$

All Atom/Ion Heat Source

$$S_{hi} = S_{hi0} + S_{hi1} T_i + S_{hi2} n_i + S_{hi3} n_i T_i \tag{D.97}$$

$$S_{hi0} = \sum_{\alpha} \left[C_{hi0\alpha} t_\alpha S_0 + C_{hi6\alpha} \max\left(0, -\Gamma_{\alpha x}\right) \right] - Q_{hiDx} \tag{D.98}$$

$$S_{hi1} = \sum_{\alpha} C_{hi1\alpha} t_\alpha S_0 - \frac{Q_{hiDx}}{T_i} \tag{D.99}$$

$$S_{hi2} = 0 \tag{D.100}$$

$$\begin{aligned} S_{hi3} = \sum_{\alpha} [\, & C_{hi2\alpha} t_\alpha c_{sD\alpha} S_0 + C_{hi3\alpha} t_\alpha c_{sD\alpha} S_1 + C_{hi4\alpha} t_\alpha c_{stD} S_0 + \\ & C_{hi5\alpha} t_\alpha c_{stD} S_1 + C_{hi7\alpha} \frac{\max\left(0, \Gamma_{\alpha x}\right)}{n_i} \Big] \end{aligned} \tag{D.101}$$

Electron Heat Source

$$S_{he} = S_{he0} + S_{he1} T_e + S_{he2} n_e + S_{he3} n_e T_e \tag{D.102}$$

$$S_{he0} = C_{he0} S_0 + C_{he6} \max\left(0, -\Gamma_{ex}\right) - Q_{heDx} \tag{D.103}$$

$$S_{he1} = C_{he1} S_0 - \frac{Q_{heDx}}{T_e} \tag{D.104}$$

$$S_{he2} = 0 \tag{D.105}$$

$$\begin{aligned} S_{he3} = \; & C_{he2} v_{th,e} S_0 + C_{he3} v_{th,e} S_1 + C_{he4} c_{stD} S_0 + \\ & C_{he5} c_{stD} S_1 + C_{he7} \frac{\max\left(0, \Gamma_{ex}\right)}{n_e} \end{aligned} \tag{D.106}$$

Electric Charge Source

$$S_{ch} = S_{ch0} + S_{ch1} \Phi + S_{ch2} n_e + S_{ch3} n_e \Phi \tag{D.107}$$

$$S_{ch0} = C_{ch7} J_{ex} - C_{ch6} J_{ix} + \max\left(C_{ch6} J_{ix}, C_{ch7} J_{ex}\right) \frac{e\Phi}{T_e} \tag{D.108}$$

$$S_{ch1} = -\frac{\max\left(C_{ch6} J_{ix}, C_{ch7} J_{ex}\right) e}{T_e} \tag{D.109}$$

$$S_{ch2} = C_{ch0} e v_{th,e} S_0 + C_{ch2} e v_{th,e} S_1 + C_{ch4} e c_{stD} S_1 \tag{D.110}$$

$$S_{ch3} = C_{ch1} \frac{e^2 v_{th,e} S_0}{T_e} + C_{ch3} \frac{e^2 v_{th,e} S_1}{T_e} + C_{ch5} \frac{e^2 c_{stD} S_1}{T_e} \tag{D.111}$$

Appendix E

Quantities stored in b2time.nc

The following scalar time-dependent quantities are stored in `b2time.nc` and can be plotted by

```
2dt tesepm
```

to plot the midplane separatrix electron temperature as a function of time, or

```
2d nesepm tesepm
```

to plot the midplane separatrix electron temperature versus the midplane separatrix electron density. The frequency at which (and whether) these quantities are written out is controlled by the `'b2mndr_b2time'` switch.

If `'b2mndr_stim'` is set to a negative value, data from the current **SOLPS-ITER** run will be appended to the existing `b2time.nc` file, otherwise the file is created anew.

In the case of double-null runs, the divertor and separatrix quantities are given in pairs. For the midplane quantities, the first value is for the outer midplane, and the second value for the inner midplane. Divertor quantities show first the bottom then the top divertor.

For double-null (DN) and lower single-null (LSN) topologies, the Western edge refers to the inboard divertor, while it points to the outboard divertor for upper single-null (USN) topologies. Conversely, the Eastern edge is the outboard divertor for DN and LSN cases, but the inboard divertor for USN cases.

In the case of linear runs, the midplane refers, by default, to the midpoint of the domain. The divertor throat quantities are not defined.

In all cases, the point of reference of the midplane quantities can be adjusted by changing the value of jxa (and jxi for double-null runs) by assigning it an explicit value in the `b2mn.dat` file.

The total internal energy flux quantities `fet...` are defined in the same way as the `fetx` **b2plot** quantity (see Section J.3).

quantity	description	units
ntstep	timestep index	
timesa	time stamp	s
fnixip	integrated poloidal main ion particle flux, Western edge	s^{-1}
feexip	integrated poloidal electron energy flux, Western edge	W
feixip	integrated poloidal ion energy flux, Western edge	W
fetxip	integrated poloidal total internal energy flux, Western edge	W
fchxip	integrated poloidal current, Western edge	A
fnixap	integrated poloidal main ion particle flux, Eastern edge	s^{-1}
feexap	integrated poloidal electron energy flux, Eastern edge	W

continued on next page

continued from previous page

quantity	description	units
feixap	integrated poloidal ion energy flux, Eastern edge	W
fetxap	integrated poloidal total internal energy flux, Eastern edge	W
fchxap	integrated poloidal current, Eastern edge	A
nesepi	separatrix electron density, Western edge	m^{-3}
tesepi	separatrix electron temperature, Western edge	eV
tisepi	separatrix ion temperature, Western edge	eV
tpsepi	separatrix target temperature, Western edge	K
posepi	separatrix potential, Western edge	V
nesepm	separatrix electron density, outer midplane	m^{-3}
tesepm	separatrix electron temperature, outer midplane	eV
tisepm	separatrix ion temperature, outer midplane	eV
posepm	separatrix potential, outer midplane	V
dnsepm	main ion density diffusion coefficient, outer midplane	$m^2.s^{-1}$
dpsepm	main neutral pressure diffusion coefficient, outer midplane	$m^2.s^{-1}$
kesepm	electron thermal diffusivity, outer midplane	$m^2.s^{-1}$
kisepm	ion thermal diffusivity, outer midplane	$m^2.s^{-1}$
vxsepm	main ion poloidal pinch velocity, outer midplane	$m.s^{-1}$
vysepm	main ion radial pinch velocity, outer midplane	$m.s^{-1}$
vssepm	main ion particle viscosity, outer midplane	$m^2.s^{-1}$
nesepa	separatrix electron density, Eastern edge	m^{-3}
tesepa	separatrix electron temperature, Eastern edge	eV
tisepa	separatrix ion temperature, Eastern edge	eV
tpsepa	separatrix target temperature, Eastern edge	K
posepa	separatrix potential, Eastern edge	V
nemxip	maximum electron density, Western edge	m^{-3}
temxip	maximum electron temperature, Western edge	eV
timxip	maximum ion temperature, Western edge	eV
tpmxip	maximum target temperature, Western edge	K
pomxip	maximum potential, Western edge	V
nemxap	maximum electron density, Eastern edge	m^{-3}
temxap	maximum electron temperature, Eastern edge	eV
timxap	maximum ion temperature, Eastern edge	eV
tpmxap	maximum target temperature, Eastern edge	K
pomxap	maximum potential, Eastern edge	V
fniyip	integrated radial main ion particle flux, main chamber	s^{-1}
feeyip	integrated radial electron energy flux, main chamber	W
feiyip	integrated radial ion energy flux, main chamber	W
fetyip	integrated radial total internal energy flux, main chamber	W
fchyip	integrated radial current, main chamber	A
fniyap	integrated radial main ion particle flux, divertor region	s^{-1}
feeyap	integrated radial electron energy flux, divertor region	W
feiyap	integrated radial ion energy flux, divertor region	W
fetyap	integrated radial total internal energy flux, divertor region	W
fchyap	integrated radial current, divertor region	A
pwmxip	maximum total power flux, Western edge	$W.m^{-2}$
pwmxap	maximum total power flux, Eastern edge	$W.m^{-2}$
tmne	total number of particles	
tmte	total electron energy	eV
tmti	total ion energy	eV
tmhacore	H_α emissivity, core	$photons.m^{-2}.sr^{-1}$?

continued on next page

continued from previous page		
quantity	description	units
tmhasol	H_α emissivity, SOL above the X-point	$photons.m^{-2}.sr^{-1}$?
tmhadiv	H_α emissivity, divertor region	$photons.m^{-2}.sr^{-1}$?
fnisip	poloidal main ion particle flux, into Western separatrix throat	s^{-1}
feesip	poloidal electron energy flux, into Western separatrix throat	W
feisip	poloidal ion energy flux, into Western separatrix throat	W
fetsip	poloidal total internal energy flux, into Western separatrix throat	W
fchsip	poloidal current, into Western separatrix throat	A
fnisap	poloidal main ion particle flux, into Eastern separatrix throat	s^{-1}
feesap	poloidal electron energy flux, into Eastern separatrix throat	W
feisap	poloidal ion energy flux, into Eastern separatrix throat	W
fetsap	poloidal total internal energy flux, into Eastern separatrix throat	W
fchsap	poloidal current, into Eastern separatrix throat	A
fnisipp	poloidal main ion particle flux, core x-pt flux region	s^{-1}
feesipp	poloidal electron energy flux, core x-pt flux region	W
feisipp	poloidal ion energy flux, core x-pt flux region	W
fetsipp	poloidal total internal energy flux, core x-pt flux region	W
fchsipp	poloidal current, core x-pt flux region	A
fnisapp	poloidal main ion particle flux, x-pt private flux region	s^{-1}
feesapp	poloidal electron energy flux, x-pt private flux region	W
feisapp	poloidal ion energy flux, x-pt private flux region	W
fetsapp	poloidal total internal energy flux, x-pt private flux region	W
fchsapp	poloidal current, x-pt private flux region	A

Table E.1: Scalar time-dependent quantities stored in b2time.nc

The following are vector time-dependent quantities stored in `b2time.nc`. These are quantities along lines of constant *ix* at the Western target (`l`) (inner for LSN, outer for USN, and lower inner for DN configurations), inner midplane (`i`), outer midplane (`a`) and Eastern target (`r`) (outer for LSN, inner for USN, and lower outer for DN configurations). The positions of the inner and outer midplanes can be selected by `b2mwti_jxi` and `b2mwti_jxa`, respectively, in `b2mn.dat`.

Again, the total internal energy flux quantities `ft3d..` are defined in the same way as the `fetx` **b2plot** quantity (see Section J.3).

Note that, when processed with `2d_profiles`, the fluxes are renormalised by the contact area to the neighbouring cells, and therefore their units must all be multiplied by m^{-2}. When using `2d_profiles_extended`, additional files are created using the name template `f?3d?P.last10` where the area used is the surface area perpendicular to the magnetic field direction.

quantity	description	units
fn3dl	poloidal particle flux of main ion, Western edge	$s^{-1}.m^{-2}$
fl3dl	poloidal electron flux, Western edge	$s^{-1}.m^{-2}$
fo3dl	poloidal total ion flux, Western edge	$s^{-1}.m^{-2}$
fe3dl	poloidal electron energy flux, Western edge	$W.m^{-2}$
fi3dl	poloidal ion energy flux, Western edge	$W.m^{-2}$
ft3dl	poloidal total internal energy flux, Western edge	$W.m^{-2}$
fc3dl	poloidal current, Western edge	$A.m^{-2}$
ne3dl	electron density, Western edge	m^{-3}
te3dl	electron temperature, Western edge	eV
continued on next page		

continued from previous page		
quantity	description	units
ti3dl	ion temperature, Western edge	eV
tp3dl	target temperature, Western edge	K
po3dl	potential, Western edge	V
an3dl	atom density of main neutral, Western edge	m^{-3}
mn3dl	molecule density of 1^{st} molecule, Western edge	m^{-3}
ne3di	electron density, inboard midplane	m^{-3}
te3di	electron temperature, inboard midplane	eV
ti3di	ion temperature, inboard midplane	eV
po3di	potential, inboard midplane	V
an3di	atom density of main neutral, inboard midplane	m^{-3}
mn3di	molecule density of 1^{st} molecule, inboard midplane	m^{-3}
dn3di	particle density diffusivity, inboard midplane	$m^2.s^{-1}$
dp3di	particle pressure diffusivity, inboard midplane	$m^2.s^{-1}$
ke3di	electron heat diffusivity, inboard midplane	$m^2.s^{-1}$
ki3di	ion heat diffusivity, inboard midplane	$m^2.s^{-1}$
vs3di	main ion viscosity, inboard midplane	$m^2.s^{-1}$
vx3di	main ion poloidal pinch velocity, inboard midplane	$m.s^{-1}$
vy3di	main ion radial pinch velocity, inboard midplane	$m.s^{-1}$
ne3da	electron density, outboard midplane	m^{-3}
te3da	electron temperature, outboard midplane	eV
ti3da	ion temperature, outboard midplane	eV
po3da	potential, outboard midplane	V
an3da	atom density of main neutral, outboard midplane	m^{-3}
mn3da	molecule density of 1^{st} molecule, outboard midplane	m^{-3}
dn3da	particle density diffusivity, outboard midplane	$m^2.s^{-1}$
dp3da	particle pressure diffusivity, outboard midplane	$m^2.s^{-1}$
ke3da	electron heat diffusivity, outboard midplane	$m^2.s^{-1}$
ki3da	ion heat diffusivity, outboard midplane	$m^2.s^{-1}$
vs3da	main ion viscosity, outboard midplane	$m^2.s^{-1}$
vx3da	main ion poloidal pinch velocity, outboard midplane	$m.s^{-1}$
vy3da	main ion radial pinch velocity, outboard midplane	$m.s^{-1}$
ne3dr	electron density, Eastern edge	m^{-3}
te3dr	electron temperature, Eastern edge	eV
ti3dr	ion temperature, Eastern edge	eV
tp3dr	target temperature, Eastern edge	K
po3dr	potential, Eastern edge	V
an3dr	atom density of main neutral, Eastern edge	m^{-3}
mn3dr	molecule density of 1^{st} molecule, Eastern edge	m^{-3}
fn3dr	poloidal particle flux of main ion, Eastern edge	$s^{-1}.m^{-2}$
fe3dr	poloidal electron energy flux, Eastern edge	$W.m^{-2}$
fi3dr	poloidal ion energy flux, Eastern edge	$W.m^{-2}$
ft3dr	poloidal total internal energy flux, Eastern edge	$W.m^{-2}$
fc3dr	poloidal current, Eastern edge	$A.m^{-2}$
fl3dr	poloidal electron flux, Eastern edge	$s^{-1}.m^{-2}$
fo3dr	poloidal total ion flux, Eastern edge	$s^{-1}.m^{-2}$

Table E.2: Vector time-dependent quantities stored in b2time.nc. Note that the conversion to "per unit area" for fluxes is done in the script "2d_profiles"; the actual data stored in b2time.nc for fluxes are the fluxes per cell. There is a similar conversion from J to eV for temperatures.

In the case of double-null geometries, additional quantities are created related to the inboard top (`tl`) and outboard top (`tr`) divertors. These are listed below in Table E.3.

quantity	description	units
ne3dtl	electron density, inboard top divertor	m^{-3}
te3dtl	electron temperature, inboard top divertor	eV
ti3dtl	ion temperature, inboard top divertor	eV
tp3dtl	target temperature, inboard top divertor	K
po3dtl	potential, inboard top divertor	V
an3dtl	atom density of main neutral, inboard top divertor	m^{-3}
mn3dtl	molecule density of 1^{st} molecule, inboard top divertor	m^{-3}
fn3dtl	poloidal particle flux of main ion, inboard top divertor	s^{-1}
fe3dtl	poloidal electron energy flux, inboard top divertor	W
fi3dtl	poloidal ion energy flux, inboard top divertor	W
ft3dtl	poloidal total internal energy flux, inboard top divertor	W
fc3dtl	poloidal current, inboard top divertor	A
fl3dtl	poloidal electron flux, inboard top divertor	s^{-1}
fo3dtl	poloidal total ion flux, inboard top divertor	s^{-1}
ne3dtr	electron density, outboard top divertor	m^{-3}
te3dtr	electron temperature, outboard top divertor	eV
ti3dtr	ion temperature, outboard top divertor	eV
tp3dtl	target temperature, outboard top divertor	K
po3dtr	potential, outboard top divertor	V
an3dtr	atom density of main neutral, outboard top divertor	m^{-3}
mn3dtr	molecule density of 1^{st} molecule, outboard top divertor	m^{-3}
fn3dtr	poloidal particle flux of main ion, outboard top divertor	s^{-1}
fe3dtr	poloidal electron energy flux, outboard top divertor	W
fi3dtr	poloidal ion energy flux, outboard top divertor	W
ft3dtr	poloidal total internal energy flux, outboard top divertor	W
fc3dtr	poloidal current, outboard top divertor	A
fl3dtr	poloidal electron flux, outboard top divertor	s^{-1}
fo3dtr	poloidal total ion flux, outboard top divertor	s^{-1}

Table E.3: Double-null vector time-dependent quantities stored in b2time.nc

Additional multi-dimensional writes to `b2time.nc` can be made using the `b2mwti_2dwrite` switch. If set to `1`, then a few 2D arrays (n_e, T_e, T_i) are output with each write to `b2time.nc`. If set to `2`, then additional quantities such as fluxes are added, see below Table E.4.

quantity	description	units
If `b2mwti_2dwrite` ≥ 1		
ne2d	electron density	m^{-3}
te2d	electron temperature	eV
ti2d	ion temperature	eV
and if `b2mwti_2dwrite` = 2		
po2d	electric potential	V
kin2d	parallel kinetic energy	J
continued on next page		

continued from previous page		
quantity	description	units
rsahi2d	ionisation energy source/sink	W
rsana2d	ionisation rate	s^{-1}
rrahi2d	recombination energy source/sink	W
rrana2d	recombination rate	s^{-1}
rcxhi2d	charge exchange energy source/sink	W
rcxna2d	charge exchange rate	s^{-1}
rqrad2d	Line radiation rate	W
rqahe2d	Electron cooling rate	W
fhe2d	Electron heat flux	W
fhi2d	Ion heat flux	W
fch2d	Electric crrent	A
fna2d	Particle flux	s^{-1}

Table E.4: Multi-dimensional time-dependent quantities stored in b2time.nc

When using the averaging scheme for `Eirene` coupled runs (see Section 5.6), obtained by activating the `'b2mndt_av'` switch, the `b2time.nc` file will also contain some averaged data, listed below in table E.5. These quantities can be visualized by means of the `2dt_av` script.

quantity	description	units
nastep	timestep index	
batchsa	time stamp for averaged batch	s
nesepi_av	averaged separatrix electron density, Western edge	m^{-3}
tesepi_av	averaged separatrix electron temperature, Western edge	eV
tisepi_av	averaged separatrix ion temperature, Western edge	eV
posepi_av	averaged separatrix potential, Western edge	V
nesepm_av	averaged separatrix electron density, outer midplane	m^{-3}
tesepm_av	averaged separatrix electron temperature, outer midplane	eV
tisepm_av	averaged separatrix ion temperature, outer midplane	eV
posepm_av	averaged separatrix potential, outer midplane	V
nesepa_av	averaged separatrix electron density, Eastern edge	m^{-3}
tesepa_av	averaged separatrix electron temperature, Eastern edge	eV
tisepa_av	averaged separatrix ion temperature, Eastern edge	eV
posepa_av	averaged separatrix potential, Eastern edge	V
nemxip_av	averaged maximum electron density, Western edge	m^{-3}
temxip_av	averaged maximum electron temperature, Western edge	eV
timxip_av	averaged maximum ion temperature, Western edge	eV
pomxip_av	averaged maximum potential, Western edge	V
nemxap_av	averaged maximum electron density, Eastern edge	m^{-3}
temxap_av	averaged maximum electron temperature, Eastern edge	eV
timxap_av	averaged maximum ion temperature, Eastern edge	eV
pomxap_av	averaged maximum potential, Eastern edge	
nesepi_std	averaged separatrix electron density, Western edge	m^{-3}
tesepi_std	averaged separatrix electron temperature, Western edge	eV
tisepi_std	averaged separatrix ion temperature, Western edge	eV
posepi_std	averaged separatrix potential, Western edge	V
nesepm_std	variance for separatrix electron density, outer midplane	m^{-3}
continued on next page		

continued from previous page		
quantity	description	units
tesepm_std	variance for separatrix electron temperature, outer midplane	eV
tisepm_std	variance for separatrix ion temperature, outer midplane	eV
posepm_std	variance for separatrix potential, outer midplane	V
nesepa_std	variance for separatrix electron density, Eastern edge	m^{-3}
tesepa_std	variance for separatrix electron temperature, Eastern edge	eV
tisepa_std	variance for separatrix ion temperature, Eastern edge	eV
posepa_std	variance for separatrix potential, Eastern edge	V
nemxip_std	variance for maximum electron density, Western edge	m^{-3}
temxip_std	variance for maximum electron temperature, Western edge	eV
timxip_std	variance for maximum ion temperature, Western edge	eV
pomxip_std	variance for maximum potential, Western edge	V
nemxap_std	variance for maximum electron density, Eastern edge	m^{-3}
temxap_std	variance for maximum electron temperature, Eastern edge	eV
timxap_std	variance for maximum ion temperature, Eastern edge	eV
pomxap_std	variance for maximum potential, Eastern edge	V

Table E.5: Scalar batch-averaged time-dependent quantities stored in b2time.nc

Appendix F

Quantities stored in b2movies.nc and eirenemovies.nc

Several relevant multi-dimensional arrays, i.e. covering the entire `B2.5` grid, are stored in `b2movies.nc`. Similarly, several relevant multi-dimensional arrays covering the entire `Eirene` triangular grid, in case of runs with `Eirene`, are stored in `eirenemovies.nc`. The real-time interval between writes of the movie frames of these quantities is controlled by the `'b2mndr_cdfmovietim'` switch.

If `'b2mndr_stim'` is set to a negative value, data from the current **SOLPS-ITER** run will be appended to the existing `b2movies.nc` and `eirenemovies.nc` files, otherwise the files are created anew.

The number of arrays outputted to `b2movies.nc` is controlled by a series of switches (which can be simultaneously used) as described in the below Table F.1.

quantity	description	units
If `cdfmovie_b25_state_variables` ≥ 1		
na	Density of fluid species (for each species)	m^{-3}
ne	Electron density	m^{-3}
te	Electron temperature	eV
ti	Ion temperature	eV
po	Electric potential	V
If `cdfmovie_b25_fluxes` ≥ 1		
fnax	Poloidal particle flux (for each fluid species)	s^{-1}
fnay	Radial particle flux (for each fluid species)	s^{-1}
fhex	Poloidal electron heat flux	W
fhey	Radial electron heat flux	W
fhix	Poloidal ion heat flux	W
fhiy	Radial ion heat flux	W
fchx	Poloidal electric current	A
fchy	Radial electric current	A
If `cdfmovie_b25_sources` ≥ 1		
rsana	Volumetric ionisation rate (for each fluid species)	$m^{-3}s^{-1}$
rsahi	Volumetric ionisation energy source/sink (for each fluid species)	Wm^{-3}
rsahisum	Total volumetric ionisation energy source/sink	Wm^{-3}
rrana	Volumetric recombination rate (for each fluid species)	$m^{-3}s^{-1}$
rrahi	Volumetric recombination energy source/sink (for each fluid species)	Wm^{-3}
rrahisum	Total volumetric recombination energy source/sink	Wm^{-3}

continued on next page

continued from previous page		
quantity	description	units
rcxna	Volumetric charge exchange rate (for each fluid species)	$m^{-3}s^{-1}$
rcxhi	Volumetric charge exchange energy source/sink (for each fluid species)	Wm^{-3}
rcxhisum	Total volumetric charge exchange energy source/sink	Wm^{-3}
rqahe	Volumetric electron cooling rate (for each fluid species)	Wm^{-3}
rqahesum	Total volumetric electron cooling rate	Wm^{-3}
rqrad	Volumetric line radiation rate (for each fluid species)	Wm^{-3}
rqahesum	Total volumetric line radiation rate	Wm^{-3}
If `cdfmovie_eirene_state_variables` ≥ 1 (if `Eirene` is used)		
dab2	Atom density (for each atom species)	m^{-3}
tab2	Atom temperature (for each atom species)	eV
dmb2	Molecule density (for each molecule species)	m^{-3}
tmb2	Molecule temperature (for each molecule species)	eV
If `cdfmovie_eirene_fluxes` ≥ 1 (if `Eirene` is used)		
rfluxa	Atomic radial flux density (for each atom species)	$m^{-2}s^{-1}$
pfluxa	Atomic poloidal flux density (for each atom species)	$m^{-2}s^{-1}$
refluxa	Atomic radial energy flux density (for each atom species)	Wm^{-2}
pefluxa	Atomic poloidal energy flux density (for each atom species)	Wm^{-2}
rfluxm	Molecular radial flux density (for each molecule species)	$m^{-2}s^{-1}$
pfluxm	Molecular poloidal flux density (for each molecule species)	$m^{-2}s^{-1}$
refluxm	Molecular radial energy flux density (for each molecule species)	Wm^{-2}
pefluxm	Molecular poloidal energy flux density (for each molecule species)	Wm^{-2}
If `cdfmovie_eirene_sources` ≥ 1 (if `Eirene` is used)		
eirene_papl_sna	Particle source density from atom-plasma collisions (for each fluid species)	$m^{-3}s^{-1}$
eirene_pmpl_sna	Particle source density from molecule-plasma collisions (for each fluid species)	$m^{-3}s^{-1}$
eirene_pppl_sna	Particle source density from recombinations (for each fluid species)	$m^{-3}s^{-1}$
eirene_eael_she	Electron energy source density from atom-plasma collisions	Wm^{-3}
eirene_emel_she	Electron energy source density from molecule-plasma collisions	Wm^{-3}
eirene_epel_she	Electron energy source density from recombinations	Wm^{-3}
eirene_eapl_shi	Ion energy source density from atom-plasma collisions	Wm^{-3}
eirene_empl_shi	Ion energy source density from molecule-plasma collisions	Wm^{-3}
eirene_eppl_shi	Ion energy source density from recombinations	Wm^{-3}
eneutrad	Volumetric radiation rate due to `Eirene` atoms (for each atom species)	Wm^{-3}
eneutradsum	Total volumetric radiation rate due to `Eirene` atoms	Wm^{-3}
emolrad	Volumetric radiation rate due to `Eirene` molecules (for each molecule species)	Wm^{-3}
emolradsum	Total volumetric radiation rate due to `Eirene` molecules	Wm^{-3}

Table F.1: Multi-dimensional time-dependent quantities stored in b2movies.nc

Users should be careful when requesting the generation of `b2movies.nc`, because it can become very heavy. The following list summarizes the additional filesize which can be roughly expected, on a 96×36 grid, for each of the requested data groups:

- If `cdfmovie_b25_state_variables` ≥ 1: 40 kB per `B2.5` fluid species per outputted timestep.

- If `cdfmovie_b25_fluxes` $\geq$ 1: 75 kB per `B2.5` fluid species per outputted timestep.
- If `cdfmovie_b25_sources` $\geq$ 1: 250 kB per `B2.5` fluid species per outputted timestep.
- If `cdfmovie_eirene_state_variables` $\geq$ 1: 60 kB per `Eirene` atom/molecule species per outputted timestep.
- If `cdfmovie_eirene_fluxes` $\geq$ 1: 120 kB per `Eirene` atom/molecule species per outputted timestep.
- If `cdfmovie_eirene_sources` $\geq$ 1: 100 kB per `B2.5` fluid species + 45 kB per `Eirene` atom/molecule species per outputted timestep.

Therefore, if the write of this file is requested for 1000 timesteps on a D+He+C case (12 `B2.5` fluid species, 3 `Eirene` atom species, and 1 `Eirene` molecule species):

- Setting the switch `cdfmovie_b25_state_variables` $\geq$ 1 will cause additional 480 MB of filesize.
- Setting the switch `cdfmovie_b25_fluxes` $\geq$ 1 will cause additional 900 MB of filesize.
- Setting the switch `cdfmovie_b25_sources` $\geq$ 1 will cause additional 3.0 GB of filesize.
- Setting the switch `cdfmovie_eirene_state_variables` $\geq$ 1 will cause additional 240 MB of filesize.
- Setting the switch `cdfmovie_eirene_fluxes` $\geq$ 1 will cause additional 480 MB of filesize.
- Setting the switch `cdfmovie_eirene_sources` $\geq$ 1 will cause additional 1.4 GB of filesize.

If all the aforementioned switches are $\geq$ 1, then a filesize of roughly 6.5 GB can be expected after 1000 outputted timesteps.

The arrays outputted to `eirenemovies.nc` are, instead, described in the below Table F.2. The file is created, in case of couples runs, only if the switch `cdfmovie_triangles` $\geq$ 1.

quantity	description	units
pdena	Atom particle density (for each atom species)	m^{-3}
pdenm	Molecule particle density (for each molecule species)	m^{-3}
edena	Atom energy density (for each atom species)	eVm^{-3}
edenm	Molecule energy density (for each molecule species)	eVm^{-3}
vxdena	Atom x-directed momentum density (for each atom species)	$kgs^{-1}m^{-2}$
vxdenm	Molecule x-directed momentum density (for each molecule species)	$kgs^{-1}m^{-2}$
vydena	Atom y-directed momentum density (for each atom species)	$kgs^{-1}m^{-2}$
vydenm	Molecule y-directed momentum density (for each molecule species)	$kgs^{-1}m^{-2}$
vzdena	Atom z-directed momentum density (for each atom species)	$kgs^{-1}m^{-2}$
vzdenm	Molecule z-directed momentum density (for each molecule species)	$kgs^{-1}m^{-2}$

Table F.2: Multi-dimensional time-dependent quantities stored in eirenemovies.nc

Similarly as `b2movies.nc`, also `eirenemovies.nc` can become very heavy. For example, on a grid consisting of 10000 triangles, the filesize can be roughly expected to be 360 kB per `Eirene` atom/molecule species per outputted timestep. Therefore, if the write of this file is requested for 1000 timesteps on a D+He+C case (3 `Eirene` atom species and 1 `Eirene` molecule species), a filesize of roughly 1.4 GB can be expected.

Appendix G

Quantities stored in b2tallies.nc

These are the tallies stored in `b2tallies.nc`. The frequency at which (and whether) these quantities are written out is controlled by the `'tallies_netcdf'` switch.

If `'b2mndr_stim'` is set to a negative value, data from the current **SOLPS-ITER** run will be appended to the existing `b2tallies.nc` file, otherwise the file is created anew.

The meaning of the dimensions are:

vreg volume region tally (see Appendix I and section 3.5.4). Here, however, region 0 stands for the whole plasma.

xreg x-directed region tally (see Appendix I)

yreg y-directed region tally (see Appendix I)

ns species index

time time index

quantity	dimensions	units	description
b2bremreg	vreg, time	W	Bremsstrahlung radiation rate
b2divua	vreg, time	W	Heating term from ∇u_a
b2divue	vreg, time	W	Heating term from ∇u_e
b2exba	vreg, time	W	Heating term to ions from $\mathbf{E} \times \mathbf{B}$
b2exbe	vreg, time	W	Heating term to electrons from $\mathbf{E} \times \mathbf{B}$
b2fraa	vreg, time	W	Heating from inter-species friction
b2joule	vreg, time	W	Joule heating
b2qie	vreg, time	W	Electron/ion exchange term
b2radreg	vreg, time	W	Line radiation rate
b2she	vreg, time	W	Incoming electron heat sources
b2she0	vreg, time	W	Electron heat sources from `b2sihs.F`
b2shi	vreg, time	W	Incoming ion heat sources
b2shi0	vreg, time	W	Ion heat sources from `b2sihs.F`
b2stbc_sch_reg	vreg, time	A	Boundary charge source
b2stbc_she_reg	vreg, time	W	Boundary electron energy source
b2stbc_shi_reg	vreg, time	W	Boundary ion energy source
b2stbc_smo_reg	vreg, ns, time	$kg.m.s^{-1}$	Boundary parallel momentum source
b2stbc_sna_reg	vreg, ns, time	s^{-1}	Boundary particle source
b2stbc_sne_reg	vreg, time	s^{-1}	Boundary electron source
b2stbr_sch_reg	vreg, time	A	Recycling charge source
b2stbr_she_reg	vreg, time	W	Recycling electron energy source

continued on next page

continued from previous page

quantity	dimensions	units	description
b2stbr_shi_reg	vreg, time	W	Recycling ion energy source
b2stbr_smo_reg	vreg, ns, time	$kg.m.s^{-1}$	Recycling parallel momentum source
b2stbr_sna_reg	vreg, ns, time	s^{-1}	Recycling particle source
b2stbr_sne_reg	vreg, time	s^{-1}	Recycling electron source
b2stbm_sch_reg	vreg, time	A	Additional charge source
b2stbm_she_reg	vreg, time	W	Additional electron energy source
b2stbm_shi_reg	vreg, time	W	Additional ion energy source
b2stbm_smo_reg	vreg, ns, time	$kg.m.s^{-1}$	Additional parallel momentum source
b2stbm_sna_reg	vreg, ns, time	s^{-1}	Additional particle source
b2stbm_sne_reg	vreg, time	s^{-1}	Additional electron source
b2sext_sch_reg	vreg, time	A	External charge source
b2sext_she_reg	vreg, time	W	External electron energy source
b2sext_shi_reg	vreg, time	W	External ion energy source
b2sext_smo_reg	vreg, ns, time	$kg.m.s^{-1}$	External parallel momentum source
b2sext_sna_reg	vreg, ns, time	s^{-1}	External particle source
b2sext_sne_reg	vreg, time	s^{-1}	External electron source
b2str	vreg, time	W	Strange term
b2visa	vreg, time	W	Ion viscous heating
b2wrong1	vreg, time	W	Obsolete
b2wrong2	vreg, time	W	Obsolete
b2wrong3	vreg, time	W	Obsolete
fchxreg	xreg, time	A	Poloidal current
fchyreg	yreg, time	A	Radial current
fhexreg	xreg, time	W	Poloidal electron thermal energy flux
fheyreg	yreg, time	W	Radial electron thermal energy flux
fhixreg	xreg, time	W	Poloidal ion thermal energy flux
fhiyreg	yreg, time	W	Radial ion thermal energy flux
fhmxreg	xreg, ns, time	W	Poloidal parallel kinetic energy flux
fhmyreg	yreg, ns, time	W	Radial parallel kinetic energy flux
fhpxreg	xreg, ns, time	W	Poloidal ionisation energy flux
fhpyreg	yreg, ns, time	W	Radial ionisation energy flux
fhjxreg	xreg, time	W	Poloidal electrostatic energy flux
fhjyreg	yreg, time	W	Radial electrostatic energy flux
fhtxreg	xreg, time	W	Poloidal total energy flux
fhtyreg	yreg, time	W	Radial total energy flux
fnaxreg	xreg, ns, time	s^-1	Poloidal particle flux
fnayreg	yreg, ns, time	s^-1	Radial particle flux
nareg	vreg, ns, time	m^{-3}	Average ion density $\frac{\int n_a dV}{\int dV}$
ne2reg	vreg, time	m^{-3}	Average weighted density $\frac{\int Z_a^2 n_a dV}{\int dV}$
nereg	vreg, time	m^{-3}	Average electron density $\frac{\int n_e dV}{\int dV}$
nireg	vreg, time	m^{-3}	Average total ion density $\frac{\int n_i dV}{\int dV}$
poreg	vreg, time	eV	Average potential $\frac{\int \Phi dV}{\int dV}$
qconvexreg	xreg, time	W	Poloidally convected electron energy
qconveyreg	yreg, time	W	Radially convected electron energy
qconvixreg	xreg, ns, time	W	Poloidally convected ion energy
qconviyreg	yreg, ns, time	W	Radially convected ion energy
rcxhireg	vreg, ns, time	W	Charge-exchange ion energy source
rcxmoreg	vreg, ns, time	$kg.m.s^{-1}$	Charge-exchange ion parallel momentum source

continued on next page

continued from previous page			
quantity	dimensions	units	description
rcxnareg	vreg, ns, time	s^{-1}	Charge-exchange particle source
rdneureg	vreg, time	W	Total radiation from `Eirene` neutrals
resco_reg	vreg, ns, time		Continuity equation residuals
reshe_reg	vreg, time	W	Electron heat equation residuals
reshi_reg	vreg, time	W	Ion heat equation residuals
resmo_reg	vreg, ns, time	$kg.m.s^{-1}$	Parallel momentum equations residuals
resmt_reg	vreg, time	$kg.m.s^{-1}$	Total parallel momentum equation residuals
respo_reg	vreg, time	eV	Potential equation residuals
rqahereg	vreg, ns, time	W	Electron cooling rate
rqbrmreg	vreg, ns, time	W	Bremsstrahlung
rqradreg	vreg, ns, time	W	Line radiation
rrahireg	vreg, ns, time	W	Recombination ion heat source
rramoreg	vreg, ns, time	$kg.m.s^{-1}$	Recombination parallel momentum source
rranareg	vreg, ns, time	s^{-1}	Recombination particle source
rsahireg	vreg, ns, time	W	Ionisation ion heat source
rsamoreg	vreg, ns, time	$kg.m.s^{-1}$	Ionisation parallel momentum source
rsanareg	vreg, ns, time	s^{-1}	Ionisation particle source
tereg	vreg, time	eV	Average electron temperature $\frac{\int T_e n_e dV}{\int n_e dV}$
tireg	vreg, time	eV	Average ion temperature $\frac{\int T_i n_i dV}{\int n_i dV}$
volreg	vreg, time	m^3	Total volume $\int dV$

Table G.1: Quantities stored in b2tallies.nc

Appendix H

Quantities printed by 'b2wdat_iout' = '4'

The switch setting 'b2wdat_iout' = '4' allows for a number of output files, developed in Saint Petersburg by Elizaveta Kaveeva (with some technical help from Sergey Voskoboynikov and Ilya Senichenkov), in order to obtain explicitly each term from the equations listed in Appendix C exactly as they are computed in the code.

For this purpose the output of each quantity is produced exactly from the subroutine where this quantity is computed, e.g. the velocity u_a (ua) is printed from the subroutine b2npmo. Each quantity is printed into a separate ASCII file as a 2D table. All these files are stored in the output/ folder inside the run directory. The general naming convention is the following (except for geometry related quantities and a few others):

```
<subroutine_name>_<quantity>[_<XXX>].dat
```

Here, <subroutine_name> is the name of subroutine (or sometimes the name of the Fortran source file containing this subroutine) from which the output is produced, while <quantity> is the quantity name, which usually coincides with the name of the corresponding variable inside the subroutine, if such a variable exists. Otherwise some association is used. The species index is denoted here as XXX (three digits), and is used for species-dependent quantities (such as na or ua), and is omitted from common quantities (such as te, ti or po). For example, the file containing the density of the species with index is = 1 would be

```
b2npco_na_001.dat
```

The content of the file looks like the following:

	-1	0	1	2	3
36	2.525043E+19	2.517455E+19	1.625484E+19	1.444400E+19	1.407206E+19
35	2.681619E+19	2.673561E+19	1.804678E+19	1.575019E+19	1.516221E+19
34	3.360574E+19	3.350655E+19	2.279144E+19	1.984601E+19	1.907786E+19
33	4.343278E+19	4.330891E+19	2.978478E+19	2.597130E+19	2.498425E+19
32	5.664300E+19	5.649203E+19	3.951054E+19	3.444186E+19	3.309993E+19
31	7.393565E+19	7.375144E+19	5.240743E+19	4.567851E+19	4.384763E+19
30	9.590684E+19	9.568531E+19	6.914344E+19	6.032204E+19	5.781704E+19
29	1.230569E+20	1.227959E+20	9.057685E+19	7.919435E+19	7.575477E+19
28	1.556381E+20	1.553424E+20	1.176512E+20	1.032794E+20	9.859909E+19
27	1.932463E+20	1.929148E+20	1.497150E+20	1.323182E+20	1.263528E+20
26	2.335330E+20	2.331599E+20	1.844608E+20	1.643623E+20	1.576109E+20
25	2.721367E+20	2.716981E+20	2.163190E+20	1.948967E+20	1.888368E+20
24	2.992329E+20	2.987071E+20	2.372826E+20	2.165452E+20	2.127225E+20
23	3.069312E+20	3.063232E+20	2.417565E+20	2.224695E+20	2.209700E+20

```
 22   2.972805E+20  2.966824E+20  2.323948E+20  2.140750E+20  2.134266E+20
 21   2.785257E+20  2.779398E+20  2.158066E+20  1.981130E+20  1.971063E+20
 20   2.592405E+20  2.586764E+20  1.986355E+20  1.815245E+20  1.801132E+20
 19   2.447657E+20  2.442036E+20  1.861651E+20  1.694604E+20  1.678487E+20
 18   2.378737E+20  2.373074E+20  1.802841E+20  1.638746E+20  1.621619E+20
 17   2.345108E+20  2.339292E+20  1.774551E+20  1.611951E+20  1.594105E+20
 16   2.302662E+20  2.297536E+20  1.742077E+20  1.580295E+20  1.561263E+20
 15   2.263556E+20  2.258411E+20  1.711906E+20  1.550573E+20  1.530871E+20
 14   2.219169E+20  2.214601E+20  1.678177E+20  1.517517E+20  1.497154E+20
 13   2.171240E+20  2.166303E+20  1.639932E+20  1.481086E+20  1.459853E+20
 12   2.121178E+20  2.116556E+20  1.601201E+20  1.444459E+20  1.422191E+20
 11   2.064163E+20  2.059699E+20  1.556076E+20  1.401716E+20  1.378837E+20
 10   2.005827E+20  2.000971E+20  1.507593E+20  1.356803E+20  1.333992E+20
  9   1.944365E+20  1.940038E+20  1.458187E+20  1.311082E+20  1.288094E+20
  8   1.872176E+20  1.868137E+20  1.399629E+20  1.256621E+20  1.233961E+20
  7   1.797522E+20  1.793348E+20  1.339149E+20  1.200777E+20  1.178179E+20
  6   1.717291E+20  1.713409E+20  1.275705E+20  1.142528E+20  1.119222E+20
  5   1.627565E+20  1.623631E+20  1.203216E+20  1.076693E+20  1.053815E+20
  4   1.534325E+20  1.530565E+20  1.128591E+20  1.008665E+20  9.870904E+19
  3   1.433210E+20  1.429840E+20  1.049081E+20  9.352627E+19  9.151503E+19
  2   1.324803E+20  1.321455E+20  9.642263E+19  8.568567E+19  8.379455E+19
  1   1.208139E+20  1.205130E+20  8.754242E+19  7.755114E+19  7.573841E+19
  0   1.083148E+20  1.080124E+20  7.790863E+19  6.878814E+19  6.712576E+19
 -1   1.015473E+20  1.012638E+20  7.310223E+19  6.428731E+19  6.265369E+19
```

Here the first row (truncated in the example above) gives the poloidal cell index `ix`, the first column (to the right) contains the radial cell index `iy`, the rest are the values of the 2D array in each cell.

The price for the possibility to output each quantity exactly from the subroutine where this quantity is calculated is the production of output files each time the corresponding subroutine is called. This gives an additional load to the filesystem due to the increased number of file writing operations and slows the computation significantly. Therefore it is NOT recommended to perform long-lasting runs with `'b2wdat_iout'` set to `'4'`. Instead, after a normal run with `'b2wdat_iout' = '0'` finishes, it is suggested to set `'b2wdat_iout' = '4'` and to continue the run for a small number of time steps, say, between 1 and 10. Then the whole set of output files may be read by any suitable tool (e.g. MATLAB) for further analysis, plotting, comparing, etc. Setting `'b2wdat_iout' = '1'` will provide an extensive set of files giving many quantities as they exist at the end of the B2.5 iteration. With `'b2wdat_iout' = '2'`, one obtains a set of profiles at the midplane and targets. With `'b2wdat_iout' = '3'`, one obtains a set of files providing a description of the primary code quantities.

Normally, the data in these files will correspond to the last time the routine writing them was called or the plasma solution at the end of the simulation. Setting `'b2wdat_append' = '1'` will instead append the data in these files at every call or every B2.5 iteration. Again, this switch should be used sparingly to avoid very large I/O load on the program.

The table below provides a listing of the various files that may be produced, the variables they contain and the equations to which they relate.

Note that some quantities (like density, velocity, temperature, potentials, magnetic field, metric coefficients, etc.) are computed at cell centers, while others (like particle, energy, and momentum fluxes, currents, and quantities which contribute to them) are computed at cell faces.

quantity	units	file(s)	equation
Main quantities			
n_a	m^{-3}	`b2npco_na[XXX].dat` or `b2npc9_na[XXX].dat` or	(C.1)
continued on next page			

continued from previous page

quantity	units	file(s)	equation
		b2npc11_na[XXX].dat	
u_a	m/s	b2npmo_ua[XXX].dat	(C.55), (C.56)
T_e	J	b2npht_te.dat or b2nph9_te.dat	(C.105)
T_i	J	b2npht_ti.dat or b2nph9_ti.dat	(C.154)
T_e	eV	te_eV.dat	(C.105)
T_i	eV	ti_eV.dat	(C.154)
φ	V	b2nppo_po.dat or b2npp7_po.dat	(C.191)
n_e	m^{-3}	ne.dat	(C.27)
Geometry and related quantities			
$\sqrt{g}$	m^3	vol.dat	
h_x	m	hx.dat	
h_y	m	hy.dat	
h_{y1}	m	hy1.dat	
h_z	m	hz.dat	
B	T	bb.dat	
B_x	T	bbx.dat	
B_z	T	bbz.dat	
R_0	m	crx0.dat	
Z_0	m	crz0.dat	
R_1	m	crx1.dat	
Z_1	m	crz1.dat	
R_2	m	crx2.dat	
Z_2	m	crz2.dat	
R_3	m	crx3.dat	
Z_3	m	crz3.dat	
Continuity equation and related quantities			
$S_a^n\sqrt{g}$	s^{-1}	b2npco_sna[XXX].dat or b2npc9_sna[XXX].dat or b2npc11_sna[XXX].dat	(C.1)
$\left(S_{I_a}^n + S_{I_{a+1}}^n\right)\sqrt{g}$	s^{-1}	b2stel_sna_ion[XXX].dat	(C.31), (C.32)
$\left(S_{R_a}^n + S_{R_{a-1}}^n\right)\sqrt{g}$	s^{-1}	b2stel_sna_rec[XXX].dat	(C.34), (C.35)
$\left(S_{CX_a}^n + S_{CX_{a-1}}^n\right)\sqrt{g}$	s^{-1}	b2stcx_sna_[XXX].dat	(C.37)
$S_a^{\text{Eirene}}\sqrt{g}$	s^{-1}	b2stbr_sna_eir[XXX].dat	
$S_a^{BC}\sqrt{g}$	s^{-1}	b2stbc_phys_sna[XXX].dat	
$\sqrt{g}\dfrac{\partial n_a}{\partial t}$	s^{-1}	b2npco_dnadt[XXX].dat or b2npc9_dnadt[XXX].dat or b2npc11_dnadt[XXX].dat	(C.1)
$\dfrac{\sqrt{g}}{h_x}\widetilde{\Gamma}_{ax}$	s^{-1}	b2npc9_fnax[XXX].dat or b2npc11_fnax[XXX].dat	(C.2)
$\dfrac{\sqrt{g}}{h_x}\widetilde{\Gamma}_{ay}$	s^{-1}	b2npc9_fnay[XXX].dat or b2npc11_fnay[XXX].dat	(C.3)
$\dfrac{\sqrt{g}}{h_x}b_x V_{a\parallel} n_a$	s^{-1}	b2tfnb_bxuanax[XXX].dat	(C.2), (C.7)

continued on next page

continued from previous page

quantity	units	file(s)	equation
$\frac{\sqrt{g}}{h_x} V_{ax}^{(E)} n_a$	s^{-1}	`b2tfnb_vaecrbnax[XXX].dat`	(C.13)
$\frac{\sqrt{g}}{h_x} V_{ay}^{(E)} n_a$	s^{-1}	`b2tfnb_vaecrbnay[XXX].dat`	(C.13)
$\frac{\sqrt{g}}{h_x} V_a^{\mathrm{corr_dpc}} n_a$	s^{-1}	`b2tfnb_dpccornax[XXX].dat`	(C.29)
$\frac{\sqrt{g}}{h_x} V_{ay}^{(AN)} n_a$	s^{-1}	`b2tfnb_cvlbnay[XXX].dat`	(C.48)
$-D_{n,a} \frac{\sqrt{g}}{h_x} \frac{1}{h_x} \frac{\partial n_a}{\partial x}$	s^{-1}	`b2tfnb_dPat_mdf_gradnax[XXX].dat`	(C.50)
$-D_{n,a} \frac{\sqrt{g}}{h_y} \frac{1}{h_y} \frac{\partial n_a}{\partial y}$	s^{-1}	`b2tfnb_dPat_mdf_gradnay[XXX].dat`	(C.50)
$-\tilde{D}_{p,ax} \frac{\sqrt{g}}{h_x} \frac{1}{h_x} \frac{\partial p_a}{\partial x}$	s^{-1}	`b2tfnb_dgradpbx[XXX].dat`	(C.53)
$-\tilde{D}_{p,ay} \frac{\sqrt{g}}{h_y} \frac{1}{h_y} \frac{\partial p_a}{\partial y}$	s^{-1}	`b2tfnb_dgradpby[XXX].dat`	(C.54)
$\frac{\sqrt{g}}{h_x} \widetilde{\Gamma}_{ax}^{(PSch)}$ [1]	s^{-1}	`b2tfnb_fnbPSchx[XXX].dat`	(C.49)
$\frac{\sqrt{g}}{h_y} \widetilde{\Gamma}_{ay}^{(PSch)}$ [2]	s^{-1}	`b2tfnb_fnbPSchy[XXX].dat`	(C.49)
$D_{n,a} \frac{\sqrt{g}}{h_x^2}$	m^3/s	`b2trno_cdnax[XXX].dat`	(C.50)
$D_{n,a} \frac{\sqrt{g}}{h_y^2}$	m^3/s	`b2trno_cdnay[XXX].dat`	(C.50)
$\tilde{D}_{ax}^{p} \frac{\sqrt{g}}{h_x^2}$	$m \cdot s/kg$	`b2trno_cdpax[XXX].dat`	(C.53)
$\tilde{D}_{ay}^{p} \frac{\sqrt{g}}{h_y^2}$	$m \cdot s/kg$	`b2trno_cdpay[XXX].dat`	(C.54)
$D_{n,a}^{AN}$	m^2/s	`b2tqna_dna0[XXX].dat`	(C.50)
$D_a{}^{p}$	m^2/s	`b2tqna_dpa0[XXX].dat`	(C.52)
$V_{ax}{}^{(E)}$	m/s	`b2tfnb_vbecrbx001.dat`	(C.44)
$V_{ay}{}^{(E)}$	m/s	`b2tfnb_vbecrby001.dat`	(C.45)
$W_x{}^{(dia)}$	m/s	`b2tfnb_wbdiax[XXX].dat`	(C.25)
$W_y{}^{(dia)}$	m/s	`b2tfnb_wbdiay[XXX].dat`	(C.25)
$\tilde{V}_{ax}^{(dia)}$	m/s	`b2tfnb_vbdiax[XXX].dat`	(C.46)
$\tilde{V}_{ay}^{(dia)}$	m/s	`b2tfnb_vbdiay[XXX].dat`	(C.47)
$\frac{\sqrt{g}}{h_x} \tilde{V}_{ax}^{(dia)} n_a$	s^{-1}	`b2tfnb_vadianax[XXX].dat`	(C.18)
$\frac{\sqrt{g}}{h_x} \tilde{V}_{ay}^{(dia)} n_a$	s^{-1}	`b2tfnb_vadianay[XXX].dat`	(C.18)
$E_{k,a}$	J	`b2tfnb_kbnrgy[XXX].dat`	(C.24)
$V_{ay}{}^{(AN)}$	m/s	`b2tqna_vla0y[XXX].dat`	(C.48)
Γ_{ax}	s^{-1}	`b2tfnb_fnbx[XXX].dat`	(C.7)
Γ_{ay}	s^{-1}	`b2tfnb_fnby[XXX].dat`	(C.8)
residuals	s^{-1}	`b2npco_resco[XXX].dat` or `b2npc9_resco[XXX].dat` or `b2npc11_resco[XXX].dat`	
Momentum balance equation and related quantities			

continued on next page

[1] Mind the sign! It is opposite in the output file and the flux balance.
[2] Mind the sign! It is opposite in the output file and the flux balance.

continued from previous page

quantity	units	file(s)	equation
$V_{\parallel a}$	m/s	b2npmo_ua[XXX].dat	(C.55), (C.56)
$S_a^m h_z\sqrt{g}$	$N\cdot m$	b2npmo_smb[XXX].dat	(C.57)
$m_a \dfrac{\partial n_a V_{\parallel a}}{\partial t} h_z\sqrt{g}$	$N\cdot m$	b2npmo_madnavadt[XXX].dat	(C.55), (C.56)
$m_a n_a \dfrac{\partial V_{\parallel a}}{\partial t} h_z\sqrt{g}$	$N\cdot m$	b2npmo_manadvadt[XXX].dat	
$m_a V_{\parallel a} \dfrac{\partial n_a}{\partial t} h_z\sqrt{g}$	$N\cdot m$	b2npmo_mavadnadt[XXX].dat	
$\dfrac{h_z\sqrt{g}}{h_x}\Gamma_{ax}^m$	$N\cdot m$	b2npmo_fmox[XXX].dat	(C.93)
$\dfrac{h_z\sqrt{g}}{h_y}\Gamma_{ay}^m$	$N\cdot m$	b2npmo_fmoy[XXX].dat	(C.94)
$-\dfrac{b_x}{h_x}\dfrac{\partial n_a T_i}{\partial x} h_z\sqrt{g}$	$N\cdot m$	b2sigp_smogpi[XXX].dat	(C.55), (C.56)
$-Z_a e n_a \dfrac{b_x}{h_x}\dfrac{\partial \Phi}{\partial x} h_z\sqrt{g}$	$N\cdot m$	b2sigp_smogpo[XXX].dat	(C.55)
$S_{CX_a}^m h_z\sqrt{g}$	$N\cdot m$	b2stcx_smq[XXX].dat	(C.69)
$S_{CF}^m h_z\sqrt{g}$	$N\cdot m$	b2npmo_smocf[XXX].dat	(C.61), (C.63)
$S_{I_a}^m h_z\sqrt{g}$	$N\cdot m$	b2stel_smq_ion[XXX].dat	(C.57), (C.63)
$S_{R_a}^m h_z\sqrt{g}$	$N\cdot m$	b2stel_smq_rec[XXX].dat	(C.57), (C.63)
$S_{Therm,ia}^m h_z\sqrt{g}$	$N\cdot m$	b2npmo_smotfia[XXX].dat	(C.78)
$S_{Therm,ea}^m h_z\sqrt{g}$	$N\cdot m$	b2npmo_smotfea[XXX].dat	(C.76)
$S_{fr,ea}^m h_z\sqrt{g}$	$N\cdot m$	b2npmo_smofrea[XXX].dat	(C.77)
$S_{fr,ia}^m h_z\sqrt{g}$	$N\cdot m$	b2npmo_smofria[XXX].dat	(C.79)
$S_{AN_a}^m h_z\sqrt{g}$	$N\cdot m$	b2npmo_smoan[XXX].dat	(C.62)
$S_{BC_a}^m h_z\sqrt{g}$	$N\cdot m$	b2stbc_phys_smo[XXX].dat	
$S_{EIRENE_{a\parallel}}^m h_z\sqrt{g}$	$N\cdot m$	b2stbr_smo_eir[XXX].dat	(C.55), (C.56)
$S_{EIRENE_{a\perp}}^m h_z\sqrt{g}$	$N\cdot m$	b2stbr_smd_eir[XXX].dat	(C.210), (C.228)
$S_{EIRENE_{a\psi}}^m h_z\sqrt{g}$	$N\cdot m$	b2stbr_smr_eir[XXX].dat	(C.210))
$\left(\nabla\cdot\overleftrightarrow{\pi}_a^{V_\parallel}\right)_\parallel^{res} h_z\sqrt{g}$	$N\cdot m$	b2npmo_smovv001.dat	(C.58), (C.59)
$\left(\nabla\cdot\overleftrightarrow{\pi}_a^{q_\parallel}\right)_\parallel h_z\sqrt{g}$	$N\cdot m$	b2npmo_smovh001.dat	(C.58), (C.60)
$\widetilde{\widetilde{\eta}}_{ax}^{(CL)} \dfrac{h_z\sqrt{g}}{h_x^2}$	$kg\cdot m/s$	b2trcl_luciani_fllim_cvsahzx[XXX].dat	(C.97)
$\widetilde{\eta}_{ax}^{(CL)} \dfrac{h_z\sqrt{g}}{h_x^2}$	$kg\cdot m/s$	b2trcl_luciani_cvsahzx[XXX].dat	(C.100)
$\dfrac{\widetilde{\widetilde{\eta}}_{ax}^{(CL)}}{\rho_a^{rel}} \dfrac{h_z\sqrt{g}}{h_x^2}$ (only for Section C.11)	$kg\cdot m/s$	b2tral_cvsahz_drhox[XXX].dat	(C.379), (C.380)
$\widetilde{\widetilde{\eta}}_{ax}^{(CL)} \dfrac{h_z\sqrt{g}}{h_x^2}$ (only for Section C.11)	$kg\cdot m/s$	b2tral_cvsahz_clx[XXX].dat	(applied in C.382)
$\widetilde{\widetilde{\eta}}_{ax}^{(CL)} \dfrac{h_z\sqrt{g}}{h_x^2}$ (only for Section C.11)	$kg\cdot m/s$	b2tral_cvsahz_drhox[XXX].dat	(C.393)
$\widetilde{\eta}_{ax}^{(CL)} \dfrac{h_z\sqrt{g}}{h_x^2}$ (only for Section C.11)	$kg\cdot m/s$	b2tral_cvsahz0_flx[XXX].dat	(C.394)
$\eta_{ax}^{(CL)} \dfrac{h_z\sqrt{g}}{h_x^2}$ (only for Section C.11)	$kg\cdot m/s$	b2tral_cvsahzx0_noflx[XXX].dat	(C.397)
$\eta_a^{(AN)} \dfrac{h_z\sqrt{g}}{h_x^2}$	$kg\cdot m/s$	b2trno_cvsahzx[XXX].dat	(C.96)
$\eta_a^{(AN)} \dfrac{h_z\sqrt{g}}{h_y^2}$	$kg\cdot m/s$	b2trno_cvsahzy[XXX].dat	(C.96)

continued on next page

continued from previous page

quantity	units	file(s)	equation
$\eta_{ax}^{CL}\frac{h_z\sqrt{g}}{h_x^2}$	$kg\cdot m/s$	b2trcl_cvsahzx[XXX].dat	(C.101)
$\frac{\widetilde{\eta}_{\alpha\beta x}^{(h_i A)}}{p_\alpha}\frac{h_z\sqrt{g}}{h_x^2}$ (only for Section C.11)	$kg\cdot m/s$	b2tral_cvsahz_hAdp_lucx[XXX]_[YYY].dat	(C.418)
$\frac{\widetilde{\eta}_{\alpha\beta x}^{(h_i A)}}{p_\alpha}\frac{\sqrt{g}}{h_x^2}$ (only for Section C.11)	$kg\cdot m/s$	b2tral_cvsa_hAdp_lucx[XXX]_[YYY].dat	(C.418)
$\frac{\eta_{\alpha\beta x}^{(h_i A)}}{p_\alpha}\frac{h_z\sqrt{g}}{h_x^2}$ (only for Section C.11)	$kg\cdot m/s$	b2tral_cvsahz0_hAdpx[XXX]_[YYY].dat	(C.418)
$\frac{\eta_{\alpha\beta x}^{(h_i A)}}{p_\alpha}\frac{\sqrt{g}}{h_x^2}$ (only for Section C.11)	$kg\cdot m/s$	b2tral_cvsa0_hAdpx[XXX]_[YYY].dat	(C.418)
$\frac{\widetilde{\eta}_{\alpha x}^{(h_i B)}}{p_\alpha}\frac{h_z\sqrt{g}}{h_x^2}$ (only for Section C.11)	$kg\cdot m/s$	b2tral_cvsahz_hBdp_lucx[XXX].dat	(C.419)
$\frac{\widetilde{\eta}_{\alpha x}^{(h_i B)}}{p_\alpha}\frac{\sqrt{g}}{h_x^2}$ (only for Section C.11)	$kg\cdot m/s$	b2tral_cvsa_hBdp_lucx[XXX].dat	(C.419)
$\frac{\eta_{\alpha x}^{(h_i B)}}{p_\alpha}\frac{h_z\sqrt{g}}{h_x^2}$ (only for Section C.11)	$kg\cdot m/s$	b2tral_cvsahz0_hBdpx[XXX].dat	(C.419)
$\frac{\eta_{\alpha x}^{(h_i B)}}{p_\alpha}\frac{\sqrt{g}}{h_x^2}$ (only for Section C.11)	$kg\cdot m/s$	b2tral_cvsa0_hBdpx[XXX].dat	(C.419)
τ_a	s	b2tqca_taua[XXX].dat	(C.102)
$\frac{h_z\sqrt{g}}{h_x}\Gamma_{ax}^{Corr}$	m^2/s	b2tfnb_fnb_fcorx[XXX].dat	(C.9)
$\frac{h_z\sqrt{g}}{h_x}\Gamma_{ay}^{Corr}$	m^2/s	b2tfnb_fnb_fcory[XXX].dat	(C.10)
$-D_a^n\frac{1}{h_x}\frac{\partial n_a}{\partial x}\frac{h_z\sqrt{g}}{h_x}$	m/s	b2tfnb_dPat_2diagradnax[XXX].dat	(C.9)
ν_a^{visc}	m^2/s	b2tqna_vsa0[XXX].dat	(C.96)
$-\frac{h_z\sqrt{g}}{h_x}\eta_{ax}\frac{\partial V_{\parallel a}}{h_x\partial x}$	kg/s	b2urmo_etaPat_graduax[XXX].dat	(C.93)
$-\frac{h_z\sqrt{g}}{h_y}\eta_{ay}\frac{\partial V_{\parallel a}}{h_y\partial y}$	$N\cdot m$	b2urmo_etaPat_graduay[XXX].dat	(C.94)
$\frac{h_z\sqrt{g}}{h_x}\left(\frac{4}{3}\widetilde{\eta}_{ax}^{(CL)}\frac{\partial\ln h_z}{h_x\partial x}\right)$	$kg\cdot m/s$	b2npmo_flubvx001.dat	(C.93)
residuals	$N\cdot m$	b2npmo_resmo[XXX].dat	
Electron heat balance equation and related quantities			
T_e	J	b2npht_te.dat or b2nph9_te.dat	(C.105)
$\frac{\sqrt{g}}{h_x}\widetilde{q}_{ex}$	W	b2nph9_fhex.dat	(C.112)
$\frac{\sqrt{g}}{h_y}\widetilde{q}_{ey}$	W	b2nph9_fhey.dat	(C.145)
$S_e^{heat}\sqrt{g}$	W	b2npht_she.dat or b2nph9_she.dat	(C.106)
$Q_R\sqrt{g}$	W	b2stel_she_rad.dat	(C.109)
$Q_\Delta\sqrt{g}$	W	b2npht_shei.dat or b2nph9_shei.dat	(C.107)
$Q_e^{\mathrm{EIRENE}}\sqrt{g}$	W	b2stbr_she_eir.dat	(C.106)
$Q_{Fei}\sqrt{g}$	W	b2sihs__shefr.dat	(C.111)

continued on next page

continued from previous page

quantity	units	file(s)	equation
$\frac{n_e T_e}{\sqrt{g}} \frac{\partial}{\partial x} \left(\frac{\sqrt{g}}{h_x} b_x V_{e\parallel} \right) \sqrt{g}$	W	`b2sihs__shedu.dat`	(C.105)
$n_e T_e \left(\nabla \cdot \vec{\mathbf{V}}^E \right) \sqrt{g}$	W	`b2sihs__shedd.dat`	(C.105)
$Q_e^{BC} \sqrt{g}$	W	`b2stbc_phys_she.dat`	
$-\frac{j_y^{(ST)}}{e n_e} \frac{\partial n_e T_e}{h_y \partial y} \sqrt{g}$	W	`b2sihs__shest.dat`	(C.105)
$\frac{3}{2} \frac{\partial n_e T_e}{\partial t} \sqrt{g}$	W	`b2npht_dnetedt.dat` or `b2nph9_dnetedt.dat`	(C.105)
$\frac{3}{2} n_e \frac{\partial T_e}{\partial t} \sqrt{g}$	W	`b2npht_nedtedt.dat` or `b2nph9_nedtedt.dat`	
$\frac{3}{2} T_e \frac{\partial n_e}{\partial t} \sqrt{g}$	W	`b2npht_tednedt.dat` or `b2nph9_tednedt.dat`	
$\frac{3}{2} \frac{\sqrt{g}}{h_x} \Gamma_{ex} T_e$	W	`b2tfhe__qe_32x.dat`	(C.116)
$\frac{3}{2} \frac{\sqrt{g}}{h_y} \Gamma_{ey} T_e$	W	`b2tfhe__qe_32y.dat`	(C.148)
$\frac{\sqrt{g}}{h_x} q_{ex}^{P.Sch}$ [3]	W	`b2tfhe__fhePSchx.dat`	(C.114)
$\frac{\sqrt{g}}{h_y} q_{ey}^{P.Sch}$ [4]	W	`b2tfhe__fhePSchy.dat`	(C.146)
$-\frac{\sqrt{g}}{h_x} \kappa_{ex} \frac{1}{h_x} \frac{\partial T_e}{\partial x}$	W	`b2tfhe__qe_ke_gTx.dat`	(C.115)
$-\frac{\sqrt{g}}{h_y} \kappa_{ey} \frac{1}{h_y} \frac{\partial T_e}{\partial y}$	W	`b2tfhe__qe_ke_gTy.dat`	(C.147)
$-\widetilde{c}_e^{(2)} \frac{j_x^{(\parallel)}}{e} T_e$ [5]	W	`b2tfhe__qe_fchpTex.dat`	(C.113)
$\alpha_{ey} \hat{E}_y T_e$	W	`b2tfhe__qe_alphaTehy.dat`	(C.150)
$\Gamma_{ax}^{(he)} \frac{\sqrt{g}}{h_x}$	s^{-1}	`b2tfnb_fnb_hex.dat`	(C.117)
$\Gamma_{ay}^{(he)} \frac{\sqrt{g}}{h_y}$	s^{-1}	`b2tfnb_fnb_hey.dat`	(C.149)
$\kappa_{ex} \frac{\sqrt{g}}{h_x^2}$	s^{-1}	`b2tfhe__chcex_Pat.dat`	(C.123)
$\kappa_{ey} \frac{\sqrt{g}}{h_y^2}$	s^{-1}	`b2tfhe__chcey_Pat.dat`	(C.151)
$\kappa_e^{(AN)} \frac{\sqrt{g}}{h_x^2}$	s^{-1}	`b2trno_chcex.dat`	(C.124)
$\kappa_e^{(AN)} \frac{\sqrt{g}}{h_y^2}$	s^{-1}	`b2trno_chcey.dat`	(C.124)
$\chi_e^{(AN)}$	m^2/s	`b2tqna_hce0.dat`	(C.124)
$\widetilde{\widetilde{\kappa}}_{ex}^{CL} \frac{\sqrt{g}}{h_x^2}$	s^{-1}	`b2trcl_luciani_fllim_chcex.dat`	(C.55)
$\widetilde{\kappa}_{ex}^{CL} \frac{\sqrt{g}}{h_x^2}$	s^{-1}	`b2trcl_luciani_chcex.dat`	(C.126)

continued on next page

[3] Mind the sign! It is opposite in the output file and the heat balance.

[4] Mind the sign! It is opposite in the output file and the heat balance.

[5] If the switch `b2tfhe_alphaTeh = 1` (and `b2tfhe_fch_pTe = 0`), then this output item contains $\alpha_{ex} \hat{E}_x T_e$

continued from previous page

quantity	units	file(s)	equation
$\kappa_{ex}^{CL}\frac{\sqrt{g}}{h_x^2}b_x^2$	s^{-1}	`b2trcl_chcex.dat`	(C.129)
τ_e	s	`b2tqce_taue.dat`	(C.131)
$V_{e\|\|}$	m/s	`b2sifrtf_ue.dat`	(C.133)
$\sigma_{\|\|}\frac{\sqrt{g}}{h_x^2}$	$\frac{1}{m\cdot S}$	`b2trcl_luciani_csigx.dat`	(C.134), (C.136)
$\sigma_{CL\|\|}\frac{\sqrt{g}}{h_x^2}$	$\frac{1}{m\cdot S}$	`b2trcl_csigx.dat`	(C.138)
$\alpha_{ex}\frac{\sqrt{g}}{h_x^2}$	$\frac{m}{V\cdot s}$	`b2trcl_luciani_fllim_calfx.dat`	(C.139)
$\widetilde{\alpha}_{CLx}\frac{\sqrt{g}}{h_x^2}$	$\frac{m}{V\cdot s}$	`b2trcl_lucian_calfx.dat`	(C.142), (C.144)
$\alpha_{CLx}\frac{\sqrt{g}}{h_x^2}$	$\frac{m}{V\cdot s}$	`b2trcl_calfx.dat`	(C.140)
$\alpha^{(ST)}$	??	`b2trno_chcey_stoch.dat`	(C.152)
$-\frac{5}{2}\frac{\sqrt{g}}{h_x}n_eT_e^2\frac{B_z}{eh_y}\frac{\partial}{\partial y}\left(\frac{1}{B^2}\right)$	W	`b2tfhe__qediax.dat`	(C.115)
$\frac{5}{2}\frac{\sqrt{g}}{h_y}n_eT_e^2\frac{B_z}{eh_x}\frac{\partial}{\partial x}\left(\frac{1}{B^2}\right)$	W	`b2tfhe__qediay.dat`	(C.147)
$r_{Q_a}^{\mathrm{brm}}\sqrt{g}$	W	`b2stel_rqbrm[XXX]`	(C.110)
$r_{Q_a}^{\mathrm{line}}\sqrt{g}$	W	`b2stel_rqrad[XXX]`	(C.110)
residuals	W	`b2npht_reshe.dat` or `b2nph9_reshe.dat`	
Ion heat balance equation and related quantities			
T_i	W	`b2npht_shi.dat` or `b2nph9_shi.dat`	(C.154)
$S_i^{heat}\sqrt{g}$	W	`b2npht_shi.dat` or `b2nph9_shi.dat`	(C.1)
$\frac{3}{2}\frac{\partial n_iT_i}{\partial t}\sqrt{g}$	W	`b2npht_dnitidt.dat` or `b2nph9_dnitidt.dat`	(C.105)
$\frac{3}{2}n_i\frac{\partial T_i}{\partial t}\sqrt{g}$	W	`b2npht_nidtidt.dat` or `b2nph9_nidtidt.dat`	
$\frac{3}{2}T_i\frac{\partial n_i}{\partial t}\sqrt{g}$	W	`b2npht_tidnidt.dat` or `b2nph9_tidnidt.dat`	
$\frac{\sqrt{g}}{h_x}\widetilde{q}_{ix}$	W	`b2nph9_fhix.dat`	(C.163)
$\frac{\sqrt{g}}{h_y}\widetilde{q}_{iy}$	W	`b2nph9_fhiy.dat`	(C.184)
$\sum_{\substack{a=0\\ z_a\neq 0}}^{ns-1} n_aT_i\left(\nabla\cdot\vec{\mathbf{V}}^E\right)\sqrt{g}$	W	`b2sihs__shidd.dat`	(C.155)
$-\sum_{a=0}^{ns-1} n_aT_i\frac{\partial}{\partial x}\left(\frac{\sqrt{g}}{h_x}V_{a\|\|}b_x\right)\sqrt{g}$	W	`b2sihs__shidu.dat`	(C.154), (C.155)
$\sum_{fluid\ species}\eta_{ax}\left(\frac{\partial V_{a\|\|}}{h_x\partial x}\right)^2\sqrt{g}$	W	`b2sihs__shivc.dat`	(C.155)

continued on next page

continued from previous page

quantity	units	file(s)	equation
$\sum_{fluid\ species} \eta_a^{(AN)} \left(\frac{\partial V_{a\parallel}}{h_y \partial y}\right)^2 \sqrt{g}$	W	`b2sihs__shiva.dat`	(C.155)
$Q_{F_{ab}} \sqrt{g}$	W	`b2sihs__shifr.dat`	(C.157), (C.158)
$Q_I^{(i)} \sqrt{g}$	W	`b2stel_shi_ion.dat`	(C.159)
$Q_R^{(i)} \sqrt{g}$	W	`b2stel_shi_rec.dat`	(C.155), (C.161)
$Q_i^{\mathrm{EIRENE}} \sqrt{g}$	W	`b2stel_shi_eir.dat`	(C.155)
$-\frac{3}{2} \frac{\sqrt{g}}{h_x} \Gamma_{ix} T_i$	W	`b2tfhi__qi_32x.dat`	(C.166)
$-\frac{3}{2} \frac{\sqrt{g}}{h_y} \Gamma_{iy} T_i$	W	`b2tfhi__qi_32y.dat`	(C.166)
$\frac{\sqrt{g}}{h_x} q_{ix}^{P.Sch}$	W	`b2tfhi__fhiPSchx.dat`	(C.164)
$\frac{\sqrt{g}}{h_y} q_{iy}^{P.Sch}$	W	`b2tfhi__fhiPSchy.dat`	(C.185)
$-\frac{\sqrt{g}}{h_x} \kappa_{ix} \frac{1}{h_x} \frac{\partial T_i}{\partial x}$	W	`b2tfhi__qi_ki_gTx.dat`	(C.163)
$-\frac{\sqrt{g}}{h_y} \kappa_{iy} \frac{1}{h_y} \frac{\partial T_i}{\partial y}$	W	`b2tfhi__qi_ki_gTy.dat`	(C.184)
$\frac{\sqrt{g}}{h_x} h_{ix}^V$ (only for Section C.11)	W	`b2tfhi__hi_velx.dat`	(C.317)
$\chi_a^{(AN)}$	m^2/s	`b2tqna_hcib[XXX].dat`	(C.172), (C.173)
$\kappa_{ix} \frac{\sqrt{g}}{h_x^2}$	s^{-1}	`b2tfhi__chcix_Pat.dat`	(C.169)
$\kappa_{iy} \frac{\sqrt{g}}{h_y^2}$	s^{-1}	`b2tfhi__chciy_Pat.dat`	(C.188)
$\kappa_{ix}^{(AN)} \frac{\sqrt{g}}{h_x^2}$	s^{-1}	`b2trno_chcix.dat`	(C.169)
$\kappa_{iy}^{(AN)} \frac{\sqrt{g}}{h_y^2}$	s^{-1}	`b2trno_chciy.dat`	(C.169)
$\widetilde{\widetilde{\kappa}}_{ix}^{(CL)} \frac{\sqrt{g}}{h_x^2}$	s^{-1}	`b2trcl_luciani_fllim_chci.dat`	(C.174)
$\widetilde{\kappa}_{ix}^{(CL)} \frac{\sqrt{g}}{h_x^2}$	s^{-1}	`b2trcl_luciani_chcix.dat`	(C.177), (C.182)
$\kappa_{ix}^{(CL)} \frac{\sqrt{g}}{h_x^2}$	s^{-1}	`b2trcl_chcix.dat`	(C.181)
$\widetilde{\widetilde{\kappa}}_{ix}^{(CL)} \frac{\sqrt{g}}{h_x^2}$ (only for Section C.11)	s^{-1}	`b2tral_chci0x.dat`	(applied in C.168)
$\widetilde{\widetilde{\kappa}}_{ix}^{(CL)} \frac{\sqrt{g}}{h_x^2}$ (only for Section C.11)	s^{-1}	`b2tral_chci0_lucx.dat`	(C.290)
$\widetilde{\kappa}_{ix}^{(CL)} \frac{\sqrt{g}}{h_x^2}$ (only for Section C.11)	s^{-1}	`b2tral_chci0_flx.dat`	(C.297)
$\kappa_{ix}^{(CL)} \frac{\sqrt{g}}{h_x^2}$ (only for Section C.11)	s^{-1}	`b2tral_chci0_noflx.dat`	(C.298)
$\widetilde{\widetilde{\kappa}}_{ax}^{(CL)} \frac{\sqrt{g}}{h_x^2}$ (only for Section C.11)	s^{-1}	`b2tral_chci_a_lucx.dat`	(C.292)
$\widetilde{\kappa}_{ax}^{(CL)} \frac{\sqrt{g}}{h_x^2}$ (only for Section C.11)	s^{-1}	`b2tral_chci0_ax.dat`	(C.300)
$\frac{5}{2} \frac{\sqrt{g}}{h_x} T_i \sum_{\substack{a=0 \\ z_a \neq 0}}^{ns-1} n_a \widetilde{V}_{ax}^{(dia)}$	W	`b2tfhi__qidiax.dat`	(C.165)

continued on next page

continued from previous page

quantity	units	file(s)	equation
$\frac{5}{2}\frac{\sqrt{g}}{h_y}T_i\sum_{\substack{a=0\\z_a\neq 0}}^{ns-1} n_a\widetilde{V}_{ay}^{(dia)}$	W	`b2tfhi__qidiay.dat`	(C.186)
residuals	W	`b2npht_reshi.dat` or `b2nph9_reshi.dat`	
Current continuity equation and related quantities			
φ	V	`b2nppo_po.dat` or `b2npp7_po.dat`	(C.191)
$\frac{\sqrt{g}}{h_x}j_x$	A	`b2nppo_fchx.dat` or `b2npp7_fchx.dat`	(C.191), (C.192)
$\frac{\sqrt{g}}{h_y}j_y$	A	`b2nppo_fchy.dat` or `b2npp7_fchy.dat`	(C.191), (C.211)
$\frac{\sqrt{g}}{h_x}j_x^{(\parallel)}$	A	`b2tfhe__fch_px.dat`	(C.134), (C.192)
$\frac{\sqrt{g}}{h_x}\widetilde{j}_x^{(dia)}$	A	`b2tfhe__fchdiax.dat`	(C.195)
$\frac{\sqrt{g}}{h_y}\widetilde{j}_y^{(dia)}$	A	`b2tfhe__fchdiay.dat`	(C.214)
$\frac{\sqrt{g}}{h_x}\widetilde{j}_x^{(in)}$	A	`b2tfch__fchinertx.dat`	(C.196)
$\frac{\sqrt{g}}{h_x}\widetilde{j}_{ax}^{(in)}$	A	`b2tfch__fchinert_ax[XXX].dat`	(C.197)
$\frac{\sqrt{g}}{h_y}j_y^{(in)}$	A	`b2tfch__fchinerty.dat`	(C.215)
$\frac{\sqrt{g}}{h_y}j_{ay}^{(in)}$	A	`b2tfch__fchinert_ay[XXX].dat`	(C.216)
$\frac{\sqrt{g}}{h_x}\widetilde{j}_x^{(vis\parallel)}$	A	`b2tfch__fchvisparx.dat`	(C.198)
$\frac{\sqrt{g}}{h_x}\widetilde{j}_{ax}^{(vis\parallel)}$	A	`b2tfch__fchvispar_ax[XXX].dat`	(C.199)
$\frac{\sqrt{g}}{h_y}\widetilde{j}_y^{(vis\parallel)}$	A	`b2tfch__fchvispary.dat`	(C.217)
$\frac{\sqrt{g}}{h_y}\widetilde{j}_{ay}^{(vis\parallel)}$	A	`b2tfch__fchvispar_ay[XXX].dat`	(C.218)
$\frac{\sqrt{g}}{h_x}\widetilde{j}_x^{(s)}$	A	`b2tfhe__fchinx.dat`	(C.192), (C.204)
$\frac{\sqrt{g}}{h_y}\widetilde{j}_y^{(s)}$	A	`b2tfhe__fchiny.dat`	(C.211), (C.226)
$\frac{\sqrt{g}}{h_x}\widetilde{j}_{ax}^{(s)}$	A	`b2tfhe__fchin_ax.dat`	(C.210)
$\frac{\sqrt{g}}{h_y}\widetilde{j}_{ay}^{(s)}$	A	`b2tfhe__fchin_ay.dat`	(C.228)
$\sigma_{aa0}b_z^2\frac{\sqrt{g}}{h_x^2}$	S	`b2tral_csiginx01_00.dat`	(C.206)
$\frac{\sqrt{g}}{h_x}\widetilde{j}_x^{(AN)}$	A	`b2tfch__fchanmlx.dat`	(C.193)
$\frac{\sqrt{g}}{h_x}\widetilde{j}_{ax}^{(AN)}$	A	`b2tfch__fchanml_ax[XXX].dat`	(C.194)

continued on next page

continued from previous page			
quantity	units	file(s)	equation
$\frac{\sqrt{g}}{h_y}\widetilde{j}_y^{(AN)}$	A	`b2tfch__fchanmly.dat`	(C.212)
$\frac{\sqrt{g}}{h_y}\widetilde{j}_{ay}^{(AN)}$	A	`b2tfch__fchanml_ay[XXX].dat`	(C.213)
σ_{ANy}	S/m^2	`b2tqna_sig0.dat`	(C.232)
$\frac{\sqrt{g}}{h_x}\widetilde{j}_{ax}^{(vis\perp)}$	A	`b2tfch__fchvisper_ax[XXX].dat`	(C.201)
$\frac{\sqrt{g}}{h_y}\widetilde{j}_y^{(vis\perp)}$	A	`b2tfch__fchvispery.dat`	(C.219)
$\frac{\sqrt{g}}{h_y}\widetilde{j}_{ay}^{(vis\perp)}$	A	`b2tfch__fchvisper_ay[XXX].dat`	(C.220)
$\frac{\sqrt{g}}{h_x}\widetilde{j}_{ax}^{(visq)}$	A	`b2tfch__fchvisq_ax[XXX].dat`	(C.203)
$\frac{\sqrt{g}}{h_y}\widetilde{j}_y^{(visq)}$	A	`b2tfch__fchvisqy.dat`	(C.224)
$\frac{\sqrt{g}}{h_y}\widetilde{j}_{ay}^{(visq)}$	A	`b2tfch__fchvisq_ay[XXX].dat`	(C.225)
$\frac{\sqrt{g}}{h_y}j_y^{(ST)}$	A	`b2tfhe__fchstochy.dat`	(C.229)
$\frac{\sqrt{g}}{h_y^2}\sigma^{(ST)}$	S	`b2trno_csigy_stoch.dat`	(C.230)
residuals	A	`b2nppo_respo.dat` or `b2npp7_respo.dat`	(C.191)
Moments of the distribution function			
$u_\parallel$ (only for Section C.11)	m/s	`b2tral_avm_u.dat`	(C.475)
$w_{\parallel a}$ (only for Section C.11)	m/s	`b2tral_w_out[XXX].dat`	(C.475)
$\frac{h_{\parallel a}^T}{p_a}$ (only for Section C.11)	m/s	`b2tral_hTdp_out[XXX].dat`	(C.475)
$\frac{h_{\parallel a}^w}{p_a}$ (only for Section C.11)	m/s	`b2tral_hwdp_out[XXX].dat`	(C.475)
$\frac{r_{\parallel a}^T\gamma_a}{p_a}$ (only for Section C.11)	m/s	`b2tral_rTdpgamma_out[XXX].dat`	(C.475)
$\frac{r_{\parallel a}^w\gamma_a}{p_a}$ (only for Section C.11)	m/s	`b2tral_rwdpgamma_out[XXX].dat`	(C.475)

Table H.1: Files produced by means of `'b2wdat_iout = 4'`, the variable they contain, and the equation to which they are related

It may be convenient to build a total energy flux $\vec{\mathbf{q}}^{\mathrm{ENERGY}}$ (cf. Eq. (C.261)) and many other quantities which contribute to it from the quantities listed in the table H.1. Note that since the energy flux is computed at cell faces, it is necessary to compute it from the quantities also computed at cell faces.

$$q_x^{\mathrm{ENERGY}} = q_{ex}^{\mathrm{ENERGY}} + q_{ix}^{\mathrm{ENERGY}} \tag{H.1}$$

$$q_y^{\mathrm{ENERGY}} = q_{ey}^{\mathrm{ENERGY}} + q_{iy}^{\mathrm{ENERGY}} \tag{H.2}$$

$$q_{ex}^{\text{ENERGY}} = \widetilde{q}_{ex} + q_{ex}^{P.Sch} + q_{ex}^{dia} + \frac{2}{3}\left(q_{ex}^{\mathbf{E}\times\mathbf{B}} + q_{ex}^{u_\parallel} + q_{ex}^{uDPC}\right) \tag{H.3}$$

$$q_{ix}^{\text{ENERGY}} = \widetilde{q}_{ix} + q_{ix}^{P.Sch} + q_{ix}^{dia} + \frac{2}{3}\left(q_{ix}^{\mathbf{E}\times\mathbf{B}} + q_{ix}^{u_\parallel} + q_{ix}^{uDPC}\right) + q_{ix}^{j} + q_{ix}^{conv} \tag{H.4}$$

$$q_{ey}^{\text{ENERGY}} = \widetilde{q}_{ey} + q_{ey}^{P.Sch} + q_{ey}^{dia} + \frac{2}{3}q_{ey}^{\mathbf{E}\times\mathbf{B}} \tag{H.5}$$

$$q_{iy}^{\text{ENERGY}} = \widetilde{q}_{iy} + q_{iy}^{P.Sch} + q_{iy}^{dia} + \frac{2}{3}q_{iy}^{\mathbf{E}\times\mathbf{B}} + q_{iy}^{conv} + q_{iy}^{j} \tag{H.6}$$

$$\widetilde{q}_{ex}^{dia} = -\frac{5}{2}T_e^{(\text{face})}\sum_a \frac{z_a^2 T_e^{(\text{face})}}{T_i^{(\text{face})}}\tilde{V}_{ax}^{(dia)}n_a \tag{H.7}$$

$$\widetilde{q}_{ey}^{dia} = -\frac{5}{2}T_e^{(\text{face})}\sum_a \frac{z_a^2 T_e^{(\text{face})}}{T_i^{(\text{face})}}\tilde{V}_{ay}^{(dia)}n_a \tag{H.8}$$

$$\widetilde{q}_{ix}^{dia} = \frac{5}{2}T_i^{(\text{face})}\sum_a \tilde{V}_{ax}^{(dia)}n_a \tag{H.9}$$

$$\widetilde{q}_{iy}^{dia} = \frac{5}{2}T_i^{(\text{face})}\sum_a \tilde{V}_{ay}^{(dia)}n_a \tag{H.10}$$

Here and below the superscript "(face)" means that the temperature (or other quantity) should be computed at the cell face as a half-sum its neighboring cell-centered values. Since $\frac{\sqrt{g}}{h_x}\tilde{V}_{ax}^{(dia)}n_a$ and $\frac{\sqrt{g}}{h_y}\tilde{V}_{ay}^{(dia)}n_a$ are the particle fluxes through the cell faces, $\frac{\sqrt{g}}{h_x}$ and $\frac{\sqrt{g}}{h_y}$ are the cell face areas, and these quantities are computed at cell faces, we need to compute temperatures at cell faces too to get heat flux densities correctly computed at cell faces. A similar approach is applied below in all equations within this Appendix.

$$q_{ex}^{\mathbf{E}\times\mathbf{B}} = \frac{3}{2}T_e^{(\text{face})}\sum_a z_a n_a V_{ax}^E \tag{H.11}$$

$$q_{ey}^{\mathbf{E}\times\mathbf{B}} = \frac{3}{2}T_e^{(\text{face})}\sum_a z_a n_a V_{ay}^E \tag{H.12}$$

$$q_{ix}^{\mathbf{E}\times\mathbf{B}} = \frac{3}{2}T_i^{(\text{face})}\sum_a n_a V_{ax}^E \tag{H.13}$$

$$q_{iy}^{\mathbf{E}\times\mathbf{B}} = \frac{3}{2}T_i^{(\text{face})}\sum_a n_a V_{ay}^E \tag{H.14}$$

$$q_{ex}^{u_\parallel} = \frac{3}{2}T_e^{(\text{face})}\left(\sum_a z_a b_x n_a V_{\parallel a} - \frac{j_x^{(\parallel)}}{e}\right) \tag{H.15}$$

$$q_{ix}^{u_\parallel} = \frac{3}{2}T_i^{(\text{face})}\sum_a b_x n_a V_{\parallel a} \tag{H.16}$$

$$q_{ex}^{uDPC} = \frac{3}{2}T_e^{(\text{face})}\sum_a z_a n_a V_a^{\text{corr_dpc}} \tag{H.17}$$

$$q_{ix}^{uDPC} = \frac{3}{2}T_i^{(\text{face})}\sum_a n_a V_a^{\text{corr_dpc}} \tag{H.18}$$

$$q_{ix}^{conv} = \sum_a \frac{m_a \left(V_{\|a}^{(\mathrm{face})}\right)^2}{2} \left(\widetilde{\Gamma}_{ax} + \widetilde{\Gamma}_{ax}^{(PSch)} + \tilde{V}_{ax}^{(dia)} n_a\right) \tag{H.19}$$

$$q_{iy}^{conv} = \sum_a \frac{m_a \left(V_{\|a}^{(\mathrm{face})}\right)^2}{2} \left(\widetilde{\Gamma}_{ay} + \widetilde{\Gamma}_{ax}^{(PSch)} + \tilde{V}_{ax}^{(dia)} n_a\right) \tag{H.20}$$

$$q_{ix}^{j} = \frac{T_i^{(\mathrm{face})}}{e} \sum_a \left(j_x^{(AN)} + j_x^{(in)} + \widetilde{j}_x^{(vis\|)} + \widetilde{j}_x^{(vis\perp)} + \widetilde{j}_x^{(visq)}\right) \tag{H.21}$$

$$q_{iy}^{j} = \frac{T_i^{(\mathrm{face})}}{e} \sum_a \left(j_y^{(AN)} + j_y^{(in)} + \widetilde{j}_y^{(vis\|)} + \widetilde{j}_y^{(vis\perp)} + \widetilde{j}_y^{(visq)}\right) \tag{H.22}$$

Additionally one can consider a flux of energy in the form of potential energy of ionisation. This energy is spent by electrons at the place of ionisation and released back at the place of recombination (or is transferred to the targets thus contributing to the target heat loads). The associated energy flux is

$$q_x^{ioniz} = \sum_a \left(\widetilde{\Gamma}_{ax} + \widetilde{\Gamma}_{ax}^{(PSch)} + \tilde{V}_{ax}^{(dia)} n_a\right) E_a^{pot} \tag{H.23}$$

$$q_y^{ioniz} = \sum_a \left(\widetilde{\Gamma}_{ax} + \widetilde{\Gamma}_{ax}^{(PSch)} + \tilde{V}_{ax}^{(dia)} n_a\right) E_a^{pot} \tag{H.24}$$

with

$$E_a^{pot} = \sum_{i=1}^{z_a} I_a^{ioniz} \tag{H.25}$$

the sum of the ionisation potentials I_a^{ioniz} of all ions with a charge state below the considered ion (inclusive) within their isonuclear sequence.

Appendix I

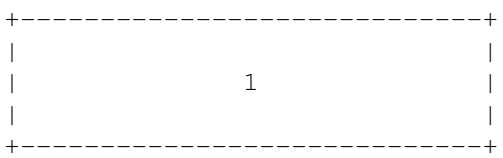
Region definitions

Three types of regions are defined in `b2agfs.F`: volume, x-directed and y-directed. The number of volume regions should also be stored in the `NREG` environment variable (or through the presence of an `NREG` ASCII file containing only the number of regions) as it is used for a variety of analysis scripts. Typical configurations for 1-D, linear, limiter, SN, DN (Connected and Disconnected), periodic stellarator island, and LFS snowflake cases are shown below:

I.1 Regions for 1-D and linear cases

For this geometry, NREG should be set to 1.

Volume regions:



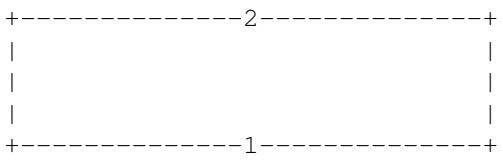

```
region 1 === plasma
```

X-directed regions:




```
region 1 === Western edge
region 2 === Eastern edge
```

Y-directed regions:




```
region 1 === core boundary
region 2 === SOL boundary
```

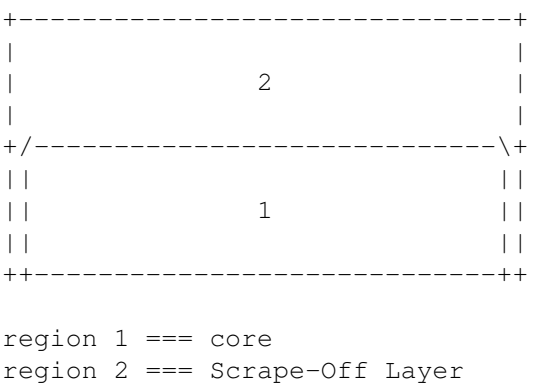
In cases where the problem of interest has only one target and a plasma entry surface, it is best if the target is located to the right of the domain and the plasma enters from the left.

An informative review of how the code geometry and physics model can be adapted to linear devices is given in [63].

I.2 Regions for limiter cases

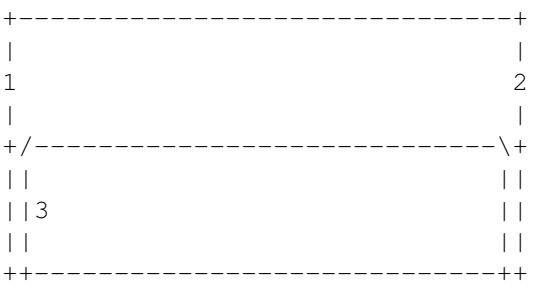
For this geometry, NREG should be set to 2.

Volume regions:



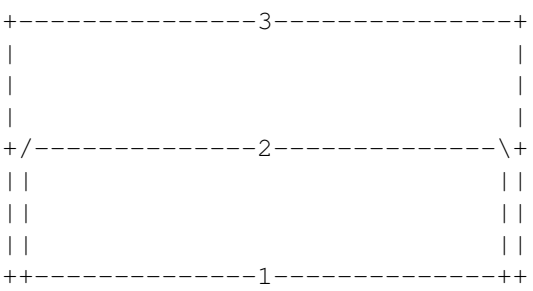

```
region 1 === core
region 2 === Scrape-Off Layer
```

X-directed regions:




```
region 1 === left target
region 2 === right target
region 3 === connection between both sides of core
```

Y-directed regions:



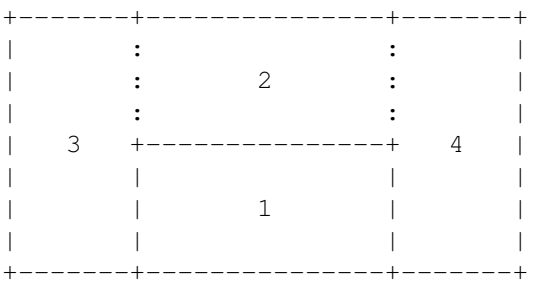

```
region 1 === core boundary
region 2 === separatrix
region 3 === SOL boundary
```

I.3 Regions for SN

For this geometry, NREG should be set to 4 (default).

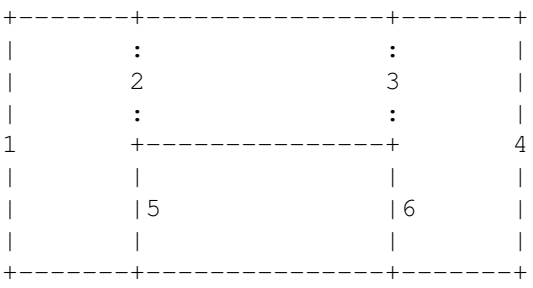
Volume regions:

See also section 3.5.4 for a specific example.



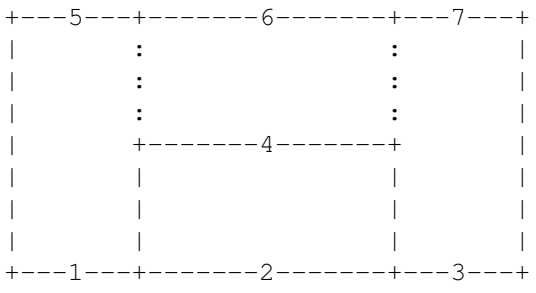

```
region 1 === core
region 2 === Scrape-Off Layer
region 3 === Western divertor (inboard for LSN, outboard for USN)
region 4 === Eastern divertor (outboard for LSN, inboard for USN)
```

X-directed regions:




```
region 1 === Western target (inboard for LSN, outboard for USN)
region 2 === entrance to Western divertor
region 3 === entrance to Eastern divertor
region 4 === Eastern target (outboard for LSN, inboard for USN)
region 5 === connection between both sides of core
region 6 === connection between both sides of divertor
```

Y-directed regions:

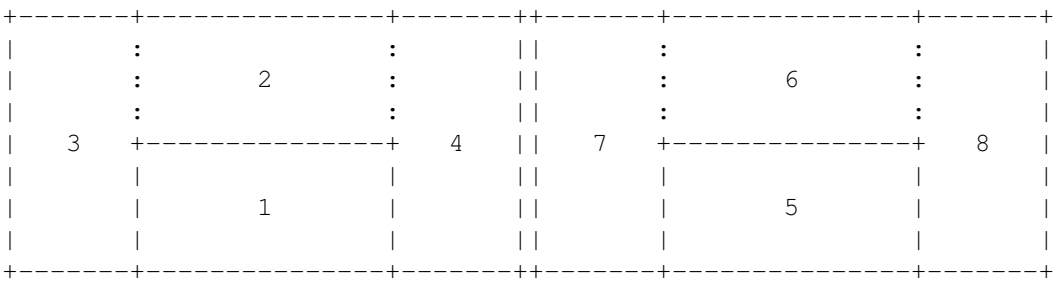



```
region 1 === Western divertor private flux wall boundary
                (inboard for LSN, outboard for USN)
region 2 === core boundary
region 3 === Eastern divertor private flux wall boundary
                (outboard for LSN, inboard for USN)
region 4 === separatrix
region 5 === Western divertor to main chamber wall boundary
                (inboard for LSN, outboard for USN)
region 6 === SOL to main chamber wall boundary
region 7 === Eastern divertor to main chamber wall boundary
                (outboard for LSN, inboard for USN)
```

I.4 Regions for CDN

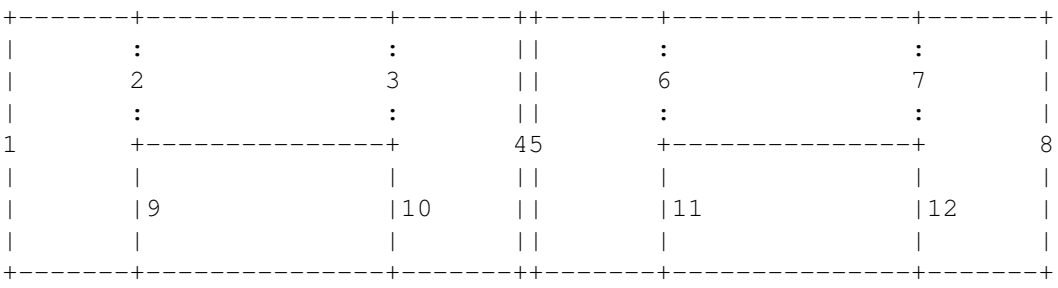
For this geometry, NREG should be set to 8.

Volume regions:



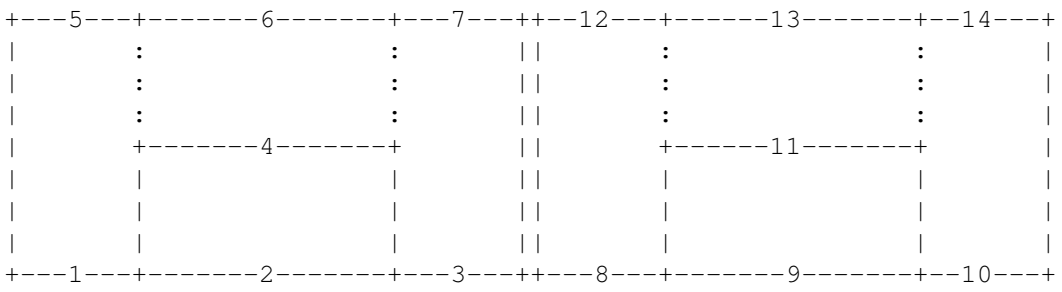

```
region 1 === left core
region 2 === left SOL
region 3 === left inboard divertor
region 4 === left outboard divertor
region 5 === right core
region 6 === right SOL
region 7 === right inboard divertor
region 8 === right outboard divertor
```

X-directed regions:




```
region 1 === left inboard divertor target
region 2 === entrance to left inboard divertor
region 3 === entrance to left outboard divertor
region 4 === left outboard target
region 5 === right inboard target
region 6 === entrance to right inboard divertor
region 7 === entrance to right outboard divertor
region 8 === right outboard target
region 9 === connection between left inboard and right outboard core
region 10 === connection between left outboard and
                                right inboard private flux region
region 11 === connection between right inboard and left outboard core
region 12 === connection between right outboard and
                                left inboard private flux region
```

Y-directed regions:




```
region 1 === left inboard divertor private flux wall boundary
region 2 === left core boundary
region 3 === left outboard divertor private flux wall boundary
region 4 === left separatrix
region 5 === left inboard divertor main chamber wall boundary
region 6 === left SOL main chamber wall boundary
region 7 === left outboard divertor main chamber wall boundary
region 8 === right inboard divertor private flux wall boundary
region 9 === right core boundary
region 10 === right outboard divertor private flux wall boundary
region 11 === right separatrix
region 12 === right inboard divertor main chamber wall boundary
region 13 === right SOL main chamber wall boundary
region 14 === right outboard divertor main chamber wall boundary
```

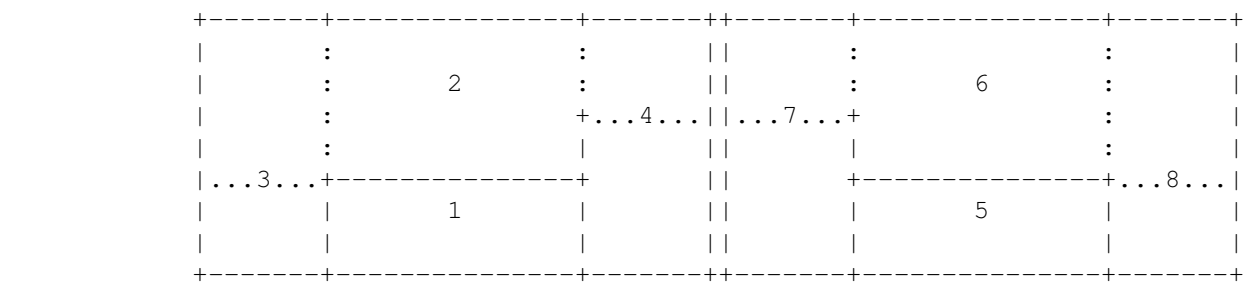
I.5 Regions for DDN

The case presented below corresponds to one where the active (or innermost) X-point is located at the bottom of the plasma (i.e. a DND case). For the opposite DNU configuration, the only change is the relative radial positions of the inner divertor boundaries between SOL-connected and PFR-connected halves and the position of the x-directed region 13 moves to being between x-directed regions 9 and 2 instead of being between x-directed regions 11 and 6.

For both cases, NREG should be set to 8.

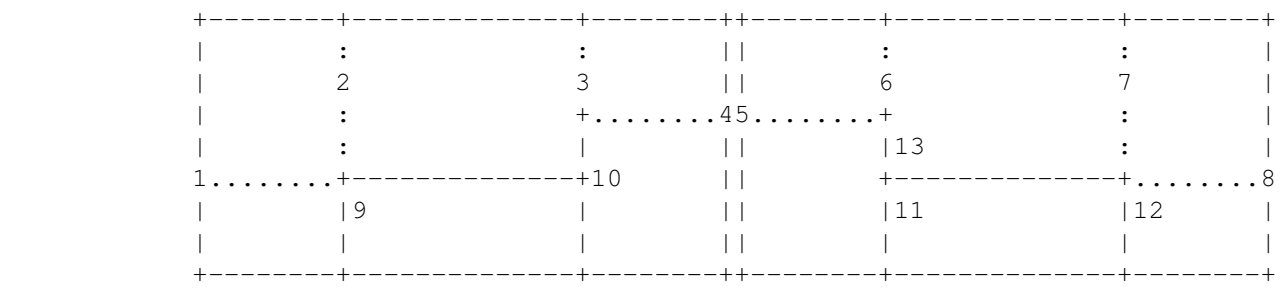
Volume regions:

See also section 3.5.4 for a specific example.



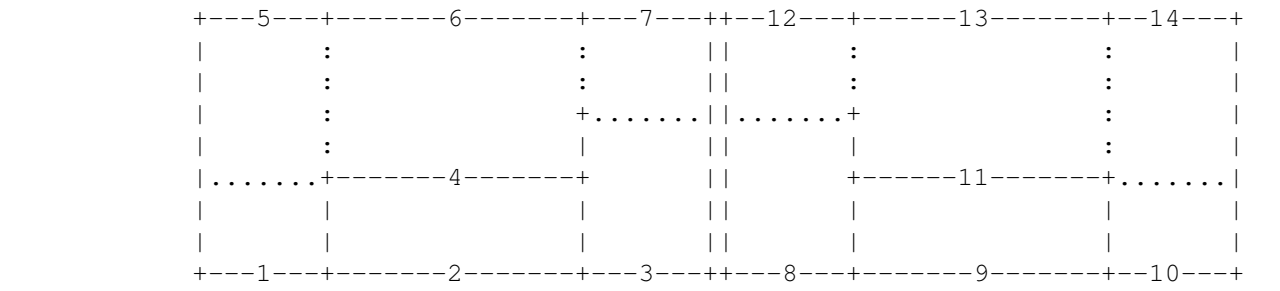

```
region 1 === left core
region 2 === left SOL
region 3 === left inboard divertor
region 4 === left outboard divertor
region 5 === right core
region 6 === right SOL
region 7 === right inboard divertor
region 8 === right outboard divertor
```

X-directed regions:




```
region 1 === left inboard divertor target
region 2 === entrance to left inboard divertor
region 3 === entrance to left outboard divertor
region 4 === left outboard target
region 5 === right inboard target
region 6 === entrance to right inboard divertor
region 7 === entrance to right outboard divertor
region 8 === right outboard target
region 9 === connection between left inboard and right outboard core
region 10 === connection between left outboard and
                                right inboard private flux region
region 11 === connection between right inboard and left outboard core
region 12 === connection between right outboard and
                                left inboard private flux region
region 13 === connection between right and left SOL
```

Y-directed regions:



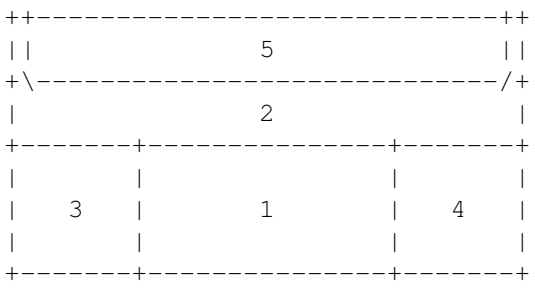

```
region 1 === left inboard divertor private flux wall boundary
region 2 === left core boundary
region 3 === left outboard divertor private flux wall boundary
region 4 === left separatrix
region 5 === left inboard divertor main chamber wall boundary
region 6 === left SOL main chamber wall boundary
region 7 === left outboard divertor main chamber wall boundary
region 8 === right inboard divertor private flux wall boundary
region 9 === right core boundary
region 10 === right outboard divertor private flux wall boundary
region 11 === right separatrix
region 12 === right inboard divertor main chamber wall boundary
region 13 === right SOL main chamber wall boundary
region 14 === right outboard divertor main chamber wall boundary
```

I.6 Regions for periodic BC stellarator island case

The reader is referred to [64] for a fuller description of this geometry. Note that the side edges of the core (volume region 1) are periodicity boundaries.

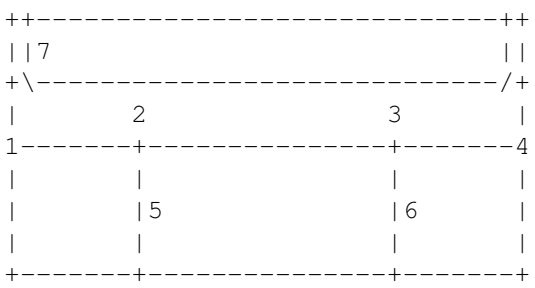
For this geometry, NREG should be set to 5.

Volume regions:



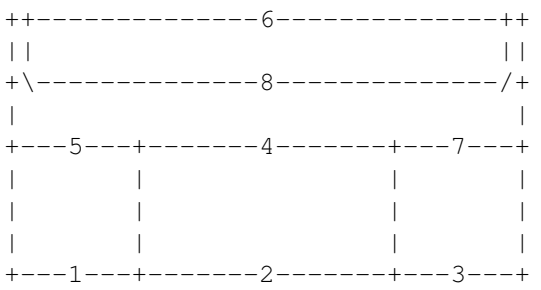

```
region 1 === core
region 2 === Scrape-Off Layer
region 3 === inboard divertor
region 4 === outboard divertor
region 5 === island
```

X-directed regions:




```
region 1 === inboard target
region 2 === entrance to inboard divertor
region 3 === entrance to outboard divertor
region 4 === outboard target
region 5 === connection between both sides of core
region 6 === connection between both sides of private flux region
region 7 === connection between both sides of island
```

Y-directed regions:



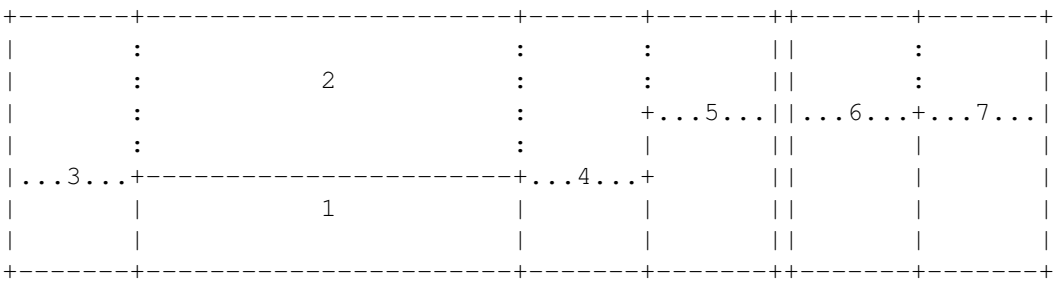

```
region 1 === inboard divertor private flux wall boundary
region 2 === core boundary
region 3 === outboard divertor private flux wall boundary
region 4 === separatrix
region 5 === entrance to inboard private flux region
region 6 === island center
region 7 === entrance to outboard private flux region
region 8 === island boundary
```

I.7 Regions for low field side snowflakes

The case presented below corresponds to one where the secondary X-point is located in the primary SOL (ie, a Snowflake-minus). For the SF-plus configuration, the only change is that the secondary X-point is located in the primary PFR. These region definitions also apply to X-point target geometries. For a fuller description, please refer to [65].

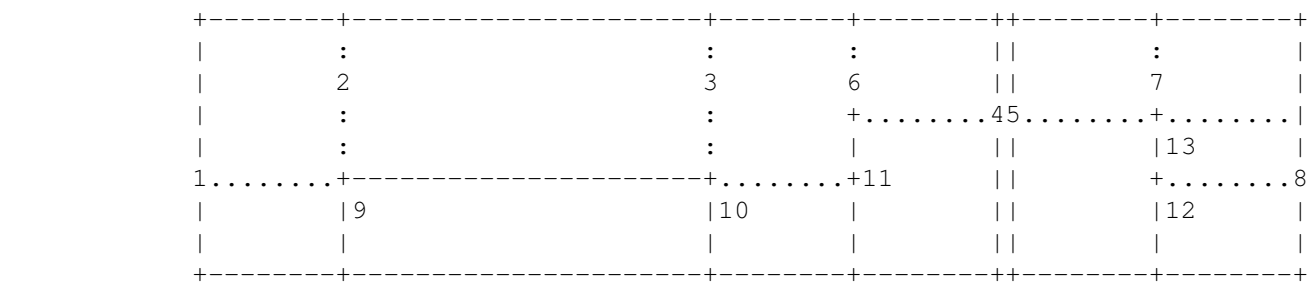
For both cases, `NREG` should be set to 7.

Volume regions:



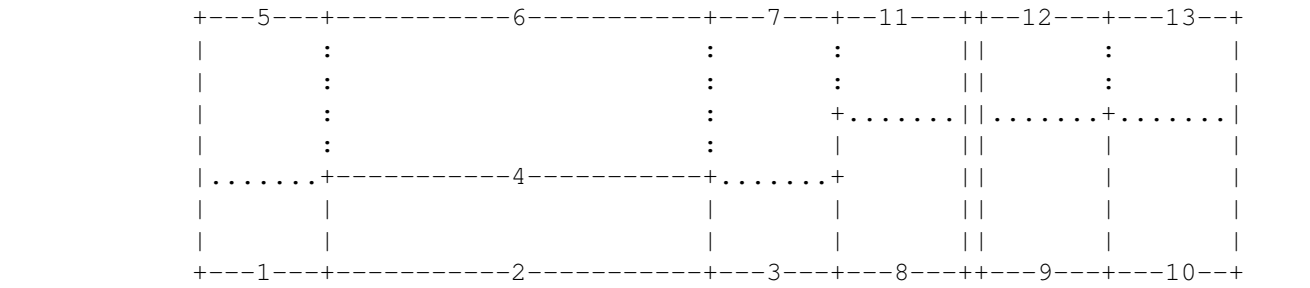

```
region 1 === core
region 2 === Scrape-Off Layer
region 3 === inboard divertor
region 4 === outboard divertor entrance
region 5 === outboard divertor leg 1
region 6 === outboard divertor leg 2
region 7 === outboard divertor leg 3
```

X-directed regions:




```
region 1 === inboard divertor target
region 2 === entrance to inboard divertor
region 3 === entrance to outboard divertor
region 4 === first outboard divertor target
region 5 === second outboard divertor target
region 6 === entrance to first outer divertor leg
region 7 === connection between 2nd and 3rd outer divertor legs
region 8 === third outboard divertor target
region 9 === connection between left and right core
region 10 === connection between left and right primary private flux region
region 11 === connection between 1st and 2nd outer divertor legs
region 12 === connection between primary PFR and third outer divertor leg
region 13 === connection between primary CFR and third outer divertor leg
```

Y-directed regions:




```
region 1 === inboard divertor private flux wall boundary
region 2 === core boundary
region 3 === outboard divertor entrance private flux wall boundary
region 4 === primary separatrix
region 5 === inboard divertor main chamber wall boundary
region 6 === SOL main chamber wall boundary
region 7 === outboard divertor entrance main chamber wall boundary
region 8 === first outboard divertor leg private flux wall boundary
region 9 === second outboard divertor leg private flux wall boundary
region 10 === third outboard divertor leg private flux wall boundary
region 11 === first outboard divertor leg main chamber wall boundary
region 12 === second outboard divertor leg main chamber wall boundary
region 13 === third outboard divertor leg main chamber wall boundary
```

Appendix J

b2plot manual

J.1 Introduction

b2plot is designed to take data from `B2`, `B2.5` or `B2.5-Eirene` and plot it under interactive control. It uses `NCAR` and so the user must have access to the `NCAR` environment. Some, but far from all, functionalities have been ported so that **b2plot** may also work in the `PLPLOT` environment. Preprocessing before compilation with `-DPLPLOT` option is then necessary (see section K.3 for more details). Data from the `B2SXDR`, and, if available, the `B2SDIAG` (for **SOLPS 4.x** runs), `b2fstate`, `b2fplasma`, `b2frates`, `b2fgmtry`, `b2mn.dat` (for **SOLPS 5.x** and **SOLPS-ITER** runs) and `fort.44` and `fort.46` (when running with `Eirene`) files are read in, and then commands are read in from `stdin`.

The basic idea in using **b2plot** is to specify the data quantity, some transformations, and then the plot command.

Note that b2plot deals with cell-face quantities by interpolating such quantities to cell-centres so that, within b2plot, all quantities are cell-centred quantities.

b2plot is stack based: seven different stacks are used. The most important stack is the "matrix" stack which is used to store quantities on the computational grid. These quantities can be $nx \times ny \times ns$ where nx and ny are the size of the grid, and ns ranges over the species. An "integer" stack is used to supply arguments for some of the commands. A "real" stack is also used to supply arguments for some commands and, importantly, for storing values to be used in doing arithmetic on the "matrix" stack. The fourth stack stores "string" values, and is also used for supplying data for some commands. The fifth stack is the "Eirene" stack, which contains quantities related to the neutrals, and computed on the `Eirene` triangular grid (only for **SOLPS-ITER** runs). The Eirene stack data is of size $ntrii \times ns$, where $ntrii$ is the number of triangles in the `Eirene` computational grid and ns ranges over the relevant species (atoms, molecules, or test ions). The sixth stack is a "wall" stack, which contains quantities that can be defined only on the edges of the domain. The wall data is of size $nbnd \times ns$, where $nbnd$ is the number of boundary cells. The seventh stack is the "target" stack, which contains data relevant to the target deposition and heat transfer models (only for **SOLPS 5.x** and **SOLPS-ITER** runs). The target data is of size $nbnd \times ndepth$, where $ndepth$ is the number of depth discretization points of the target elements in the direction normal to the vessel surface. As input lines are parsed, each input entity is identified as being a command, a real value, an integer value or a string value. If insufficient data is stored on the appropriate stacks for command execution, an error message will be returned and no other action will be taken. The real, integer or string values are placed on the "real", "integer" or "string" stacks, respectively. Integers and reals are distinguished by the latter having a decimal point, or having an `e` or `E` indicating exponential notation. The "integer", "real" and "string" stacks have a capacity of 1000 values each, and the strings can be up to 256 characters long. If a stack overflow occurs, an error is returned and no action is taken. The "matrix", "wall" and "target" stacks have a capacity of 10 data sets each. If a stack overflow occurs, the data set at the bottom of the stack is dropped, and the command can proceed. Stack overflows can be avoided by using the appropriate **drop** (see J.5.4) commands once a stack quantity is no longer needed.

J.2 Environment variables

B2PLOT_DEV is used to specify plotting to any combination of `x11`, `ps` or `cgm` (default is `ps`). Under `tcsh`, use

```
setenv B2PLOT_DEV 'x11 ps'
```

SOLPSTOP used to specify the top of the `SOLPS` tree

SOLPSTOP_FORCE used to override the value of `$SOLPSTOP` to force execution of code executables from a different tree (not defined by default)

Environment variables can also be used in the **extl** command (see J.6.1). If not explicitly set by the user, they inherit their value from the `setup.csh.*` files in the `${SOLPSTOP}` directory.

J.3 Commands

Commands are distinguished from strings by the latter being surrounded by single quote marks. Several types of commands are distinguished (sometimes somewhat arbitrarily): available data quantities (plasma, neutrals, wall, or target), available commands acting on stack quantities on or the plots, setting real plot parameters (like maxima and minima), string valued commands, and storing constants onto the stacks.

The available plasma data quantities are given in table J.1. These are quantities like the electron temperature, or the particle densities and typically store the results on the "matrix" stack. Some of these commands also take arguments when calculating derived data.

quantity	description $(units)$
ahal	Atomic H_α emissivity $(photons.m^{-3}.s^{-1})$
alf	Thermo-electric coefficient $(\Omega^{-1}.m^{-1})$
alph	Ratio of pressure gradient to critical ballooning value
b	Magnetic field components (T)
b2br	Bremsstrahlung losses (W)
b2eatcr	`Eirene` atomic species index
b2espcr	Nuclear sequence index
b2im	Momentum ionisation/recombination losses (s^{-1}) (**SOLPS 4.x** runs)
b2in	Particle ionisation/recombination losses (s^{-1}) (**SOLPS 4.x** runs)
b2io	Ionisation/recombination losses (W) (**SOLPS 4.x** runs)
b2ne	Electron energy neutral losses (W) (**SOLPS 4.x** runs)
b2ni	Ion energy neutral losses (W) (**SOLPS 4.x** runs)
b2nm	Momentum neutral losses (s^{-1}) (**SOLPS 4.x** runs)
b2nn	Particle neutral losses (s^{-1}) (**SOLPS 4.x** runs)
b2ra	Line radiation losses (W)
bb	Total magnetic field (T)
bc	Boundary condition number
bcch	Boundary charge source *b2stbc_sch* (A)
bche	Boundary electron heat source *b2stbc_she* (W)
bchi	Boundary ion heat source *b2stbc_shi* (W)
bcmo	Boundary parallel momentum source *b2stbc_smo* (N)
bcna	Boundary particle source *b2stbc_sna* $(particles.s^{-1})$
bix	Bottom neighbour *ix* coordinate
biy	Bottom neighbour *iy* coordinate
brch	Recycling charge source *b2stbr_sch* (A) (including any `Eirene` contributions)

continued on next page

continued from previous page

quantity	description $(units)$
brhe	Recycling electron heat source *b2stbr_she* (W) (including any `Eirene` contributions)
brhi	Recycling ion heat source *b2stbr_shi* (W) (including any `Eirene` contributions)
brmo	Recycling parallel momentum source *b2stbr_smo* (N) (including any `Eirene` contributions)
brna	Recycling particle source *b2stbr_sna* $(particles.s^{-1})$ (including any `Eirene` contributions)
bvis	Braams viscosity piece $(m.kg^{-1}.s^{-1})$
bx, bpol	Poloidal magnetic field (T)
by, brad	Radial magnetic field (T)
bz, btor	Toroidal magnetic field (T)
cd	Poloidal component of diamagnetic current $(m.s^{-1})$
ceqp	Equipartition term multiplier
chhichii	Chang-Hinton χ_i $(m^2.s^{-1})$
chineo	Neoclassical heat conductivity χ $(m^2.s^{-1})$
coni	Coronal equilibrium densities
conn	Connection length (m)
cora	Coronal equilibrium radiation rate (per isonuclear sequence) (W)
cpa	Pressure correction (Pa)
cpo	Electric potential correction (V)
cr	Radial coordinate (m)
crx	Corner radial coordinate (m)
cry	Corner vertical coordinate (m)
cte	Electron temperature correction (eV)
cti	Ion temperature correction (eV)
cua	Parallel velocity correction $(m.s^{-1})$
cut	Total momentum correction $(kg.m.s^{-1})$
cvsx	Parallel classical viscosity $(m.kg^{-1}.s^{-1})$
cvsy	Radial classical viscosity $(m.kg^{-1}.s^{-1})$
cvv	*CORVV* (**SOLPS 4.x** runs)
cz	Vertical coordinate (m)
d	Particle density diffusion coefficient $(m^2.s^{-1})$
dab2	Atomic density (m^{-3})
dd	Radial component of diamagnetic current $(m.s^{-1})$
deltab	Ratio of collision frequency to gyro-frequency
deltal	Ratio of Larmor radius to minimum gradient length
deltapar	Ratio of radial to parallel dynamics
deltaw	Ratio of banana width to minimum gradient length
deltawn	Ratio of banana width to density gradient length
deltawt	Ratio of banana width to temperature gradient length
dfhe	Divergence of the electron energy flux (W)
dfhi	Divergence of the ion energy flux (W)
dfhj	Divergence of the electrostatic energy flux (A)
dfhm	Divergence of the parallel kinetic energy flux (W)
dfhp	Divergence of the potential energy flux (W)
dfmo	Divergence of the momentum flux $(N.s)$
dfna	Divergence of the particle flux (s^{-1})
dhet, dfhet	Divergence of the electron total energy flux (W)
dhit, dfhit	Divergence of the electron total energy flux (W)

continued on next page

continued from previous page

quantity	description $(units)$
dib2	Test ion density (m^{-3})
dmb2	Molecular density (m^{-3})
dnneo	Neoclassical particle diffusivity D_n $(m^2.s^{-1})$
dp	Particle pressure diffusion coefficient $(m^2.s^{-1})$
dvua	Ion heat source *b2sihs_divua* (W)
dvue	Electron heat source *b2sihs_divue* (W)
dx	Cell poloidal length *dx* (m) (**SOLPS 4.x** runs)
dy	Cell radial width *dy* (m) (**SOLPS 4.x** runs)
ebal	Energy balance (W)
ebld	Energy balance (-divergence of fluxes) (W)
ebls	Energy balance (sources) (W)
edissml	Power to atoms due to molecule dissociation (W)
ehx	*ehx* (V)
ehy	*ehy* (V)
exba	$\mathbf{E} \times \mathbf{B}$ ion heating term *b2sihs_exba* (W)
exbe	$\mathbf{E} \times \mathbf{B}$ electron heating term *b2sihs_exbe* (W)
exch	External charge source *ext_sch* (A)
exhe	External electron heat source *ext_she* (W)
exhi	External ion heat source *ext_shi* (W)
exmo	External parallel momentum source *ext_smo* (N)
exna	External particle source *ext_sna* $(particles.s^{-1})$
fac_exb	$\mathbf{E} \times \mathbf{B}$ drift multiplier *fac_ExB*
fac_vis	Viscosity drift terms multiplier *fac_vis*
facdrift	Diamagnetic drift terms multiplier *facdrift*
fax_ext	External species poloidal flux (s^{-1})
fay_ext	External species radial flux (s^{-1})
fchx	Poloidal current (A)
fchx32	Convective poloidal current (A) (Does not include drift contributions)
fchx52	Conductive poloidal current (A) (Does not include drift contributions)
fchxn	Poloidal component of inertial current *fchin* (A)
fchxvspa	Poloidal component of parallel viscosity current *fchvispar* (A)
fchxvspe	Poloidal component of perpendicular viscosity current *fchvisper* (A)
fchp	Parallel current (A)
fchy	Radial current (A)
fchy32	Convective radial current (A) (Does not include drift contributions)
fchy52	Conductive radial current (A) (Does not include drift contributions)
fchyn	Radial component of inertial current *fchin* (A)
fchyvspa	Radial component of parallel viscosity current *fchvispar* (A)
fchyvspe	Radial component of perpendicular viscosity current *fchvisper* (A)
fetx	Total poloidal internal energy flux (W)
fety	Total radial internal energy flux (W)
ffbz	Toroidal magnetic flux function *ffbz* (Wb/m)
fhetx	Poloidal total electron energy flux (W)
fhety	Radial total electron energy flux (W)
fhex	Poloidal electron energy flux (W)
fhey	Radial electron energy flux (W)
fhitx	Poloidal total ion energy flux (W)
fhity	Radial total ion energy flux (W)
fhix	Poloidal ion energy flux (W)
fhiy	Radial ion energy flux (W)

continued on next page

continued from previous page

quantity	description $(units)$
fhjx	Poloidal electrostatic energy flux (W)
fhjy	Radial electrostatic energy flux (W)
fhmx	Poloidal parallel kinetic energy flux (W)
fhmy	Radial parallel kinetic energy flux (W)
fhpx	Poloidal potential energy flux (W)
fhpy	Radial potential energy flux (W)
fhtx	Poloidal total energy flux (W)
fhty	Peprendicular total energy flux (W)
flal	Thermo-electric classical flux limit factor
flalc	Thermo-electric coefficient limit factor
fle	Electron heat flux limit factor
fli	Ion heat flux limit factor
flke	Electron heat diffusivity classical flux limit factor
flki	Ion heat diffusivity classical flux limit factor
fluca	Thermo-electric Luciani flux limit factor
fluce	Electron heat diffusivity Luciani flux limit factor
fluci	Ion heat diffusivity Luciani flux limit factor
flucs	Electric conductivity Luciani flux limit factor
flucv	Viscosity Luciani flux limit factor
flvisc	Viscosity flux limit factor
fmox	Poloidal momentum flux (N)
fmoy	Radial momentum flux (N)
fmx	Poloidal mass flux $(kg.s^{-1})$
fmy	Radial mass flux $(kg.s^{-1})$
fnax	Poloidal particle flux (s^{-1})
fnax32	Convective poloidal particle flux (s^{-1})
fnax52	Conductive poloidal particle flux (s^{-1})
fnaxnodr	Poloidal particle flux without drifts (s^{-1})
fnay	Radial particle flux (s^{-1})
fnay32	Convective radial particle flux (s^{-1})
fnay52	Conductive radial particle flux (s^{-1})
fnaynodr	Radial particle flux without drifts (s^{-1})
fnex	Poloidal electron flux (s^{-1})
fnex32	Convective poloidal electron flux (s^{-1}) (Does not include drift contributions)
fnex52	Conductive poloidal electron flux (s^{-1}) (Does not include drift contributions)
fney	Radial electron flux (s^{-1})
fney32	Convective radial electron flux (s^{-1}) (Does not include drift contributions)
fney52	Conductive radial electron flux (s^{-1}) (Does not include drift contributions)
fnix32	Convective poloidal total ion flux (s^{-1}) (Does not include drift contributions)
fnix52	Conductive poloidal total ion flux (s^{-1}) (Does not include drift contributions)
fniy32	Convective radial total ion flux (s^{-1}) (Does not include drift contributions)
fniy52	Conductive radial total ion flux (s^{-1}) (Does not include drift contributions)
fraa	Friction heating term *b2sihs_fraa* (W)
gdcl	Guard cells
hx	Poloidal cell length (m)
hy	Radial cell width (m)
hy1	Shear-corrected radial cell width (m)
iix	Poloidal cell index *ix*
iiy	Radial cell index *iy*
ionrad	Power loss due to molecular ion radiation (W)

continued on next page

continued from previous page

quantity	description $(units)$
iseq	Species index *is* is equal to
isge	Species index *is* is greater than or equal to
isgt	Species index *is* is greater than
isle	Species index *is* is less than or equal to
islt	Species index *is* is less than
isne	Species index *is* is not equal to
ix	Poloidal cell index *ix* (normalized)
ixeq	Poloidal cell index *ix* is equal to
ixge	Poloidal cell index *ix* is greater than or equal to
ixgt	Poloidal cell index *ix* is greater than
ixle	Poloidal cell index *ix* is less than or equal to
ixlt	Poloidal cell index *ix* is less than
ixne	Poloidal cell index *ix* is not equal to
iy	Radial cell index *iy* (normalized)
iyeq	Radial cell index *iy* is equal to
iyge	Radial cell index *iy* is greater than or equal to
iygt	Radial cell index *iy* is greater than
iyle	Radial cell index *iy* is less than or equal to
iylt	Radial cell index *iy* is less than
iyne	Radial cell index *iy* is not equal to
joul	Joule heating term *b2sihs_joule* (W)
ke	Parallel kinetic energy (J)
kye	Electron heat diffusivity χ_e $(m^2.s^{-1})$
kyi	Ion heat diffusivity χ_i $(m^2.s^{-1})$
kyi0	Total ion heat diffusivity $(m^2.s^{-1})$
lix	Left neighbour poloidal index *leftix*
liy	Left neighbour radial index *leftiy*
lnlam	Coulomb logarithm
lnn	Density gradient length (m)
ltt	Temperature gradient length (m)
mach	Mach number
mach4	Mach number (**SOLPS 4.x** calculation)
mch2	Normalized parallel velocity
mhal	Molecular H_α emissivity $(photons.m^{-3}.s^{-1})$
mi	Ion mass (kg)
mi_ext	External species mass (kg)
molrad	Power loss due to molecule radiation (W)
na	Ion density (m^{-3}) (**SOLPS 5.x** and **SOLPS-ITER** runs)
na0	Ion density *na* at previous time step (m^{-3})
na_ext	External species density (m^{-3})
ne	Electron density (m^{-3})
ne2	ne2 (m^{-3})
ne_ext	Electron density associated with external species (m^{-3})
neutrad	Power loss due to atom radiation (W)
nflfhix	Neutral flux limits for poloidal ion energy flux
nflfhiy	Neutral flux limits for radial ion energy flux
nflfnax	Neutral flux limits for poloidal particle flux
nflfnay	Neutral flux limits for radial particle flux
nh	Total atomic hydrogen density (m^{-3})
ni	Total density (m^{-3}) (**SOLPS 5.x** and **SOLPS-ITER** runs)

continued on next page

continued from previous page

quantity	description $(units)$
ni	Ion density (m^{-3}) (**SOLPS 4.x** runs)
ni0	Density *ni* at previous time step (m^{-3})
niso	Isolating region number
nreg	Region number
nua	Total collision frequency (s^{-1})
nuab	Two species collision frequency (s^{-1})
nustar	Normalized collisionality
omega	Gyro-frequency $(rad.s^{-1})$
p*	Dynamic pressure (Pa)
pbal	Particle balance
pefa	Poloidal atomic energy flux $(W.m^{-2})$
pefm	Poloidal molecular energy flux $(W.m^{-2})$
pfla	Poloidal atomic flux $(s^{-1}.m^{-2})$
pflm	Poloidal molecular flux $(s^{-1}.m^{-2})$
phi	Plasma potential $\varphi(V)$ (**SOLPS 4.x** runs)
pit, pitch	Pitch
po	Electric potential $\varphi(V)$
po0	Electric potential *po* at previous time step (V)
pot	Potential energy (cumulative) (eV)
pot*	Ionisation energy [shifted down one] (eV)
poti	Ionisation energy (eV)
pr	Static pressure (Pa)
psi	Poloidal magnetic flux function Ψ (Wb)
pz	Plasma pressure (Pa)
q	Safety factor
qc	Cosine of the angle between flux line direction and left cell face
qz0	Sine of the angle away from orthogonality of the grid
qz1	Cosine of the angle away from orthogonality of the grid
rc	Recycling condition number
rco	Continuity residuals
rcxi	Ion energy neutral losses (*rcxhi*) (W)
rcxm	Ion momentum neutral losses (*rcxmo*) (N)
rcxn	Particle neutral losses (*rcxna*) (s^{-1})
rdi	*RESDI* (**SOLPS 4.x** runs)
rdi1	*RESDI1* (**SOLPS 4.x** runs)
ree	Electron energy equation residuals (W)
refa	Radial atomic energy flux $(W.m^{-2})$
refm	Radial molecular energy flux $(W.m^{-2})$
rei	Ion energy equation residuals (W)
rfla	Radial atomic flux $(s^{-1}.m^{-2})$
rflm	Radial molecular flux $(s^{-1}.m^{-2})$
rhol	Full-B Larmor radius (m)
rix	Right neighbour poloidal index *rightix*
riy	Right neighbour radial index *rightiy*
rlcl	Real cells
rmo	Momentum equation residuals $(kg.m.s^{-1})$
rmt	Total momentum equation residuals $(kg.m.s^{-1})$
ro	Mass density $(kg.m^{-3})$
rpo	Potential equation residuals (V)
rqae	Electron cooling rate (*rqahe*) (W)

continued on next page

continued from previous page

quantity	description $(units)$
rrai	Recombination ion energy losses (*rrahi*) (W)
rram	Recombination ion momentum losses (*rramo*) (N)
rran	Recombination ion particle losses (*rrana*) (s^{-1})
rsai	Ionisation ion energy losses (*rsahi*) (W)
rsam	Ionisation ion momentum losses (*rsamo*) (N)
rsan	Ionisation ion particle losses (*rsana*) (s^{-1})
rud	*RESUD* (**SOLPS 4.x** runs)
rudc	*RESUDC* (**SOLPS 4.x** runs)
rvd	*RESVD* (**SOLPS 4.x** runs)
s	Magnetic shear
sch	Charge sources (C)
she	Electron energy sources (W) (excluding equipartition)
shi	Ion energy sources (W) (excluding equipartition)
sig	Anomalous conductivity $(\Omega^{-1}.m^{-1})$
smaf	Parallel momentum source from friction *b2npmo_smaf* (N)
smag	Parallel momentum source from pressure gradient *b2npmo_smag* (N)
smav	Parallel momentum source from additional viscosity *b2npmo_smav* (N)
smfr	Friction force (N)
smo	Momentum sources (N)
smp	Self-pressure force (N) (**SOLPS 4.x** runs)
smpr	Thermal force (proportional to *exhp*, not valid when *b2sigp_style = 2*) (N)
smpt	Thermal force (proportional to T_i and T_e gradients, not valid when *b2sigp_style = 2*) (N)
smq	Total momentum sources (N)
smth	Thermal force (N)
sna	Particle sources $(particles.s^{-1})$
srcml	Molecule dissociation rate $(molecules.s^{-1})$
str	Heating from strange term *b2sihs_str* (W)
sx	Poloidal area of contact for cell (m^2)
sxperp	Poloidal area perpendicular to x-direction (m^2)
sy	Radial area of contact for cell (m^2)
sz	Poloidal cross-sectional area for cell (m^2)
tab2	Neutral atom temperature (eV)
ta_ext	External species temperature (eV)
te	Electron temperature (eV)
te0	Electron temperature *te* at previous time step (eV)
ti	Ion temperature (eV)
ti0	Ion temperature *ti* at previous time step (eV)
tib2	Test ion temperature (eV)
tix	Top neighbour poloidal index *topix*
tiy	Top neighbour radial index *topiy*
tmb2	Neutral molecule temperature (eV)
ua	Parallel velocity (*ua*) $(m.s^{-1})$ (**SOLPS 5.x** and **SOLPS-ITER** runs)
ua0	Parallel velocity *ua* at previous time step $(m.s^{-1})$
ua_ext	External species parallel velocity $(m.s^{-1})$
ud	Poloidal component of drift velocity $(m.s^{-1})$
ue	Parallel electron velocity (*ue*) $(m.s^{-1})$
up	Parallel velocity (*up*) $(m.s^{-1})$ (**SOLPS 4.x** runs)
up	Poloidal velocity (*up*) $(m.s^{-1})$ (**SOLPS 5.x** and **SOLPS-ITER** runs)
uu	Poloidal velocity (*uu*) $(m.s^{-1})$ (**SOLPS 4.x** runs)

continued on next page

continued from previous page	
quantity	description $(units)$
vadia.x	Poloidal component of divergent part of ion diamagnetic drift velocity $(m.s^{-1})$
vadia.y	Radial component of divergent part of ion diamagnetic drift velocity $(m.s^{-1})$
valfven	Alfvén velocity $v_{Alfvén}$ $(m.s^{-1})$
vcon	v_{con} $(m.s^{-1})$ (**SOLPS 4.x** runs)
vd	Radial component of drift velocity $(m.s^{-1})$
vedia.x	Poloidal component of divergent part of electron diamagnetic drift velocity $(m.s^{-1})$
vedia.y	Radial component of divergent part of electron diamagnetic drift velocity $(m.s^{-1})$
veexb.x	Poloidal component of electron $\mathbf{E} \times \mathbf{B}$ drift velocity $(m.s^{-1})$
veexb.y	Radial component of electron $\mathbf{E} \times \mathbf{B}$ drift velocity $(m.s^{-1})$
vexb.x	Poloidal component of ion $\mathbf{E} \times \mathbf{B}$ drift velocity $(m.s^{-1})$
vexb.y	Radial component of ion $\mathbf{E} \times \mathbf{B}$ drift velocity $(m.s^{-1})$
visa	Viscous heating term *b2sihs_visa* (W)
visc	Viscosity $(m.kg^{-1}.s^{-1})$
vlax	Poloidal pinch velocity $(m.s^{-1})$
vlay	Radial pinch velocity $(m.s^{-1})$
vol	Cell volume (m^3)
vnneo	Neoclassical particle pinch v_n $(m.s^{-1})$
vth	Thermal velocity v_{th} $(m.s^{-1})$
vthe	Electron thermal velocity $v_{th,e}$ $(m.s^{-1})$
vthi	Ion thermal velocity $v_{th,i}$ $(m.s^{-1})$
vv	Radial velocity (*vv*) $(m.s^{-1})$
vv0	Radial velocity *vv* at previous time step $(m.s^{-1})$
vwneo	Neoclassical heat pinch v_w $(m.s^{-1})$
wadia.x	Poloidal component of total diamagnetic drift velocity $(m.s^{-1})$
wadia.y	Radial component of total diamagnetic drift velocity $(m.s^{-1})$
wb	Width of the banana orbit (m)
wce	Electron cyclotron frequency ω_{ce} $(rad.s^{-1})$
wci	Ion cyclotron frequency ω_{ci} $(rad.s^{-1})$
wpe	Plasma frequency ω_{pe} $(rad.s^{-1})$
ww	Toroidal velocity (*ww*) $(m.s{-}1)$
xreg	x-directed flux region number
xymap	Index of wall segment *xymap*
yreg	y-directed flux region number
za_ext	External species charge (amu)
za2_ext	External species squared charge (amu)
zeff	Z effective (amu)
zi, za	Ion charge (amu)
zn	Nuclear charge (amu)
zn_ext	External species nuclear charge (amu)

Table J.1: Available quantities

Note that the **nreg**, **niso**, **xreg**, and **yreg** commands require an integer argument ⟨`ireg`⟩. If ⟨`ireg`⟩ = 0, the arrays will contain the region number of each cell. If ⟨`ireg`⟩ is greater than `0`, the cells in region with index ⟨`ireg`⟩ will contain `1.0`, and the rest will contain zero. If ⟨`ireg`⟩ is less than `0`, the cells in region with index −⟨`ireg`⟩ will contain zero, and the rest will contain `1.0`.

The source arrays **sna**, **smo**, **smq**, **she**, **shi**, and **sch** can take an optional integer argument ⟨`isel`⟩, varying from `0` to `3` (`0` to `1` for **sna**). Then the data stack contains the linearized source data according to the description found in Appendix A.3. In the absence of this integer argument, the data stack will contain the total source terms. The same

treatment applies to **smaf**, **smag**, and **smav**.

For the corner coordinates arrays **crx** and **cry**, an integer argument must be provided to determine the corner index according to the following:

0 Bottom left

1 Bottom right

2 Top left

3 Top right

where the topology is that of the computational domain. This means that for a cell with indices (ix, iy), the left face is shared with $(ix - 1, iy)$, the right face with $(ix + 1, iy)$, the top face with $(ix, iy + 1)$, and the bottom face with $(ix, iy - 1)$.

For the Coulomb logarithm **lnlam**, an optional integer argument can be provided to choose which formula is being used. The allowed values are:

0 Braginskii formula [default if the argument is not provided]

1 Wesson formula for ion-ion collisions (Eq. (C.81))

2 Wesson formula for electron-electron collisions (Eq. (C.108))

3 Wesson formula for electron-ion collisions (Eq. (C.132))

The arrays **ixlt**, **ixle**, **ixeq**, **ixne**, **ixgt**, **ixge**, **iylt**, **iyle**, **iyeq**, **iyne**, **iygt**, **iyge**, **islt**, **isle**, **iseq**, **isne**, **isgt**, and **isge** also all require an integer argument $\langle$`i`$\rangle$, with which the logical comparison is done. If the comparison test is true, the arrays will contain 1.0, otherwise they will contain zero.

Please note that the sign convention for the parallel ion (**ua**) and electron (**ue**) velocities depends on the sign of the poloidal magnetic field. The poloidal field is positive if it points if the direction of increasing poloidal index ix. The parallel velocities are positive if they point in the *same direction as the poloidal magnetic field*. Note further that this sign convention *does not apply* to the parallel current **fchp**, for which the standard sign convention applies, i.e. it is positive if pointing in the direction of increasing poloidal index ix.

A note about total energy fluxes. These are provided in two slightly different forms, denoted **fet[xy]** and **fht[xy]**. Both formulations contain the **fhe**, **fhi**, **fhm**, and **fhp** terms. If the code was solving internal energy equations (i.e. `b2news_BoRiS` set to `0.0` [default]), we add viscous correction terms and an extra factor of particle flux times temperature. For the **fht** formulation, we also add the electrostatic energy flux term **fhj**. This term is not included in **fet**, but, instead, **fet** will include any contributions from species provided by an additional external code (for example, `IMPGYRO`). These external species do not include the ones followed by `Eirene`, however.

The commands acting on the matrix stack are given in table J.2. These involve some manipulation or use of the data on the "matrix" stack. The stack requirements of each command are shown in the table for the integer ('i'), real ('r'), string ('s'), matrix ('m'), wall ('w') and target ('t') stacks. Whenever these stack requirements are not met, an error is returned and no action is taken.

command	stack requirements							section	description
	'i'	'r'	's'	'm'	'e'	'w'	't'		
abs	0	0	0	1	0	0	0	J.5.2	Absolute value
arox	0	0	0	1	0	0	0	J.5.8	Use as array for arrow X-component
aroy	0	0	0	1	0	0	0	J.5.8	Use as array for arrow Y-component
ave	1	0	0	0	0	0	0	J.5.10	Perform average
bala	0	0	0	1	0	0	0	J.5.11	Compute balance
chs	0	0	0	1	0	0	0	J.5.2	Change sign
dase	0	0	0	1	0	0	0	J.5.10	Set as *dab2* array
ddx	0	0	0	1	0	0	0	J.5.2	Differentiate with respect to x

continued on next page

continued from previous page

command	stack requirements							section	description
	'i'	'r'	's'	'm'	'e'	'w'	't'		
ddy	0	0	0	1	0	0	0	J.5.2	Differentiate with respect to y
drop	0	0	0	1	0	0	0	J.5.4	Drop array on top of stack
dup	0	0	0	1	0	0	0	J.5.4	Duplicate array on top of stack
eirc	0	0	0	0	0	0	0	J.5.10	Copy `Eirene` data onto `B2.5` data
f.lx	1	0	0	1	0	0	0	J.5.1	Plot against $\log(x)$
f.ly	1	0	0	1	0	0	0	J.5.1	Plot against $\log(y)$
f.x	1	0	0	1	0	0	0	J.5.1	Plot against x
f.y	1	0	0	1	0	0	0	J.5.1	Plot against y
fpsi	0	0	0	0	0	0	0	J.5.6	Plot data against poloidal magnetic flux (Ψ) value
intx	0	0	0	1	0	0	0	J.5.2	Integrate along x
inty	0	0	0	1	0	0	0	J.5.2	Integrate along y
line	1	0	0	0	0	0	0	J.5.1	Choose x- or y-directed graphs
list	0	0	0	1	0	0	0	J.5.4	List the contents of the matrix stack
ln	0	0	0	1	0	0	0	J.5.2	Take natural logarithm of array
log	0	0	0	1	0	0	0	J.5.2	Take log_{10} of array
m*	0	0	0	2	0	0	0	J.5.3	Matrix multiplication
m+	0	0	0	2	0	0	0	J.5.3	Matrix addition
m-	0	0	0	2	0	0	0	J.5.3	Matrix subtraction
m/	0	0	0	2	0	0	0	J.5.3	Matrix division
mdims	0	0	0	0	0	0	0	J.5.4	Prints the dimensions and contents of the matrix stack
meq	0	0	0	2	0	0	0	J.5.3	Matrix equal to
mge	0	0	0	2	0	0	0	J.5.3	Matrix greater than or equal to
mgt	0	0	0	2	0	0	0	J.5.3	Matrix greater than
mle	0	0	0	2	0	0	0	J.5.3	Matrix less than or equal to
mlt	0	0	0	2	0	0	0	J.5.3	Matrix less than
mmax	0	0	0	2	0	0	0	J.5.3	Maximum of two matrices
mmid	0	0	0	0	0	0	0	J.5.6	Map data to midplane
mmin	0	0	0	2	0	0	0	J.5.3	Minimum of two matrices
mne	0	0	0	2	0	0	0	J.5.3	Matrix not equal to
mpsi	0	0	0	0	0	0	0	J.5.6	Plot data against normalised poloidal magnetic flux (ψ) value
nese	0	0	0	0	0	0	0	J.5.10	Overwrite onto electron density array
nspe, nspec	3	0	0	0	0	0	0	J.5.10	New spectrometer
over	0	0	0	2	0	0	0	J.5.4	Copy over second top element of matrix stack
pne	0	0	0	1	0	0	0	J.5.2	Divide by electron density
pos	0	0	0	1	0	0	0	J.5.3	Is positive?
psx	0	0	0	1	0	0	0	J.5.2	Divide by poloidal contact area
psxperp	0	0	0	1	0	0	0	J.5.2	Divide by area perpendicular to flux tube
psy	0	0	0	1	0	0	0	J.5.2	Divide by radial contact area
pvol	0	0	0	1	0	0	0	J.5.2	Divide by cell volume
rigz	0	0	0	0	0	0	0	J.5.2	Toggles use of ignorable cells (**SOLPS 4.x** runs)
rm*	0	1	0	1	0	0	0	J.5.3	Matrix multiplication by real
rm**	0	1	0	1	0	0	0	J.5.3	Matrix taken to a real power
rm+	0	1	0	1	0	0	0	J.5.3	Real added to matrix

continued on next page

continued from previous page

command	stack requirements							section	description
	'i'	'r'	's'	'm'	'e'	'w'	't'		
rm-	0	1	0	1	0	0	0	J.5.3	Real substracted from matrix
rm/	0	1	0	1	0	0	0	J.5.3	Matrix divided by real
rmma, rmmax	0	1	0	1	0	0	0	J.5.3	Maximum between real and matrix
rmmi, rmmin	0	1	0	1	0	0	0	J.5.3	Minimum between real and matrix
rpsi	0	0	0	0	0	0	0	J.5.6	Plot data against ρ_{pol} $\left(= \sqrt{\psi}\right)$ value
spec	0	0	0	0	0	0	0	J.5.10	Old spectrometer
sqrt	0	0	0	1	0	0	0	J.5.2	Square root of matrix
stoe	0	0	0	1	0	0	0	J.5.4	Copy top of matrix stack to Eirene stack
stow	0	0	0	1	0	0	0	J.5.4	Copy top of matrix stack to wall stack
sumx	0	0	0	1	0	0	0	J.5.2	Sum over x
sumy	0	0	0	1	0	0	0	J.5.2	Sum over y
sumz	2	0	0	1	0	0	0	J.5.2	Sum over species
swap	0	0	0	2	0	0	0	J.5.4	Swap top two matrices on stack
t_#p	1	0	0	0	0	0	0	J.5.1	Number of trajectories
tese	0	0	0	1	0	0	0	J.5.10	Copy top of stack onto electron temperature
traj	7+*	0	0	1	0	0	0	J.5.1	Trajectories
uuse	0	0	0	1	0	0	0	J.5.1	Set X-component of trajectory flows
vvse	0	0	0	1	0	0	0	J.5.1	Set Y-component of trajectory flows
wtos	0	0	0	0	0	1	0	J.5.4	Copy wall stack array onto matrix stack
xsel	1	0	0	1	0	0	0	J.5.2	Select X value
xtow	1	0	0	0	0	0	0	J.5.4	Copy X-row of matrix onto wall stack
ysel	1	0	0	1	0	0	0	J.5.2	Select Y value
ytow	1	0	0	0	0	0	0	J.5.4	Copy Y-column of matrix onto wall stack
zrng	2	0	0	1	0	0	0	J.5.2	Select species range
zsel	1	0	0	1	0	0	0	J.5.2	Select species index

Table J.2: Commands acting on the "matrix" stack together with their stack requirements.

The commands acting on the "matrix" stack plotting are given in table J.3. These modify some behavior or appearance of the "matrix" stack plot. Their arguments requirements are indicated in the table.

command	stack requirements							section	description
	'i'	'r'	's'	'm'	'e'	'w'	't'		
a4l	0	0	0	0	0	0	0	J.5.5	A4 landscape
a4p	0	0	0	0	0	0	0	J.5.5	A4 portrait
acol	1	0	0	0	0	0	0	J.5.8	Arrow colour
arow	0	1	0	0	0	0	0	J.5.8	Set arrow length scale
arww	0	1	0	0	0	0	0	J.5.8	Set arrow line width
aspc	0	0	0	0	0	0	0	J.5.6	Toggle aspect ratio
bcol	1	0	0	0	0	0	0	J.5.8	Colour of label bar title
bold	0	0	0	0	0	0	0	J.5.7	Toggles bold graph frame
both	0	0	0	0	0	0	0	J.5.6	Computational and physical domain plots

continued on next page

continued from previous page

command	stack requirements							section	description
	'i'	'r'	's'	'm'	'e'	'w'	't'		
bw	0	0	0	0	0	0	0	J.5.7	Black-and-white plots only
clvd	0	3	0	0	0	0	0	J.5.9	Customize contour levels (delta)
clvl	0	0	0	0	0	0	0	J.5.9	Turn off contour levels
clvn	1	2	0	0	0	0	0	J.5.9	Customize contour levels (number)
clvp	2	0	0	1	0	0	0	J.5.9	Customize contour levels (proportion)
clvs	1	*	0	0	0	0	0	J.5.9	Customize contour levels (specified)
col	0	0	0	0	0	0	0	J.5.7	Colour plots
comp	0	0	0	0	0	0	0	J.5.6	Computational domain plots only
cont	0	0	0	1	0	0	0	J.5.1	Contour plot
dcol	1	0	0	0	0	0	0	J.5.8	Colour for matrix data line plots
eclv	0	0	0	0	0	0	0	J.5.9	Empty contour levels
ecol	1	0	0	0	0	0	0	J.5.8	Experimental data colour
expl	0	0	0	0	0	0	0	J.5.8	Toggle line or scatter experimental data plot
exps	1	0	0	0	0	0	0	J.5.8	Symbol for experimental scatter plot
ewid	0	1	0	0	0	0	0	J.5.8	Experimental data line width
fclv	0	0	0	0	0	0	0	J.5.9	Fill contour levels
fcol	1	0	0	0	0	0	0	J.5.8	Color for graph frame, axes, ticks and labels
filo	1	0	0	0	0	0	0	J.5.9	Fill option
font	1	0	0	0	0	0	0	J.5.7	Set font
gcol	1	0	0	0	0	0	0	J.5.8	Colour of graph title
glob	0	0	0	0	0	0	0	J.5.7	Copy *extralabel* to the global label field
glsz	0	1	0	0	0	0	0	J.5.7	Global header font size
grid	0	0	0	0	0	0	0	J.5.8	Add grid to plot
hcol	1	0	0	0	0	0	0	J.5.8	Colour of global header
i*	2	0	0	0	0	0	0	J.5.3	Integer multiplication
i**	2	0	0	0	0	0	0	J.5.3	Integer to the power
i+	2	0	0	0	0	0	0	J.5.3	Integer addition
i-	2	0	0	0	0	0	0	J.5.3	Integer substraction
i/	2	0	0	0	0	0	0	J.5.3	Integer division
idrop	1	0	0	0	0	0	0	J.5.4	Drop top element from the integer stack
idup	1	0	0	0	0	0	0	J.5.4	Duplicate top element from the integer stack
imp	1	0	0	0	0	0	0	J.5.10	Set jxi (inner midplane coordinate)
iover	2	0	0	0	0	0	0	J.5.4	Copy over second element from the integer stack to top
iprt	1	0	0	0	0	0	0	J.5.4	Outputs the integer stack
iswap	2	0	0	0	0	0	0	J.5.4	Swap top two elements from the integer stack
itoe	1	0	0	0	0	0	0	J.5.4	Copy integer onto Eirene stack
itom	1	0	0	0	0	0	0	J.5.4	Copy integer onto matrix stack
itor	1	0	0	0	0	0	0	J.5.4	Copy integer onto real stack
itostr	1	0	0	0	0	0	0	J.5.4	Write integer onto string stack
itot	1	0	0	0	0	0	0	J.5.4	Copy integer onto target stack
itow	1	0	0	0	0	0	0	J.5.4	Copy integer onto wall stack
lacl	0	0	0	0	0	0	0	J.5.9	Labels along contour plots
lbof	0	0	0	0	0	0	0	J.5.9	Labels off
lbon	0	0	0	0	0	0	0	J.5.9	Labels on

continued on next page

continued from previous page

command	stack requirements							section	description
	'i'	'r'	's'	'm'	'e'	'w'	't'		
lbsa	0	0	0	0	0	0	0	J.5.9	Toggle label lines
lbsz	0	1	0	0	0	0	0	J.5.7	Label size
lfsz	1	0	0	0	0	0	0	J.5.7	Line label font size
lhor	0	0	0	0	0	0	0	J.5.9	Horizontal labels
linf	0	0	0	0	0	0	0	J.5.6	Linear functions plot
lint	0	0	0	0	0	0	0	J.5.1	Linear time for trajectories plot
linx	0	0	0	0	0	0	0	J.5.6	Linear X scale
logf	0	0	0	0	0	0	0	J.5.6	Logarithmic functions plot
logt	0	0	0	0	0	0	0	J.5.1	Logarithmic time for trajectories plot
logx	0	0	0	0	0	0	0	J.5.6	Logarithmic X scale
lwid	0	1	0	0	0	0	0	J.5.7	Line width
mcol	1	0	0	0	0	0	0	J.5.8	Mesh (and triangular grid) colour
mesh	0	0	0	0	0	0	0	J.5.1	Plot mesh
mvsk	1	0	0	0	0	0	0	J.5.10	Movie skip
ndec	1	0	0	0	0	0	0	J.5.6	Number of logarithmic decades
new	0	0	0	0	0	0	0	J.5.7	New page
neo_ic	1	0	0	0	0	0	0	J.5.10	Option for neoclassical calculation
nice	0	0	0	0	0	0	0	J.5.7	Toggle whether nice axes will be used
noax	0	0	0	0	0	0	0	J.5.7	No axes
nocp	0	0	0	0	0	0	0	J.5.9	No contour plot
ocol	1	0	0	0	0	0	0	J.5.8	Outline colour
omp	1	0	0	0	0	0	0	J.5.10	Set jxa (outer midplane coordinate)
outl	0	0	0	0	0	0	0	J.5.8	Plot outline
owid	0	1	0	0	0	0	0	J.5.8	Outline line width
page	2	0	0	0	0	0	0	J.5.5	Number of graphs per page
part	0	1	0	0	0	0	0	J.5.11	Partition colour map
phys	0	0	0	0	0	0	0	J.5.6	Physical domain plots only
r*	0	2	0	0	0	0	0	J.5.3	Real multiplication
r**	0	2	0	0	0	0	0	J.5.3	Real to the power
r+	0	2	0	0	0	0	0	J.5.3	Real addition
r-	0	2	0	0	0	0	0	J.5.3	Real substraction
r/	0	2	0	0	0	0	0	J.5.3	Real division
rdrop	0	1	0	0	0	0	0	J.5.4	Drop top element from the real stack
rdup	0	1	0	0	0	0	0	J.5.4	Duplicate top element from the real stack
reso	0	0	0	0	0	0	0	J.5.10	Reset output
rigz	0	0	0	0	0	0	0	J.5.2	Toggle use of ignore regions
rover	0	2	0	0	0	0	0	J.5.4	Copy over second element from the real stack to top
rprt	0	1	0	0	0	0	0	J.5.4	Outputs the real stack
rswap	0	2	0	0	0	0	0	J.5.4	Swap top two elements from the real stack
rtom	0	1	0	0	0	0	0	J.5.4	Copy real onto Eirene stack
rtom	0	1	0	0	0	0	0	J.5.4	Copy real onto matrix stack
rtostr	0	1	0	0	0	0	0	J.5.4	Write real onto string stack
rtot	0	1	0	0	0	0	0	J.5.4	Copy real onto target stack
rtow	0	1	0	0	0	0	0	J.5.4	Copy real onto wall stack
same	0	0	0	0	0	0	0	J.5.7	Next graph on same axes
scol	1	0	0	0	0	0	0	J.5.8	Separatrix colour
sep	0	0	0	0	0	0	0	J.5.8	Plot separatrices

continued on next page

continued from previous page

command	stack requirements							section	description
	'i'	'r'	's'	'm'	'e'	'w'	't'		
sepw	0	1	0	0	0	0	0	J.5.8	Separatrix line width
shco	0	0	0	0	0	0	0	J.5.11	
spln	0	0	0	1	0	0	0	J.5.9	Spline plot
sq	0	0	0	0	0	0	0	J.5.5	Square aspect ratio plot
stdl	0	0	0	0	0	0	0	J.5.7	Set standard label as *extralabel*
surf	0	0	0	1	0	0	0	J.5.1	Surface plot
swid	1	0	0	0	0	0	0	J.5.10	Set `WINID`
t_r+	0	0	0	0	0	0	0	J.5.11	Set ratio to `.true.`
t_r-	0	0	0	0	0	0	0	J.5.11	Set ratio to `.false.`
target	0	0	0	0	0	0	1	J.5.8	Add target data to plot
tcol	1	0	0	0	0	0	0	J.5.8	Colour for target data line plots
trgoff	0	0	0	0	0	0	0	J.5.8	Turn off target data in plot
trig	0	0	0	0	0	0	0	J.5.8	Add triangular grid to plot
trigmesh	0	0	0	0	0	0	0	J.5.1	Plot triangular grid
vcol	1	0	0	0	0	0	0	J.5.8	Vessel colour
vesl	0	0	0	0	0	0	0	J.5.8	Plot vessel
vwid	0	1	0	0	0	0	0	J.5.8	Vessel line width
wall	0	0	0	0	0	1	0	J.5.8	Add wall data to plot
wcol	1	0	0	0	0	0	0	J.5.8	Colour for wall data line plots
writ , write	0	0	0	0	0	0	0	J.5.10	Write output file
xcol	1	0	0	0	0	0	0	J.5.8	Colour of extra label
xpar	0	0	0	0	0	0	0	J.5.6	Set X coordinate to parallel length of fieldline
xpol	0	0	0	0	0	0	0	J.5.6	Set X coordinate to poloidal length
xyrz	0	0	0	0	0	0	0	J.5.10	Output plot ranges and boundaries
yref	1	0	0	0	0	0	0	J.5.10	Set $yref$ (radial reference coordinate)
ytra	1	0	0	0	0	0	0	J.5.8	Transform Y coordinate for comparison to experiment
zcol	1	0	0	0	0	0	0	J.5.8	Colour of zero axes
zero	0	0	0	0	0	0	0	J.5.8	Add zero axes to plot
zwid	0	1	0	0	0	0	0	J.5.8	Zero axes line width

Table J.3: Commands acting on the "matrix" stack plotting together with their stack requirements.

The maxima and minima for plotting can be set by changing the real variables listed in table J.4. These commands take one value from the "real" stack and store it in an associated global variable.

name	use $(units)$
a0	R radius of the ellipse in `chord.F` (m)
b0	R radius of the ellipse in `chord.F` (m)
emax	Maximum function value for plotting Eirene stack data
emin	Minimum function value for plotting Eirene stack data
fmax	Maximum function value for plotting matrix data
fmin	Minimum function value for plotting matrix data
omax	Maximum ordinate for 2-D plots
omin	Minimum ordinate for 2-D plots

continued on next page

continued from previous page	
name	use $(units)$
r0	R position of the ellipse in `chord.F` (m)
rmax	Maximum R for plotting in the physical domain (m)
rmin	Minimum R for plotting in the physical domain (m)
rshi	Relative R shift to be applied to chords (m)
smax	Maximum ordinate for wall plots
smin	Minimum ordinate for wall plots
t_pt	Total flight time of particles in `caltraj` (s)
trgfmax	Maximum function value for plotting in the target domain
trgfmin	Minimum function value plotting in the target domain
trgomax	Maximum ordinate for target domain 2-D plots
trgomin	Minimum ordinate for target domain 2-D plots
trgrmax	Maximum R for plotting in the target domain (m)
trgrmin	Minimum R for plotting in the target domain (m)
trgxmax	Maximum x for plotting in the target domain
trgxmin	Minimum x for plotting in the target domain
trgymax	Maximum y for plotting in the target domain
trgymin	Minimum y for plotting in the target domain
trgzmax	Maximum Z for plotting in the target domain (m)
trgzmin	Minimum Z for plotting in the target domain (m)
tt0	Function value inside the ellipse in `chord.F`
wmax	Maximum function value for plotting in wall domain
wmin	Minimum function value for plotting in wall domain
xmax	Maximum x for plotting in the computational domain
xmin	Minimum x for plotting in the computational domain
xref	Reference position in x: 0.0=left, 0.5=mid, 1.0=right
ymax	Maximum y for plotting in the computational domain
ymin	Minimum y for plotting in the computational domain
z0	Z position of the ellipse in `chord.F` (m)
zmax	Maximum Z for plotting in the physical domain (m)
zmin	Minimum Z for plotting in the physical domain (m)
zshi	Relative Z shift to be applied to chords (m)

Table J.4: Commands assigning real variables from stack for use in plotting

Values for the text variables mentioned in tables J.5 can also be set. These text variables often form the argument for a command. Typically these commands take one value off the "string" stack.

name	stack requirements							section	use
	'i'	'r'	's'	'm'	'e'	'w'	't'		
adas	0	0	1	0	0	0	0	J.6.2	Access `ADAS` data
amju , amjuel	0	0	4	0	0	0	0	J.6.2	Access `AMJUEL` data
cam	0	0	1	0	0	0	0	J.6.2	Camera plot
chbx	0	0	1	0	0	0	0	J.6.2	Boxcar average of chords
chmp	0	0	1	0	0	0	0	J.6.2	Chord contributions to the mesh element
chor	0	0	1	0	0	0	0	J.6.2	Filename containing chords (for integration)
chpl	0	0	1	0	0	0	0	J.6.2	Filename containing chords (to plot)
continued on next page									

continued from previous page

name	stack requirements							section	use
	'i'	'r'	's'	'm'	'e'	'w'	't'		
chvl	0	0	1	0	0	0	0	J.6.2	Value at positions
cltb	0	0	1	0	0	0	0	J.6.2	Filename for a colour table
cmd	0	0	1	0	0	0	0	J.6.2	Filename from which to take commands
cmdcdf	0	0	1	0	0	0	0	J.6.2	Loop command file on `CDF` data
ecam	0	0	1	0	1	0	0	J.6.2	Camera plot for Eirene stack data
echbx	0	0	1	0	1	0	0	J.6.2	Boxcar average of chords for Eirene stack data
echmp	0	0	1	0	1	0	0	J.6.2	Chord contributions to the mesh element from Eirene stack data
echor	0	0	1	0	1	0	0	J.6.2	Filename containing chords for integration of Eirene stack data
echvl	0	0	1	0	1	0	0	J.6.2	Value at positions of Eirene stack data
expf	0	0	1	0	0	0	0	J.5.8	Experimental data file
extl	0	0	1	0	0	0	0	J.6.1	Extra label
flux	0	0	1	0	0	0	0	J.6.2	Call `WOSlib` flux routine
glbl	0	0	1	1	0	0	0	J.5.7	Set graph label
movi	0	0	1	0	0	0	0	J.6.2	Movie
movie, nmovie	2	0	1	0	0	0	0	J.6.2	New movie option
mprt	0	0	1	1	0	0	0	J.5.10	Output top of matrix stack to a file
mtab	4	0	1	1	0	0	0	J.5.10	Output part of the matrix on top of the stack to a file
omovie	2	0	2	0	0	0	0	J.6.2	Old movie option
oson	0	0	1	0	0	0	0	J.6.2	Output geometry in `Sonnet` format
outp	2	0	1	1	0	0	0	J.5.10	Prints value of specific element
pdev	0	0	1	0	0	0	0	J.5.10	`PLPLOT` device name
saw	2	0	1	0	0	0	0	J.5.4	Read external data onto the matrix stack
saws	3	0	1	0	0	0	0	J.5.4	Read external fluid species data onto the matrix stack
sawx	3	0	1	0	0	0	0	J.5.4	Read external additional species data onto the matrix stack
scat	0	0	2	0	0	0	0	J.5.4	Concatenate two strings
sdrop	0	0	1	0	0	0	0	J.5.4	Drop top of string stack
sdup	0	0	1	0	0	0	0	J.5.4	Duplicate top of string stack
slbl	1	0	1	1	0	0	0	J.5.7	Set species label
sonn	0	0	1	0	0	0	0	J.6.2	Read a new `Sonnet` geometry
sover	0	0	2	0	0	0	0	J.5.4	Copy over the second element of the string stack
sprt	0	0	1	0	0	0	0	J.5.4	Outputs the string stack
sswap	0	0	2	0	0	0	0	J.5.4	Swap the top two elements of the string stack
tlbl	0	0	1	0	0	0	1	J.5.7	Set target graph label
trgblbl	0	0	1	0	0	0	0	J.5.7	Set target bar label
trgprt	0	0	1	0	0	0	1	J.5.10	Output top of target stack to a file
trgtab	4	0	1	0	0	0	1	J.5.10	Output part of the top of the target stack to a file
uedgegrd	0	1	1	0	0	0	0	J.6.2	Output geometry in `UEDGE` format
wallblbl	0	0	1	0	0	0	0	J.5.7	Set wall bar label
wlbl	0	0	1	0	0	1	0	J.5.7	Set wall graph label

continued on next page

continued from previous page

name	stack requirements							section	use
	'i'	'r'	's'	'm'	'e'	'w'	't'		
wprt	0	0	1	0	0	1	0	J.5.10	Output top of wall stack to a file
wslbl	1	0	1	0	0	1	0	J.5.7	Set wall species label
wtab	2	0	1	0	0	1	0	J.5.10	Output part of the top of the wall stack to a file
xlab	0	0	1	0	0	0	0	J.6.1	Set x-axis label
ylab	0	0	1	0	0	0	0	J.6.1	Set y-axis label

Table J.5: Commands taking character strings as parameters. Note that the string should be delimited by single quotes.

In table J.6, we give the list of quantities that are available for use on the "Eirene" stack, containing values defined on the triangular grid defined for the `Eirene` computations. These quantities are only available for coupled **SOLPS-ITER** runs.

name	description $(units)$
edena	Atoms energy density $(eV.m^{-3})$
edeni	Test ions energy density $(eV.m^{-3})$
edenm	Molecules energy density $(eV.m^{-3})$
massa	Atom mass (kg)
massi	Molecular ion mass (kg)
massm	Molecule mass (kg)
pdena	Atoms particle density (m^{-3})
pdeni	Test ions particle density (m^{-3})
pdenm	Molecules particle density (m^{-3})
pux	x-component of poloidal unit vector at triangle center
puy	y-component of poloidal unit vector at triangle center
pvx	x-component of radial unit vector at triangle center
pvy	y-component of radial unit vector at triangle center
tindex	Triangle index
tr	Triangle centroid R coordinate (m)
triarea	Triangle area (m^2)
triavol	Triangle volume (m^3)
tz	Triangle centroid z coordinate (m)
vxa	Atoms x-directed (Cartesian) velocity $(m.s^{-1})$
vxdena	Atoms x-directed (Cartesian) momentum density $(kg.m^{-2}.s^{-1})$
vxdeni	Test ions x-directed (Cartesian) momentum density $(kg.m^{-2}.s^{-1})$
vxdenm	Molecules x-directed (Cartesian) momentum density $(kg.m^{-2}.s^{-1})$
vxi	Test ions x-directed (Cartesian) velocity $(m.s^{-1})$
vxm	Molecules x-directed (Cartesian) velocity $(m.s^{-1})$
vya	Atoms y-directed (Cartesian) velocity $(m.s^{-1})$
vydena	Atoms y-directed (Cartesian) momentum density $(kg.m^{-2}.s^{-1})$
vydeni	Test ions y-directed (Cartesian) momentum density $(kg.m^{-2}.s^{-1})$
vydenm	Molecules y-directed (Cartesian) momentum density $(kg.m^{-2}.s^{-1})$
vyi	Test ions y-directed (Cartesian) velocity $(m.s^{-1})$
vym	Molecules y-directed (Cartesian) velocity $(m.s^{-1})$
vza	Atoms z-directed (Cartesian) velocity $(m.s^{-1})$
vzdena	Atoms z-directed (Cartesian) momentum density $(kg.m^{-2}.s^{-1})$

continued on next page

continued from previous page	
name	description $(units)$
vzdeni	Test ions z-directed (Cartesian) momentum density $(kg.m^{-2}.s^{-1})$
vzdenm	Molecules z-directed (Cartesian) momentum density $(kg.m^{-2}.s^{-1})$
vzi	Test ions z-directed (Cartesian) velocity $(m.s^{-1})$
vzm	Molecules z-directed (Cartesian) velocity $(m.s^{-1})$

Table J.6: Quantities that can be stored in the "Eirene" stack.

Note that the neutral pressure is readily obtainable as $\frac{2}{3}$ of the energy density.

In table J.7, we give the list of commands that act on the "Eirene" stack or affect graphs of those data.

name	stack requirements							section	use
	'i'	'r'	's'	'm'	'e'	'w'	't'		
eabs	0	0	0	0	1	0	0	J.5.2	Absolute value
earow	0	1	0	0	0	0	0	J.5.8	Set arrow length scale for Eirene data plots
earox	0	0	0	0	1	0	0	J.5.8	Use as Eirene array for arrow x-component
earoy	0	0	0	0	1	0	0	J.5.8	Use as Eirene array for arrow y-component
eavg	0	8	0	0	1	0	0	J.5.10	Perform a volume average of Eirene data over a rectangular region
echs	0	0	0	0	1	0	0	J.5.2	Change sign of Eirene data
eclvd	0	3	0	0	0	0	0	J.5.9	Customize Eirene contour levels (delta)
eclvn	1	2	0	0	0	0	0	J.5.9	Customize Eirene contour levels (number)
eclvp	2	0	0	0	1	0	0	J.5.9	Customize Eirene contour levels (proportion)
eclvs	1	*	0	0	0	0	0	J.5.9	Customize Eirene contour levels (specified)
edims	0	0	0	0	0	0	0	J.5.4	Prints the dimensions and contents of the Eirene stack
edrop	0	0	0	0	1	0	0	J.5.4	Drop top of Eirene stack
edup	0	0	0	0	1	0	0	J.5.4	Duplicate top of Eirene stack
eexp	0	0	0	0	1	0	0	J.5.2	Take exponential of Eirene data
elbl	0	0	1	0	0	0	0	J.5.7	Provide a label for the Eirene data graph
elist	0	0	0	0	1	0	0	J.5.4	Lists the contents of the Eirene stack
eln	0	0	0	0	1	0	0	J.5.2	Take natural logarithm of Eirene data
elog	0	0	0	0	1	0	0	J.5.2	Take log_{10} of Eirene data
em*	0	0	0	0	2	0	0	J.5.3	Eirene stack multiplication
em+	0	0	0	0	2	0	0	J.5.3	Eirene stack addition
em-	0	0	0	0	2	0	0	J.5.3	Eirene stack substraction
em/	0	0	0	0	2	0	0	J.5.3	Eirene stack division
emeq	0	1	0	0	1	0	0	J.5.3	Eirene matrix element is equal to
emge	0	1	0	0	1	0	0	J.5.3	Eirene matrix element is greater than or equal to
emgt	0	1	0	0	1	0	0	J.5.3	Eirene matrix element is greater than
emle	0	1	0	0	1	0	0	J.5.3	Eirene matrix element is less than or equal to
emlt	0	1	0	0	1	0	0	J.5.3	Eirene matrix element is less than
emmax	0	0	0	0	2	0	0	J.5.3	Maximum of top two elements on the Eirene stack
emmin	0	0	0	0	2	0	0	J.5.3	Minimum of top two elements on the Eirene stack
emne	0	1	0	0	1	0	0	J.5.3	Eirene matrix element is not equal to
eoutp	1	0	1	0	1	0	0	J.5.10	Output the value of a specific Eirene triangle
continued on next page									

continued from previous page

name	stack requirements							section	use
	'i'	'r'	's'	'm'	'e'	'w'	't'		
eover	0	0	0	0	2	0	0	J.5.4	Copy over the second element of the Eirene stack
epos	0	0	0	0	1	0	0	J.5.3	Is Eirene data element positive?
eprt	0	0	1	0	1	0	0	J.5.10	Prints the Eirene data onto a file
esel	1	0	0	0	1	0	0	J.5.2	Select Eirene triangle
esqrt	0	0	0	0	1	0	0	J.5.2	Take square root of Eirene data
esumz	2	0	0	0	1	0	0	J.5.2	Sum over species
esurf	0	0	0	0	1	0	0	J.5.1	Plot Eirene data on the triangular grid
eswap	0	0	0	0	2	0	0	J.5.4	Swap top two elements of Eirene stack
etal	0	0	1	0	0	0	0	J.5.10	Read Eirene tally output file onto the Eirene stack
etos	0	0	0	0	1	0	0	J.5.4	Copy Eirene data onto matrix stack
evac	0	1	0	0	1	0	0	J.5.10	Fill vacuum region of Eirene stack with a value
wzrng	2	0	0	0	1	0	0	J.5.2	Select species range in Eirene data
ezsel	1	0	0	0	1	0	0	J.5.2	Select species in Eirene data
rem*	0	1	0	0	1	0	0	J.5.3	Eirene data * real
rem**	0	1	0	0	1	0	0	J.5.3	Eirene data ** real
rem+	0	1	0	0	1	0	0	J.5.3	Eirene data + real
rem-	0	1	0	0	1	0	0	J.5.3	Eirene data - real
rem/	0	1	0	0	1	0	0	J.5.3	Eirne data / real
remma	0	1	0	0	1	0	0	J.5.3	Maximum of Eirene data and real
remmi	0	1	0	0	1	0	0	J.5.3	Minimum of Eirene data and real
psta	0	0	0	0	1	0	0	J.5.2	Divide Eirene data per triangle area
pstv	0	0	0	0	1	0	0	J.5.2	Divide Eirene data per triangle volume
sumt	0	0	0	0	1	0	0	J.5.2	Sum over triangular grid

Table J.7: Commands related to the "Eirene" stack and triangular grid data plots.

In table J.8, we give the list of quantities that are available for use on the "wall" stack, containing values defined at the boundary surface at the edge of the computational domain.

name	description $(units)$
backscae	Backscattering energy fraction
backscat	Backscattering fraction
bbdyemis	Blackbody emissivity
c_cp_b1	Coating heat capacity b_1 factor $(J.kg^{-1}.K^{-1})$
c_cp_b2	Coating heat capacity b_2 factor $(J.kg^{-1}.K^{-2})$
c_cp_b3	Coating heat capacity b_3 factor (either $(J.kg^{-1}.K)$ [old format] or $(J.kg^{-1}.K^{-3})$ [new format])
c_cp_b4	Coating heat capacity b_4 factor $(J.kg^{-1}.K^{-4})$
ckappa_a	Coating factor a in $\kappa = 1/(aT + b)$ $(m.W^{-1})$
ckappaa1	Coating factor a_1 in κ polynomial fit $(W.m^{-1}.K^{-1})$
ckappaa2	Coating factor a_2 in κ polynomial fit $(W.m^{-1}.K^{-2})$
ckappaa3	Coating factor a_3 in κ polynomial fit $(W.m^{-1}.K^{-3})$
ckappaa4	Coating factor a_4 in κ polynomial fit $(W.m^{-1}.K^{-4})$
ckappaa5	Coating factor a_5 in κ polynomial fit $(W.m^{-1}.K^{-5})$

continued on next page

continued from previous page

name	description $(units)$
ckappa_b	Coating factor b in $\kappa = 1/(aT + b)$ $(m.K.W^{-1})$
ckapppa1	Coating factor a_1 in $\kappa_\perp$ polynomial fit $(W.m^{-1}.K^{-1})$
ckapppa2	Coating factor a_2 in $\kappa_\perp$ polynomial fit $(W.m^{-1}.K^{-2})$
ckapppa3	Coating factor a_3 in $\kappa_\perp$ polynomial fit $(W.m^{-1}.K^{-3})$
ckapppa4	Coating factor a_4 in $\kappa_\perp$ polynomial fit $(W.m^{-1}.K^{-4})$
ckapppa5	Coating factor a_5 in $\kappa_\perp$ polynomial fit $(W.m^{-1}.K^{-5})$
coathick	Wall element coating thickness (m)
cooltemp	Coolant temperature on back side of wall element (K)
cp_b1	Heat capacity b_1 factor $(J.kg^{-1}.K^{-1})$
cp_b2	Heat capacity b_2 factor $(J.kg^{-1}.K^{-2})$
cp_b3	Heat capacity b_3 factor (either $(J.kg^{-1}.K)$ [old format] or $(J.kg^{-1}.K^{-3})$ [new format])
cp_b4	Heat capacity b_4 factor $(J.kg^{-1}.K^{-4})$
ctgdns	Wall element coating material density $(kg.m^{-3})$
depo_ml	Monolayer deposition $(monolayers)$
deposit	Gross deposition $(particles)$
east	Wall elements corresponding to the East boundary
ei	Incident energy (eV)
erel	Chemical sputtering factor *erel*
ero_ml	Monolayer erosion $(monolayers)$
erosion	Gross erosion $(particles)$
ibnd	Wall element number
imapx	Wall element *ix* position
imapy	Wall element *iy* position
isweq	Species index *is* equal to
iswge	Species index *is* greater than or equal to
iswgt	Species index *is* greater than
iswle	Species index *is* less than or equal to
iswlt	Species index *is* less than
iswne	Species index *is* not equal to
iweq	Wall elements whose index is equal to
iwge	Wall elements whose index is greater than or equal to
iwgt	Wall elements whose index is greater than
iwle	Wall elements whose index is less than or equal to
iwlt	Wall elements whose index is less than
iwne	Wall elements whose index is not equal to
kappa_a	Factor a in $\kappa = 1/(aT + b)$ $(m.W^{-1})$
kappa_a1	Factor a_1 in κ polynomial fit $(W.m^{-1}.K^{-1})$
kappa_a2	Factor a_2 in κ polynomial fit $(W.m^{-1}.K^{-2})$
kappa_a3	Factor a_3 in κ polynomial fit $(W.m^{-1}.K^{-3})$
kappa_a4	Factor a_4 in κ polynomial fit $(W.m^{-1}.K^{-4})$
kappa_a5	Factor a_5 in κ polynomial fit $(W.m^{-1}.K^{-5})$
kappa_b	Factor b in $\kappa = 1/(aT + b)$ $(m.K.W^{-1})$
kappapa1	Factor a_1 in $\kappa_\perp$ polynomial fit $(W.m^{-1}.K^{-1})$
kappapa2	Factor a_2 in $\kappa_\perp$ polynomial fit $(W.m^{-1}.K^{-2})$
kappapa3	Factor a_3 in $\kappa_\perp$ polynomial fit $(W.m^{-1}.K^{-3})$
kappapa4	Factor a_4 in $\kappa_\perp$ polynomial fit $(W.m^{-1}.K^{-4})$
kappapa5	Factor a_5 in $\kappa_\perp$ polynomial fit $(W.m^{-1}.K^{-5})$
lcp	Layer heat capacity $(J.kg^{-1}.K^{-1})$
ldepth	Layer depth (m)
lemis	Layer blackbody emissivity

continued on next page

continued from previous page	
name	description $(units)$
lkappa	Layer heat conductivity $(W.m^{-1}.K^{-1})$
lrho	Layer mass density $(kg.m^{-3})$
lthick	Layer thickness (m)
north	Wall elements corresponding to the North boundary
nseg	Wall segment number
plathick	Wall element thickness (m)
qevap	Energy lost by vaporisation $(W.m^{-2})$
qrad	Energy lost by blackbody radiation $(W.m^{-2})$
r1	Start R position of wall element (m)
r2	End R position of wall element (m)
south	Wall elements corresponding to the South boundary
sput	Physical sputtering yield
sput2	Physical sputtering energy yield
sputchem	Chemical sputtering yield
sputfrac	Layer fraction
sputres	Radiation enhanced sputtering yield
stckcoef	Sticking coefficient
sublheat	Heat of sublimation $(J.kg^{-1})$
tgdns	Wall element material density $(kg.m^{-3})$
thermev	Thermal evaporation flux $(m^{-2}.s^{-1})$
time	Wall element time (s)
vapprs	Vapor pressure (Pa)
vapprs_a	Vapor pressure A factor (K)
vapprs_b	Vapor pressure B factor
vaprtemp	Vaporisation temperature of wall material (K)
wallarea	Wall area (m^2)
walldist	Distance along wall (m)
wallflux	Wall particle flux (s^{-1})
wallhflx	Wall heat flux $(W.m^{-2})$
wallngth	Wall element length (m)
walltemp	Wall surface temperature (K)
west	Wall elements corresponding to the West boundary
z1	Start Z position of wall element (m)
z2	End Z position of wall element (m)

Table J.8: Quantities that can be stored in the "wall" stack.

Note that the **nseg** command requires an integer argument ⟨`iseg`⟩. If ⟨`iseg`⟩ $= 0$, the arrays will contain the wall segment number of each element. If ⟨`iseg`⟩ is larger than `0`, the elements in segment with index ⟨`iseg`⟩ will contain `1.0`, and the rest will contain zero. If ⟨`iseg`⟩ is less than `0`, the elements in segment with index $-$⟨`iseg`⟩ will contain zero, and the rest will contain `1.0`.

The arrays **iwlt**, **iwle**, **iweq**, **iwne**, **iwge**, **iwgt**, **iswlt**, **iswle**, **isweq**, **iswne**, **iswge**, and **iswgt** all require an integer argument ⟨`i`⟩, with which the logical comparison is done. If the comparison test is true, the arrays will contain `1.0`, otherwise they will contain zero.

In table J.9, we give the list of commands that act on the "wall" stack or affect wall graphs.

name	stack requirements							section	use
	'i'	'r'	's'	'm'	'e'	'w'	't'		
ddw	0	0	0	0	0	1	0	J.5.2	Take derivative along wall length
intw	0	0	0	0	0	1	0	J.5.2	Integrate over walls (per segment)
intw*	0	0	0	0	0	1	0	J.5.2	Integrate over walls (all segments)
linw	0	0	0	0	0	1	0	J.5.8	Use linear scale for wall plot
linxw	0	0	0	0	0	1	0	J.5.8	Use linear abscissa for wall plot
logw	0	0	0	0	0	0	0	J.5.8	Use logarithmic scale for wall plot
logxw	0	0	0	0	0	0	0	J.5.8	Use logarithmic abscissa for wall plot
outpw	1	0	1	0	0	1	0	J.5.10	Output the value of a specific wall element
plwl	0	0	0	0	0	1	0	J.5.1	Plot in wall domain
pswa	0	0	0	0	0	1	0	J.5.2	Divide per wall element area
pwne	0	0	0	0	0	1	0	J.5.2	Divide by electron density
rwm*	0	1	0	0	0	1	0	J.5.3	Wall data * real
rwm**	0	1	0	0	0	1	0	J.5.3	Wall data ** real
rwm+	0	1	0	0	0	1	0	J.5.3	Wall data + real
rwm-	0	1	0	0	0	1	0	J.5.3	Wall data - real
rwm/	0	1	0	0	0	1	0	J.5.3	Wall data / real
rwmma	0	1	0	0	0	1	0	J.5.3	Maximum of wall data and real
rwmmi	0	1	0	0	0	1	0	J.5.3	Minimum of wall data and real
sumw	0	0	0	0	0	1	0	J.5.2	Sum over walls
wabs	0	0	0	0	0	1	0	J.5.2	Absolute value
wall	0	0	0	0	0	1	0	J.5.8	Add wall data to contour plot
wall_ank	1	0	0	0	0	0	0	J.5.11	Add ANK wall data to contour plot
wallf	0	0	1	0	0	0	0	J.5.8	Add experimental data to wall plot
wchs	0	0	0	0	0	1	0	J.5.2	Change sign of wall data
wclvd	0	3	0	0	0	0	0	J.5.9	Customize wall contour levels (delta)
wclvn	1	2	0	0	0	0	0	J.5.9	Customize wall contour levels (number)
wclvp	2	0	0	0	0	1	0	J.5.9	Customize wall contour levels (proportion)
wclvs	1	*	0	0	0	0	0	J.5.9	Customize wall contour levels (specified)
wdrop	0	0	0	0	0	1	0	J.5.4	Drop top of wall stack
wdup	0	0	0	0	0	1	0	J.5.4	Duplicate top of wall stack
wexp	0	0	0	0	0	1	0	J.5.2	Take exponential of wall data
wlist	0	0	0	0	0	1	0	J.5.4	Lists the contents of the wall stack
wln	0	0	0	0	0	1	0	J.5.2	Take natural logarithm of wall data
wlog	0	0	0	0	0	1	0	J.5.2	Take log_{10} of wall data
wm*	0	0	0	0	0	2	0	J.5.3	Wall stack multiplication
wm+	0	0	0	0	0	2	0	J.5.3	Wall stack addition
wm-	0	0	0	0	0	2	0	J.5.3	Wall stack substraction
wm/	0	0	0	0	0	2	0	J.5.3	Wall stack division
wmeq	0	1	0	0	0	1	0	J.5.3	Wall matrix element is equal to
wmge	0	1	0	0	0	1	0	J.5.3	Wall matrix element is greater than or equal to
wmgt	0	1	0	0	0	1	0	J.5.3	Wall matrix element is greater than
wmle	0	1	0	0	0	1	0	J.5.3	Wall matrix element is less than or equal to
wmlt	0	1	0	0	0	1	0	J.5.3	Wall matrix element is less than
wmmax	0	0	0	0	0	2	0	J.5.3	Maximum of top two elements on the wall stack
wmmin	0	0	0	0	0	2	0	J.5.3	Minimum of top two elements on the wall stack
wmne	0	1	0	0	0	1	0	J.5.3	Wall matrix element is not equal to
woff	0	0	0	0	0	0	0	J.5.8	Turn off wall plots

continued on next page

continued from previous page

name	stack requirements							section	use
	'i'	'r'	's'	'm'	'e'	'w'	't'		
wover	0	0	0	0	0	2	0	J.5.4	Copy over the second element of the wall stack
wpos	0	0	0	0	0	1	0	J.5.3	Is wall data element positive?
wprt	0	0	1	0	0	1	0	J.5.10	Prints the wall data onto a file
wsaw	1	0	1	0	0	0	0	J.5.4	Read external data onto the wall stack
wsel	1	0	0	0	0	1	0	J.5.2	Select wall element
wsqrt	0	0	0	0	0	1	0	J.5.2	Take square root of wall data
wsumz	2	0	0	0	0	1	0	J.5.2	Sum over species
wswap	0	0	0	0	0	2	0	J.5.4	Swap top two elements of wall stack
wtos	0	0	0	0	0	1	0	J.5.4	Copy wall data onto matrix stack
wtot	0	0	0	0	0	1	0	J.5.4	Copy wall data onto target stack
wwid	0	1	0	0	0	0	0	J.5.8	Wall linewidth
wzrng	2	0	0	0	0	1	0	J.5.2	Select species range in wall data
wzsel	1	0	0	0	0	1	0	J.5.2	Select species in wall data

Table J.9: Commands related to the "wall" stack and wall plots.

In table J.10, we give the list of quantities that are available for use on the "target" stack, containing values defined inside the target elements, if the appropriate **SOLPS 5.x** models have been activated. Note that by default, the "target" stack contains the target temperature (in K) upon **b2plot** start-up.

name	description $(units)$
cp	Target specific heat capacity $(J.kg^{-1}.K^{-1})$
dkappadt	Derivative of target heat conductivity $(W.m^{-1}.K^{-2})$
itrgeq	Is the target element index equal to?
itrgge	Is the target element index greater than or equal to?
itrggt	Is the target element index greater than?
itrgle	Is the target element index less than or equal to?
itrglt	Is the target element index less than?
itrgne	Is the target element index not equal to?
kappa	Target heat conductivity (from polynomial fit) $(W.m^{-1}.K^{-1})$
kappa_ab	Target heat conductivity (from $1/(aT+b)$ fit) $(W.m^{-1}.K^{-1})$
kappap	Perpendicular target heat conductivity $(W.m^{-1}.K^{-1})$
kappapdt	Perpendicular target heat conductivity derivative $(W.m^{-1}.K^{-2})$
trgdepth	Target element depth (m)
trgeast	Target elements pertaining to the East boundary
trgibnd	Target element index
trgiz	Target depth index
trgnorth	Target elements pertaining to the North boundary
trgr1	Radial coordinate of first side of target element (m)
trgr2	Radial coordinate of second side of target element (m)
trgseg	Target wall segment index
trgsouth	Target elements pertaining to the South boundary
trgtemp	Target temperature (K) (default)
trgvol	Target element volume (m^3)
trgwest	Target elements pertaining to the West boundary
trgwidth	Target element width (m)

continued on next page

continued from previous page	
name	description $(units)$
trgz1	Z coordinate of first side of target element (m)
trgz2	Z coordinate of second side of target element (m)
ztrgeq	Is the target element depth index equal to?
ztrgge	Is the target element depth index greater than or equal to?
ztrggt	Is the target element depth index greater than?
ztrgle	Is the target element depth index less than or equal to?
ztrglt	Is the target element depth index less than?
ztrgne	Is the target element depth index not equal to?

Table J.10: Available target quantities for plotting.

The arrays **itrglt**, **itrgle**, **itrgeq**, **itrgne**, **itrgge**, **itrggt**, **ztrglt**, **ztrgle**, **ztrgeq**, **ztrgne**, **ztrgge**, and **ztrggt** all require an integer argument ⟨`i`⟩, with which the logical comparison is done. If the comparison test is true, the arrays will contain `1.0`, otherwise they will contain zero.

The **trgseg** command requires an integer argument ⟨`iseg`⟩. If ⟨`iseg`⟩ $= 0$, the arrays will contain the wall segment number of each element. If ⟨`iseg`⟩ is larger than `0`, the elements in segment with index ⟨`iseg`⟩ will contain `1.0`, and the rest will contain zero. If ⟨`iseg`⟩ is less than `0`, the elements in segment with index $-$⟨`iseg`⟩ will contain zero, and the others will contain `1.0`.

In table J.11, we give the list of commands that act on the "target" stack or affect target graphs, with their respective stack requirements.

name	stack requirements							section	use
	'i'	'r'	's'	'm'	'e'	'w'	't'		
inttrgw	0	0	0	0	0	0	1	J.5.2	Integrate along walls (all segments)
inttrgw*	0	0	0	0	0	0	1	J.5.2	Integrate along walls (per segment)
inttrgz	0	0	0	0	0	0	1	J.5.2	Integrate in depth
lintrg	0	0	0	0	0	0	0	J.5.6	Linear target plot
linxtrg	0	0	0	0	0	0	0	J.5.6	Linear abscissae target plot
logtrg	0	0	0	0	0	0	0	J.5.6	Logarithmic target plot
logxtrg	0	0	0	0	0	0	0	J.5.6	Logarithmic abscissae target plot
outptrg	2	0	1	0	0	0	1	J.5.10	Outputs an element of target data
rtrgm*	0	1	0	0	0	0	1	J.5.3	Multiply the target data with a real
rtrgm**	0	1	0	0	0	0	1	J.5.3	Take the target data to a real power
rtrgm+	0	1	0	0	0	0	1	J.5.3	Add a real to the target data
rtrgm-	0	1	0	0	0	0	1	J.5.3	Substract a real from the target data
rtrgm/	0	1	0	0	0	0	1	J.5.3	Divide the target data by a real
rtrgmma	0	1	0	0	0	0	1	J.5.3	Take the maximum of the target data and a real
rtrgmmi	0	1	0	0	0	0	1	J.5.3	Take the minimum of the target data and a real
sumtrgw	0	0	0	0	0	0	1	J.5.2	Sum along wall coordinate
sumtrgz	0	0	0	0	0	0	1	J.5.2	Sum over target depth
tclvd	0	3	0	0	0	0	0	J.5.9	Customize target contour levels (delta)
tclvn	1	2	0	0	0	0	0	J.5.9	Customize target contour levels (number)
tclvp	2	0	0	0	0	0	1	J.5.9	Customize target contour levels (proportion)
tclvs	1	*	0	0	0	0	0	J.5.9	Customize target contour levels (specified)

continued on next page

continued from previous page

name	stack requirements							section	use
	'i'	'r'	's'	'm'	'e'	'w'	't'		
trg.w	1	0	0	0	0	0	1	J.5.1	Line target plot against wall element
trg.z	1	0	0	0	0	0	1	J.5.1	Line target plot against target depth
trgabs	0	0	0	0	0	0	1	J.5.2	Absolute value of target data
trgchs	0	0	0	0	0	0	1	J.5.2	Change sign of target data
trgddp	0	0	0	0	0	0	1	J.5.2	Take derivative of target data along perpendicular direction
trgddw	0	0	0	0	0	0	1	J.5.2	Take derivative of target data along wall coordinate
trgddz	0	0	0	0	0	0	1	J.5.2	Take derivative of target data along target depth
trgdrop	0	0	0	0	0	0	1	J.5.4	Drop the top element from the target stack
trgdup	0	0	0	0	0	0	1	J.5.4	Duplicate the top element from the target stack
trgeq	0	1	0	0	0	0	1	J.5.3	Target data is equal to
trgexp	0	0	0	0	0	0	1	J.5.2	Take exponential of target data
trgge	0	1	0	0	0	0	1	J.5.3	Target data is greater than or equal to
trggt	0	1	0	0	0	0	1	J.5.3	Target data is greater than
trgle	0	1	0	0	0	0	1	J.5.3	Target data is less than or equal to
trglist	0	0	0	0	0	0	1	J.5.4	List the contents of the target stack
trgln	0	0	0	0	0	0	1	J.5.2	Take natural logarithm of target data
trglog	0	0	0	0	0	0	1	J.5.2	Take log_{10} of target data
trglt	0	1	0	0	0	0	1	J.5.3	Target data is less than
trgm*	0	0	0	0	0	0	2	J.5.3	Target stack multiplication
trgm+	0	0	0	0	0	0	2	J.5.3	Target stack addition
trgm-	0	0	0	0	0	0	2	J.5.3	Target stack substraction
trgm/	0	0	0	0	0	0	2	J.5.3	Target stack division
trgmmax	0	0	0	0	0	0	2	J.5.3	Maximum of top two elements of target stack
trgmmin	0	0	0	0	0	0	2	J.5.3	Minimum of top two elements of target stack
trgne	0	1	0	0	0	0	1	J.5.3	Target data is not equal to
trgover	0	0	0	0	0	0	2	J.5.4	Copy over second element on top of target stack
trgpos	0	0	0	0	0	0	1	J.5.3	Is target data positive?
trgsqrt	0	0	0	0	0	0	1	J.5.2	Take square root of target data
trgsurf	0	0	0	0	0	0	1	J.5.1	Create target data contour plot
trgswap	0	0	0	0	0	0	2	J.5.4	Swap top two elements of target stack
trgtow	0	0	0	0	0	0	1	J.5.4	Copy target data onto wall stack
trgwsel	1	0	0	0	0	0	1	J.5.2	Select wall element in target data
trgzsel	1	0	0	0	0	0	1	J.5.2	Select depth element in target data

Table J.11: Commands related to the "target" stack and target plots.

Additionally, some real and integer constants used by the code can be read into the "real" and "integer" stacks of **b2plot** for further manipulation. These are given in the table J.12.

name	description $(units)$
avogr	Avogadro's constant N_A (mol^{-1})
c	Speed of light c (m/s)

continued on next page

continued from previous page	
name	description $(units)$
eps0	Electric permittivity of vacuum ϵ_0 (F)
ev	Electron-Volt eV (J)
ismain	Index of the main plasma species
ixref	Index of reference poloidal location
iyref	Index of reference radial location
jsep	Radial index of outermost closed flux surface (just inside the separatrix)
jxa	Outer midplane index
jxi	Inner midplane index
kbolt	Boltzmann's constant k_B $(J.K^{-1})$
me	Electron mass m_e (kg)
mp	Proton mass m_p (kg)
mu0	Magnetic permittivity of vacuum μ_0 $(F.m^{-1})$
natm	Number of atomic species followed by `Eirene`
nbnd	Number of boundary cells
ndepth	Size of target arrays in depth direction
nion	Number of molecular ion species followed by `Eirene`
nmol	Number of molecular species followed by `Eirene`
ns	Number of species
nspecies	Number of isonuclear sequences
nx	Mesh size in poloidal direction
nxtl	Position index of top inner divertor in DN cases
nxtr	Position index of top outer divertor in DN cases
ny	Mesh size in radial direction
pi	π
qe	Electron charge q_e (C)

Table J.12: Available **b2plot** constants.

In table J.13 some additional commands are given.

name	use
coro	Print coronal rates
debu , debug	Set debugging information level
eof	Indicate end of file
f30	Create a `fort.30` file
f31	Create a `fort.31` file
help	Print available commands
tranprof	Create `b2.transport.inputfile` file
user	Available for user-defined functions
wlld	Produces ANK wall loading files
quit , exit, end	Terminate program

Table J.13: Additional commands.

The **coro** command, with integer argument ⟨`is`⟩ and real argument ⟨`T`⟩ prints to `stdout` the coronal radiation rates for the isonuclear sequence of species ⟨`is`⟩ at temperature ⟨`T`⟩ (in eV).

The **debu** command, with integer argument ⟨level⟩, sets up the level of debugging information to ⟨level⟩ (default 0). The higher the value of ⟨level⟩, the more output is produced. **debug** is an alias for **debu**.

The **wlld** command requires an integer argument ⟨ig⟩. Let ⟨ig⟩= JKLM. If M >0, then the file radsrc.b2e is produced, containing the radiation sources (in MW/m^3). If L >0, then the file wlld.dat is produced, which contains the power loading from the plasma onto the Eirene walls (including splitting of the targets by the grid). If K >0, then a set of files is produced containing the power loading on the targets, provided the necessary preparatory data exists in the DG model. These files are:

ld_tg_*.dat provides the power loading at the targets, including contributions from plasma particles, neutrals, and radiation.

fp_tg_*.dat provides the particle fluxes at the targets

peak_power provides the location and value of the peak power at the lower targets

peak_power_u provides the location and value of the peak power at the upper targets

peak_power_w gives the width of the power profiles

where the asterisk (*) stands for *i* (inner target), *iu* (inner upper target), *ou* (outer upper target) or *o* (outer target), respectively. If J >0, then the codes produces a set of wlld_*.dat, where the asterisk (*) is the name of each "plot zone" defined in the DG model. Each such file contains the power loading onto the set of walls defined in that plot zone. If LM = 99 and J >0, additional diagnostics are produced that indicate the radiation split by isonuclear sequence. If LM = 98 and J >0, additional diagnostics are produced that indicate the radiation coming from the core additional sources. Note that the radiation loading is computed using a cylindrical approximation, i.e. neglecting the toroidal curvature.
If the user wants to use the heat transfer model for the monoblock and/or the reflector plates, then plot zones for the inner/outer target, the reflector plates, and/or the dome must be defined in the DG model, using the labels trg[i-o], [i-o]rp, and/or dome, respectively. The wlld_*.dat files produced should then be postprocessed by either b2ymb (see 3.18), b2yrp (3.19), or b2ydm (3.20).

J.4 Common errors

Amongst the most common errors are

- not distinguishing between real and integer values — real values have a decimal point in them or are written in exponential notation, integers consist only of numerals (preceded by a sign, if necessary);
- not preceding commands by their arguments — the whole program is structured to use a prefix notation (think HP calculator!);
- not delimiting character strings by single quotes.

J.5 Available Functions and Operators in b2plot

Section J.5.1 describes the commands used to produce a plot. Sections J.5.2 describes commands that manipulate quantities on the "matrix" stack, and J.5.3 describes the commands that can be used to do arithmetic. Stack manipulating commands are described in section J.5.4. Graph appearance is covered in sections J.5.5 (page format), J.5.6 (graph format) and J.5.7 (graph appearance). Overlays to contour plots are covered in section J.5.8 and changing contour parameters in section J.5.9. There is a grab bag of miscellaneous commands in section J.5.10.

J.5.1 Data plotting commands

These commands are typically used to produce a plot.

See section J.5.7 for a list of commands that can be used to change the graph appearance (text font, label size, overlays), and section J.5.6 for commands changing the graph format (physical or computational domain, aspect ratio, linear or logarithmic).

See also the **chor** (J.6.2), **chpl** (J.6.2), **chbx** (J.6.2), **chvl** (J.6.2), **flux** (J.6.2), **chmp** (J.6.2), **cam** (J.6.2), **ecam** (J.6.2), **echbx** (J.6.2), **echmp** (J.6.2), **echor** (J.6.2), **echvl** (J.6.2) and **movi** (J.6.2) commands.

A simple example is:

```
echo a4l 2 1 page 2.0 lbsz stdl glob vesl sep logf 3 ndec phys ni cont dab2 surf | b2plot
```

which produced figure J.1.

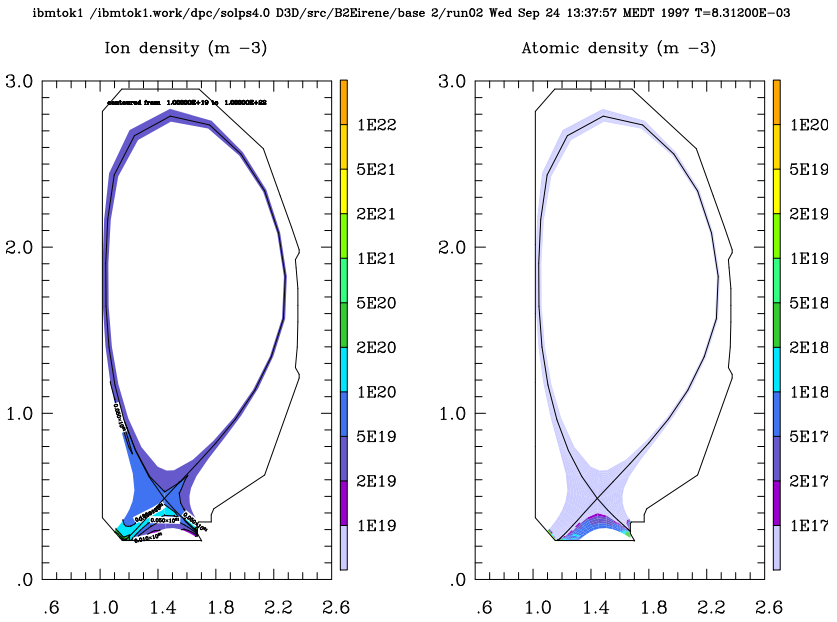


Figure J.1: Ion and atomic densities for DIII–D.

cont

Plots contour plots of the current data. See figure J.1. See section J.5.9 for a list of commands that can be used to change the appearance of contour plots. Section J.5.5 covers global changes to the plotting (square, landscape or portrait, number of graphs per page). The use of **surf** (see J.5.1) is preferred over that of **cont**.

<pos,I> f.x

Plots the 2-D graph of the matrix data at $iy = \langle \mathrm{pos} \rangle$ versus x. Vertical lines will be added to the plot to delimit any eventual unconnected domain boundaries. Use of **xpol** (default, see J.5.6) or **xpar** (see J.5.6) commands allows one to toggle between plotting against poloidal or parallel distance along the field line.

<pos,I> f.y

Plots the 2-D graph of the matrix data at $ix = \langle \mathtt{pos} \rangle$ versus y. Use of the **mmid** (see J.5.6), **fpsi** (see J.5.6), **mpsi** (see J.5.6) or **rpsi** (see J.5.6) commands allows for plotting versus the outer midplane (or another position chosen by the user) radial coordinate, full poloidal magnetic flux, normalized flux, or ρ_{pol}, respectively.

<n1,I> f.lx

Plots the 2-D graph of the matrix data at $iy = \langle \mathtt{pos} \rangle$ versus $\log(x)$. Equivalent to a combination of **logx** (see J.5.6) and **f.x** above, but remembers ths old setting of linear or logarithmic abscissa scale.

<n1,I> f.ly

Plots the 2-D graph of the matrix data at $ix = \langle \mathtt{pos} \rangle$ versus $\log(y)$. Equivalent to a combination of **logx** (see J.5.6) and **f.y** above, but remembers ths old setting of linear or logarithmic abscissa scale.

plwl

Plots in the "wall" domain (applicable to wall data only). This creates 2-D plots, with vertical lines separating the various wall segments.

<pos,I> trg.w

Plots the 2-D graph of target data at $iz = \langle \mathtt{pos} \rangle$ versus wall coordinate. Vertical lines will be added to the plot to delimit the various wall segments.

<pos,I> trg.z

Plots the 2-D graph of target data at element $i = \langle \mathtt{pos} \rangle$ versus z.

<n1,I> line

Chooses between plotting against x-coordinate (default, $\langle \mathtt{n1} \rangle = 1$) or y-coordinate ($\langle \mathtt{n1} \rangle = 2$). Not sure this works.

mesh

Plots the mesh. The colour of the mesh can be chosen with the **mcol** (see J.5.8) command.

trigmesh

Plots the triangular grid. The colour of the mesh can be chosen with the **mcol** (see J.5.8) command.

surf

Produces a "patch" plot where each cell is plotted in the colour associated with its value. This can be useful when the data is noisy and contour plots take excessive time. See figure J.1. Recommended.

esurf

Produces a ”patch” plot where each triangle from the `Eirene` grid is plotted in the colour associated with its value.

trgsurf

Produces a ”patch” plot of the target domain. Note that the side in contact with the plasma is at the bottom of the target computational domain.

traj

Plots trajectories. The following command

```
echo phys a4p vesl sep outl 70 1 70 36 1000 0 1 1 traj | b2plot
```

produced the results shown in figure J.2.

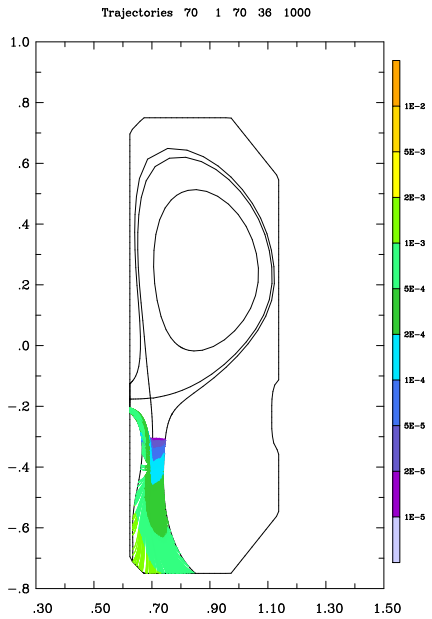


Figure J.2: Trajectory plot.

The command takes a variable number of integer arguments, the minimum number being 8.

The first 4 numbers specify the bottom left and top right coordinates of a rectangle in which the test particles are to be started (ix_{start} iy_{start} ix_{end} iy_{end}) (0’s will be replaced by the appropriate limit). The next number specifies the amount of test particles to be started (if positive the particles are started weighted by volume, if negative then the weighting function is taken off the matrix stack). This is then followed by a flag (1 for a movie, otherwise normal trajectories), and then the species indices come next. The last argument specifies the number of species. The use of **uuse** and **vvse** allows one to replace the particle velocities matrices used by default to compute the trajectories with data from the matrix stack.

The following example uses 2000 particles, weighted by particle number, over the entire domain, for the 6 charge states of carbon.

```
stdl glob
a4p
```

```
phys
-1.2 zmin 1.2 zmax
1.0 lbsz
'rgb.pal' cltb
vesl sep outl
2000 t_#p
1e-3 t_pt
na vol m*
0 0 0 0 -2000 1 2 3 4 5 6 7 6 traj
drop
```

From the code documentation:

```
C     This routine is used to plot trajectories of particles using the
C     velocity fields uu and vv. To call this routine
C     one must specify at least 8 parameters: ixmin, iymin, ixmax, iymax,
C     # of particles, 1 or 0 for movie or lines, species indices (at least
C     one), and last but not least the total number of species indices one
C     has given in the step before. A valid call could be:
C     0 0 95 35 100 0 0 1 traj
C     This should plot the trajectories of 100 particles which start
C     in any of the cells between cell (0,0) (southwest corner) and
C     cell (95,35) (northeast corner) as lines for hydrogen (0=H).
C     Note that the cell and species indices start at zero!
C     The number of time steps (plots) is given by the 't_#p' command.
C     The total flight time of the particles is given by the 't_pt' command.
C     The velocity fields uu and vv can be overwritten using the
C     'uuse' and 'vvse' commands. Their defaults values are ua and fna/na/sy
C     respectively. If the number of particles is positive, then the
C     particles are born randomly, with the probability of being born in a
C     cell being proportional to the cell volume. This probability weighting
C     can be changed by calling 'traj' with a negative number of particles.
C     Then the weighting function is whatever field is on top of the matrix
C     stack. For example, this set of commands does the following:
C     fnax na m/ psxperp 0 zsel uuse  ! uutraj set to poloidal
C                                     !   neutral flow velocity
C     fnay na m/ psy 0 zsel vvse      ! vvtraj set to radial
C                                     !   neutral flow velocity
C     1000 t_#p                       ! 1000 time steps
C     1e-4 t_pt                       ! followed for a total time of 1.e-4 s.
C     na 0 zsel                       ! weighted with neutral density
C     0 0 0 0 -1000 0 0 1 traj        ! over the entire physical domain
C                                     !   (0 0 0 0)
C               ! 1000 particles using non-default weighting function (-1000)
C               ! trajectories plotted (0)
C               ! for species 0 (=H neutrals) (0)
C               ! total of species considered is 1 (1)
C     ======================================================================
```

The trajectories are coloured according to the length of time the particle has traveled along it (illustrated by the label bar to the right of the graph). This label bar can be set to be logarithmic (default) with the **logt** command, or linear with the **lint** command.

J.5.2 Data manipulating functions

abs

Take the absolute value of the matrix data values.

eabs

Take the absolute value of the Eirene data values.

wabs

Take the absolute value of the wall data values.

trgabs

Take the absolute value of the target data values.

chs

Change sign of matrix data.

echs

Change sign of Eirene data.

trgchs

Change sign of target data.

wchs

Change sign of wall data.

ddw

Differentiates the wall data with respect to coordinate along wall perimeter.

ddx

Differentiates the matrix data with respect to poloidal coordinate x.

ddy

Differentiates the matrix data with respect to radial coordinate y.

trgddp

Differentiates the target data with respect to perpendicular coordinate.

trgddw

Differentiates the target data with respect to coordinate along wall perimeter.

trgddz

Differentiates the target data with respect to depth coordinate z.

inttrgw

Integrates data from the target stack along wall direction, summing over all wall segments.

inttrgw*

Integrates data from the target stack along wall direction, each segment being done separately.

inttrgz

Integrates data from the target stack in depth direction.

intw

Integrates data from the wall stack, summing over all wall segments.

intw*

Integrates data from the wall stack, each segment being done separately.

intx

Integrates data from the matrix stack along the x-direction, taking connectivities into account.

inty

Integrates data from the matrix stack along the y-direction.

pne

Divides the matrix data by the electron density n_e.

pwne

Divides the wall data by the local electron density n_e.

pswa

Divides the wall data by the wall element area.

psx

Divides the matrix data by s_x — the contact cell area between cells in the x-direction.

psxperp

Divides the matrix data by $s_x \times qc$ — the cell area perpendicular to the x-direction.

psy

Divides the matrix data by s_y — the cell area perpendicular to the y-direction.

pvol

Divides the matrix data by the volume of the cells.

psta

Divides the Eirene data by the triangular element area.

pstv

Divides the Eirene data by the triangular element volume.

sumt

Sums the Eirene data over all triangles, and then copies the sum to all elements.

sumtrgw

Sums the target data along the wall coordinate, and then copies the sum to all elements.

sumtrgz

Sums the target data depth-wise, and then copies the sum to all z-positions.

sumw

Sums the wall data, and then copies the sum to all elements.

sumx

Sums the matrix data over the entire x-direction, and then copies the sum to all x-positions.

sumy

Sums the matrix data over the entire y-direction, and then copies the sum to all y-positions.

<is$_{\text{start}}$,I> <is$_{\text{end}}$,I> sumz

Sums the matrix data over the z-direction from $\langle \texttt{is}_{\texttt{start}} \rangle$ to $\langle \texttt{is}_{\texttt{end}} \rangle$, and then copies the sum to the first z position. If $\langle \texttt{is}_{\texttt{start}} \rangle = \langle \texttt{is}_{\texttt{end}} \rangle = 0$, then the sum is done over all species.

<is$_{\text{start}}$,I> <is$_{\text{end}}$,I> esumz

Sums the Eirene data over the z-direction from $\langle \texttt{is}_{\texttt{start}} \rangle$ to $\langle \texttt{is}_{\texttt{end}} \rangle$, and then copies the sum to the first z position. If $\langle \texttt{is}_{\texttt{start}} \rangle = \langle \texttt{is}_{\texttt{end}} \rangle = 0$, then the sum is done over all species.

<is$_{\text{start}}$,I> <is$_{\text{end}}$,I> wsumz

Sums the wall data over the z-direction from $\langle \texttt{is}_{\texttt{start}} \rangle$ to $\langle \texttt{is}_{\texttt{end}} \rangle$, and then copies the sum to the first z position. If $\langle \texttt{is}_{\texttt{start}} \rangle = \langle \texttt{is}_{\texttt{end}} \rangle = 0$, then the sum is done over all species.

<is$_{\text{start}}$,I> <is$_{\text{end}}$,I> zrng

Selects the subset over species extended from species index $\langle \texttt{is}_{\texttt{start}} \rangle$ to species index $\langle \texttt{is}_{\texttt{end}} \rangle$ from the matrix data.

<is$_{\text{start}}$,I> <is$_{\text{end}}$,I> ezrng

Selects the subset over species extended from species index $\langle \texttt{is}_{\texttt{start}} \rangle$ to species index $\langle \texttt{is}_{\texttt{end}} \rangle$ from the Eirene data.

<is$_{\text{start}}$,I> <is$_{\text{end}}$,I> wzrng

Selects the subset over species extended from species index $\langle \texttt{is}_{\texttt{start}} \rangle$ to species index $\langle \texttt{is}_{\texttt{end}} \rangle$ from the wall data.

<i,I> esel

Propagates the Eirene data value from element $\langle \texttt{i} \rangle$ to all other elements.

<i,I> wsel

Propagates the wall data value from element $\langle \texttt{i} \rangle$ to all other elements.

<ix,I> xsel

Propagates the matrix data value in the specified position $\langle \texttt{ix} \rangle$ to all other x values.

<iy,I> ysel

Propagates the matrix data value in the specified position $\langle \texttt{iy} \rangle$ to all other y values.

<is,I> zsel

Propagates the matrix data value in the specified position $\langle \texttt{is} \rangle$ to all other z (species) values.

<i,I> trgwsel

Propagates the target data value from the specified element $\langle \texttt{i} \rangle$ to all other elements.

<is,I> ezsel

Propagates the Eirene data value in the specified position $\langle \texttt{is} \rangle$ to all other z (species) values.

<iz,I> trgzsel

Propagates the target data value at the specified depth $\langle \texttt{iz} \rangle$ to all other z values.

<is,I> wzsel

Propagates the wall data value in the specified position $\langle \texttt{is} \rangle$ to all other z (species) values.

rigz

Toggles the inclusion of ignorable cells in the plots. The ignorable cells are the guard cells at the boundaries and cells from isolated regions.

log

Computes logarithm base 10 of a matrix.

ln

Computes natural logarithm of a matrix.

sqrt

Takes square root of a matrix.

exp

Computes exponential of a matrix.

elog

Computes logarithm base 10 of Eirene data.

eln

Computes natural logarithm of Eirene data.

esqrt

Takes square root of Eirene data.

eexp

Computes exponential of Eirene data.

trglog

Computes logarithm base 10 of target data.

trgln

Computes natural logarithm of target data.

trgsqrt

Takes square root of target data.

trgexp

Computes exponential of target data.

wlog

Computes logarithm base 10 of wall data.

wln

Computes natural logarithm of wall data.

wsqrt

Takes square root of wall data.

wexp

Computes exponential of wall data.

J.5.3 Arithmetic commands

m+

Adds the two arrays on the top of the matrix stack.

m-

Subtracts the array on the top of the matrix stack from the array second from the top.

m*

Multiplies the two arrays on the top of the matrix stack.

m/

Divides the array second from the top of the matrix stack by the array on the top. Division by a zero element will yield zero.

mmin

Minimum of two top matrices on the stack.

mmax

Minimum of two top matrices on the stack.

em+

Adds the two arrays on the top of the Eirene stack.

em-

Subtracts the array on the top of the Eirene stack from the array second from the top.

em*

Multiplies the two arrays on the top of the Eirene stack.

em/

Divides the array second from the top of the Eirene stack by the array on the top. Division by a zero element will yield zero.

emmin

Minimum of two top arrays on the Eirene stack.

emmax

Minimum of two top arrays on the Eirene stack.

<r1,R> <r2,R> r+

Adds the two real values ⟨r1⟩ and ⟨r2⟩ on the top of the real stack.

<r1,R> <r2,R> r-

Subtracts the real value on the top of the real stack ⟨r2⟩ from the real value second from the top ⟨r1⟩.

<r1,R> <r2,R> r*

Multiplies the two real values ⟨r1⟩ and ⟨r2⟩ on the top of the real stack.

<r1,R> <r2,R> r/

Divides the real value second from the top of the real stack ⟨r1⟩ by the real value on the top of the real stack ⟨r2⟩. Division by 0 will yield an error and do nothing.

<r1,R> <r2,R> r**

Raises the real value second from the top of the real stack ⟨r1⟩ to the power of the real value on the top ⟨r2⟩. 0^0 will yield an error and do nothing.

<number,R> rm+

Adds ⟨number⟩ to the array on the top of the matrix stack.

<number,R> rm-

Substracts ⟨number⟩ from the array on the top of the matrix stack.

<number,R> rm*

Multiplies the array on the top of the matrix stack by ⟨number⟩.

<number,R> rm/

Divides the array on the top of the matrix stack by ⟨number⟩. Division by zero will yield an error and do nothing.

<number,R> rm

Raises each element of the array on the top of the matrix stack to the ⟨number⟩-th power. If ⟨number⟩ = 0, the result will be 1.0 for all cells.

<number,R> rmmi

Minimum of a matrix and ⟨number⟩.

<number,R> rmma

Maximum of a matrix and ⟨number⟩.

<number,R> rem+

Adds ⟨number⟩ to the array on the top of the Eirene stack.

<number,R> rem-

Substracts ⟨number⟩ from the array on the top of the Eirene stack.

<number,R> rem*

Multiplies the array on the top of the Eirene stack by ⟨number⟩.

<number,R> rem/

Divides the array on the top of the Eirene stack by ⟨number⟩. Division by zero will yield an error and do nothing.

<number,R> rem

Raises each element of the array on the top of the Eirene stack to the ⟨number⟩-th power. If ⟨number⟩ = 0, the result will be 1.0 for all cells.

<number,R> remmi

Minimum of an Eirene array and ⟨number⟩.

<number,R> remma

Maximum of an Eirene array and ⟨number⟩.

<number,R> rtrgm+

Adds ⟨number⟩ to the array on the top of the target stack.

<number,R> rtrgm-

Substracts ⟨number⟩ from the array on the top of the target stack.

<number,R> rtrgm*

Multiplies the array on the top of the target stack by ⟨number⟩.

<number,R> rtrgm/

Divides the array on the top of the target stack by ⟨number⟩. Division by zero will yield an error and do nothing.

<number,R> rtrgm**

Raises each element of the array on the top of the target stack to the ⟨number⟩-th power. If ⟨number⟩ $= 0$, the result will be `1.0` for all elements.

<number,R> rtrgmmi

Minimum of target data and ⟨number⟩.

<number,R> rtrgmma

Maximum of target data and ⟨number⟩.

<number,R> rwm+

Adds ⟨number⟩ to the array on the top of the wall stack.

<number,R> rwm-

Substracts ⟨number⟩ from the array on the top of the wall stack.

<number,R> rwm*

Multiplies the array on the top of the wall stack by ⟨number⟩.

<number,R> rwm/

Divides the array on the top of the wall stack by ⟨number⟩. Division by zero will yield an error and do nothing.

<number,R> rwm**

Raises each element of the array on the top of the wall stack to the ⟨number⟩-th power. If ⟨number⟩ = 0.0, the result will be `1.0` for all elements.

<number,R> rwmmi

Minimum of wall data and ⟨number⟩.

<number,R> rwmma

Maximum of wall data and ⟨number⟩.

<number,R> meq

If a matrix element is equal to ⟨number⟩, sets this matrix element to `1.0`. Otherwise, sets it to zero.

<number,R> mne

If a matrix element is not equal to ⟨number⟩, sets this matrix element to `1.0`. Otherwise, sets it to zero.

<number,R> mge

If a matrix element is greater than or equal to ⟨number⟩, sets this matrix element to `1.0`. Otherwise, sets it to zero.

<number,R> mgt

If a matrix element is greater than ⟨number⟩, sets this matrix element to `1.0`. Otherwise, sets it to zero.

<number,R> mle

If a matrix element is less than or equal to ⟨number⟩, sets this matrix element to `1.0`. Otherwise, sets it to zero.

<number,R> mlt

If a matrix element is less than ⟨number⟩, sets this matrix element to `1.0`. Otherwise, sets it to zero.

pos

If a matrix element is positive, sets this matrix element to `1.0`. Otherwise, sets it to zero.

<number,R> emeq

If an Eirene array element is equal to ⟨number⟩, sets this element to `1.0`. Otherwise, sets it to zero.

<number,R> emne

If an Eirene array element is not equal to ⟨number⟩, sets this element to 1.0. Otherwise, sets it to zero.

<number,R> emge

If an Eirene array element is greater than or equal to ⟨number⟩, sets this element to 1.0. Otherwise, sets it to zero.

<number,R> emgt

If an Eirene array element is greater than ⟨number⟩, sets this element to 1.0. Otherwise, sets it to zero.

<number,R> emle

If an Eirene array element is less than or equal to ⟨number⟩, sets this element to 1.0. Otherwise, sets it to zero.

<number,R> emlt

If an Eirene array element is less than ⟨number⟩, sets this element to 1.0. Otherwise, sets it to zero.

epos

If an Eirene array element is positive, sets this element to 1.0. Otherwise, sets it to zero.

<i1,I> <i2,I> i+

Adds the two integer values ⟨i1⟩ and ⟨i2⟩ on the top of the integer stack.

<i1,I> <i2,I> i-

Subtracts the integer value on the top of the integer stack ⟨i2⟩ from the integer value second from the top ⟨i1⟩.

<i1,I> <i2,I> i*

Multiplies the two integer values ⟨i1⟩ and ⟨i2⟩ on the top of the integer stack.

<i1,I> <i2,I> i/

Divides the integer value second from the top of the integer stack ⟨i1⟩ by the integer value on top of the integer stack ⟨i2⟩. This is integer division! Division by 0 will yield an error and do nothing.

<i1,I> <i2,I> i**

Raises the integer value second from the top of the integer stack ⟨i1⟩ to the power of the integer value on the top ⟨i2⟩. 0^0 will yield an error and do nothing.

<number,R> trgeq

If a target element is equal to ⟨`number`⟩, sets this target element to `1.0`. Otherwise, sets it to zero.

<number,R> trgne

If a target element is not equal to ⟨`number`⟩, sets this target element to `1.0`. Otherwise, sets it to zero.

<number,R> trgge

If a target element is greater than or equal to ⟨`number`⟩, sets this target element to `1.0`. Otherwise, sets it to zero.

<number,R> trggt

If a target element is greater than ⟨`number`⟩, sets this target element to `1.0`. Otherwise, sets it to zero.

<number,R> trgle

If a target element is less than or equal to ⟨`number`⟩, sets this target element to `1.0`. Otherwise, sets it to zero.

<number,R> trglt

If a target element is less than ⟨`number`⟩, sets this target element to `1.0`. Otherwise, sets it to zero.

trgmmin

Minimum of two top matrices on the target stack.

trgmmax

Maximum of two top matrices on the target stack.

trgm+

Adds the two arrays on the top of the target stack.

trgm-

Subtracts the array on the top of the target stack from the array second from the top.

trgm*

Multiplies the two arrays on the top of the target stack.

trgm/

Divides the array second from the top of the target stack by the array on the top. Division by a zero element will yield zero.

trgpos

If a target element is positive, sets this target element to 1.0. Otherwise, sets it to zero.

wm+

Adds the two arrays on the top of the wall stack.

wm-

Subtracts the array on the top of the wall stack from the array second from the top.

wm*

Multiplies the two arrays on the top of the wall stack.

wm/

Divides the array second from the top of the wall stack by the array on the top. Division by a zero element will yield zero.

wmmin

Minimum of two top data sets on the wall stack.

wmmax

Minimum of two top data sets on the wall stack.

<number,R> wmeq

If a wall element is equal to ⟨`number`⟩, sets this wall element to `1.0`. Otherwise, sets it to zero.

<number,R> wmne

If a wall element is not equal to ⟨`number`⟩, sets this wall element to `1.0`. Otherwise, sets it to zero.

<number,R> wmge

If a wall element is greater than or equal to ⟨`number`⟩, sets this wall element to `1.0`. Otherwise, sets it to zero.

<number,R> wmgt

If a wall element is greater than ⟨`number`⟩, sets this wall element to `1.0`. Otherwise, sets it to zero.

<number,R> wmle

If a wall element is less than or equal to ⟨`number`⟩, sets this wall element to `1.0`. Otherwise, sets it to zero.

<number,R> wmlt

If a wall element is less than ⟨`number`⟩, sets this wall element to `1.0`. Otherwise, sets it to zero.

wpos

If a wall element is positive, sets this wall element to `1.0`. Otherwise, sets it to zero.

J.5.4 Stack manipulating commands

drop

Deletes the array on the top of the matrix stack.

dup

Duplicates the array on the top of the matrix stack.

swap

Swaps the top two arrays on the matrix stack.

over

Copies the array second from the top to the top of the matrix stack.

list

Lists the contents of the matrix stack by printing the titles of the stored matrices to `stdout`.

mdims

Prints the dimensions and contents (i.e. data set titles) of the matrix stack to `stdout`.

rdrop

Deletes the real on the top of the real stack.

rdup

Duplicates the real on the top of the real stack.

rswap

Swaps the top two reals on the real stack.

rover

Copies the real second from the top to the top of the real stack.

rtostr

Writes the real on top of the real stack to the top of the string stack.

rtom

Copies the real on top of the real stack to the top of the matrix stack.

rtoe

Copies the real on top of the real stack to the top of the Eirene stack.

rtow

Copies the real on top of the real stack to the top of the wall stack.

rtot

Copies the real on top of the real stack to the top of the target stack.

rprt

Outputs the contents of the real stack to `stdout`, and to `b2plot.write` (in the **SOLPS 4.x** environment) or `b2pl.exe.dir/b2plot.write` (in the **SOLPS 5.x** environment) if **writ** is active.

idrop

Deletes the integer on the top of the integer stack.

idup

Duplicates the integer on the top of the integer stack.

iswap

Swaps the top two integers on the integer stack.

iover

Copies the integer second from the top to the top of the integer stack.

itor

Copies the integer on top of the integer stack to the top of the real stack.

itostr

Writes the integer on top of the integer stack to the top of the string stack.

itom

Copies the integer on top of the integer stack to the top of the matrix stack.

itoe

Copies the integer on top of the integer stack to the top of the Eirene stack.

itow

Copies the integer on top of the integer stack to the top of the wall stack.

itot

Copies the integer on top of the integer stack to the top of the target stack.

iprt

Outputs the contents of the integer stack to `stdout`, and to `b2plot.write` (in the **SOLPS 4.x** environment) or `b2pl.exe.dir/b2plot.write` (in the **SOLPS 5.x** environment) if **writ** is active.

scat

Concatenates the top two strings of the string stack.

sdrop

Deletes the string on the top of the string stack.

sdup

Duplicates the string on the top of the string stack.

sswap

Swaps the top two strings on the string stack.

sover

Copies the string second from the top to the top of the string stack.

sprt

Outputs the contents of the string stack to `stdout`, and to `b2plot.write` (in the **SOLPS 4.x** environment) or `b2pl.exe.dir/b2plot.write` (in the **SOLPS 5.x** environment) if **writ** is active.

<nx,I> <ny,I> <filename,S> saw

Reads the matrix of size $(\langle \texttt{nx} \rangle + 2) \times (\langle \texttt{ny} \rangle + 2)$ from file $\langle \texttt{filename} \rangle$ into the matrix stack.

<nx,I> <ny,I> <nz,I> <filename,S> saws

Reads the matrix of size $(\langle \texttt{nx} \rangle + 2) \times (\langle \texttt{ny} \rangle + 2) \times \langle \texttt{nz} \rangle$ from file $\langle \texttt{filename} \rangle$ into the matrix stack. Expects nz to match ns and that the data refers to the fluid plasma species.

<nx,I> <ny,I> <nz,I> <filename,S> sawx

Reads the matrix of size $(\langle \texttt{nx} \rangle + 2) \times (\langle \texttt{ny} \rangle + 2) \times \langle \texttt{nz} \rangle$ from file $\langle \texttt{filename} \rangle$ into the matrix stack. Expects nz to match ns_ext and that the data refers to the additional species followed by the external code to which **SOLPS** is coupled.

<nwall,I> <filename,S> wsaw

Reads the array of size $\langle \texttt{nwall} \rangle$ from file $\langle \texttt{filename} \rangle$ into the wall stack.

stoe

Copies the array on top of the matrix stack to the Eirene stack. Only the matrix elements from internal cells will be copied over, onto the corresponding triangular elements.

stow

Copies the array on top of the matrix stack to the wall data stack. Only the matrix elements from boundary cells will be copied over, onto the corresponding wall elements.

edrop

Deletes the array on the top of the Eirene stack.

edup

Duplicates the array on the top of the Eirene stack.

eswap

Swaps the top two arrays on the Eirene stack.

eover

Copies the array second from the top to the top of the Eirene stack.

elist

Lists the contents of the Eirene stack by printing the titles of the stored arrays to `stdout`.

edims

Prints the dimensions and contents (i.e. data set titles) of the Eirene stack to `stdout`.

etos

Copies the array on top of the Eirene stack to the matrix stack. The average value of the Eirene elements from internal cells will be copied over, onto the corresponding matrix elements.

trgdrop

Deletes the array on the top of the target stack.

trgdup

Duplicates the array on the top of the target stack.

trgswap

Swaps the top two arrays on the target stack.

trgover

Copies the array second from the top to the top of the target stack.

trglist

Lists the contents of the target stack by printing the titles of the stored matrices to `stdout`.

trgtow

Copies the matrix on top of the target data stack onto the wall data stack. Only the top surface target data values will be copied onto the corresponding wall elements.

wdrop

Deletes the array on the top of the wall stack.

wdup

Duplicates the array on the top of the wall stack.

wswap

Swaps the top two arrays on the wall stack.

wover

Copies the array second from the top to the top of the wall stack.

wlist

Lists the contents of the wall stack by printing the titles of the stored matrices to `stdout`.

wtos

Copies the array on top of the wall data stack onto the matrix stack. Each wall element value is copied onto the corresponding boundary cell in the matrix stack. Inner cells of the matrix stack will contain zero.

wtot

Copies the array on top of the wall data stack onto the target stack. Each wall element value is copied onto the corresponding column of cells in the target stack.

<ix,I> xtow

Copies the data from the ix-th row of the array on top of the matrix stack to the wall data stack.

<iy,I> ytow

Copies the data from the iy-th column of the array on top of the matrix stack to the wall data stack.

J.5.5 Page format parameters

a4l

Sets the plot area for A4 landscape. Use `ctrans -d ps.mono.land | a4page | lpr`.

a4p

Sets the plot area for A4 portrait. Use `ctrans -d ps.mono | a4page | lpr`.

sq

Sets the plot area to be square. This ensures that the$x$ and y scaling factors are the same.

<nx,I> <ny,I> page

Plots $\langle$`nx`$\rangle$ graphs in the x-direction and $\langle$`ny`$\rangle$ in the y-direction on the same page.

J.5.6 Graph format commands

both

Plots on both the physical and computational meshes.

comp

Plots on computational mesh only.

phys

Plots on the physical mesh only.

aspc

Toggles whether aspect ratio should be preserved for physical domain plots (default `.true.`). If `.false.`, the plot region limits will be those defined from the $rmin$ $rmax$ $zmin$ $zmax$, etc... values (perhaps modified to get "nice" axes intervals).

linf

Sets linear scaling for the function values (default).

logf

Sets logarithmic scaling for the function values.

lintrg

Sets linear scaling for the target data values (default).

logtrg

Sets logarithmic scaling for the target data values.

linxtrg

Sets linear abscissae for the target data values (default).

logxtrg

Sets logarithmic abscissae scaling for the target data values.

linx

Sets linear scaling for the abscissae values (default).

logx

Sets logarithmic scaling for the abscissae values.

xpar

Plots 2-D plots against parallel x distance.

xpol

Plots 2-D plots against poloidal x distance (default).

<n,I> ndec

Sets number of decades to $\langle \mathrm{n} \rangle$ for logarithmic plots (default 10). If $n = 0$, restores the default value.

mmid

Toggles mapping to midplane coordinate for 2-D plots. The midplane is assumed to be located at $ix = jxa$. See commands **omp** (section J.5.10) and **jxa** (section J.5.10) for setting and knowing its value. Special cases: For double-null cases, HFS data is mapped to the inner midplane $ix = jxi$, and for private flux data, it is mapped to the distance below the X-point.

fpsi

Toggles plotting 2-D radial plots against the full poloidal flux coordinate Ψ. If turned on, then turns off **mpsi** and **rpsi**.

mpsi

Toggles plotting 2-D radial plots against the normalised poloidal flux coordinate ψ. If turned on, then turns off **fpsi** and **rpsi**.

rpsi

Toggles plotting 2-D radial plots against the square root of normalised poloidal flux coordinate ($\rho_{pol} = \sqrt{\Psi}$). If turned on, then turns off **fpsi** and **mpsi**.

J.5.7 Graph appearance commands

bw

Want black and white for some plots.

col

Want colour for some plots.

bold

Toggles having the graph frames, axes, ticks and labels be in bold type.

<number,I> font

Changes the current font type to ⟨number⟩

- -25 TIMES ROMAN
- -26 TIMES BOLD
- -21 HELVETICA
- -22 HELVETICA BOLD
- -29 COURIER
- -30 COURIER BOLD

There are even more available fonts — refer to NCAR manpage.

In general, it is possible to change fonts within a text string to be printed on the graph, for the commands below taking a string as argument. For example, the command

```
'D-\\Ga\\R (photons m^{-3} s^{-1})' extl
```

would give an extra graph label containing 'D-α (photons m^{-3} s^{-1})'.

To switch font properties inside the string, use a double backslash followed by one of the letters:

R Switch to Roman characters

G Switch to Greek characters

d Switch to default characters

g Switch to Greek font

r Switch to Roman font

n Switch to normal level

h Resize font 130%

t Resize font 70%

m Switch to math font

Exponents can be requested with a string such as '^ {exp}', subscripts with '_{sub}', environment variables with '${var}', or the result of shell commands with ''cmd''.

stdl

Sets standard label as *extralabel*:

```
'\${HOST} \${PWD} `date` T=\${TIME}'
```

<label, S> glbl

Sets the current graph label to ⟨label⟩.

<label, S> elbl

Sets the current Eirene data graph label to ⟨label⟩.

<number,I> <label, S> slbl

Sets the species label of species index ⟨number⟩ to ⟨label⟩ for the current graph.

<label, S> tlbl

Sets the current target graph label to ⟨label⟩.

<label, S> trgblbl

Sets the current target bar label to ⟨label⟩.

<label, S> wlbl

Sets the current wall graph label to ⟨label⟩.

<number,I> <label, S> wslbl

Sets the species label of species index ⟨number⟩ to ⟨label⟩ for the current wall graph.

<label, S> wallblbl

Sets the current wall bar label to ⟨`label`⟩.

glob

Copies the contents of *extralabel* into the global header for all graphs.

<factor,R> glsz

Sets font size for global header to ⟨`factor`⟩ times default size (default is `1.0`).

<factor,R> lbsz

Sets label size to ⟨`factor`⟩ times default size (default is `1.0`). Does not apply to the global header.

<number,I> lfsz

Sets line label font size to ⟨`number`⟩ (default is `0`). If ⟨`number`⟩ is between `0` and `3`, then the size represents `1.0`, `1.5`, `2.0`, and `3.0` times an 8-plotter address unit width. Otherwise, it represents the character width in plotter address units.

nice

Toggles whether graphs will use nice axes with rounded values (`.true.` by default). Even with this set to `.false.`, some massaging of the plot range occurs.

noax

Does not plot axes.

<factor,R> lwid

Changes line width to ⟨`factor`⟩ times default width.

same

The next graphs occur on the same set of axes. Terminated with **new**.

new

Starts graphs on a new set of axes.

J.5.8 Overlays to contour and line plots

zero

Toggles the plotting of dotted axes lines at zero.

<factor,R> zwid

Changes width of dotted axes lines at zero to ⟨factor⟩ times default width.

<ncol,I> zcol

Plots axes line at zero with colour ⟨ncol⟩ (default is 1). The colours available by default are:

0 White (background)
1 Black (default)
2 Lavender
3 Dark Violet
4 Slate Blue
5 Royal Blue
6 Deep Sky Blue
7 Aqua
8 Green
9 Celeste
10 Chartreuse
11 Green Yellow
12 Yellow
13 Golden
14 Orange
15 Orange Red
16 Red
17 Orchid

These values can be overridden by use of the **cltb** command.

<ncol,I> dcol

Colour to be used for matrix data line plots (default 1). If **dcol** is set to a negative value, the data plot lines will use a succession of colours starting from |⟨dcol⟩| with increasing index. See **zcol** above for a list of available colours.

<ncol,I> tcol

Colour to be used for target data line plots (default 1). If **tcol** is set to a negative value, the data plot lines will use a succession of colours starting from $|tcol|$ with increasing index. See **zcol** above for a list of available colours.

<ncol,I> wcol

Colour to be used for wall data line plots (default 1). If **wcol** is set to a negative value, the data plot lines will use a succession of colours starting from $|wcol|$ with increasing index. See **zcol** above for a list of available colours.

<ncol,I> bcol

Colour to be used for printing label bar title (default 1). See **zcol** above for a list of available colours.

<ncol,I> fcol

Colour to be used for printing the graph frame, axes, ticks and labels (default 1). See **zcol** above for a list of available colours.

<ncol,I> gcol

Colour to be used for printing graph title (default 1). See **zcol** above for a list of available colours.

<ncol,I> hcol

Colour to be used for printing global header (default 1). See **zcol** above for a list of available colours.

<ncol,I> xcol

Colour to be used for printing extra label (default 1). See **zcol** above for a list of available colours.

<length,R> arow

Produces flow chart of arrays *up* and *vv* using a maximum arrow length of ⟨`length`⟩ metres. The use of *up* and *vv* can be superseded by the commands **arox** and **aroy**, respectively, which will then use data from the matrix stack for computing the arrow directions instead.

<length,R> earow

Produces flow chart of Eirene array *vxa* and *vya* using a maximum arrow length of ⟨`length`⟩ metres. The use of *vxa* and *vya* can be superseded by the commands **earox** and **earoy**, respectively, which will then use data from the Eirene stack for computing the arrow directions instead.

<widthfactor,R> arww

Changes default arrow width by ⟨`widthfactor`⟩ (default is `1.0`).

<ncol,I> acol

Plots overlaid arrows with colour ⟨ncol⟩ (default is 1). See **zcol** above for a list of available colours.

<data file,S> expf

File from which to take experimental data to be plotted with code data.

expl

Toggles between plotting the experimental data as a line (default) or as a scatter plot.

<expsymbol,I> exps

Choice of the symbol to be used forthe experimental data scatter plots (default 5). The numbering convention is the same as used in the libgr library from Eirene.

<data file,S> wallf

File from which to take experimental data to be plotted with wall data.

<ncol,I> ecol

Plots experimental data overlay with colour ⟨ncol⟩ (default is 1). If **ecol** is set to a negative value, the data plot lines will use a succession of colours starting from |ecol| with increasing index. See **zcol** above for a list of available colours.

<widthfactor,R> ewid

Changes the experimental data overlay line width by ⟨widthfactor⟩ (default is 1.0).

<n,I> ytra

Choose transformation function (⟨n⟩.ne.0) for radial coordinate to compare with experimental data, according to n.s.r.z.a_inner and n.s.r.z.a_outer files in the ${SOLPSTOP}/data.local or ${SOLPSTOP}/data directories.

sep

Toggles drawing of the separatrices.

<ncol,I> scol

Plots separatrices with colour ⟨ncol⟩ (default is 1). Also applies to the vertical separators drawn for 2-D plots. See **zcol** above for a list of available colours.

<widthfactor,R> sepw

Changes the default width of the separatrices line by ⟨`widthfactor`⟩ (default is `1.0`). Also applies to the vertical separators drawn for 2-D plots.

vesl

Toggles drawing of the outline of vacuum vessel. The data for this option is obtained from the file `mesh.extra` if it exists. If the `mesh.extra` file is not found, does nothing.

<ncol,I> vcol

Plots overlaid vacuum vessel outline with colour ⟨`ncol`⟩ (default is `1`). See **zcol** above for a list of available colours.

<widthfactor,R> vwid

Changes the default width of the vessel outline by ⟨`widthfactor`⟩ (default is `1.0`).

grid

Toggles drawing of overlaid mesh.

trig

Toggles drawing of overlaid triangular grid.

<ncol,I> mcol

Sets colour of overlaid mesh and triangular grid to ⟨`ncol`⟩ (default is `1`). See **zcol** above for a list of available colours.

outl

Toggles plotting of plasma domain outline. When target elements overlays are also plotted, then the target elements outline is activated as well.

<ncol,I> ocol

Sets colour of overlaid plasma outline to ⟨`ncol`⟩ (default is `1`). See **zcol** above for a list of available colours.

<widthfactor,R> owid

Changes the default width of the overlaid plasma outline by ⟨`widthfactor`⟩ (default is `1.0`).

wall

Toggles plotting of additional wall data.

woff

Turns off plotting of additional wall data (default).

<widthfactor,R> wwid

Changes the default width of the wall data overlay line by ⟨`widthfactor`⟩ (default is `10.0`).

logw

Selects logarithmic scale for wall plots.

linw

Selects linear scale for wall plots (default).

logxw

Selects logarithmic abscissae for wall plots.

linxw

Selects linear abscissae for wall plots (default).

target

Turns on plotting of additional target data.

trgoff

Turns off plotting of additional target data (default).

J.5.9 Changing contour parameters

nocp

Does not plot contour plot.

<level$_{min}$,R> <level$_{max}$,R> <delta,R> clvd

Starts contour levels at ⟨`level`$_{min}$⟩ increasing by ⟨`delta`⟩ up to ⟨`level`$_{max}$⟩.

<level$_{min}$,R> <level$_{max}$,R> <n,I> clvn

Produces ⟨`n`⟩ contour levels going from ⟨`level`$_{min}$⟩ to ⟨`level`$_{max}$⟩.

<n,I> <level$_1$,R> ... <level$_n$,R> clvs

Sets ⟨n⟩ contour levels to the values specified by ⟨label$_i$⟩.

`0 clvs`

can be used to go back to the default contour labelling scheme after use of **clvd**, **clvn**, **clvp** or **clvs**.

<npplvm,I> <npplvp,I> clvp

Sets up contour levels according to their proportion inside the plasma domain. The levels are arranged so that the integral of the absolute value of the data over the region between each two neighbouring level lines is the same. If ⟨npplvm⟩`.lt.0`, then this integral is equal (approximately) to $\frac{1}{\|\langle \text{npplvp} \rangle\|}$ of the integral over the whole volume. If ⟨npplvm⟩`.ge.0`, then the regions of positive and negative data values are treated separately, with $\|\langle \texttt{npplvp} \rangle\|$ data intervals for the positive and ⟨npplvm⟩ for the negative data range. In this case, if ⟨npplvp⟩`.ge.0` then `0` is added to the level list, otherwise, the negative and positive values closest to zero fall in the same zone.

<level$_{min}$,R> <level$_{max}$,R> <delta,R> eclvd

Starts contour levels at ⟨level$_{min}$⟩ increasing by ⟨delta⟩ up to ⟨level$_{max}$⟩ for Eirene data contour plot.

<level$_{min}$,R> <level$_{max}$,R> <n,I> eclvn

Produces ⟨n⟩ contour levels going from ⟨level$_{min}$⟩ to ⟨level$_{max}$⟩ for Eirene data contour plot.

<n,I> <level$_1$,R> ... <level$_n$,R> eclvs

Sets ⟨n⟩ contour levels to the values specified by ⟨label$_i$⟩ for Eirene data contour plots. As above, the command `0 eclvs` can be used to revert back to default behaviour.

<npplvm,I> <npplvp,I> eclvp

Sets up contour levels according to their proportion inside the Eirene domain, as per the **clvp** logic above.

<level$_{min}$,R> <level$_{max}$,R> <delta,R> tclvd

Starts contour levels at ⟨level$_{min}$⟩ increasing by ⟨delta⟩ up to ⟨level$_{max}$⟩ for target data contour plot.

<level$_{min}$,R> <level$_{max}$,R> <n,I> tclvn

Produces ⟨n⟩ contour levels going from ⟨level$_{min}$⟩ to ⟨level$_{max}$⟩ for target data contour plot.

<n,I> <level$_1$,R> ... <level$_n$,R> tclvs

Sets ⟨n⟩ contour levels to the values specified by ⟨label$_i$⟩ for target data contour plots. As above, the command `0 tclvs` can be used to revert back to default behaviour.

<npplvm,I> <npplvp,I> tclvp

Sets up contour levels according to their proportion inside the target domain, as per the **clvp** logic above.

<level$_{min}$,R> <level$_{max}$,R> <delta,R> wclvd

Selects threshold levels at ⟨level$_{min}$⟩ increasing by ⟨delta⟩ up to ⟨level$_{max}$⟩, for use in the wall data colour scheme.

<level$_{min}$,R> <level$_{max}$,R> <n,I> wclvn

Produces ⟨n⟩ threshold levels starting from ⟨level$_{min}$⟩ to ⟨level$_{max}$⟩ for the wall data colour scheme.

<n,I> <level$_1$,R> ... <level$_n$,R> wclvs

Sets ⟨n⟩ threshold levels to the values specified by ⟨label$_i$⟩ for the wall data colour scheme. As above, the command `0 wclvs` can be used to revert back to default behaviour.

<npplvm,I> <npplvp,I> wclvp

Sets threshold levels for the wall data colour scheme, as per the **cvlp** logic above.

clvl

Switches off contour level lines to be drawn into filled contour plot.

eclv

Switches to empty contour levels (for output on BW-Laser).

fclv

Switches to filled contour levels (default).

lacl

Switches to labels along contour lines.

lhor

Switches to horizontal labels (default).

lbon

Draws niveau labels into contour plot. (default).

lbof

Does not draw labels into contour plot.

lbsa

Toggles labeling of lines for 2-D plots with multiple curves (default `.TRUE.`).

spln

Uses spline interpolation for plot (very slow !!!).

<fill_opt,I> filo

Sets the fill option to ⟨`fill_opt`⟩. If it equals `0`, then colours are selected from the available colour range and adjacent levels might have the same value, otherwise modulo selection of colours is enabled and colour cycling might occur.

J.5.10 miscellaneous

neo_ic

The command takes a single integer argument and uses this to choose which transport contributions to include from NEOART.

The choices are:

0 NEOART: classical,

1 NEOART: banana plateau,

2 NEOART: Pfirsch-Schlueter,

3 NEOART: banana plateau + Pfirsch-Schlueter, and

4 NEOART: all.

The available neoclassical quantities are:

dnneo Neoclassical particle diffusivity D_n $(m^2.s^{-1})$,

chineo Neoclassical heat conductivity χ $(m^2.s^{-1})$,

vnneo Neoclassical particle pinch v_n $(m.s^{-1})$, and

vwneo Neoclassical heat pinch v_w $(m.s^{-1})$.

Related quantities are:

lnn Density gradient length (m),

ltt Temperature gradient length (m),

nuab Two species collision frequency (s^{-1}),

nua Total collision frequency (s^{-1}),

nustar Normalized collisionality,

vth Thermal velocity v_{th} $(m.s^{-1})$,

rhol Full-B Larmor radius (m),

deltal Ratio of Larmor radius to minimum gradient length,

wb Width of the banana orbit (m),

deltaw Ratio of banana width to minimum gradient length

deltawn Ratio of banana width to density gradient length

deltawt Ratio of banana width to temperature gradient length

omega Gyro-frequency $(rad.s^{-1})$

deltab Ratio of collision frequency to gyro-frequency, and

deltapar Ratio of perpendicular to parallel dynamics.

dase

Overwrites the *dab2* array with the top of the matrix stack.

eirc

Copies `Eirene` data onto `B2.5` data. Will only be performed once. Specifically, the densities of neutrals species (when a match between `B2.5` and `Eirene` species can be found) computed from `Eirene` (*dab2*) are added to the fluid neutral densities from `B2.5` (*na*). Same for the mass density field *ro*. If the atomic particle fluxes *pfla* and *rfla* are available, then the fluxes and velocities for the neutral species are also updated, meaning the *fnax*, *fnay*, *ua*, *up*, and *vv* fields, along with the default values of the *uutraj*, *vvtraj*, *arrowx*, and *arrowy* arrays.

<filename,S> etal

Copies data from the Eirene tally output file ⟨`filename`⟩ to the Eirene data stack. Such files are written from the volume tally output in block 11 of the Eirene input file. This function assumes the tally file contains 2-D data on the Eirene triangular grid.

<value,R> evac

Assigns real value ⟨`value`⟩ to all triangles from the top of the Eirene data stack that do not map to a `B2.5` grid cell.

writ

Writes out the labels and values for all simple 2-D plots to `b2plot.write` (in the **SOLPS 4.x** environment) or `b2pl.exe.dir/b2plot.write` (in the **SOLPS 5.x** environment). **write** is an alias for **writ**.

<filename,S> mprt

Writes out the top of the matrix stack to ⟨`filename`⟩ (in the **SOLPS 4.x** environment) or (in the **SOLPS 5.x** environment) `b2pl.exe.dir/`⟨`filename`⟩.

<filename,S> eprt

Writes out the top of the Eirene stack to ⟨`filename`⟩ (in the **SOLPS 4.x** environment) or (in the **SOLPS 5.x** environment) `b2pl.exe.dir/`⟨`filename`⟩.

<x1,R> <y1,R> <x2,R> <y2,R> <x3,R> <y3,R> <x4,R> <y4,R> eavg

Writes out the volume average of the data on top of the Eirene stack over the quadrangular region delimited by the four doublets of coordinates [($x1$,$y1$);($x2$,$y2$);($x3$,$y3$);($x4$,$y4$)].

<ix1,I> <ix2,I> <iy1,I> <iy2,I> <filename,S> mtab

Writes out a table extracted from the matrix on the top of the stack to ⟨`filename`⟩ (in the **SOLPS 4.x** environment) or (in the **SOLPS 5.x** environment) `b2pl.exe.dir/`⟨`filename`⟩. 0 0 pairs for the ix or iy values will output the entire ix or iy range, respectively.

<filename,S> wprt

Writes out the top of the wall stack to ⟨`filename`⟩ (in the **SOLPS 4.x** environment) or (in the **SOLPS 5.x** environment) `b2pl.exe.dir/`⟨`filename`⟩.

<iw1,I> <iw2,I> <filename,S> wtab

Writes out a table extracted from the array on the top of the wall stack to ⟨`filename`⟩ (in the **SOLPS 4.x** environment) or (in the **SOLPS 5.x** environment) `b2pl.exe.dir/`⟨`filename`⟩. A 0 0 pair for the iw value will output the entire range.

<filename,S> trgprt

Writes out the top of the target stack to ⟨`filename`⟩ (in the **SOLPS 4.x** environment) or (in the **SOLPS 5.x** environment) `b2pl.exe.dir/`⟨`filename`⟩.

<iw1,I> <iw2,I> <depth1,I> <depth2,I> <filename,S> trgtab

Writes out a table extracted from the matrix on the top of the target stack to ⟨`filename`⟩ (in the **SOLPS 4.x** environment) or (in the **SOLPS 5.x** environment) `b2pl.exe.dir/`⟨`filename`⟩. 0 0 pairs for the iw or $depth$ values will output the entire wall element or depth ranges, respectively.

<ix,I> <iy,I> <filename,S> outp

Writes out data from element (⟨`ix`⟩, ⟨`iy`⟩) from the top of the matrix stack to ⟨`filename`⟩ (in the **SOLPS 4.x** environment) or `b2pl.exe.dir/`⟨`filename`⟩ (in the **SOLPS 5.x** environment).

<i,I> <filename,S> eoutp

Writes out data from triangle element ⟨`i`⟩ from the top of the matrix stack to ⟨`filename`⟩ (in the **SOLPS 4.x** environment) or `b2pl.exe.dir/`⟨`filename`⟩ (in the **SOLPS 5.x** environment).

<i,I> <iz,I> <filename,S> outptrg

Writes out data from element (⟨i⟩, ⟨iz⟩) from the top of the target stack to ⟨filename⟩ (in the **SOLPS 4.x** environment) or b2pl.exe.dir/⟨filename⟩ (in the **SOLPS 5.x** environment).

<i,I> <filename,S> outpw

Writes out data from element ⟨i⟩ from the top of the wall stack to ⟨filename⟩ (in the **SOLPS 4.x** environment) or (in the **SOLPS 5.x** environment) to b2pl.exe.dir/⟨filename⟩.

f30

Produces the fort.30 file containing the geometry for use within Eirene.

f31

Produces the fort.31 file containing the plasma state for use within Eirene.

tranprof

Outputs the transport profile information as a b2.transport.inputfile file.

reso

Resets plotting output.

spec

Generates the spectral line emissivities from the Summers' data stored in ${SOLPSTOP}/modules/B2.5/Database/Summers (which can be superseded by the contents of ${SOLPSTOP}/data.local/Summers). One pre-selected diagnostic line per ionisation stage is provided. Data is only provided for Helium, Boron, Carbon, Oxygen and Neon.

<is,I> <lambda,I> <delta,I> nspe

Generates the spectral line emissivities according to the ADAS database [5], stored in ${SOLPSTOP}/modules/adas or ${SOLPSTOP}/data.local/adas, for species index ⟨is⟩, centered around wavelength ⟨lambda⟩ (in $\AA$), with width ⟨delta⟩ (in $\AA$). **nspec** is an alias for **nspe**.

<n,I> mvsk

Sets the movie skip factor to ⟨n⟩ (default is 1).

nese

Overwrites the n_e array with the top of the matrix stack.

tese

Overwrites the T_e array with the top of the matrix stack.

<ix,I> imp

Overwrites the inner midplane location index *jxi* with ⟨ix⟩. See description of b2mwti_jxi switch (see Appendix A.4) for the computation of its default value.

<ix,I> omp

Overwrites the outer midplane location index *jxa* with ⟨ix⟩. See description of b2mwti_jxa switch (see Appendix A.4) for the computation of its default value.

<iy,I> yref

Overwrites the radial reference location index *iyref* with ⟨iy⟩. See description of set_transport_iyref switch (see Appendix A.4) for the computation of its default value (usually the separatrix location).

<n,I> ave

Averages over ⟨n⟩ chords for **chor** and **chvl**.

<device name,S> pdev

Sets value of PLPLOT device name to ⟨device name⟩.

<winid,I> swid

Sets value of WIN_ID environment variable (part of TCL/TK interface) to ⟨winid⟩.

xyrz

Outputs *xmin*, *xmax*, *ymin*, *ymax*, *rmin*, *rmax*, *zmin*, and *zmax* to stdout.

<ig,I> user

Optional user routine. The choice of user routine is governed by ⟨ig⟩. Currently coded functions are:

ig = -20 Produces a file diagno.dat with coordinates of the grid cell centers and densities, temperatures for all species

ig = -19 Produces a file brooks.dat with coordinates of the grid cell centers and the plasma parameters (density, temperatures, drift velocity) in these points

ig = -18 Produces input files for a DIVIMP run. Requires one field on the matrix stack and one filename on the string stack to indicate which field is to be produced and where it is to be written

ig = -17 Prints detailed particle balance for the cells around the lower x-point in file x-point.dat

ig = -16 Produces a file `krieger.dat` with the plasma densities, temperatures, fluid velocities in the cell centers

ig = -15 Produces data files for the neutral fluxes on the specified surfaces. File names are `ndt_*.dat` (not yet available)

ig = -14 Produces data files for the cell-face-integrated fluxes of ions on every target. File names are `fi_tg_*.dat`

ig = -13 Produces a file `radsources.dat` with coordinates of the grid cell centers and the radiation intensity (total and components) in these points - for bolometry optimisation

ig = -12 Produces a file with coordinates of the grid cell centers (`wuerz-g.dat`) and the plasma parameters (densities, temperatures, fluid velocities) (`wuerz-d.dat`) in the Eastern divertor

ig = -11 Creates tables with coordinates of the grid cell corners along the separatrices and along the magnetic surfaces, together with the wall geometry in specified zones for separate plotting, in files `gm_sep.dat`, `gm_sep_i.dat`, `gm_sep_o.dat`, `gm_SOL.dat`, `gm_SOL_i.dat`, `gm_SOL_o.dat`, `gm_PFR.dat`, `gm_PFR_u.dat`, and `gm_core.dat`, according to magnetic topology

ig = -10 Creates table of fluxes and temperatures at the grid edges in file `gridedge.dat`

ig = -9 Prints out the total number of nuclei of each kind in the system in file `nuclei.tot`

ig = -8 Creates files `pr.dat` (plasma pressure), `prn.dat` (pressure due to hydrogen isotope *atoms*), `prt.dat` (sum of the previous two) and `sm.dat` (neutral momentum source)

ig = -7 Prints out the neutral data read from `fort.44` for debugging

ig = -6 Produces a file `zaveraev.dat` with coordinates of the grid cell centers and the plasma parameters (density, temperatures, neutral density) in these points

ig = -5 Writes the CII sources in the plasma for DIVIMP in file `CII_source.dvi`

ig = -4 Produces a map of model radiation emissivity on a rectangular grid for 3D calculation of the wall loading in files `fpri.dat`, `fpro.dat`, `fprp.dat`, `plenter.dat`, `plenter.grd`. Requires 11 numbers on the real stack for providing the emissivity profile shape and overall magnitude (see code in `b2pplent.F` routine for details)

ig = -3 Prints out the particle flux integrals onto the targets in file `particle.flx`

ig = -2 Produces a file `horton.dat` with coordinates of the grid cell centers and the plasma parameters (density, temperatures, neutral density, and so on) in these points

ig = -1 Produces a file `abramov.dat` with coordinates of the grid cell centers and the plasma parameters (density, temperatures, neutral density) in these points

$0 \leq$ ig $\leq$ ny Produces a file `oleg.dat` with plasma parameters (density, temperature, parallel flow velocity, parallel heat flux) against the distance along the magnetic field along flux surface index $\langle$`ig`$\rangle$

ny $<$ ig $<$ 1000 Produces two files with estimated helium concentration (`fraction.he`) and neutral pressure (`pressure.ntr`) in the PFR for summary plots

ig $\geq$ 1000 Adds a line to the file `../sum_tmp/spec_pwr.dat` with the power entering both divertors, power radiated along the divertor legs, power radiated near the targets, areas of the separatrix branches and mean specific power radiated along the divertor legs per square meter. $\langle$`ig`$\rangle$ -1000 cells are treated as "close to the targets"

J.5.11 Still need to do

bala Computes departure from pressure balance along field line

shco currently not used (meant for `TCL/TK` interface)

t_r+ set *ratio* to `.true.` (default)

t_r- set *ratio* to `.false.`. Not sure this does anything.

part partition colour map

wall_ank Add ANK wall data to contour plot

eof, exit, quit, end

Use of any of **eof**, **exit**, **end** or **quit** will terminate the **b2plot** run.

J.6 Commands taking a string parameter

J.6.1 Manipulating text appearing on graphs

<extra label> extl

Specifies an additional label to appear on the graphs.

Options to the 'extl' string

In your configuration file you can specify additional "text" which is added to the title of a plot. There are several TEX-like commands available:

- $\backslash R$ Switch to the "Roman" part of the `PWRITX` database
- $\backslash G$ Switch to the "Greek" part of the `PWRITX` database
- $\backslash d$ switches to default typeface
- $\backslash g$ switches to Greek typeface
- $\backslash r$ switches to standard roman typeface
- $\backslash h$ changes font size to huge
- $\backslash t$ changes font size to tiny
- ^ switches to superscript, e.g. word^{super}
- _ switches to subscript, e.g. word$_{sub}$
- $\backslash n$ returns to normal level
- $\backslash h$ increases font size by 130 percent
- $\backslash t$ decreases font size by 70 percent

It is however not possible to iterate super–/subscript with any of the other options like Roman or Greek font type! This can be achieved by using ^ and _, closing super- or subscript with $\backslash n$:
^$\backslash glt\backslash n\backslash$r text expands to $^{\lambda\tau}$ `text`

Environment variables can also be included by using, for example ${PWD}. `PWD` could be any environment variable except `TIME`, `LABEL` or `SONNETFILE` (either lower or uppercase) where the values stored in the `B2SXDR` file are used (for **SOLPS 4.x** runs).

Commands to be executed externally by a shell and returning a value can be entered between `s, e.g.

```
`date`
```

See section J.5.7 for the label set by **stdl** for an example using environment variables as well as shell expansions.

<x-axis label> xlab

Specifies a label for the x-axis. The same text options as for J.6.1 are available.

A (not to be taken too seriously) example for usage is

```
echo 'a4p phys 16 font "R [m]" xlab "Z [m]" ylab mesh' | b2plot -s
```

<y-axis label> ylab

Specifies a label for the y-axis. The same text options as for J.6.1 are available.

An example is provided in J.6.1.

J.6.2 Commands using external files

<palette file> cltb

Reads in the specifications for an alternative palette. The data format is RGB triplets for each colour.

If the file does not exist in the current working directory, the directories `${SOLPSTOP}/data.local/palettes`, `${SOLPSTOP}/scripts.local/palettes` and `${SOLPSTOP}/scripts/palettes` are searched in order.

Many different palettes are available and the user can easily create its own. For instance, the `blue_to_red` color maps are useful for data fields that have a zero in the middle (such as Mach numbers or velocities). The integer number at the end of the filename indicates how many colours the palette contains. Figure J.3 was produced using the commands below, and shows examples of the `jet`, `blue_to_red`, and `rainbow` palettes.

```
3 1 page a4l
sep outl comp
2.0 lbsz 3.0 sepw
0. 192. 17 clvn te
'jet_16.pal' cltb 'jet palette' extl
surf

ti
'blue_to_red_16.pal' cltb 'blue_to_red palette' extl
surf

mach
-1.2 1.2 17 clvn
'rainbow_16.pal' cltb 'rainbow palette' extl
surf
```

<chord file> chor

Reads in chords from the specified file and calculates the line integrals for the currently specified data. The data format in the file consists of a label (in single quotes) followed by an integer specifying the number of points to be sampled along the line. Following this line are lines consisting of pairs of X, Y, and Z coordinates and a numeric value. The numeric value is used as the ordinate when plotting the chord integrals.

If the file does not exist in the current working directory, the directories `${SOLPSTOP}/data.local/chords` and `${SOLPSTOP}/data/chords` are searched in order.

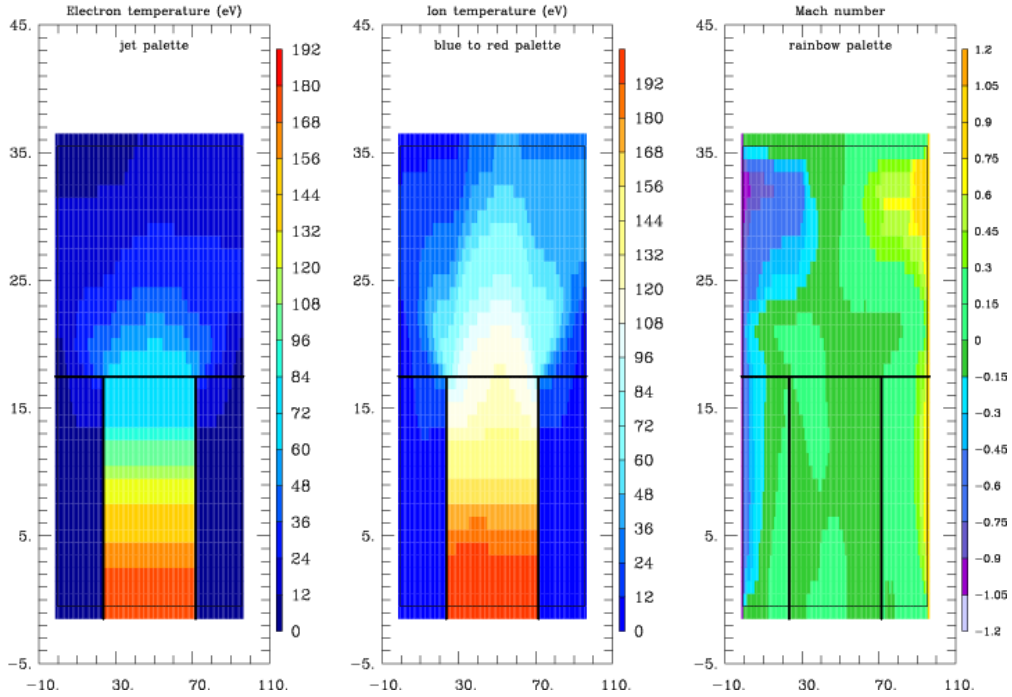


Figure J.3: Demonstration of various color palettes available in **b2plot**

<chord file> echor

Same as **chor** above, but does the line integral over the triangular grid.

<chord file> chpl

The chords specified by the chord file are plotted in the physical plane. Data format for the file is as for **chor** (see J.6.2).

Currently the following chord file examples (from AUG diagnostics) are provided (in the `${SOLPSTOP}/doc/solps/chords` directory):

- BLS, figure J.4(a), use:

  ```
  spec 'BLS.chords' chor
  ```

- BOL, figure J.4(b), use:

  ```
  b2ra pvol abs 0 0 sumz 'BOL.chords' chor
  ```

- DCN, figure J.4(c), use:

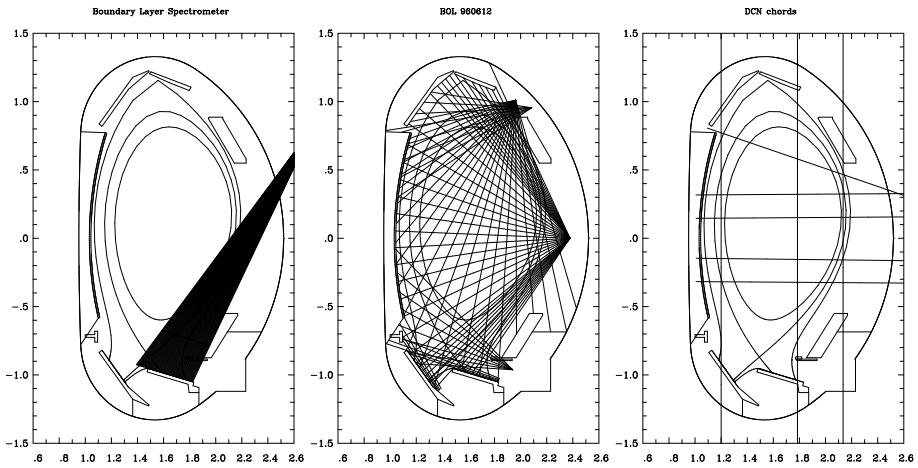
  ```
  ne 'DCN.chords' chor
  ```

- DIV, figure J.4(d), use:

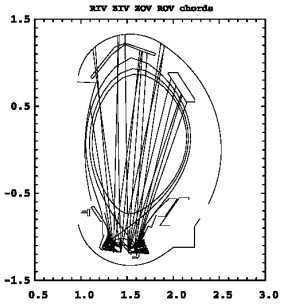
  ```
  spec 'DIV.chords' chor
  ```

- HAL, figure J.5(a), J.5(b), J.5(c), J.5(d) and J.5(e):

```
ahal mhal m+ 'HAL.chords' chor
```



(a) BLS.chords (b) BOL.chords (c) DCN.chords



(d) DIV.chords

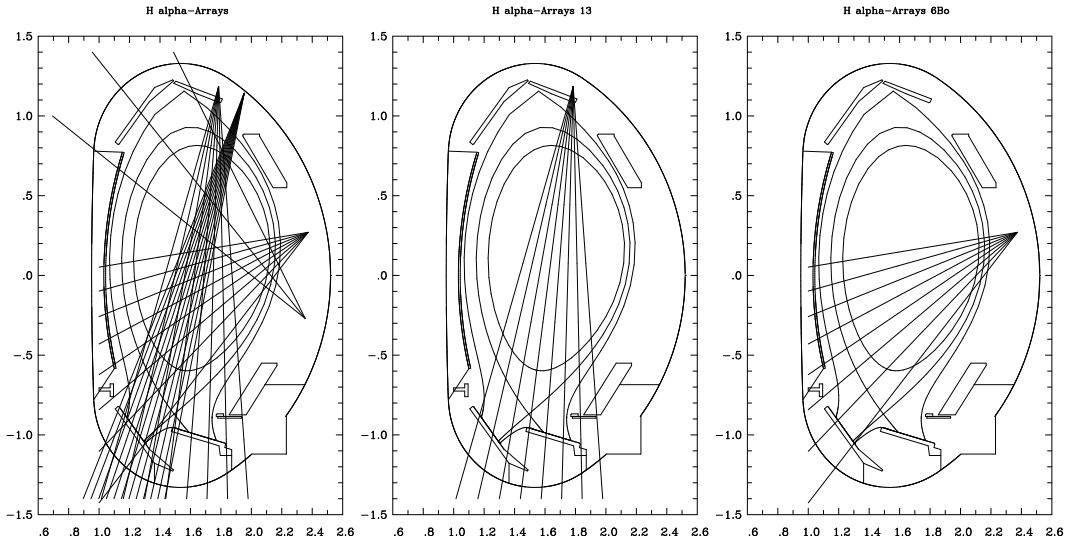
Figure J.4: BLS, BOL, DCN and DIV chords.

<command file> cmd

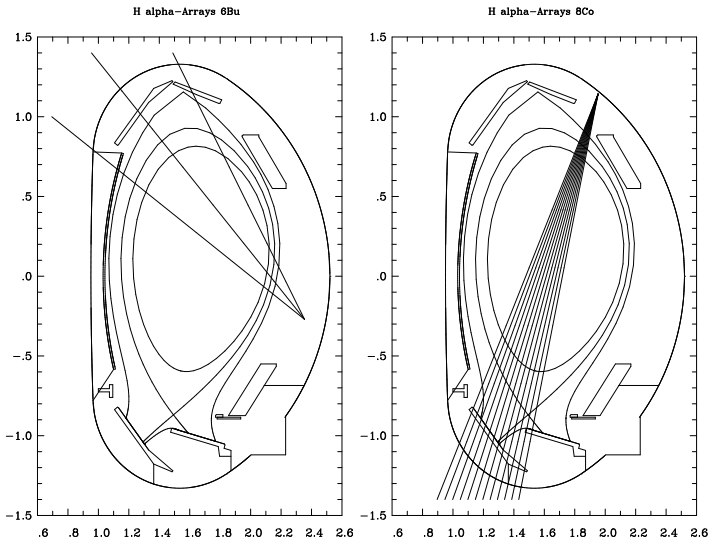
Further commands are taken from the specified file. If the file does not exist in the current working directory, the directories `${SOLPSTOP}/scripts.local/commands` and `${SOLPSTOP}/scripts/commands` are searched in order. The command file may contain comment lines, starting with `'*'`, `'!'` or `'#'`.

<CDF command file> cmdcdf

`CDF` commands used repetitively (for movie frames) are taken from the specified file. If the file does not exist in the current working directory, the directories `${SOLPSTOP}/scripts.local/commands` and `${SOLPSTOP}/scripts/commands` are searched in order.



(a) HAL.chords (b) HAL_13.chords (c) HAL_6Bo.chords



(d) HAL_6Bu.chords (e) HAL_8Co.chords

Figure J.5: HAL chords.

<chord file> chbx

Chord integral averaged over three chords. Equivalent to using the **3 ave** command.

<position file> chvl

The value is plotted at the specified positions.

<flux info> flux

Needs to be compiled with `-DWOS`. String should contain shot number, time within the shot, and the diagnostic name. Then causes the equilibrium for this shot to overlay subsequent plots.

<chord file> chmp

Plots a contour plot of the chord contribution from the mesh elements.

<chord file> cam

Camera plot. The chord file needs to contain some additional information on the first line.

<chord file> echbx

Chord integral of Eirene data averaged over three chords. Equivalent to using the **3 ave** command.

<position file> echvl

The Eirene stack data value is plotted at the specified positions.

<chord file> echmp

Plots a contour plot of the chord contribution from the triangles.

<chord file> ecam

Camera plot of Eirene data. The chord file needs to contain some additional information on the first line.

<CDF file name> movi

Used for specifying the `CDF` file if a movie is to be made.

<istart,I> <iend,I> <varname, S> movie

Creates a movie for timeslice ⟨`istart`⟩ to time slice ⟨`iend`⟩ of variable ⟨`varname`⟩. The list of available quantities in given in Appendix E. The time interval between frames can be controlled with the **mvsk** command. **nmovie** is an alias for **movie**.

<istart,I> <iend,I> <filename, S> <varname, S> omovie

Creates a movie for timeslice ⟨`istart`⟩ to time slice ⟨`iend`⟩ of variable ⟨`varname`⟩ read from file ⟨`filename`⟩.

<sonnetfile> sonn

Reads in a new `Sonnet` geometry file.

<**output_sonnet file**> **oson**

Outputs the geometry in `Sonnet ASCII` format.

<z_{shift},**R**> <**UEDGE grid file**> **uedgegrd**

Outputs the geometry in `UEDGE` ASCII format. The grid can be shifted in the vertical direction by an amount of $\langle z_{shift} \rangle$ metres.

<**filename**> <**H123**> <**reac**> <**crc**> **amju**

Computes a reaction rate as per the `Eirene` specification. Consult the `Eirene` manual for details. **amjuel** is an alias for **amju**.

<**reac**> **adas**

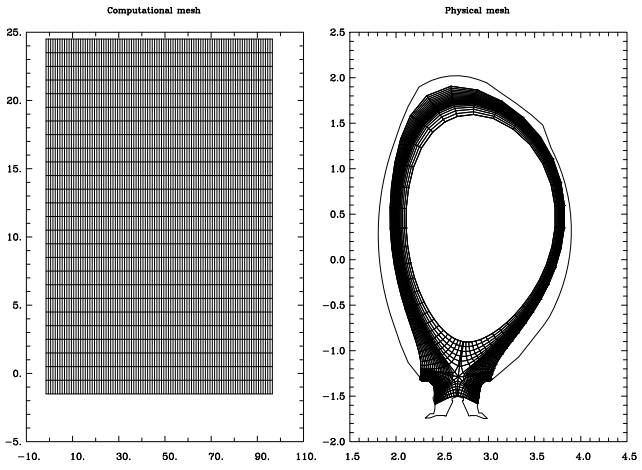
Computes a reaction rate as per the `ADAS` specification. The $\langle$`reac`$\rangle$ string should be something of the form 'acd89'. The matrix obtained contains then all the rates from these files, for all species in the run, extracted from the files stored in `${SOLPSTOP}/modules/adas` or `${SOLPSTOP}/data.local/adas`.

J.7 Examples

Some simple (and not so simple) examples of plots that can be obtained with **b2plot** commands are shown in the next section.

J.7.1 Mesh plot

Here (figure J.6) we have two plots of the computational and physical grids, captioned with the commands used to obtain them.

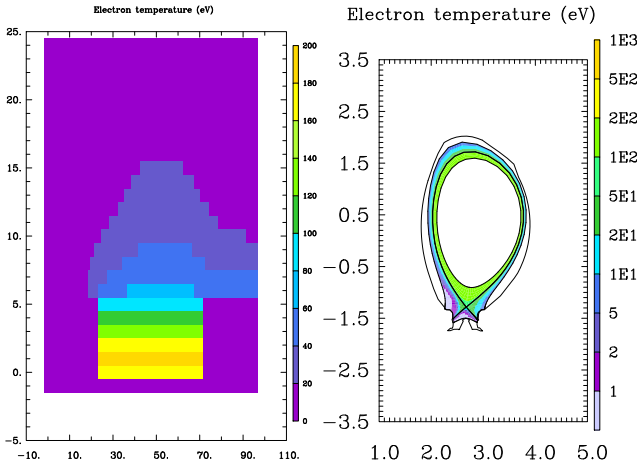


(a) comp a4p mesh (b) phys a4p mesh

Figure J.6: Mesh plots.

J.7.2 Plotting plasma solution data

Data from the plasma solution, stored on the ”matrix” stack, can be visualized as per figure J.7.



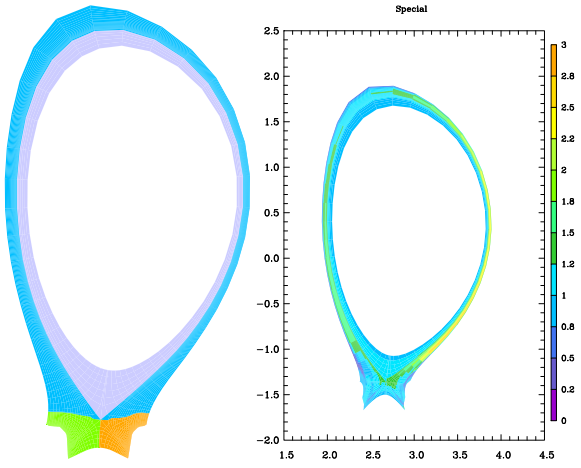
(a) comp a4p te surf

(b) phys a4p -2.2 zmin 2.2 zmax 2.0 lbsz sep outl vesl logf 3 ndec te surf

Figure J.7: Electron temperature plots.

Note that here the physical domain plot (**phys** figure J.7(b)) has been modified to have more readable labels (**2.0 lbsz**), a logarithmic scale (**logf**) covering 3 decades (**3 ndec**), includes the magnetic separatrix (**sep**), the plasma outline (**outl**), the vacuum vessel (**vesl**), and has an extended vertical range (**-2.2 zmin 2.2 zmax**). Note however that the plotting program always attempts to maintain ”nice” axis bounds and conserve the graph aspect ration (unless **aspc** is toggled).

Other examples:



(a) phys a4p noax 0 nreg

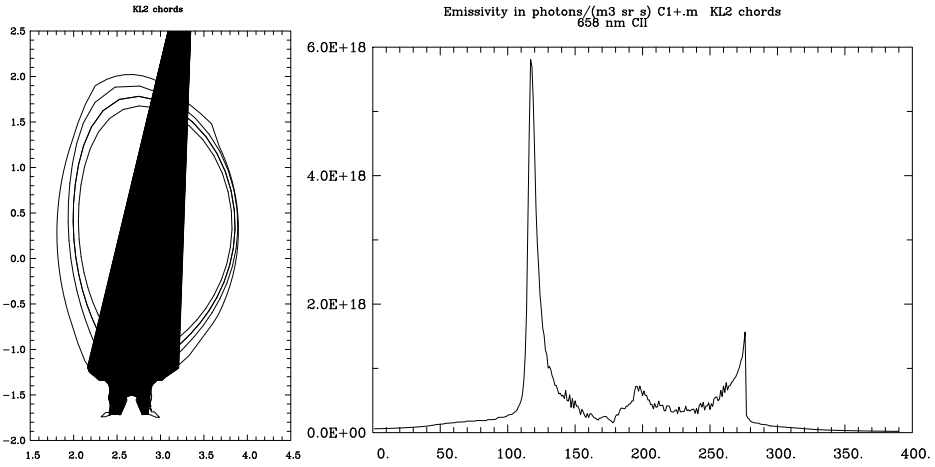
(b) phys a4p ti te m/ surf

Figure J.8: Other solution data plots.

In figure J.8(b), we obtain the ratio of ion to electron temperature, while figure J.8(a) is a region plot (like the ones in Fig. 3.2, obtained with **sep vesl outl 0 nreg surf**, and Fig. 3.3, obtained with **sep 0 nreg surf**).

J.7.3 Chord plots

In figure J.9 we have an example of chord plots as applied to the KL2 diagnostic on JET.



(a) a4p vesl outl sep'KL2.chords' chpl

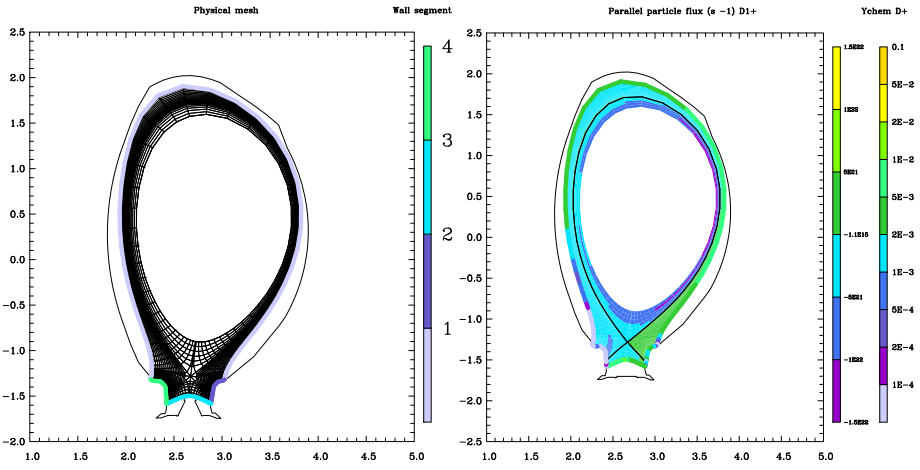
(b) a4l phys 3 6581 1 nspec '658 nm CII' extl 'KL2.chords' chor

Figure J.9: Chord plots applied to JET.

The first **chpl** plot (J.9(a)) gives the physical position of the chords, as obtained from the `KL2.chords` file. The second **chpl** plot (J.9(b)), in landscape mode **a4l**, shows the emissivity of a C^+ line (species 3) at $\lambda = 6581 \pm 1$ *Å* (**3 6581 1 nspec**). The **extl** instruction gives the extra label on the second title line. The chords file will be looked for, in succession, in the local directory, one level up, in `${SOLPSTOP}/data.local/chords`, and in `${SOLPSTOP}/data/chords`. The file used for this example can be found in `${SOLPSTOP}/doc/solps/chords`.

J.7.4 Wall plots

In figure J.10, we show examples of wall plots.



(a) phys 0 nseg 4 1. 2. 3. 4. wclvs10.0 wwid wall mesh

(b) vesl sep phys wall 10.0 wwid sputchem 1 wzsel 3 ndec logw fnax 1 zsel 'Ychem D+' wallblbl surf

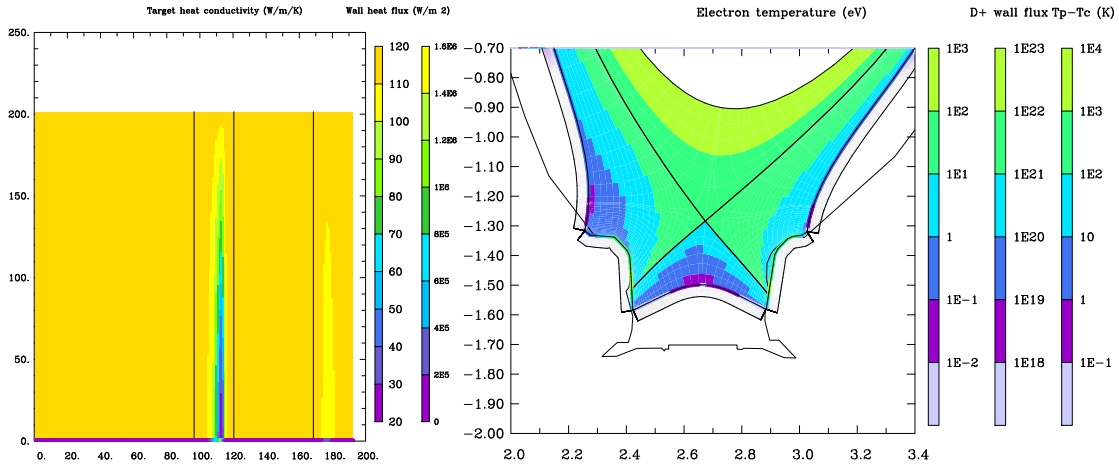
Figure J.10: Wall plots.

The wall segment indices are illustrated in figure J.10(a), obtained with the **0 nseg** command. We impose the contour levels to be at the values 1 through 4 with the **4 1. 2. 3. 4. wclvs** command. The thickness of the drawn wall line is given by **10.0 wwid**. **wall** asks for the wall data overlay on top of the requested **mesh** plot.

Another more complex example is given in figure J.10(b). It combines a **surf** plot of the D^+ poloidal particle flux (**fnax 1 zsel**, as D^+ has species index 1), with a **wall** plot of the chemical sputtering yield for D^+ (**sputchem 1 wzsel**). The latter is done on a logarithmic scale (**logw**), covering 3 decades (**3 ndec**). The label for the colour bar for the wall data is shortened with **'Ychem D+' wallblbl**. The wall colour bar is the rightmost one. Not choosing the **zsel** and **wzsel** options would have given a series of plots, one per species, with their individual poloidal particle flux and chemical sputtering yield. If only the **wzsel** option is used, one obtains a series of **surf** plots with all the same "wall" data overlay. If only the **zsel** option is used, then the **wall** overlay will come from the first species, namely D^0 in this case. It is therefore recommended to always make sure that the species dimensions of the "matrix" and "wall" data match!

J.7.5 Target plots

In figure J.11, we show examples of target plots.



(a) comp sep 10.0 wwid wallhflx wall kappa trgsurf

(b) a4l outl vesl 2.2 rmin 3.2 rmax -1.0 zmax -1.75 zmin sep 8.0 wwid logw te logf 5 ndec cooltemp wtot trgm- phys 'Tp-Tc (K)' trgblbl wallflux 1 wzsel 'D+ wall flux' wallblbl wall target logtrg surf

Figure J.11: Target plots.

The first plot (figure J.11(a)) shows the target heat conductivity (**kappa**) in the target computational domain (**comp** and **trgsurf**). The vertical separators (**sep**) indicate the segment boundaries. The overlay line at y=0, i.e. at the target surface in contact with the plasma is the **wall** data, here the incoming plasma heat flux (**wallhflx**).

The second plot (figure J.11(b)) is a triple plot, containing a plasma quantity from the "matrix" stack, (here **te**), with **wall** and **target** overlays. The "matrix" (**logf**), "wall" (**logw**) and "target" (**trglog**) data are all plotted on a logarithmic scale. The target data is obtained with the command

```
cooltemp wtot trgm-
```

which takes the "wall" quantity **cooltemp**, the temperature at the cooled end of the target plate, copies it over to the "target" stack (**wtot**), and subtracts it from the preceding "target" stack value (here the target temperature, which is loaded onto the stack by default at startup). So we obtain the amount by which the target has been heated by the plasma, and label it with **'Tp-Tc (K)' trgblbl**. The wall overlay data is the D^+ particle flux to the wall (**wallflux 1 wzsel**), labeled with **'D+ wall flux' wallblbl**. The colour bars, from left to right, correspond to the "matrix", "wall" and "target" data arrays.

J.7.6 Radiation plots

The following was used to produce a plot (figure J.12) of the emissivity arising out of line radiation for C and Ne.

```
! b2plot < b2rad.plot ; ctrans -d ps.land.color gmeta | a4page > ! tmp.ps
a4l 2 1 page lbof logf phys
b2ra pvol abs 2  8 sumz 1.0e4 fmin 1.0e8 fmax 'Carbon' extl surf
b2ra pvol abs 9 19 sumz 1.0e4 fmin 1.0e8 fmax 'Neon' extl surf
```

The first line is a comment indicating how **b2plot** should be run (the file containing the above lines was called `b2rad.plot`), and how the PostScript file should be generated.

The second line sets up an A4 landscape format with two graphs next to each other, sets labelling of contour levels off, logarithmic scaling and plotting on the physical domain only.

The third and fourth lines plot the line radiation data summed over species indices 2 to 8 and 9 to 19, respectively, assuming a D+C+Ne case, setting the data minimum to 10^4 and maximum to 10^8, labelling the graphs "Carbon" and "Neon" respectively.

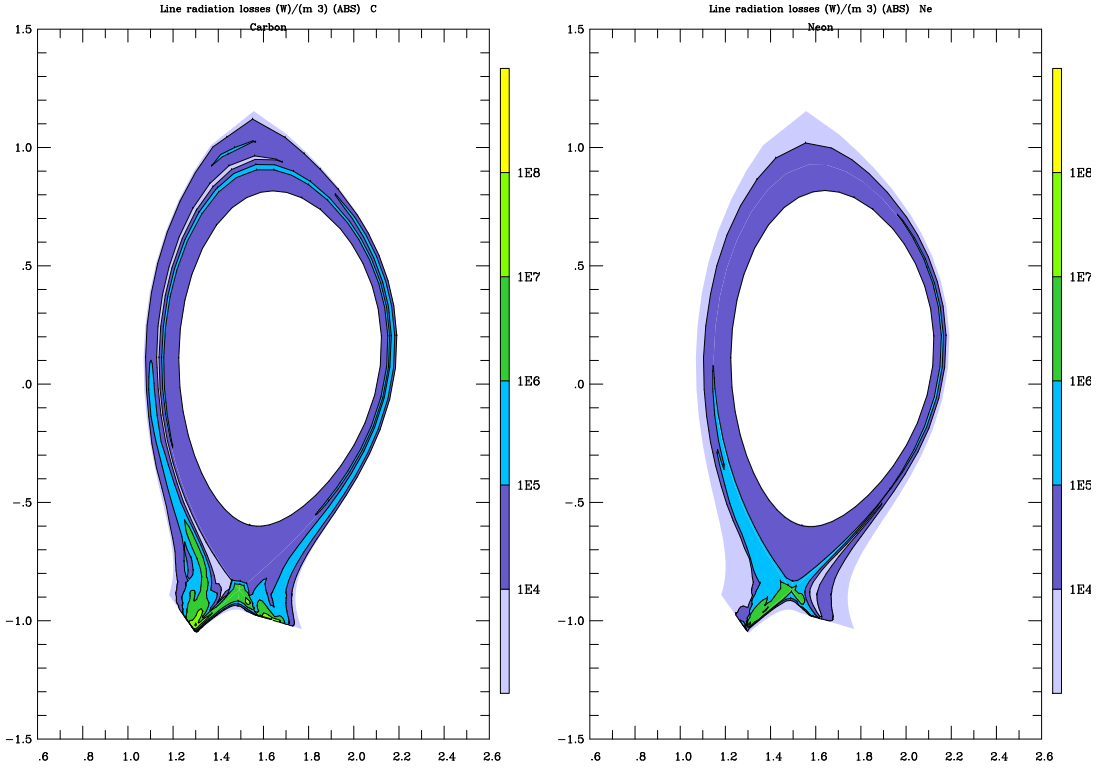


Figure J.12: C and Ne radiation.

Please note that **b2ra** only contains the line radiation emanating from the fluid species followed by `B2.5`. If you wish to also include the radiation emanating from the neutral kinetic species computed by `Eirene`, you need to add contributions from **neutrad**, **molrad**, and **ionrad**, which correspond to the radiation due to atoms, molecules, and molecular ions, respectively. Please be mindful however that **neutrad** has dimensions $natmi$ (**molrad** $nmoli$ and **ionrad** $nioni$) while **b2ra** has dimensions ns. Moreover, if you are intending to compute the total radiation rate, you should also include the bremsstrahlung contribution from **b2br**. You would then have as your set of commands, now using a D+He+C example:

```
a4l 3 1 page lbof logf phys sep vesl outl 1.5 lbsz 5 ndec
b2ra b2br m+ abs 0  1 sumz
```

```
neutrad 0 zsel m+ molrad  0 zsel m+ ionrad  0 zsel m+ pvol 0 ' ' slbl
'Total radiation loss (W/m^{3})' glbl 'Hydrogenic species' extl surf
b2ra b2br m+  abs 2  4 sumz
neutrad 1 zsel m+ pvol 0 ' ' slbl
'Total radiation loss (W/m^{3})' glbl 'Helium species' extl surf
b2ra b2br m+  abs 5 11 sumz
neutrad 2 zsel m+ pvol 0 ' ' slbl
'Total radiation loss (W/m^{3})' glbl 'Carbon species' extl surf
```

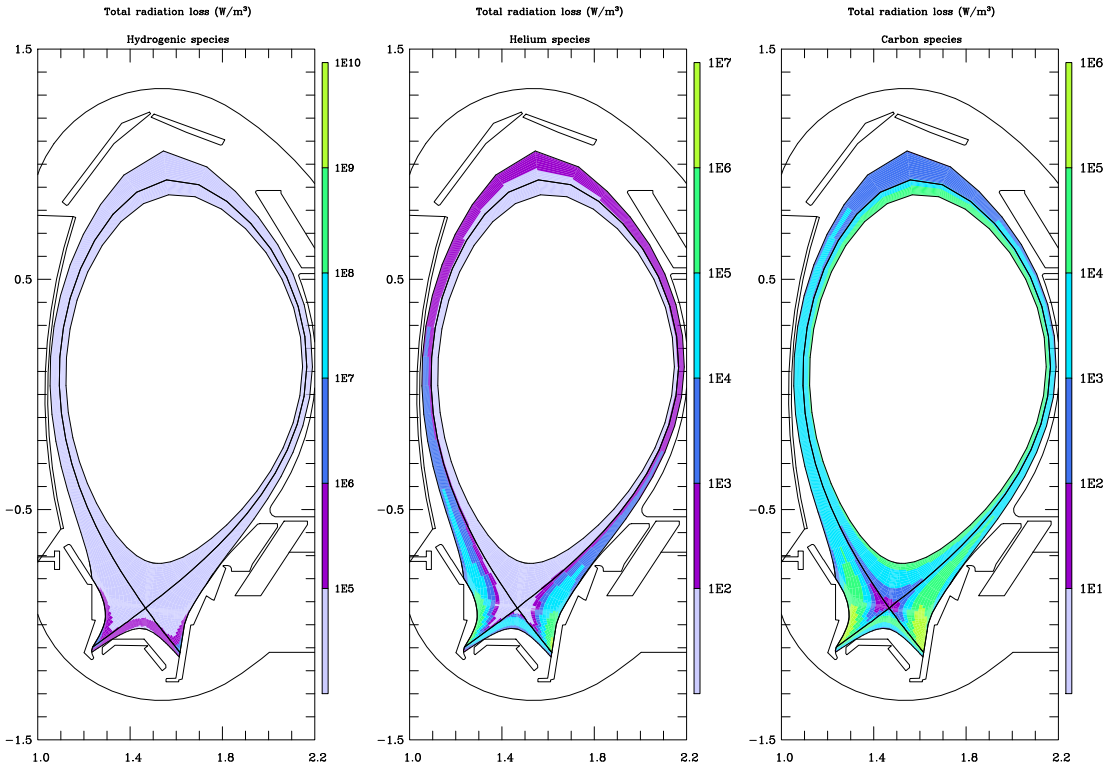


Figure J.13: Total radiation for a D+He+C example case.

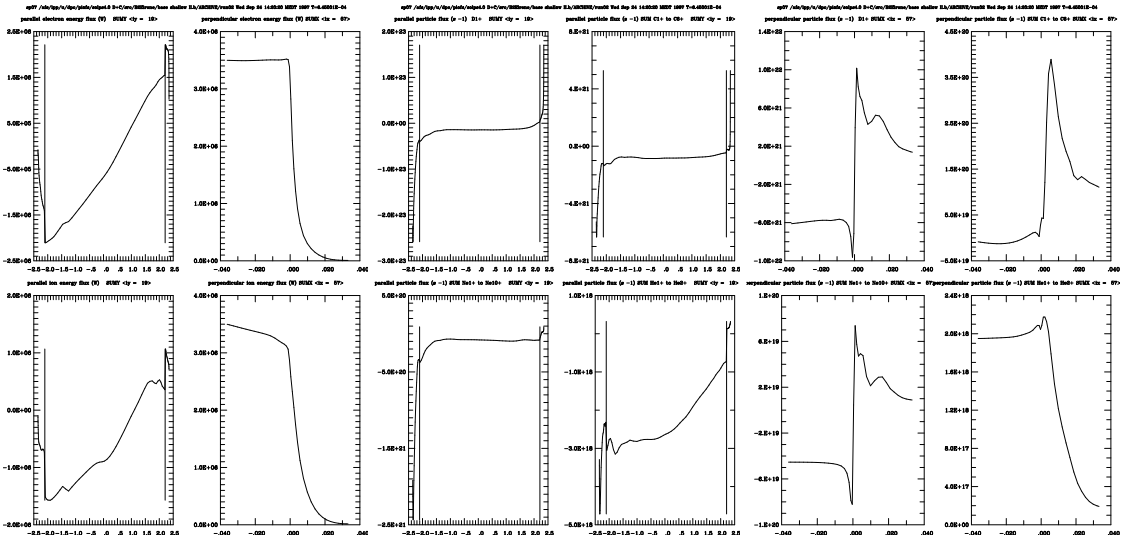
J.7.7 Fluxes and values plot

The following was used to calculate and plot integrated values of fluxes and some local quantities along various lines. See figure J.14.

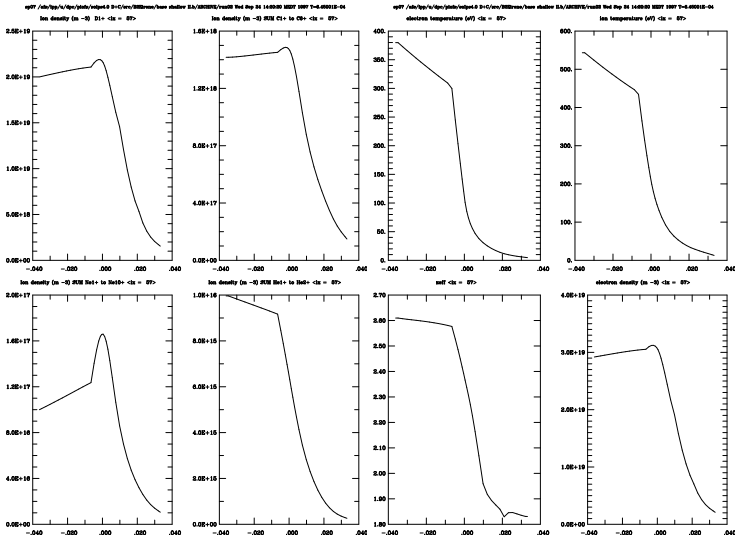
```
'${HOST} ${PWD} `date` T=${TIME}' extl glob
a4p 2 2 page
phys
feex sumy 19 f.x drop
feey sumx 57 f.y drop
feix sumy 19 f.x drop
feiy sumx 57 f.y drop
fnix 1 1 sumz sumy 19 f.x drop
fnix 2 7 sumz sumy 19 f.x drop
fnix 8 17 sumz sumy 19 f.x drop
fnix 18 19 sumz sumy 19 f.x drop
fniy 1 1 sumz sumx 57 f.y drop
```

```
fniy 2 7 sumz sumx 57 f.y drop
fniy 8 17 sumz sumx 57 f.y drop
fniy 18 19 sumz sumx 57 f.y drop
phys
ni 1 1 sumz 0.0 fmin 0.0 fmax 57 f.y drop
ni 2 7 sumz 0.0 fmin 0.0 fmax 57 f.y drop
ni 8 17 sumz 0.0 fmin 0.0 fmax 57 f.y drop
ni 18 19 sumz 0.0 fmin 0.0 fmax 57 f.y drop
te 0.0 fmin 0.0 fmax 57 f.y drop
ti 0.0 fmin 0.0 fmax 57 f.y drop
zeff 0.0 fmin 0.0 fmax 57 f.y drop
ne 0.0 fmin 0.0 fmax 57 f.y drop
```

This example plots a series of line plots in a 2×2 format on an A4 portrait page. Liberal use is made of **sumx** and **sumy** to produce appropriate sums, and of **sumz** to sum over the same species. **f.x** and **f.y** are used to plot the line plots. Setting both **fmin** and **fmax** to 0.0 means each plot is scaled separately — otherwise the scaling is done on all of the data.



(a) Page 1 (b) Page 2 (c) Page 3



(d) Page 4 (e) Page 5

Figure J.14: Line plots of fluxes and other quantities.

Appendix K

Miscellaneous

K.1 b2mod_dimensions.F

The file `${SOLPSTOP}/modules/B2.5/src/modules/b2mod_dimensions.F` and its local version `${SOLPSTOP}/modules/B2.5/src/modules.local/b2mod_dimensions.F` contain key dimensions that determine the sizes of arrays that can be defined by namelists in `b2.*.parameters` files (for compatibility with certain compilers).

Upon the first creation of the local version of the `b2mod_dimensions.F` file, one must make the `Makefile` aware of its presence by doing the following:

```
stop
gmake listobj
gmake tags
gmake depend
gmake
```

so that the local file is used. The last `gmake` command may of course be replaced by `gmake solps(_debug)`, `gmake solps_mpi(_debug)`, `gmake all(_debug)` or `gmake all_mpi(_debug)` as needed.

The quantities defined in `b2mod_dimensions.F` are:

DEF_ISOEXTRA the maximum number of extra isolating cells

DEF_MAXCHORDS Maximum number of chords to be considered (optional, internal default value is 10000)

DEF_NADD the maximum number of additional vacuum cells outside the gridded domain in `Eirene`

DEF_NATM the maximum number of neutral atomic species to be handled by `Eirene`

DEF_NBC the maximum number of boundary conditions provided in `b2.boundary.parameters` (optional, internal default value is 20)

DEF_NCHEN the maximum number of energy bins for the diagnostic chords in `Eirene`

DEF_NCUT the maximum number of "cuts" (2 for single null cases, 4 for double nulls)

DEF_NFL the maximum number of charged fluids passed from `B2.5` to `Eirene`

DEF_NGSTAL the maximum number of grids on which spatially resolved surface tallies are computed

DEF_NION the maximum number of molecular ion species to be handled by `Eirene`

DEF_NISO the maximum number of isolated regions within the `B2.5` computational domain (optional, internal default value is 10)

DEF_NLIM the maximum number of additional surfaces to be handled by DivGeo

DEF_NMOL the maximum number of neutral molecular species to be handled by Eirene

DEF_NPHID the maximum number of toroidal standard surfaces

DEF_NPLS the maximum number of background plasma species to be handled by Eirene

DEF_NPRNL the maximum number of particles that can be saved in the time-dependent arrays

DEF_NREAC the maximum number of reaction cards that can be created by Uinp

DEF_NRCX the maximum number of charge exchange reaction cards that can be created by Uinp

DEF_NREC the maximum number of recombination reaction cards that can be created by Uinp

DEF_NREI the maximum number of electron impact reaction cards that can be created by Uinp

DEF_NREL the maximum number of elastic collision reaction cards that can be created by Uinp

DEF_NSD the maximum number of plasma species handled in B2.5

DEF_NSPCSRFG the maximum number of groups of Eirene surfaces for which the data can be transferred from B2.5 (DG specification "Surface special")

DEF_NSRFS the maximum number of surfaces in a stratum

DEF_NSTRA the maximum number of Eirene strata

DEF_NSTS the maximum number of non-default standard surfaces to be handled by Eirene

DEF_NTOR the maximum number of toroidal periodicity boundaries

DEF_NXD the maximum number of cells in the x (parallel or poloidal) direction not including two guard cells

DEF_NYD the maximum number of cells in the radial direction not including two guard cells

HITS_PER_CHORD Maximum number of chord points per cell (optional, internal default value is 50)

The current default settings of b2mod_dimensions.F, which allow for a 128 $\times$ 36 DN geometry case including species H, D, T, He, Be and C, are:

```
      module b2mod_dimensions
      implicit none
!  Common dimensions
!
!  version : 01.12.98 21:42
!
#ifndef DIMENSIONS_DEFINED
#define DIMENSIONS_DEFINED

!
! parameters that are unique to B2
!

      integer, parameter :: DEF_NSD = 42
! DEF_NCUT is common to Eirene as well but needs to be defined here first
      integer, parameter :: DEF_NCUT = 4
      integer, parameter :: DEF_NBC = 6*(DEF_NCUT/2+1)
      integer, parameter :: DEF_NISO = 10

!
! parameters that are common to Eirene and B2
!

! NOTE: DEF_NXD should not include the additional cells to handle the cuts
      integer, parameter :: DEF_NXD = 128
      integer, parameter :: DEF_ISOEXTRA = 0
      integer, parameter :: DEF_NYD = 36
```

```
      integer, parameter :: DEF_NPHID = 1
      integer, parameter :: DEF_NATM = 6
      integer, parameter :: DEF_NMOL = 3
      integer, parameter :: DEF_NION = 3
      integer, parameter :: DEF_NSTRA = (5*DEF_NCUT/2+2)*DEF_NATM+1
      integer, parameter :: DEF_NFL = DEF_NSD
      integer, parameter :: DEF_NPLS = (DEF_NSD-DEF_NATM)+
     &                                 (DEF_NATM+DEF_NMOL)*
     &                                 (DEF_NATM+DEF_NMOL)
      integer, parameter :: DEF_NLIM = 300
      integer, parameter :: DEF_NSTS = 50
      integer, parameter :: DEF_NTOR = 60
!*** Max. number of groups of Eirene surfaces for which the data can
!*** be transferred from B2 (DG specification "Surface special")
      integer, parameter :: DEF_NSPCSRFG = 40

! new! [2002.04.22]
      integer, parameter :: DEF_NADD = 4
      integer, parameter :: DEF_NPRNL = 50000
! new! [2002.06.14]
      integer, parameter :: DEF_NSRFS = 4

!
! parameters that are unique to Eirene
!

      integer, parameter :: DEF_NGSTAL = 1

!
! parameters needed by uinp
!

      integer, parameter :: DEF_NREI = DEF_NATM+DEF_NMOL+DEF_NION+
     &                                 DEF_NFL
      integer, parameter :: DEF_NREC = DEF_NATM
      integer, parameter :: DEF_NRCX = DEF_NATM*(DEF_NION+DEF_NFL)
      integer, parameter :: DEF_NREL = (DEF_NATM+DEF_NMOL)*
     &                                 (DEF_NATM+DEF_NMOL)
      integer, parameter :: DEF_NREAC = DEF_NREI+DEF_NREC+DEF_NRCX+
     &                                  DEF_NREL

#endif
      end module b2mod_dimensions
```

K.2 Making movies

K.3 Compiling B2.5

(See also section 2.2.1, page 37.)

The following is the `Makefile` provided with the Master in the `sbb` directory:

```
# Test whether necessary environment variables are defined; if not, exit
ifndef HOST_NAME
  $(error HOST_NAME not defined)
endif
ifndef COMPILER
  $(error COMPILER not defined)
endif

SRCB2     = ${PWD}
SRCDIR    = ${SRCB2}/src
DOCDIR    = ${SRCDIR}/documentation
```

```
PYTHON  ?= python
TAGSLIST =
SOLPSINCLUDE ?=

MAKETAGS ?= ctags -e -f

MAKES = ${SRCB2}/Makefile
DEFINES = ${B25_DEFINES} ${SOLPS_CPP}
# Include global SOLPS compiler settings
ifndef SOLPS_CPP
  NODENAME = $(shell echo `hostname`)
  ifeq ($(shell [ -e ${SOLPSTOP}/SETUP/config.${HOST_NAME}.${COMPILER} ] && echo yes || echo no )
      ,yes)
    include ${SOLPSTOP}/SETUP/config.${HOST_NAME}.${COMPILER}
    MAKES += ${SOLPSTOP}/SETUP/setup.csh.${HOST_NAME}.${COMPILER} ${SOLPSTOP}/SETUP/config.${
        HOST_NAME}.${COMPILER}
  else
    $(warning ${SOLPSTOP}/SETUP/config.${HOST_NAME}.${COMPILER} not found. Assuming stand-alone
        compilation.)
  endif
  ifeq ($(shell [ -e ${SOLPSTOP}/SETUP/config.common.${COMPILER} ] && echo yes || echo no ),yes)
    include ${SOLPSTOP}/SETUP/config.common.${COMPILER}
    MAKES += ${SOLPSTOP}/SETUP/config.common.${COMPILER}
  endif
  ifeq ($(shell [ -e ${SOLPSTOP}/SETUP/config.${HOST_NAME}.${COMPILER}.local ] && echo yes ||
      echo no ),yes)
    include ${SOLPSTOP}/SETUP/config.${HOST_NAME}.${COMPILER}.local
    MAKES += ${SOLPSTOP}/SETUP/config.${HOST_NAME}.${COMPILER}.local
  endif
else
  MAKES += ${SOLPSTOP}/Makefile ${SOLPSTOP}/SETUP/setup.csh.${HOST_NAME}.${COMPILER} ${SOLPSTOP}/
      SETUP/config.${HOST_NAME}.${COMPILER}
  ifeq ($(shell [ -e ${SOLPSTOP}/SETUP/config.common.${COMPILER} ] && echo yes || echo no ),yes)
    MAKES += ${SOLPSTOP}/SETUP/config.common.${COMPILER}
  endif
  ifeq ($(shell [ -e ${SOLPSTOP}/SETUP/config.${HOST_NAME}.${COMPILER}.local ] && echo yes ||
      echo no ),yes)
    MAKES += ${SOLPSTOP}/SETUP/config.${HOST_NAME}.${COMPILER}.local
  endif
endif
ifeq ($(shell [ -e ${SOLPSTOP}/SETUP/setup.csh.${HOST_NAME}.${COMPILER}.local ] && echo yes ||
    echo no ),yes)
  MAKES += ${SOLPSTOP}/SETUP/setup.csh.${HOST_NAME}.${COMPILER}.local
endif

ifdef USE_EIRENE
  ifndef SOLPSTOP
    $(error SOLPSTOP not defined, but trying to compile with Eirene)
  endif
endif
ifdef SOLPS_MPI
  ifndef USE_MPI
    USE_MPI = -DUSE_MPI
  endif
endif
ifdef SOLPS_OPENMP
  ifndef USE_OPENMP
    USE_OPENMP = -D_OPENMP
  endif
endif

# Default prefix for OBJDIR: standalone
PREF_OBJDIR = standalone
ifdef USE_EIRENE
PREF_OBJDIR = couple_SOLPS-ITER
endif
```

```
# Extensions for object directories when various options are used
ifdef USE_IMPGYRO
EXT_IMPGYRO = .ig
else
ifdef SOLPS_MPI
EXT_MPI = .mpi
endif
endif
ifdef SOLPS_OPENMP
EXT_OPENMP = .openmp
endif
ifdef SOLPS_DEBUG
EXT_DEBUG = .debug
IMAS_AMNS_DEBUG = yes
else
IMAS_AMNS_DEBUG = no
endif
TOOLCHAIN = ${COMPILER}${EXT_OPENMP}${EXT_MPI}${EXT_IMPGYRO}${EXT_DEBUG}

# Directory where objectcode/binaries will be created
OBJDIR = ${SRCB2}/builds/${PREF_OBJDIR}.${HOST_NAME}.${TOOLCHAIN}
OBNDIR = ${SRCB2}/builds/standalone.${HOST_NAME}.${COMPILER}${EXT_DEBUG}

# If compiling with Eirene, look in default place for Eirene sources/lib
ifdef USE_EIRENE
  SRCEIR = ${SOLPSTOP}/modules/Eirene/src
  EIRDIR = ${SOLPSTOP}/modules/Eirene/builds/couple_SOLPS-ITER.${HOST_NAME}.${TOOLCHAIN}
endif
ifdef SOLPSTOP
  NCSDIR = ${SOLPSTOP}/scripts/nc2text_simple
  NCODIR = ${SOLPSTOP}/scripts/${HOST_NAME}.${COMPILER}${EXT_DEBUG}
  NCXDIR = ${SOLPSTOP}/scripts/${HOST_NAME}.${COMPILER}
endif

# Defines MPI_VERSION, which is the MPI version number
ifdef USE_IMPGYRO
  USE_MPI ?= -DUSE_MPI
  include ${OBJDIR}/mpiversion.mk
else
 ifdef SOLPS_MPI
  include ${OBJDIR}/mpiversion.mk
 endif
endif

ifeq ($(shell [ -e ${OBJDIR}/LISTOBJ ] && echo yes || echo no ),yes)
  include ${OBJDIR}/LISTOBJ
endif
include ${SRCB2}/config/compile
MAKES += ${SRCB2}/config/compile ${SRCB2}/config/config.${HOST_NAME}.${COMPILER}
ifeq ($(shell [ -e ${SRCB2}/config/config.${HOST_NAME}.${COMPILER}.local ] && echo yes || echo no
    ),yes)
  include ${SRCB2}/config/config.${HOST_NAME}.${COMPILER}.local
  MAKES += ${SRCB2}/config/config.${HOST_NAME}.${COMPILER}.local
endif

# Verify that some needed variables are defined
ifndef LD_MSCL
ifndef SOLPS_CPP
$(error LD_MSCL not defined!)
else
$(error LD_MSCL not defined. Run the install_dependencies script first.)
endif
endif

# Add external includes first
ifdef NCDIR
SOLPSINCLUDE += -I${NCDIR}/include
```

```
ifeq ($(UNAME),Darwin)
SOLPSINCLUDE += -I${NCFDIR}/include
endif
endif

ifdef USE_IMPGYRO
SOLPSINCLUDE += ${MPI_CPP}
else
ifdef SOLPS_MPI
SOLPSINCLUDE += ${MPI_CPP}
else
SOLPSINCLUDE += -I${SRCDIR}/mpi_dummy
endif
endif

ifdef MDSPLUS_DIR
ifndef LD_MDSPLUS
LD_MDSPLUS ?= -L${MDSPLUS_DIR}/lib -lMdsLib_client
endif
SOLPSINCLUDE += -I${MDSPLUS_DIR}/include
ifndef SOLPS_CPP
DEFINES += -DMDSPLUS
endif
endif

ifdef IMAS_PREFIX
ifeq ($(shell test ${IMAS_MINOR_VERSION} -ge 12; echo $$?),0)
ifeq ($(shell test ${GGD_MAJOR_VERSION} -eq 0; echo $$?),0)
$(warning Asking for an IMAS build but missing a GGD module: build may be incomplete)
endif
endif
endif

# If compiling with Paraview Catalyst
ifdef LD_CATALYST
SRCCAT = ${SRCDIR}/catalyst
PARAVIEW_MAJOR_VERSION ?= `paraview --version | cut -d '.' -f 1 | cut -d ' ' -f 3`
CMAKE_MAJOR_VERSION ?= `cmake --version | head -1 | cut -d ' ' -f 3 | cut -d '.' -f 1`
ifeq ($(shell test ${CMAKE_MAJOR_VERSION} -gt 2; echo $$?),0)
PARAVIEW_INCLUDE ?= $(shell paraview-config --cppflags)
else
PARAVIEW_INCLUDE ?= $(shell paraview-config --include)
endif
else
PARAVIEW_INCLUDE =
endif

# Local includes
BASEDIR = ${OBJDIR}
INCLOCAL = ${SRCDIR}/include.local
MODLOCAL = ${SRCDIR}/modules.local
SRCLOCAL = ${SRCB2}/src.local
SOLPS4 = ${SOLPSTOP}/modules/solps4-5/src/solps4_B2
EIR4 = ${SOLPSTOP}/modules/solps4-5/src/Eirene_modules
ifeq ($(shell [ -d ${MODLOCAL} ] && echo yes || echo no ),yes)
SOLPSINCLUDE += -I${MODLOCAL}
TAGSLIST += ${MODLOCAL}/*.F
endif
ifeq ($(shell [ -d ${SRCLOCAL} ] && echo yes || echo no ),yes)
SOLPSINCLUDE += -I${SRCLOCAL}
TAGSLIST += ${SRCLOCAL}/*.F
endif
ifeq ($(shell [ -d ${INCLOCAL} ] && echo yes || echo no ),yes)
SOLPSINCLUDE += -I${INCLOCAL}
TAGSLIST += ${INCLOCAL}/*.*
endif
SOLPSINCLUDE += -I${SRCDIR}/common -I${SRCDIR}/include
```

```
SOLPS4INCLUDE = -I${SOLPSTOP}/modules/solps4-5/src/B2_include
TAGSLIST += ${SRCDIR}/include/*.* ${SRCDIR}/common/*.* ${SRCDIR}/common/COUPLE/*.F ${SRCDIR}/*/*.
    F ${SRCDIR}/*/*.F90 ${DOCDIR}/*.xml ${DOCDIR}/*.py

ifdef USE_MPI
DPFINES += ${USE_MPI}
else
SOLPS4INCLUDE += -I${SOLPSTOP}/modules/solps4-5/src/Eirene_commons
endif
ifdef USE_OPENMP
DPFINES += ${USE_OPENMP}
endif
ifdef USE_IMPGYRO
DPFINES += ${USE_IMPGYRO}
endif
ifdef PERFMON
DEFINES += ${PERFMON}
endif
ifdef USE_EIRENE
DEFINES += ${USE_EIRENE}
endif
ifdef SOLPS_DEBUG
DEFINES += -DDBG
endif

# Switches to disable individual OpenMP regions, for debugging
# uncomment to activate
#
# It is not necessary to define all these flags to completely disable the
# OpenMP parallelization, in order to do that, just compile without OpenMP
# compiler options (ifort -qopenmp or similar)

ifdef SOLPS_OPENMP
#DPFINES += -DNO_OPENMP_B2XPFE
#DPFINES += -DNO_OPENMP_B2SIFRTF_LOOP1
#DPFINES += -DNO_OPENMP_B2SIFRTF_LOOP2
#DPFINES += -DNO_OPENMP_B2SIHS
#DPFINES += -DNO_OPENMP_B2SPCX
#DPFINES += -DNO_OPENMP_B2SPEL
#DPFINES += -DNO_OPENMP_B2SQCX
#DPFINES += -DNO_OPENMP_B2SQEL
#DPFINES += -DNO_OPENMP_B2SRDT
#DPFINES += -DNO_OPENMP_B2SRST
#DPFINES += -DNO_OPENMP_B2STBR
#DPFINES += -DNO_OPENMP_B2STCX
#DPFINES += -DNO_OPENMP_B2STEL
#DPFINES += -DNO_OPENMP_MYBLAS
#DPFINES += -DNO_OPENMP_SFILL
#DPFINES += -DNO_OPENMP_B2NEWS_LOOP1
#DPFINES += -DNO_OPENMP_B2NEWS_LOOP2
#DPFINES += -DNO_OPENMP_B2NEWS_LOOP3
#DPFINES += -DNO_OPENMP_B2NEWS_LOOP4
#DPFINES += -DNO_OPENMP_B2NEWS_UNDERSCORE_LOOP1
#DPFINES += -DNO_OPENMP_B2NEWS_UNDERSCORE_LOOP2
#DPFINES += -DNO_OPENMP_B2NEWS_UNDERSCORE_LOOP3
#DPFINES += -DNO_OPENMP_B2NEWS_UNDERSCORE_LOOP4
#DPFINES += -DNO_OPENMP_B2NPMO
endif

VHEAD =
ifeq ($(shell [ -d ${MODLOCAL} ] && echo yes || echo no ),yes)
VHEAD+=${MODLOCAL}:
endif
ifeq ($(shell [ -d ${SRCLOCAL} ] && echo yes || echo no ),yes)
VHEAD+=${SRCLOCAL}:
endif
# Required spacing variables for substitution
```

```
space :=
space +=
empty :=
#VPATH=$(subst $(space),$(empty),${VHEAD}${SRCDIR}/modules:${SRCDIR}/b2aux:${SRCDIR}/convert:${
    SRCDIR}/documentation:${SRCDIR}/driver:${SRCDIR}/equations:${SRCDIR}/input:${SRCDIR}/output:$
    {SRCDIR}/postprocessing:${SRCDIR}/preprocessing:${SRCDIR}/solvers:${SRCDIR}/sources:${SRCDIR
    }/transport:${SRCDIR}/utility:${SRCDIR}/b2plot:${SRCDIR}/user)
VPATH=${VHEAD}${SRCDIR}/modules:${SRCDIR}/b2aux:${SRCDIR}/convert:${SRCDIR}/documentation:${
    SRCDIR}/driver:${SRCDIR}/equations:${SRCDIR}/input:${SRCDIR}/output:${SRCDIR}/postprocessing:
    ${SRCDIR}/preprocessing:${SRCDIR}/solvers:${SRCDIR}/sources:${SRCDIR}/transport:${SRCDIR}/
    utility:${SRCDIR}/b2plot:${SRCDIR}/user
FPATH := ${VPATH}
VPATH += :${SRCDIR}/test
VPATH += :${SRCDIR}/ids
VPATH += :${SRCDIR}/ids/archive
FFPATH += :${SRCDIR}/ids
FFPATH += :${SRCDIR}/modules
FFPATH += :${SRCDIR}/user

MODLIST =
MODLISTF =
MODLISTF90 =
ifeq ($(shell [ -d ${MODLOCAL} ] && echo yes || echo no ),yes)
MODLIST += ${MODLOCAL}/*.F
MODLISTF += ${MODLOCAL}/*.F
endif
MODLIST += ${SRCDIR}/*/b2mod_*.F ${SRCDIR}/*/b2mod_*.F90 ${SRCDIR}/ids/*.F90
MODLISTF += ${SRCDIR}/*/b2mod_*.F
MODLISTF90 += ${SRCDIR}/*/b2mod_*.F90 ${SRCDIR}/ids/*.F90

ifeq ($(shell [ -d ${SOLPS4} ] && echo yes || echo no ),yes)
S4LIST = ${SOLPS4}/*.F
endif
ifeq ($(shell [ -d ${EIR4} ] && echo yes || echo no ),yes)
ifdef USE_EIRENE
E4LIST = ${EIR4}/precision.F ${EIR4}/parmmod.F ${EIR4}/braeir.F ${EIR4}/ccoupl.F ${EIR4}/clgin.F
    ${EIR4}/eirdiag.F ${EIR4}/ceirsrt.F
else
E4LIST = ${EIR4}/*.F
endif
endif

ifdef LD_CATALYST
ALLOBJS = ${OBJS:%.o=${OBJDIR}/%.o} ${OBJDIR}/cxxAdaptor.o
else
ALLOBJS = ${OBJS:%.o=${OBJDIR}/%.o}
endif

PROG_GE = b2pl.exe
PROG_GR = b2yg.exe b2yi.exe b2ym.exe b2yn.exe b2yp.exe b2yq.exe b2yr.exe
PROG_MN = b2mn.exe b2mnastra.exe
PROG_AM = b2ar.exe
PROG_XD = b2xd.exe
PROG_OE = b2ag.exe b2fu.exe b2ts.exe b2uf.exe b2ye.exe b2yt.exe b2ymb.exe b2yrp.exe b2ydm.exe
    b2plasmastate_inspect.exe calc_atomic_data.exe
PROG_CO = b2co.exe
PROG_OT = b2ah.exe b2ai.exe b2yi_gnuplot.exe b2yh.exe b2yv.exe b2fgmtry_mod.exe
PROG_90 = check_b2_output.exe
PROG_OP = b2op.exe
PROG_OQ = b2mn_opt.exe
PROG_MD = b2md.exe b2rd.exe
PROG_ID = b2_ual_write.exe b2_ual_rewrite.exe b2_ual_write_b2mod.exe
PROG_TT = test_shrink_label.exe
PROG_NC = nc2text_simple.exe
PROG_NR = nc_reduce.exe

EXCLUDELIST = ${patsubst %.exe, %\\.o, ${PROG_GE} ${PROG_GR} ${PROG_MN} ${PROG_AM} ${PROG_XD} ${
```

```
    PROG_OE} ${PROG_CO} ${PROG_OT} ${PROG_90} ${PROG_MD} ${PROG_OP} ${PROG_OQ} ${PROG_ID} ${
    PROG_TT}}
EXELIST = ${patsubst %.exe, %.o, ${PROG_GE} ${PROG_GR} ${PROG_MN} ${PROG_AM} ${PROG_XD} ${PROG_OE
    } ${PROG_CO} ${PROG_OT} ${PROG_MD} ${PROG_OP} ${PROG_OQ}}
EX90LIST = ${patsubst %.exe, %.o, ${PROG_90} ${PROG_ID}}

GEEXE = ${patsubst %.exe, ${OBJDIR}/%.exe, ${PROG_GE}}
GREXE = ${patsubst %.exe, ${OBJDIR}/%.exe, ${PROG_GR}}
XDEXE = ${patsubst %.exe, ${OBJDIR}/%.exe, ${PROG_XD}}
MNEXE = ${patsubst %.exe, ${OBJDIR}/%.exe, ${PROG_MN}}
AMEXE = ${patsubst %.exe, ${OBJDIR}/%.exe, ${PROG_AM}}
OEEXE = ${patsubst %.exe, ${OBJDIR}/%.exe, ${PROG_OE}}
COEXE = ${patsubst %.exe, ${OBJDIR}/%.exe, ${PROG_CO}}
OTEXE = ${patsubst %.exe, ${OBJDIR}/%.exe, ${PROG_OT}}
O9EXE = ${patsubst %.exe, ${OBJDIR}/%.exe, ${PROG_90}}
OPEXE = ${patsubst %.exe, ${OBJDIR}/%.exe, ${PROG_OP}}
OQEXE = ${patsubst %.exe, ${OBJDIR}/%.exe, ${PROG_OQ}}
MDEXE = ${patsubst %.exe, ${OBJDIR}/%.exe, ${PROG_MD}}
IDEXE = ${patsubst %.exe, ${OBJDIR}/%.exe, ${PROG_ID}}
TTEXE = ${patsubst %.exe, ${OBJDIR}/%.exe, ${PROG_TT}}
NCEXE = ${patsubst %.exe, ${NCODIR}/%.exe, ${PROG_NC}}
NREXE = ${patsubst %.exe, ${NCODIR}/%.exe, ${PROG_NR}}

.PHONY: DEFAULT NOPLOT ALL VERSION mods clean depend listobj tags echo local force test
    nc2text_simple nc2text

DEFAULT: VERSION ${MNEXE} ${AMEXE} ${OEEXE} ${COEXE} ${OTEXE} ${O9EXE}
ALL: VERSION ${MNEXE} ${AMEXE} ${OEEXE} ${COEXE} ${OTEXE} ${O9EXE} ${XDEXE}
NOPLOT: VERSION ${MNEXE} ${AMEXE} ${OEEXE} ${COEXE} ${OTEXE} ${O9EXE}
ifdef NCARG_ROOT
ifeq ($(strip ${GLI_HOME}),)
$(warning B2.5 graphical post-processing programs may not work because GLI_HOME is not defined.)
endif
DEFAULT: ${GEEXE} ${GREXE}
ALL: ${GEEXE} ${GREXE}
else
$(warning B2.5 graphical post-processing programs will not be compiled because NCARG_ROOT is not
    defined.)
endif
ifdef MDSPLUS_DIR
DEFAULT: ${MDEXE}
ALL: ${MDEXE}
NOPLOT: ${MDEXE}
endif
ifdef PAR_OPT
DEFAULT: ${OPEXE} ${OQEXE}
ALL: ${OPEXE} ${OQEXE}
NOPLOT: ${OPEXE} ${OQEXE}
endif
ifdef IMAS_VERSION
DEFAULT: ${IDEXE}
ALL: ${IDEXE}
ids: ${IDEXE}
NOPLOT: ${IDEXE}
endif
ifdef SOLPS_CPP
ifdef LD_NETCDF
DEFAULT: ${NCEXE} ${NREXE} nc2text
ALL: ${NCEXE} ${NREXE} nc2text
NOPLOT: ${NCEXE} ${NREXE} nc2text
endif
endif
MAIN: VERSION ${MNEXE}

ifdef USE_EIRENE
VPATH+=${SRCEIR}/modules:${SRCEIR}/interfaces/couple_SOLPS-ITER
MODLIST+=${SRCEIR}/modules/*.f ${SRCEIR}/modules/*.[fF]90 ${SRCEIR}/interfaces/couple_SOLPS-ITER/
```

```
    eirmod_*.f ${SRCEIR}/interfaces/couple_SOLPS-ITER/eirmod_*.F90
MODLISTF+=${SRCEIR}/modules/*.f ${SRCEIR}/interfaces/couple_SOLPS-ITER/eirmod_*.f
MODLISTF90+=${SRCEIR}/modules/*.[fF]90 ${SRCEIR}/interfaces/couple_SOLPS-ITER/eirmod_*.F90
MNEXTRA=${EIRDIR}/libeirene.a ${EIRDIR}/libgr_dummy.a
EIRLIBS=${MNEXTRA} ${LD_JSON}
DIMSDIR=${SRCDIR}/modules
ifeq ($(shell [ -s ${SRCDIR}/modules.local/b2mod_dimensions.F ] && echo yes || echo no),yes)
DIMSDIR=${SRCDIR}/modules.local
endif
DEFINES += -DDIMENSIONS_MODULE
else
# MNEXTRA=${EIRDIR}/eirmod_balanced_strategy.o ${EIRDIR}/eirmod_braeir.o ${EIRDIR}/eirmod_brascl.
    o ${EIRDIR}/eirmod_braspoi.o ${EIRDIR}/eirmod_cadgeo.o ${EIRDIR}/eirmod_cai.o ${EIRDIR}/
    eirmod_calstr_buffered.o ${EIRDIR}/eirmod_ccona.o ${EIRDIR}/eirmod_ccoupl.o ${EIRDIR}/
    eirmod_ccrm.o ${EIRDIR}/eirmod_cestim.o ${EIRDIR}/eirmod_cfplk.o ${EIRDIR}/eirmod_cgeom.o ${
    EIRDIR}/eirmod_cgrid.o ${EIRDIR}/eirmod_cgrptl.o ${EIRDIR}/eirmod_cinit.o ${EIRDIR}/
    eirmod_clast.o ${EIRDIR}/eirmod_clgin.o ${EIRDIR}/eirmod_clogau.o ${EIRDIR}/eirmod_comnnl.o $
    {EIRDIR}/eirmod_comprt.o ${EIRDIR}/eirmod_comsig.o ${EIRDIR}/eirmod_comsou.o ${EIRDIR}/
    eirmod_comspl.o ${EIRDIR}/eirmod_comusr.o ${EIRDIR}/eirmod_comxs.o ${EIRDIR}/eirmod_coutau.o
    ${EIRDIR}/eirmod_cpes.o ${EIRDIR}/eirmod_cpl3d.o ${EIRDIR}/eirmod_cplmsk.o ${EIRDIR}/
    eirmod_cplot.o ${EIRDIR}/eirmod_cpolyg.o ${EIRDIR}/eirmod_crand.o ${EIRDIR}/eirmod_crech.o ${
    EIRDIR}/eirmod_cref.o ${EIRDIR}/eirmod_crefmod.o ${EIRDIR}/eirmod_csdvi.o ${EIRDIR}/
    eirmod_csdvi_bgk.o ${EIRDIR}/eirmod_csdvi_cop.o ${EIRDIR}/eirmod_cspei.o ${EIRDIR}/
    eirmod_cspez.o ${EIRDIR}/eirmod_cstep.o ${EIRDIR}/eirmod_ctetra.o ${EIRDIR}/eirmod_ctext.o ${
    EIRDIR}/eirmod_ctrcei.o ${EIRDIR}/eirmod_ctrig.o ${EIRDIR}/eirmod_ctsurf.o ${EIRDIR}/
    eirmod_cupd.o ${EIRDIR}/eirmod_cvarusr.o ${EIRDIR}/eirmod_czt1.o ${EIRDIR}/eirmod_eirbra.o ${
    EIRDIR}/eirmod_eirdiag.o ${EIRMOD}/eirmod_improved_strategy.o ${EIRDIR}/eirmod_infcop.o ${
    EIRMOD}/eirmod_json.o ${EIRMOD}/eirmod_mcarlo.o ${EIRDIR}/eirmod_module_avltree.o ${EIRDIR}/
    eirmod_mpi.o ${EIRDIR}/eirmod_octree.o ${EIRDIR}/eirmod_openfile.o ${EIRDIR}/eirmod_openmp.o
    ${EIRDIR}/eirmod_parmmod.o ${EIRDIR}/eirmod_precision.o ${EIRDIR}/eirmod_reflec.o ${EIRDIR}/
    eirmod_refusr.o ${EIRDIR}/eirmod_sheath.o ${EIRDIR}/eirmod_solps.o ${EIRDIR}/eirmod_sputer.o
    ${EIRDIR}/eirmod_statis.o
# EXCLUDELIST += ${patsubst ${OBJDIR}/%.o, %.o, ${MNEXTRA} }
endif
ifdef LD_CATALYST
VPATH += :${SRCDIR}/catalyst
FFPATH += :${SRCDIR}/catalyst
endif

IDSMODS = ${PROG_ID:%.exe=${OBJDIR}/%.${MOD}}
MODULES = ${patsubst %.f90,%.o,${patsubst %.F90,%.o,${patsubst %.f,%.o,${patsubst %.F,%.o,$(
    notdir ${shell echo ${MODLIST} } ) } } } }
MODOBJS = ${MODULES:%.o=${OBJDIR}/%.o}
MODMODS = $(filter-out ${IDSMODS},${MODOBJS:%.o=%.${MOD}})
SOLPS4OBJS = ${patsubst ${SOLPS4}/%.F,${OBJDIR}/%.o,${shell echo ${S4LIST} } }
EIR4MODS = ${patsubst ${EIR4}/%.F,${OBJDIR}/%.${MOD},${shell echo ${E4LIST} } }
EIR4OBJS = ${patsubst ${EIR4}/%.F,${OBJDIR}/%.o,${shell echo ${E4LIST} } }

ifeq (${MOD},o)
LIBOBJS = $(filter-out ${MODOBJS},${ALLOBJS})
else
LIBOBJS = ${ALLOBJS}
endif

mods : ${MODMODS}

${DOCDIR}/b2cdci.F: ${DOCDIR}/b2input.xml ${DOCDIR}/b2cdci.py
        -cd ${DOCDIR}; ${PYTHON} b2cdci.py || echo "! Error building b2cdci.F from b2input.xml" >
            ${DOCDIR}/b2cdci.F

${DOCDIR}/b2cdcn.F: ${DOCDIR}/b2input.xml ${DOCDIR}/b2cdcn.py
        -cd ${DOCDIR}; ${PYTHON} b2cdcn.py || echo "! Error building b2cdcn.F from b2input.xml" >
            ${DOCDIR}/b2cdcn.F

ifdef USE_EIRENE
${OBJDIR}/libgr_dummy.a:
        ln -sf ${EIRDIR}/libgr_dummy.a ${OBJDIR}
```

```
${OBJDIR}/libeirene.a:
	ln -sf ${EIRDIR}/libeirene.a ${OBJDIR}

${OBJDIR}/eirmod_extrab25.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_extrab25.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_wneutrals.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_wneutrals.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_balanced_strategy.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_balanced_strategy.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_braeir.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_braeir.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_brascl.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_brascl.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_braspoi.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_braspoi.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_cadgeo.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_cadgeo.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_cai.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_cai.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_calstr_buffered.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_calstr_buffered.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_ccona.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_ccona.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_ccoupl.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_ccoupl.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_ccrm.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_ccrm.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_cestim.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_cestim.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_cfplk.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_cfplk.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_cgeom.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_cgeom.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_cgrid.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_cgrid.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_cgrptl.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_cgrptl.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_cinit.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_cinit.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_clast.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_clast.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_clgin.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_clgin.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_clogau.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_clogau.${MOD} ${OBJDIR}
```

```
${OBJDIR}/eirmod_clmsur.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_clmsur.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_comnnl.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_comnnl.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_comprt.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_comprt.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_comsig.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_comsig.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_comsou.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_comsou.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_comspl.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_comspl.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_comusr.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_comusr.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_comxs.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_comxs.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_coutau.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_coutau.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_cpes.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_cpes.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_cpl3d.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_cpl3d.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_cplmsk.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_cplmsk.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_cplot.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_cplot.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_cpolyg.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_cpolyg.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_crand.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_crand.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_crech.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_crech.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_cref.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_cref.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_crefmod.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_crefmod.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_csdvi.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_csdvi.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_csdvi_bgk.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_csdvi_bgk.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_csdvi_cop.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_csdvi_cop.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_cspei.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_cspei.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_cspez.${MOD}:
```

```
	@ln -sf ${EIRDIR}/eirmod_cspez.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_cstep.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_cstep.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_ctetra.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_ctetra.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_ctext.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_ctext.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_ctrcei.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_ctrcei.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_ctrig.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_ctrig.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_ctsurf.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_ctsurf.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_cupd.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_cupd.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_cvarusr.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_cvarusr.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_czt1.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_czt1.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_eirbra.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_eirbra.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_eirdiag.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_eirdiag.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_improved_strategy.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_improved_strategy.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_infcop.${MOD}: ${OBJDIR}/eirmod_cplot.${MOD} ${OBJDIR}/eirmod_openfile.${MOD} ${
    OBJDIR}/eirmod_refusr.${MOD}
	@ln -sf ${EIRDIR}/eirmod_infcop.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_json.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_json.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_mcarlo.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_mcarlo.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_module_avltree.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_module_avltree.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_mpi.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_mpi.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_octree.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_octree.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_openfile.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_openfile.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_openmp.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_openmp.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_parmmod.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_parmmod.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_precision.${MOD}:
```

```
	@ln -sf ${EIRDIR}/eirmod_precision.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_reflec.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_reflec.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_refusr.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_refusr.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_sheath.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_sheath.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_solps.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_solps.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_sputer.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_sputer.${MOD} ${OBJDIR}

${OBJDIR}/eirmod_statis.${MOD}:
	@ln -sf ${EIRDIR}/eirmod_statis.${MOD} ${OBJDIR}
else
${OBJDIR}/eirmod_balanced_strategy.${MOD}:
	touch ${OBJDIR}/eirmod_balanced_strategy.${MOD}

${OBJDIR}/eirmod_braeir.${MOD}:
	touch ${OBJDIR}/eirmod_braeir.${MOD}

${OBJDIR}/eirmod_ccona.${MOD}:
	touch ${OBJDIR}/eirmod_ccona.${MOD}

${OBJDIR}/eirmod_ccoupl.${MOD}:
	touch ${OBJDIR}/eirmod_ccoupl.${MOD}

${OBJDIR}/eirmod_cestim.${MOD}:
	touch ${OBJDIR}/eirmod_cestim.${MOD}

${OBJDIR}/eirmod_cinit.${MOD}:
	touch ${OBJDIR}/eirmod_cinit.${MOD}

${OBJDIR}/eirmod_clgin.${MOD}:
	touch ${OBJDIR}/eirmod_clgin.${MOD}

${OBJDIR}/eirmod_clogau.${MOD}:
	touch ${OBJDIR}/eirmod_clogau.${MOD}

${OBJDIR}/eirmod_comprt.${MOD}:
	touch ${OBJDIR}/eirmod_comprt.${MOD}

${OBJDIR}/eirmod_comusr.${MOD}:
	touch ${OBJDIR}/eirmod_comusr.${MOD}

${OBJDIR}/eirmod_coutau.${MOD}:
	touch ${OBJDIR}/eirmod_coutau.${MOD}

${OBJDIR}/eirmod_cpes.${MOD}:
	touch ${OBJDIR}/eirmod_cpes.${MOD}

${OBJDIR}/eirmod_ctrcei.${MOD}:
	touch ${OBJDIR}/eirmod_ctrcei.${MOD}

${OBJDIR}/eirmod_ctrig.${MOD}:
	touch ${OBJDIR}/eirmod_ctrig.${MOD}

${OBJDIR}/eirmod_eirbra.${MOD}:
	touch ${OBJDIR}/eirmod_eirbra.${MOD}

${OBJDIR}/eirmod_eirdiag.${MOD}:
	touch ${OBJDIR}/eirmod_eirdiag.${MOD}
```

```
${OBJDIR}/eirmod_extrab25.${MOD}:
	touch ${OBJDIR}/eirmod_extrab25.${MOD}

${OBJDIR}/eirmod_improved_strategy.${MOD}:
	touch ${OBJDIR}/eirmod_improved_strategy.${MOD}

${OBJDIR}/eirmod_infcop.${MOD}:
	touch ${OBJDIR}/eirmod_infcop.${MOD}

${OBJDIR}/eirmod_json.${MOD}:
	touch ${OBJDIR}/eirmod_json.${MOD}

${OBJDIR}/eirmod_mcarlo.${MOD}:
	touch ${OBJDIR}/eirmod_mcarlo.${MOD}

${OBJDIR}/eirmod_mpi.${MOD}:
	touch ${OBJDIR}/eirmod_mpi.${MOD}

${OBJDIR}/eirmod_openmp.${MOD}:
	touch ${OBJDIR}/eirmod_openmp.${MOD}

${OBJDIR}/eirmod_parmmod.${MOD}:
	touch ${OBJDIR}/eirmod_parmmod.${MOD}

${OBJDIR}/eirmod_precision.${MOD}:
	ln -s ${OBJDIR}/precision.${MOD} ${OBJDIR}/eirmod_precision.${MOD}

${OBJDIR}/eirmod_reflec.${MOD}:
	touch ${OBJDIR}/eirmod_reflec.${MOD}

${OBJDIR}/eirmod_refusr.${MOD}:
	touch ${OBJDIR}/eirmod_refusr.${MOD}

${OBJDIR}/eirmod_sheath.${MOD}:
	touch ${OBJDIR}/eirmod_sheath.${MOD}

${OBJDIR}/eirmod_solps.${MOD}:
	touch ${OBJDIR}/eirmod_solps.${MOD}

${OBJDIR}/eirmod_sputer.${MOD}:
	touch ${OBJDIR}/eirmod_sputer.${MOD}

${OBJDIR}/eirmod_statis.${MOD}:
	touch ${OBJDIR}/eirmod_statis.${MOD}

${OBJDIR}/eirmod_wneutrals.${MOD}:
	touch ${OBJDIR}/eirmod_wneutrals.${MOD}
endif

ifeq ($(COMPILER),gfortran)
ifeq ($(shell test ${GFORTRAN_MAJOR_VERSION} -ge 10; echo $$?),0)
${OBJDIR}/b2mod_mdsplus.o: b2mod_mdsplus.F
	@- /bin/rm -f ${OBJDIR}/b2mod_mdsplus.f ${OBJDIR}/b2mod_mdsplus.o ${OBJDIR}/b2mod_mdsplus
	    .${MOD}
ifeq ($(strip $(CPP)),)
	${FC} ${FCOPTS} ${FPOPTS} -fallow-argument-mismatch ${FFLAGSEXTRA} ${DEFINES} ${DPFINES}
	    ${EQUIVS} ${SOLPSINCLUDE} -c $<
else
ifeq ($(strip $(SED)),)
	-${CPP} ${DEFINES} ${DPFINES} ${EQUIVS} -P ${SOLPSINCLUDE} $< ${OBJDIR}/b2mod_mdsplus.f
else
	-${CPP} ${DEFINES} ${DPFINES} ${EQUIVS} -P ${SOLPSINCLUDE} $< | ${SED} > ${OBJDIR}/
	    b2mod_mdsplus.f
endif
	${FC} ${FCOPTS} ${FPOPTS} -fallow-argument-mismatch ${FFLAGSEXTRA} -c ${MODINCLUDE} ${
	    INCMODS} -o ${OBJDIR}/b2mod_mdsplus.o ${OBJDIR}/b2mod_mdsplus.f
```

```
endif
        @if [ -f b2mod_mdsplus.o ] ; then /bin/mv b2mod_mdsplus.o ${OBJDIR}/ ; fi
        @if [ -f b2mod_mdsplus.${MOD} ] ; then /bin/mv b2mod_mdsplus.${MOD} ${OBJDIR}/ ; fi

ifneq (${MOD},o)
${OBJDIR}/b2mod_mdsplus.${MOD}: b2mod_mdsplus.F
        @- /bin/rm -f ${OBJDIR}/b2mod_mdsplus.f ${OBJDIR}/b2mod_mdsplus.o ${OBJDIR}/b2mod_mdsplus
            .${MOD}
ifeq ($(strip $(CPP)),)
        ${FC} ${FCOPTS} ${FPOPTS} -fallow-argument-mismatch ${FFLAGSEXTRA} ${DEFINES} ${DPFINES}
            ${EQUIVS} ${SOLPSINCLUDE} -c $<
else
ifeq ($(strip $(SED)),)
        -${CPP} ${DEFINES} ${DPFINES} ${EQUIVS} -P ${SOLPSINCLUDE} $< ${OBJDIR}/b2mod_mdsplus.f
else
        -${CPP} ${DEFINES} ${DPFINES} ${EQUIVS} -P ${SOLPSINCLUDE} $< | ${SED} > ${OBJDIR}/
            b2mod_mdsplus.f
endif
        ${FC} ${FCOPTS} ${FPOPTS} -fallow-argument-mismatch ${FFLAGSEXTRA} -c ${MODINCLUDE} ${
            INCMODS} -o ${OBJDIR}/b2mod_mdsplus.o ${OBJDIR}/b2mod_mdsplus.f
endif
        @if [ -f b2mod_mdsplus.o ] ; then /bin/mv b2mod_mdsplus.o ${OBJDIR}/ ; fi
ifeq ($(strip $(LINK_MOD)),)
        @if [ -f b2mod_mdsplus.${MOD} ] ; then /bin/mv b2mod_mdsplus.${MOD} ${OBJDIR}/ ; fi
else
        @[ -f b2mod_mdsplus.${MOD} ] && [ ! -L b2mod_mdsplus.${MOD} ] && /bin/mv b2mod_mdsplus.${
            MOD} ${OBJDIR}/ && /bin/ln -s ${OBJDIR}/b2mod_mdsplus.${MOD} .
endif
endif
endif
endif

ifeq ($(COMPILER),ifort64)
ifdef SOLPS_OPENMP
ifdef USE_EIRENE
${OBJDIR}/b2ytdr.o : b2ytdr.F
        @- /bin/rm -f $*.f $*.o $*.${MOD}
ifeq ($(strip $(CPP)),)
        ${FC} ${FCOPTS} ${FPOPTS} -qoverride-limits ${FFLAGSEXTRA} ${DEFINES} ${DPFINES} ${EQUIVS
            } ${SOLPSINCLUDE} -c $<
else
ifeq ($(strip $(SED)),)
        -${CPP} ${DEFINES} ${DPFINES} ${EQUIVS} -P ${SOLPSINCLUDE} $< $*.f
else
        -${CPP} ${DEFINES} ${DPFINES} ${EQUIVS} -P ${SOLPSINCLUDE} $< | ${SED} > $*.f
endif
        ${FC} ${FCOPTS} ${FPOPTS} -qoverride-limits ${FFLAGSEXTRA} -c ${MODINCLUDE} ${INCMODS} -
            module ${OBJDIR} -o $*.o $*.f
endif
endif
endif
endif

${MNEXE}: ${OBJDIR}/%.exe: ${OBJDIR}/%.o ${OBJDIR}/libb2.a ${MNEXTRA} ${MAKES}
        ${LD} ${LDOPTS} ${LPOPTS} ${FFLAGSEXTRA} -o $@ ${OBJDIR}/$*.o ${OBJDIR}/libb2.a ${EIRLIBS
            } ${IMASLIBS} ${PLLIBES} ${LDLIBES} ${LD_CATALYST} ${LDOPTSend}

${AMEXE}: ${OBJDIR}/%.exe: ${OBJDIR}/%.o ${OBJDIR}/libb2.a ${MAKES}
        ${LD} ${LDOPTS} ${LPOPTS} ${FFLAGSEXTRA} -o $@ ${OBJDIR}/$*.o ${OBJDIR}/libb2.a ${
            IMASLIBS} ${LDLIBES} ${LDOPTSend}

${OEEXE}: ${OBJDIR}/%.exe: ${OBJDIR}/%.o ${OBJDIR}/libb2.a ${MNEXTRA} ${MAKES}
        ${LD} ${LDOPTS} ${LPOPTS} ${FFLAGSEXTRA} -o $@ ${OBJDIR}/$*.o ${OBJDIR}/libb2.a ${EIRLIBS
            } ${LDLIBES} ${LDOPTSend}

${COEXE}: ${OBJDIR}/%.exe: ${OBJDIR}/%.o ${OBJDIR}/libb2.a ${MNEXTRA} ${MAKES}
        ${LD} ${LDOPTS} ${LPOPTS} ${FFLAGSEXTRA} -o $@ ${OBJDIR}/$*.o ${OBJDIR}/libb2.a ${EIRLIBS
```

```
            } ${PLLIBES} ${LDLIBES} ${LDOPTSend}

${OPEXE}: ${OBJDIR}/%.exe: ${OBJDIR}/%.o ${OBJDIR}/libb2.a ${MAKES}
        ${LD} ${LDOPTS} ${LPOPTS} ${FFLAGSEXTRA} -o $@ ${OBJDIR}/$*.o ${OBJDIR}/libb2.a ${LDLIBES
            } ${LDOPTSend}

${OQEXE}: ${OBJDIR}/%.exe: ${OBJDIR}/%.o ${OBJDIR}/libb2.a ${MNEXTRA} ${MAKES}
        ${LD} ${LDOPTS} ${LPOPTS} ${FFLAGSEXTRA} -o $@ ${OBJDIR}/$*.o ${OBJDIR}/libb2.a ${EIRLIBS
            } ${LDLIBES} ${LD_CATALYST} ${LDOPTSend}

${OTEXE}: ${OBJDIR}/%.exe: ${OBJDIR}/%.o ${OBJDIR}/libb2.a ${MAKES}
        ${LD} ${LDOPTS} ${LPOPTS} ${FFLAGSEXTRA} -o $@ ${OBJDIR}/$*.o ${OBJDIR}/libb2.a ${LDLIBES
            } ${LDOPTSend}

${O9EXE}: ${OBJDIR}/%.exe: ${OBJDIR}/%.o ${OBJDIR}/libb2.a ${MNEXTRA} ${MAKES}
        ${LD} ${LDOPTS} ${LPOPTS} ${FFLAGSEXTRA} -o $@ ${OBJDIR}/$*.o ${OBJDIR}/libb2.a ${EIRLIBS
            } ${LDLIBES}

${GEEXE}: ${OBJDIR}/%.exe: ${OBJDIR}/%.o ${OBJDIR}/libb2.a ${MNEXTRA} ${MAKES}
        ${LD} ${LDOPTS} ${LPOPTS} ${FFLAGSEXTRA} -o $@ ${OBJDIR}/$*.o ${OBJDIR}/libb2.a ${EIRLIBS
            } ${PLLIBES} ${GRLIBES} ${LDLIBES} ${LDOPTSend}

${GREXE}: ${OBJDIR}/%.exe: ${OBJDIR}/%.o ${OBJDIR}/libb2.a ${MAKES}
        ${LD} ${LDOPTS} ${LPOPTS} ${FFLAGSEXTRA} -o $@ ${OBJDIR}/$*.o ${OBJDIR}/libb2.a ${GRLIBES
            } ${LDLIBES} ${LDOPTSend}

${XDEXE}: ${OBJDIR}/%.exe: ${OBJDIR}/%.o ${OBJDIR}/libb2.a ${MNEXTRA} ${OBJDIR}/libsolps4.a ${
    MAKES}
        ${LD} ${LDOPTS} ${LPOPTS} ${FFLAGSEXTRA} -o $@ ${OBJDIR}/$*.o ${OBJDIR}/libb2.a ${EIRLIBS
            } ${OBJDIR}/libsolps4.a ${LDLIBES} ${LDEXTRA} ${LDOPTSend}

${MDEXE}: ${OBJDIR}/%.exe: ${OBJDIR}/%.o ${OBJDIR}/libb2.a ${MNEXTRA} ${MAKES}
        ${LD} ${LDOPTS} ${LPOPTS} ${FFLAGSEXTRA} -o $@ ${OBJDIR}/$*.o ${OBJDIR}/libb2.a ${EIRLIBS
            } ${LDLIBES} ${LD_MDSPLUS} ${LDOPTSend}

${IDEXE}: ${OBJDIR}/%.exe: ${OBJDIR}/%.o ${OBJDIR}/libb2.a ${MNEXTRA} ${MAKES}
        ${LD} ${LDOPTS} ${LPOPTS} ${FFLAGSEXTRA} -o $@ ${OBJDIR}/$*.o ${OBJDIR}/libb2.a ${EIRLIBS
            } ${IMASLIBS} ${PLLIBES} ${LDLIBES} ${LD_CATALYST} ${LDOPTSend}

${TTEXE}: ${OBJDIR}/%.exe: ${OBJDIR}/%.o ${OBJDIR}/libb2.a ${MAKES}
        ${LD} ${LDOPTS} ${LPOPTS} ${FFLAGSEXTRA} -o $@ ${OBJDIR}/$*.o ${OBJDIR}/libb2.a ${LDLIBES
            } ${LDOPTSend}

${NCEXE}: ${NCODIR}/%.exe: ${NCODIR}/%.o ${MAKES}
ifdef LD_NETCDF
ifndef COMPILER_REDEF
        ${LN} ${LDOPTS} ${FFLAGSEXTRA} -o $@ ${NCODIR}/$*.o ${LD_NETCDF}
        @-ln -sf $@ ${NCXDIR}/$*
        @-ln -sf ${NCXDIR}/nc2text_simple ${NCXDIR}/nc2text
else
        $(warning Compiler type was redefined: skipping!)
endif
else
        $(warning nc2text_simple compilation skipped because netCDF library not present)
endif

${NREXE}: ${NCODIR}/%.exe: ${NCODIR}/%.o ${OBNDIR}/cdf_routines.o ${OBNDIR}/chcase.o ${OBNDIR}/
    ifill.o ${OBNDIR}/isadigit.o ${OBNDIR}/machsfr.o ${OBNDIR}/nagsubst.o ${OBNDIR}/open_file.o $
    {OBNDIR}/prgend.o ${OBNDIR}/prgini.o ${OBNDIR}/prvrt.o ${OBNDIR}/prvrti.o ${OBNDIR}/sfill.o $
    {OBNDIR}/streql.o ${OBNDIR}/sysend.o ${OBNDIR}/sysini.o ${OBNDIR}/xerrab.o ${OBNDIR}/xertst.o
     ${MAKES}
ifdef LD_NETCDF
ifndef COMPILER_REDEF
        ${LN} ${LDOPTS} ${FFLAGSEXTRA} -o $@ ${NCODIR}/$*.o ${OBNDIR}/b2mod_ipmain.o ${OBNDIR}/
            b2mod_lwimai.o ${OBNDIR}/b2mod_lwmain.o ${OBNDIR}/b2mod_math.o ${OBNDIR}/b2mod_openmp
            .o ${OBNDIR}/b2mod_stack.o ${OBNDIR}/b2mod_subsys.o ${OBNDIR}/b2mod_xerset.o ${OBNDIR
            }/cdf_routines.o ${OBNDIR}/chcase.o ${OBNDIR}/ifill.o ${OBNDIR}/isadigit.o ${OBNDIR}/
```

```
            machsfr.o ${OBNDIR}/nagsubst.o ${OBNDIR}/open_file.o ${OBNDIR}/prgend.o ${OBNDIR}/
            prgini.o ${OBNDIR}/prvrt.o ${OBNDIR}/prvrti.o ${OBNDIR}/sfill.o ${OBNDIR}/streql.o ${
            OBNDIR}/sysend.o ${OBNDIR}/sysini.o ${OBNDIR}/xerrab.o ${OBNDIR}/xertst.o ${LD_NETCDF
            } ${LD_NAG}
ifndef SOLPS_DEBUG
        @-ln -sf $@ ${NCXDIR}/$*
endif
else
        $(warning Compiler type was redefined: skipping!)
endif
else
        $(warning nc_reduce compilation skipped because netCDF library not present)
endif

${OBJDIR}/libb2.a: ${LIBOBJS} ${SRCDIR}/include/git_version_B25.h ${DOCDIR}/b2cdci.F ${DOCDIR}/
    b2cdcn.F
        @${BLD} $@ ${LIBOBJS}

test:   ${TTEXE}

nc2text: ${NCEXE}

nc2text_simple: ${NCEXE}

nc_reduce: ${NREXE}

${NCODIR}/nc2text_simple.o: ${NCSDIR}/nc2text_simple.F90
ifdef LD_NETCDF
ifndef COMPILER_REDEF
        @-mkdir -p ${NCODIR}
        -${CPP} ${DEFINES} ${EQUIVS} -P ${SOLPSINCLUDE} $< $*.F90
        ${FN} ${FCOPTS} ${FFLAGSEXTRA} -c -o $*.o $*.F90
endif
endif

${NCODIR}/nc_reduce.o: ${NCSDIR}/nc_reduce.F90 ${OBNDIR}/b2mod_ipmain.${MOD} ${OBNDIR}/
    b2mod_lwimai.${MOD} ${OBNDIR}/b2mod_lwmain.${MOD} ${OBNDIR}/b2mod_math.${MOD} ${OBNDIR}/
    b2mod_openmp.${MOD} ${OBNDIR}/b2mod_stack.${MOD} ${OBNDIR}/b2mod_subsys.${MOD} ${OBNDIR}/
    b2mod_types.${MOD} ${OBNDIR}/b2mod_xerset.${MOD}
ifdef LD_NETCDF
ifndef COMPILER_REDEF
        @-mkdir -p ${NCODIR}
        -${CPP} ${DEFINES} ${EQUIVS} -P ${SOLPSINCLUDE} $< $*.F90
        ${FN} ${FCOPTS} ${FFLAGSEXTRA} -c ${MODINCLUDE} ${INCMODN} -o $*.o $*.F90
endif
endif

${SOLPS4OBJS}: ${OBJDIR}/%.o: ${SOLPS4}/%.F
        @- /bin/rm -f ${OBJDIR}/$*.f ${OBJDIR}/$*.o
ifeq ($(strip $(CPP)),)
        ${FC} ${FCOPTS} ${FPOPTS} ${FFLAGSEXTRA} ${DEFINES} ${DPFINES} ${EQUIVS} ${SOLPSINCLUDE}
            ${SOLPS4INCLUDE} -c $<
else
ifeq ($(strip $(SED)),)
        -${CPP} ${DEFINES} ${DPFINES} -DSOLPS_ITER -DDIMENSIONS_MODULE ${EQUIVS} -P ${
            SOLPSINCLUDE} ${SOLPS4INCLUDE} $< ${OBJDIR}/$*.f
else
        -${CPP} ${DEFINES} ${DPFINES} -DSOLPS_ITER -DDIMENSIONS_MODULE ${EQUIVS} -P ${
            SOLPSINCLUDE} ${SOLPS4INCLUDE} $< | ${SED} > ${OBJDIR}/$*.f
endif
        ${FC} ${FCOPTS} ${FPOPTS} ${FFLAGSEXTRA} -c ${MODINCLUDE} ${INCMODS} ${OBJDEST} ${OBJDIR
            }/$*.f
endif
        @if [ -f $*.o ] ; then /bin/mv $*.o ${OBJDIR}/ ; fi

${EIR4OBJS}: ${OBJDIR}/%.o: ${EIR4}/%.F
        ${FC} ${FCOPTS} ${FPOPTS} ${FFLAGSEXTRA} ${DEFINES} ${DPFINES} -DDIMENSIONS_MODULE ${
```

```
            EQUIVS} ${SOLPSINCLUDE} ${OBJDEST} -c $?

${EIR4MODS}: ${OBJDIR}/%.${MOD}: ${EIR4}/%.F
        @- /bin/rm -f ${OBJDIR}/$*.f ${OBJDIR}/$*.o
ifeq ($(strip $(CPP)),)
        ${FC} ${FCOPTS} ${FPOPTS} ${FFLAGSEXTRA} ${DEFINES} ${DPFINES} ${EQUIVS} ${SOLPSINCLUDE}
            ${SOLPS4INCLUDE} -c $<
else
ifeq ($(strip $(SED)),)
        -${CPP} ${DEFINES} ${DPFINES} -DDIMENSIONS_MODULE ${EQUIVS} -P ${SOLPSINCLUDE} ${
            SOLPS4INCLUDE} $< ${OBJDIR}/$*.f
else
        -${CPP} ${DEFINES} ${DPFINES} -DDIMENSIONS_MODULE ${EQUIVS} -P ${SOLPSINCLUDE} ${
            SOLPS4INCLUDE} $< | ${SED} > ${OBJDIR}/$*.f
endif
        ${FC} ${FCOPTS} ${FPOPTS} ${FFLAGSEXTRA} -c ${MODINCLUDE} ${INCMODS} ${SOLPS4INCLUDE} ${
            OBJDEST} ${OBJDIR}/$*.f
        @touch ${OBJDIR}/$*.${MOD}
endif

ifdef USE_EIRENE
${OBJDIR}/libsolps4.a: ${SOLPS4OBJS} ${EIR4MODS}
else
${OBJDIR}/libsolps4.a: ${SOLPS4OBJS}
endif
        ${BLD} $@ ${SOLPS4OBJS}

${OBJDIR}/b2rw.o: ${OBJDIR}/eirdiag.${MOD}
${OBJDIR}/default.o: ${OBJDIR}/ceirsrt.${MOD}

ifneq (${MOD},o)
${OBJDIR}/adsp.${MOD}: ${OBJDIR}/cadgeo.${MOD} ${OBJDIR}/clogau.${MOD} ${OBJDIR}/comusr.${MOD} ${
    OBJDIR}/cpes.${MOD} ${OBJDIR}/ctrcei.${MOD} ${OBJDIR}/comprt.${MOD}
${OBJDIR}/avltree.${MOD}: ${OBJDIR}/ccona.${MOD}
${OBJDIR}/caprmc.${MOD}: ${OBJDIR}/cgrid.${MOD} ${OBJDIR}/comxs.${MOD} ${OBJDIR}/comsou.${MOD}
${OBJDIR}/ccflux.${MOD}: ${OBJDIR}/ctrig.${MOD} ${OBJDIR}/cgeom.${MOD}
${OBJDIR}/ccrm.${MOD}: ${OBJDIR}/cestim.${MOD} ${OBJDIR}/csdvi.${MOD} ${OBJDIR}/czt1.${MOD} ${
    OBJDIR}/photon.${MOD}
${OBJDIR}/comxs.${MOD}: ${OBJDIR}/cupd.${MOD}
${OBJDIR}/cpes.${MOD}: ${OBJDIR}/comprt.${MOD} ${OBJDIR}/ctrcei.${MOD} ${OBJDIR}/eirmod_precision
    .${MOD}
${OBJDIR}/cupd.${MOD}: ${OBJDIR}/comsig.${MOD}
${OBJDIR}/eirdiag.${MOD}: ${OBJDIR}/precision.${MOD} ${OBJDIR}/parmmod.${MOD} ${OBJDIR}/braeir.${
    MOD} ${OBJDIR}/ccoupl.${MOD} ${OBJDIR}/clgin.${MOD}
${OBJDIR}/eirgrid_lib.${MOD}: ${OBJDIR}/eirmap.${MOD}
endif
${OBJDIR}/adsp.o: ${OBJDIR}/cadgeo.o ${OBJDIR}/clogau.o ${OBJDIR}/comusr.o ${OBJDIR}/cpes.o ${
    OBJDIR}/ctrcei.o ${OBJDIR}/comprt.o
${OBJDIR}/avltree.o: ${OBJDIR}/ccona.o
${OBJDIR}/caprmc.o: ${OBJDIR}/cgrid.o ${OBJDIR}/comxs.o ${OBJDIR}/comsou.o
${OBJDIR}/ccflux.o: ${OBJDIR}/ctrig.o ${OBJDIR}/cgeom.o
${OBJDIR}/ccrm.o: ${OBJDIR}/cestim.o ${OBJDIR}/csdvi.o ${OBJDIR}/czt1.o ${OBJDIR}/photon.o
${OBJDIR}/comxs.o: ${OBJDIR}/cupd.o
${OBJDIR}/cpes.o: ${OBJDIR}/comprt.o ${OBJDIR}/ctrcei.o ${OBJDIR}/eirmod_precision.o
${OBJDIR}/cupd.o: ${OBJDIR}/comsig.o
${OBJDIR}/eirdiag.o: ${OBJDIR}/precision.o ${OBJDIR}/parmmod.o ${OBJDIR}/braeir.o ${OBJDIR}/
    ccoupl.o ${OBJDIR}/clgin.o
${OBJDIR}/eirgrid_lib.o: ${OBJDIR}/eirmap.o

# target 'clean' cleans up the directory.
clean :
        -mkdir ${OBJDIR}/.delete
        -mv -i ${OBJDIR}/*.o ${OBJDIR}/*.f ${OBJDIR}/*.f90 ${OBJDIR}/*.a ${OBJDIR}/*.exe ${SRCDIR
            }/include/git_version_B25.h ${OBJDIR}/LISTOBJ ${OBJDIR}/dependencies ${OBJDIR}/
            mpiversion.mk ${OBJDIR}/.delete >& /dev/null
ifneq (${MOD},o)
        -mv -i ${OBJDIR}/*.${MOD} ${OBJDIR}/.delete >& /dev/null
```

```
endif
ifdef SOLPSTOP
ifdef LD_NETCDF
	-rm -f ${NCODIR}/*.f90 ${NCODIR}/*.o ${NCODIR}/*.exe
endif
endif
	-rm -rf ${OBJDIR}/.delete &

depend: ${OBJDIR}/LISTOBJ ${B2OBJS:.o=.F} ${B2F90OBJS:.o=.F90} ${EXELIST:.o=.F} ${EX90LIST:.o=.
    F90}
	@`which makedepend` -p'$${OBJDIR}/' ${DEFINES} -f- ${SOLPSINCLUDE} $^ | \
	sed 's,^$${OBJDIR}/[^ ][^ ]*/,\$${OBJDIR}/,' | \
	sed 's,: ${SOLPSTOP},: $${SOLPSTOP},' > ${OBJDIR}/dependencies
	@echo '# 1' >> ${OBJDIR}/dependencies
ifneq (${MOD},o)
	@`which makedepend` -p'$${OBJDIR}/' ${DEFINES} -f- ${SOLPSINCLUDE} ${MODLIST} -o.${MOD} |
	    \
	sed 's,^$${OBJDIR}/[^ ][^ ]*/,\$${OBJDIR}/,' | \
	sed 's,: ${SOLPSTOP},: $${SOLPSTOP},' >> ${OBJDIR}/dependencies
	@echo '# 2' >> ${OBJDIR}/dependencies
endif
ifeq ($(shell [ -d ${SRCLOCAL} ] && echo yes || echo no ),yes)
	@egrep -aiH '^ {6,}use ' ${SRCLOCAL}/*.F | grep -v 'IGNORE' | tr , ' ' | awk '{sub("\\.F
	    :",".o:",$$1);sub("^.*/","$${OBJDIR}/",$$1); print $$1,"$${OBJDIR}/"tolower($$3)".${
	    MOD}"}' >> ${OBJDIR}/dependencies
	@echo '# 3' >> ${OBJDIR}/dependencies
endif
	@egrep -aiH '^ {6,}use ' ${SRCDIR}/*/*.F | grep -v 'IGNORE' | tr , ' ' | awk '{sub("\\.F
	    :",".o:",$$1);sub("^.*/","$${OBJDIR}/",$$1); print $$1,"$${OBJDIR}/"tolower($$3)".${
	    MOD}"}' >> ${OBJDIR}/dependencies
	@echo '# 4a' >> ${OBJDIR}/dependencies
	@egrep -aiH '^ {0,}use ' ${SRCDIR}/*/*.F90 | grep -v 'IGNORE' | tr , ' ' | awk '{sub("\\.
	    F90:",".o:",$$1);sub("^.*/","$${OBJDIR}/",$$1); print $$1,"$${OBJDIR}/"tolower($$3)".
	    ${MOD}"}' >> ${OBJDIR}/dependencies
	@echo '# 4b' >> ${OBJDIR}/dependencies
ifneq (${MOD},o)
	@egrep -aiH '^ {6,}use ' ${MODLISTF} | grep -v 'IGNORE' | tr , ' ' | awk '{sub("\\.F:",".
	    ${MOD}:",$$1);sub("\\.f:",".${MOD}:",$$1);sub("^.*/","$${OBJDIR}/",$$1); print $$1,"
	    $${OBJDIR}/"tolower($$3)".${MOD}"}' >> ${OBJDIR}/dependencies
	@echo '# 5a' >> ${OBJDIR}/dependencies
ifdef MODLISTF90
	@egrep -aiH '^ {0,}use ' ${MODLISTF90} | grep -v 'IGNORE' | tr , ' ' | awk '{sub("\\.F90
	    :",".${MOD}:",$$1);sub("\\.f90:",".${MOD}:",$$1);sub("^.*/","$${OBJDIR}/",$$1); print
	     $$1,"$${OBJDIR}/"tolower($$3)".${MOD}"}' >> ${OBJDIR}/dependencies
	@echo '# 5b' >> ${OBJDIR}/dependencies
endif
endif

tags:
	rm -f ${SRCB2}/TAGS ; ${MAKETAGS} ${SRCB2}/TAGS ${TAGSLIST} || touch ${SRCB2}/TAGS

listobj: ${OBJDIR}/dependencies ${DOCDIR}/b2cdci.F ${DOCDIR}/b2cdcn.F
ifdef USE_EIRENE
	@rm -f ${OBJDIR}/LISTOBJ; touch ${OBJDIR}/LISTOBJ; l="OBJS ="; \
	for d in `echo "${FPATH}" | tr : \ `; do \
	        l="$$l `(cd $$d > /dev/null; echo *.F)`"; \
	done; \
	for d in `echo "${FFPATH}" | tr : \ `; do \
	        l="$$l `(cd $$d > /dev/null; echo *.F90)`"; \
	done; \
	E="-e 's/ \*\.F90//g' -e 's/ \*\.F//g' -e 's/ eirmod_\*\.F90//g' -e 's/eirmod_\*\.f//g' -
	    e 's/\.F90/\.o/g' -e 's/\.F/\.o/g' -e 's/\.f/\.o/g'" ; for f in ${EXCLUDELIST}; do \
	        E="$$E -e 's/ $$f//'"; \
	done; \
	echo "$$l" | eval sed "$$E" > ${OBJDIR}/LISTOBJ
else
	@rm -f ${OBJDIR}/LISTOBJ; touch ${OBJDIR}/LISTOBJ; l="OBJS ="; \
```

```
        for d in `echo "${FPATH}" | tr : \ `; do \
                l="$$l `(cd $$d > /dev/null; echo *.F)`"; \
        done; \
        for d in `echo "${FFPATH}" | tr : \ `; do \
                l="$$l `(cd $$d > /dev/null; echo *.F90)`"; \
        done; \
        E="-e 's/ \*\.F90//g' -e 's/ \*\.F//g' -e 's/\.F90/\.o/g' -e 's/\.F/\.o/g'" ; for f in ${
            EXCLUDELIST}; do \
                E="$$E -e 's/ $$f//'"; \
        done; \
        echo "$$l" | eval sed "$$E" > ${OBJDIR}/LISTOBJ
endif
        @ll="B2OBJS ="; \
        for d in `echo "${FPATH}" | tr : \ `; do \
                ll="$$ll `(cd $$d > /dev/null; echo *.F)`"; \
        done; \
        E="-e 's/ \*\.F//g' -e 's/\.F/\.o/g'" ; for f in ${EXCLUDELIST}; do \
                E="$$E -e 's/ $$f//'"; \
        done; \
        echo "$$ll" | eval sed "$$E" >> ${OBJDIR}/LISTOBJ
        @lll="B2F90OBJS ="; \
        for d in `echo "${FFPATH}" | tr : \ `; do \
                lll="$$lll `(cd $$d > /dev/null; echo *.F90)`"; \
        done; \
        E="-e 's/ \*\.F90//g' -e 's/\.F90/\.o/g'" ; for f in ${EXCLUDELIST}; do \
                E="$$E -e 's/ $$f//'"; \
        done; \
        echo "$$lll" | eval sed "$$E" >> ${OBJDIR}/LISTOBJ

${OBJDIR}/LISTOBJ: listobj

VERSION: ${SRCDIR}/include/git_version_B25.h

${SRCDIR}/include/git_version_B25.h: force
        @echo "      character*32 :: git_version_B25 =" > ${SRCDIR}/include/git_version_new.h
        @echo "     . ''git describe --tags --dirty --always | cut -c 1-32`'" >> ${SRCDIR}/
            include/git_version_new.h
ifdef SOLPS_CPP
        @echo "      character*32 :: git_version_ADAS =" >> ${SRCDIR}/include/git_version_new.h
        @echo "     . ''( cd $${SOLPSTOP}/modules/adas ; git describe --tags --dirty --always |
            cut -c 1-32 )`'" >> ${SRCDIR}/include/git_version_new.h
        @echo "      character*32 :: git_version_SOLPS =" >> ${SRCDIR}/include/git_version_new.h
        @echo "     . ''( cd $${SOLPSTOP} ; git describe --tags --dirty --always | cut -c 1-32 )
            `'" >> ${SRCDIR}/include/git_version_new.h
else
        @echo "      character*32 :: git_version_ADAS = '0.0.0-0-g0000000'" >> ${SRCDIR}/include/
            git_version_new.h
        @echo "      character*32 :: git_version_SOLPS = '0.0.0-0-g0000000'" >> ${SRCDIR}/include
            /git_version_new.h
endif
        @if cmp -s ${SRCDIR}/include/git_version_new.h ${SRCDIR}/include/git_version_B25.h; then
            rm ${SRCDIR}/include/git_version_new.h; else mv ${SRCDIR}/include/git_version_new.h $
            {SRCDIR}/include/git_version_B25.h; fi

${OBJDIR}/dependencies: ${SRCDIR}/modules/.new_modules
ifeq ($(shell [ -d ${OBJDIR} ] && echo yes || echo no ),no)
        -mkdir -p ${OBJDIR}
endif
        touch ${OBJDIR}/dependencies
        ${MAKE} tags
        ${MAKE} VERSION
        ${MAKE} local
        ${MAKE} listobj
        ${MAKE} depend

include ${OBJDIR}/dependencies
ifeq ($(shell [ -e ${SRCB2}/config/dependencies.local ] && echo yes || echo no ),yes)
```

```
include ${SRCB2}/config/dependencies.local
endif

ifeq ($(COMPILER),g77)
${OBJDIR}/b2stbc.o : b2stbc.F
	@- /bin/rm -f ${OBJDIR}/b2stbc.f ${OBJDIR}/b2stbc.o
ifeq ($(strip $(CPP)),)
	${FC} ${FCOPTS} ${FPOPTS} ${FFLAGSEXTRA} ${DEFINES} ${DPFINES} ${EQUIVS} ${SOLPSINCLUDE}
	    -c $<
else
ifeq ($(strip $(SED)),)
	-${CPP} ${DEFINES} ${DPFINES} ${EQUIVS} -P ${SOLPSINCLUDE} $< ${OBJDIR}/b2stbc.f
else
	-${CPP} ${DEFINES} ${DPFINES} ${EQUIVS} -P ${SOLPSINCLUDE} $< | ${SED} > ${OBJDIR}/b2stbc
	    .f
endif
	${FC} ${FCOPTS} ${FPOPTS} ${FFLAGSEXTRA} -O1 -c ${INCMOD}${OBJDIR} ${OBJDIR}/b2stbc.f
endif
	@if [ -f b2stbc.o ] ; then /bin/mv b2stbc.o ${OBJDIR}/ ; fi
endif

echo:
	@echo LD_CATALYST=${LD_CATALYST}
	@echo SOLPSINCLUDE=${SOLPSINCLUDE}
	@echo LDLIBES=${LDLIBES}
	@echo DEFINES=${DEFINES}
	@echo DPFINES=${DPFINES}
	@echo EQUIVS=${EQUIVS}
	@echo VPATH=${VPATH}
	@echo FPATH=${FPATH}
	@echo OBJDIR=${OBJDIR}
	@echo OBJS=${OBJS}
	@echo SOLPS4OBJS=${SOLPS4OBJS}
	@echo EIR4MODS=${EIR4MODS}
	@echo EIR4OBJS=${EIR4OBJS}
	@echo MODLIST=${MODLIST}
	@echo MODLISTF=${MODLISTF}
	@echo MODLISTF90=${MODLISTF90}
	@echo MODULES=${MODULES}
	@echo MODOBJS=${MODOBJS}
	@echo MODMODS=${MODMODS}
	@echo MODLOCAL=${MODLOCAL}
	@echo $(filter-out ${MODOBJS},${ALLOBJS})
	@echo EXCLUDELIST=${EXCLUDELIST}
	@echo EXELIST=${EXELIST}
	@echo EX90LIST=${EX90LIST}
	@echo GREXE=${GREXE}
	@echo MNEXE=${MNEXE}
	@echo AMEXE=${AMEXE}
	@echo XDEXE=${XDEXE}
	@echo OTEXE=${OTEXE}
	@echo O9EXE=${O9EXE}
	@echo IDEXE=${IDEXE}
	@echo NCEXE=${NCEXE}
	@echo NREXE=${NREXE}

local: ${SRCLOCAL}/b2local.F ${MODLOCAL}/b2mod_local.F ${INCLOCAL}/b2local.h

${SRCLOCAL}/b2local.F:
	mkdir -p ${SRCLOCAL}
	echo "      subroutine b2local" > ${SRCLOCAL}/b2local.F
	echo "c" >> ${SRCLOCAL}/b2local.F
	echo "c store local or locally modified subroutines in this directory" >> ${SRCLOCAL}/
	    b2local.F
	echo "c" >> ${SRCLOCAL}/b2local.F
	echo "      use b2mod_local" >> ${SRCLOCAL}/b2local.F
	echo '#include "b2local.h"' >> ${SRCLOCAL}/b2local.F
```

```
        echo "c" >> ${SRCLOCAL}/b2local.F
        echo "      end" >> ${SRCLOCAL}/b2local.F

${MODLOCAL}/b2mod_local.F:
        mkdir -p ${MODLOCAL}
        echo "      module b2mod_local" > ${MODLOCAL}/b2mod_local.F
        echo "c" >> ${MODLOCAL}/b2mod_local.F
        echo "c store local or locally modified modules in this directory" >> ${MODLOCAL}/
            b2mod_local.F
        echo "c" >> ${MODLOCAL}/b2mod_local.F
        echo "      end" >> ${MODLOCAL}/b2mod_local.F

${INCLOCAL}/b2local.h:
        mkdir -p ${INCLOCAL}
        echo "c" > ${INCLOCAL}/b2local.h
        echo "c store local or locally modified include files in this directory" >> ${INCLOCAL}/
            b2local.h
        echo "c" >> ${INCLOCAL}/b2local.h

${OBJDIR}/mpiversion.mk: ${MAKES}
ifdef NO_MPI
        echo 'MPI_VERSION=0' > ${OBJDIR}/mpiversion.mk
else
        printf "use mpi\nWRITE(*,fmt='(A12,I1)') 'MPI_VERSION=', MPI_VERSION\nWRITE(*,fmt='(A9)')
             'MPI_MOD=1'\nEND\n" > ${OBJDIR}/mpi_version.f90
        ( ${FC} ${FCOPTS} ${FPOPTS} ${SOLPSINCLUDE} -o ${OBJDIR}/mpi_version ${OBJDIR}/
            mpi_version.f90 ${LD_MPI} && ( ${OBJDIR}/mpi_version | tail -n2 ) || \
        ( printf "include 'mpif.h'\nWRITE(*,fmt='(A12,I1)') 'MPI_VERSION=', MPI_VERSION\nWRITE(*,
            fmt='(A9)') 'MPI_MOD=0'\nEND\n" > ${OBJDIR}/mpi_version.f90 ; \
        ${FC} ${FCOPTS} ${FPOPTS} ${SOLPSINCLUDE} -o ${OBJDIR}/mpi_version ${OBJDIR}/mpi_version.
            f90 ${LD_MPI} && ( ${OBJDIR}/mpi_version | tail -n2 ) || ( echo MPI_VERSION=0 ; echo
            MPI_MOD=0 ) ) ) > ${OBJDIR}/mpiversion.mk
endif
```

and, as a prototypical example, the `config.ITER.ifort64` and `config.common.ifort64` files from the `config` subdirectory:

config.ITER.ifort64

```
# Test whether necessary variables are set
OS         ?= $(shell grep -ml NAME /etc/os-release | cut -d '"' -f 2 | cut -d ' ' -f 1 )
NODENAME   ?= $(shell echo `hostname`)
LD_MSCL    ?= `pkg-config mscl --libs`
ifdef NCARG_ROOT
#LD_NCARG ?= -L$(NCARG_ROOT)/lib -lncarg -lncarg_gks -lncarg_c -lm -lX11
LD_NCARG   ?= -L$(NCARG_ROOT)/lib -lncarg -lncarg_gks -lncarg_c -lm -lX11 -lXext `pkg-config cairo
     --libs` -lfontconfig -lfreetype -lpthread -lXrender -lbz2 -lgfortran
GLI_HOME   ?= ${EBROOTGLI}
endif
ifeq ($(shell `pkg-config --exists netcdf-fortran`; echo $$?),0)
NCDIR      ?= ${EBROOTNETCDFMINFORTRAN}
ifeq (,$(findstring CentOS,$(OS)))
LD_NETCDF  = `pkg-config netcdf-fortran --libs`
else
LD_NETCDF ?= `nf-config --flibs`
endif
ifndef SOLPS_CPP
DEFINES    += -DCDF_MOVIE
INCMODS    += `pkg-config netcdf-fortran --cflags`
endif
else
ifndef SOLPS_CPP
DEFINES    += -DNO_CDF
endif
endif
ifdef NAGDIR
LD_NAG     ?= -lnag_nag
ifndef SOLPS_CPP
```

```
DEFINES    += -DNAG22 -DNO_F01AAF
endif
else
LD_NAG     ?= `pkg-config openblas --libs`
ifndef SOLPS_CPP
DEFINES    += -DNO_NAG
endif
endif

DEFINES    += -DNO_OPT

CPP         = /lib/cpp -traditional

# Avoid usage of ifx compiler until its issues are fixed
# ifeq (,$(findstring CentOS,$(OS)))
# MPI_FC     ?= mpiifort -fc=mpiifx
# FN          = ifx
# else
# MPI_FC     ?= mpiifort
# endif
ifdef SOLPS_MPI
FC          = ${MPI_FC}
ifndef MPDIR
MPDIR       = ${EBROOTIMPI}
LD_MPI     += -Wl,-rpath,${MPDIR}/lib64
endif
else
ifdef USE_IMPGYRO
FC          = ${MPI_FC}
ifndef MPDIR
MPDIR       = ${EBROOTIMPI}
LD_MPI     += -Wl,-rpath,${MPDIR}/lib64
endif
# else
# ifeq (,$(findstring CentOS,$(OS)))
# FC          = ifx
# endif
endif
endif

FCOPTS      = -assume no2underscore -align commons -fPIC
ifdef SOLPS_DEBUG
FCOPTS     += -g -C -debug all -traceback -check all -fpe0
LDOPTS     += -check all
else
FCOPTS     += -O2
ifeq (,$(findstring CentOS,$(OS)))
FCOPTS     += -xHost
else
FCOPTS     += -march=sandybridge
endif
endif
ifdef SOLPS_MPI
FPOPTS     += ${MPI_CPP}
else
ifdef USE_IMPGYRO
FPOPTS     += ${MPI_CPP}
endif
endif

LDLIBES     = -lz

ifeq ($(shell `pkg-config --exists ggd`; echo $$?),0)
  GGD_VERSION ?= `pkg-config ggd --modversion`
else
  GGD_VERSION ?= '0.0.0'
endif
```

```
ifdef SOLPS_MPI
LDLIBES    += ${LD_MPI}
else
ifdef USE_IMPGYRO
LDLIBES    += ${LD_MPI}
endif
endif


#!!! Local Variables:
#!!! mode: Makefile
#!!! End:
```

config.common.ifort64

```
ifeq (${FC},f77)
FC = ifort
endif
ifeq (${FN},)
FN = ifort
endif
# Use of ifx OpenMP is unsafe at the moment
ifneq (,$(findstring ifx,${FC}))
  ifdef SOLPS_OPENMP
    FN = ifort
    MPI_FC = mpiifort
    ifdef SOLPS_MPI
      FC = ${MPI_FC}
    else
      FC = ifort
    endif
  endif
endif
EQUIVS    = -Dsgbtrf=dgbtrf -Dsgbtrs=dgbtrs -Dsgetrf=dgetrf -Dsgetri=dgetri

CPPOPTS   = -P -I${srcdir} ${EQUIVS}
SED       = sed s:\#PREC\#:d:g
SED       =
LINK_MOD =

DBLFC     = ${FC}
DBLSED    = sed -e s:\#PREC\#:d:g
DBLSED    =

ifndef COMPILER_REDEF
 INCMOD = -I
 OBJDEST  = -module ${OBJDIR} -o ${OBJDIR}/$*.o
 COMPILER_MAJOR_VERSION ?= $(shell ${FC} -V 2>&1 | grep -o "Version [0-9]*" | cut -d ' ' -f 2 )
 ifeq ($(shell test ${COMPILER_MAJOR_VERSION} -ge 15; echo $$?),0)
  FCOPTS += -fp-model=precise
 endif
 ifdef SOLPS_DEBUG
  FCOPTS  += -diag-disable:10182,10440
  LDOPTS  += -diag-disable:10182
  ifeq ($(shell test ${COMPILER_MAJOR_VERSION} -ge 2023; echo $$?),0)
   FCOPTS += -check nouninit
   LDOPTS += -check nouninit
  endif
 endif
 ifeq ($(shell test ${COMPILER_MAJOR_VERSION} -le 12; echo $$?),0)
  FCOPTS  += -vec_report0
  LDOPTS  += -i_dynamic
 endif
 ifeq ($(shell test ${COMPILER_MAJOR_VERSION} -ge 13; echo $$?),0)
  FCOPTS  += -align array64byte
 endif
 ifeq ($(shell test ${COMPILER_MAJOR_VERSION} -gt 17; echo $$?),0)
```

```
  ifeq ($(shell test ${COMPILER_MAJOR_VERSION} -ge 2023; echo $$?),0)
   FCOPTS  += -mprefer-vector-width=512
  else
   FCOPTS  += -qopt-zmm-usage=high
  endif
 endif
 ifdef SOLPS_OPENMP
  ifeq ($(shell test ${COMPILER_MAJOR_VERSION} -ge 15; echo $$?),0)
   FPOPTS += -qopenmp
   LPOPTS += -qopenmp
  else
   FPOPTS += -openmp
   LPOPTS += -openmp
  endif
 endif
else
 INCMOD = -I
 OBJDEST = -J${OBJDIR} -o ${OBJDIR}/$*.o
 OBNDEST = -J${OBNDIR} -o ${OBNDIR}/$*.o
 GFORTRAN_MAJOR_VERSION = $(shell ${FC} --version | head -1 | awk '{printf "%s ",$$NF,"\n"}' |
      cut -d '.' -f 1 )
 GFORTRAN_MINOR_VERSION = $(shell ${FC} --version | head -1 | awk '{printf "%s ",$$NF,"\n"}' |
      cut -d '.' -f 2 )
 DEFINES += -DGFORTRAN -DCOMPILER_MAJOR_VERSION=${GFORTRAN_MAJOR_VERSION} -
      DCOMPILER_MINOR_VERSION=${GFORTRAN_MINOR_VERSION} -DNOQUAD -DNO_IEEE
 FCOPTS += -fno-second-underscore -fno-unsafe-math-optimizations -frounding-math -fsignaling-nans
       -frecursive
 ifeq ($(shell test ${GFORTRAN_MAJOR_VERSION} -gt 7; echo $$?),0)
  FCOPTS += -mprefer-vector-width=512
 endif
 ifdef SOLPS_OPENMP
  FPOPTS += -fopenmp
  LPOPTS += -fopenmp
 endif
 FPOPTS += ${MPI_CPP}
 LPOPTS += ${LD_MPI}
 COMPILER_MAJOR_VERSION ?= $(shell ${FN} -V 2>&1 | grep -o "Version [0-9]*" | cut -d ' ' -f 2 )
endif

ifndef SOLPS_CPP
  ifeq ($(shell `pkg-config --exists fundamental-constants`; echo $$?),0)
    DEFINES += -DCONSTANTS_PROVIDED
    IMAS_PKG_LIST = fundamental-constants
    SOLPSINCLUDE += `pkg-config fundamental-constants --cflags`
  endif
  ifdef IMAS_VERSION
    ifdef AL_VERSION
      AL_MAJOR_VERSION ?= $(shell echo ${AL_VERSION} | cut -d '.' -f1 )
      AL_MINOR_VERSION ?= $(shell echo ${AL_VERSION} | cut -d '.' -f2 )
      AL_MICRO_VERSION ?= $(shell echo ${AL_VERSION} | cut -d '.' -f3 | cut -d '-' -f1 )
    else
      AL_MAJOR_VERSION ?= $(shell echo ${UAL_VERSION} | cut -d '.' -f1 )
      AL_MINOR_VERSION ?= $(shell echo ${UAL_VERSION} | cut -d '.' -f2 )
      AL_MICRO_VERSION ?= $(shell echo ${UAL_VERSION} | cut -d '.' -f3 | cut -d '-' -f1 )
    endif
    IMAS_MAJOR_VERSION ?= $(shell echo ${IMAS_VERSION} | cut -d '.' -f1 )
    IMAS_MINOR_VERSION ?= $(shell echo ${IMAS_VERSION} | cut -d '.' -f2 )
    IMAS_MICRO_VERSION ?= $(shell echo ${IMAS_VERSION} | cut -d '.' -f3 )
    GGD_MAJOR_VERSION ?= $(shell echo ${GGD_VERSION} | cut -d '.' -f1 )
    GGD_MINOR_VERSION ?= $(shell echo ${GGD_VERSION} | cut -d '.' -f2 )
    GGD_MICRO_VERSION ?= $(shell echo ${GGD_VERSION} | cut -d '.' -f3 | cut -d '-' -f1 )
    ifeq ($(shell test ${AL_MAJOR_VERSION} -ge 5; echo $$?),0)
      IMAS_PKG_LIST += al-fortran al-identifiers-fortran
    else
      ifndef COMPILER_REDEF
        IMAS_PKG_LIST += imas-ifort
        ifeq ($(shell test ${IMAS_MAJOR_VERSION} -ge 4; echo $$?),0)
```

```
        IMAS_PKG_LIST += imas-identifiers-ifort
      else
        ifeq ($(shell test ${IMAS_MINOR_VERSION} -ge 31; echo $$?),0)
          IMAS_PKG_LIST += imas-identifiers-ifort
        endif
      endif
    else
      IMAS_PKG_LIST += imas-gfortran
      ifeq ($(shell test ${IMAS_MAJOR_VERSION} -ge 4; echo $$?),0)
        IMAS_PKG_LIST += imas-identifiers-gfortran
      else
        ifeq ($(shell test ${IMAS_MINOR_VERSION} -ge 31; echo $$?),0)
          IMAS_PKG_LIST += imas-identifiers-gfortran
        endif
      endif
    endif
  endif
  ifdef AMNS_PREFIX
    DEFINE += -DAMNS
    IMAS_PKG_LIST += amns
  endif
  ifeq ($(shell test ${GGD_MAJOR_VERSION} -gt 0; echo $$?),0)
    ifeq ($(shell test ${IMAS_MAJOR_VERSION} -ge 4; echo $$?),0)
      IMAS_PKG_LIST += ggd
    else
      ifeq ($(shell test ${IMAS_MINOR_VERSION} -ge 17; echo $$?),0)
        IMAS_PKG_LIST += ggd
      else
        ifeq ($(shell test ${IMAS_MINOR_VERSION} -ge 12; echo $$?),0)
          ifndef COMPILER_REDEF
            LD_GGD ?= -lggd-ifort
          else
            LD_GGD := -lggd-gfortran
          endif
        else
          IMAS_PKG_LIST += imas-lowlevel
        endif
      endif
    endif
  endif
  LD_AL := `pkg-config ${IMAS_PKG_LIST} --libs`
  ifdef LD_GGD
    LD_AL += ${LD_GGD}
  endif
  DEFINES += -DIMAS -DAL_MAJOR_VERSION=${AL_MAJOR_VERSION} -DAL_MINOR_VERSION=${
      AL_MINOR_VERSION} -DAL_MICRO_VERSION=${AL_MICRO_VERSION} -DIMAS_MAJOR_VERSION=${
      IMAS_MAJOR_VERSION} -DIMAS_MINOR_VERSION=${IMAS_MINOR_VERSION} -DIMAS_MICRO_VERSION=${
      IMAS_MICRO_VERSION} -DGGD_MAJOR_VERSION=${GGD_MAJOR_VERSION} -DGGD_MINOR_VERSION=${
      GGD_MINOR_VERSION} -DGGD_MICRO_VERSION=${GGD_MICRO_VERSION}
 endif
endif

ifdef IMAS_PKG_LIST
 INCMODS += `pkg-config ${IMAS_PKG_LIST} --cflags`
 IMASLIBS = ${LD_AL}
 ifeq ($(shell test ${COMPILER_MAJOR_VERSION} -ge 2021; echo $$?),0)
   FCOPTS += -diag-disable:5462
 endif
endif

ifdef SOLPS_MPI
 DPFINES += -DMPI_VERSION=${MPI_VERSION} -DMPI_MOD=${MPI_MOD}
endif

MOD      = mod

BLD      = ar rucv
```

```
GRLIBES   = ${LD_NCARG}
PLLIBES   = ${LD_MSCL}
LDLIBES  += ${LD_NETCDF} ${LD_NAG}

ifdef LD_CATALYST
CXX       = c++
CXXFLAGS = -std=c++11
endif

ifdef PERFMON
LDLIBES += -L${SOLPSLIB} -lperf
endif

LD        = ${FC}
LN        = ${FN}


#!!! Local Variables:
#!!! mode: Makefile
#!!! End:
```

If the user wishes to modify the compiling options used, the creation of a `config.${HOST_NAME}.${COMPILER}.local` file in the `config` directory is recommended, in which to store the modified settings. Since the `Makefile` reads those after the `config.${HOST_NAME}.${COMPILER}` and `config.common.${COMPILER}` files, they need only contain the modified definitions.

K.4 Defines

K.4.1 SOLPS-ITER

We list here the possible optional pre-processor definitions that can be used when compiling the various components of the **SOLPS-ITER** code suite. The optional variables below must therefore, if desired, be declared as part of the `SOLPS_CPP` environment variable, which is defined in the `${SOLPSTOP}/SETUP/config.${HOST_NAME}.${COMPILER}(.local)` and `${SOLPSTOP}/SETUP/config.common.${COMPILER}` files. Additionally, system-wide compiler flags can be provided by means of the `FFLAGSEXTRA` environment variable in the same files.

AL_MAJOR_VERSION Major version number of the AL (IMAS Access Layer) library to which the code is linked (set internally by the configuration files)

AL_MICRO_VERSION Micro version number of the AL (IMAS Access Layer) library to which the code is linked (set internally by the configuration files)

AL_MINOR_VERSION Minor version number of the AL (IMAS Access Layer) library to which the code is linked (set internally by the configuration files)

CATALYST Turns on the Paraview Catalyst utility (see the SOLPS-GUI manual for full details)

DIMENSIONS_MODULE Indicates whether the **SOLPS-ITER** version is one that uses a `b2mod_dimensions.F` module file instead of the `DIMENSIONS.F` include file (set internally by the configuration files)

COMPILER_MAJOR_VERSION Major version number of the compiler being used (set internally by the configuration files)

COMPILER_MINOR_VERSION Minor version number of the compiler being used (set internally by the configuration files)

CONSTANTS_PROVIDED Indicates whether the `Fundamental-Constants` module is being at compilation time (set internally by the configuration files)

CRAY To be used with the Cray Fortran compiler

DBG Turns on additional debugging output (set internally by the `Makefile` when requesting a `_debug` compilation)

F77 Identifies specific code to `Fortran-77` compilers

GFORTRAN To be used with `gfortran` compilation

GGD_MAJOR_VERSION Major version number of the IMAS GGD library to which the code is linked (set internally by the configuration files)

GGD_MICRO_VERSION Micro version number of the IMAS GGD library to which the code is linked (set internally by the configuration files)

GGD_MINOR_VERSION Minor version number of the IMAS GGD library to which the code is linked (set internally by the configuration files)

G95 To be used with `g95` compilation

HPUX To be used when compiling on HP Unix systems

IBMaix To be used when compiling on IBM AIX computers

IMAS To be used when compiling within the IMAS framework

IMAS_MAJOR_VERSION Major version number of the IMAS library to which the code is linked (set internally by the configuration files)

IMAS_MICRO_VERSION Micro version number of the IMAS library to which the code is linked (set internally by the configuration files)

IMAS_MINOR_VERSION Minor version number of the IMAS library to which the code is linked (set internally by the configuration files)

ITM To be used when compiling within the ITM framework

ITM_ENVIRONMENT_LOADED To be used when wishing to compile with ITM CPOs

LEGACYCOMP For use with legacy compilers which do not support Fortran 2003 features

MDSPLUS Enables use of `MDSplus` trees

NAGF95 To be used when compiling with the `NAG` F95 compiler

NAG22 Specific to using the Nag Mark 22 or higher mathematical library

NCAR4 Used to specify NCAR4 rather than NCAR3

NO_CDF No `CDF` library available

NO_GETENV No `GETENV` system call available

NO_LONG_LINES Identify those compilers which have a limitation to relatively short output lines

NO_NAG No `NAG` libraries available

PGF90 To be used when compiling with the Portland Group `Fortran-90` compiler

PSR Specific to the Pacific Sierra VAST compiler

USE_MPI Set internally by the `Makefile` and used to identify code necessary for MPI applications

USE_PXFGETENV Uses `PXFGETENV` instead of `GETENV`

WINDOWS To be used when compiling on a Windows system

If you want to activate any of these options, you should add `-DOPTION` (where `OPTION` might, for example, be `NO_NAG`) to the `SOLPS_CPP` environment variable with

```
setenv SOLPS_CPP '-DOPTION'
```

and then recompile all of the routines affected (if unsure which routines are affected, recompile them all with

```
gmake clean
gmake depend
gmake
```

). If SOLPS_CPP already had a value, you would want to do

```
setenv SOLPS_CPP "$SOLPS_CPP -DOPTION"
```

or

```
SOLPS_CPP += -DOPTION
```

so that the new option is added to the list. ksh or bash users would use something like

```
export SOLPS_CPP="$SOLPS_CPP -DOPTION"
```

K.4.2 Carre

We list here additional possible compiler options that can be used when compiling Carre.

BUILDING_CARRE Flags code for extended grids handling routine specific to Carre

EXTRA_GRAPHS Turns on additional graphical output of intermediate steps of the Carre grid generation

MODIF Allows for manual modifications to the grid generation process during the Carre run

READOPT Allows for format-free keyboard input

We list below where the Carre compilation switches are used.

name	used in
DBG	carre/doubld.F
DBG	carre/saute.F
DBG	fcrr/fcrprp.F
DIMENSIONS_MODULE	carre/carre.F
DIMENSIONS_MODULE	carre/change.F
DIMENSIONS_MODULE	carre/defaut.F
DIMENSIONS_MODULE	carre/derive.F
DIMENSIONS_MODULE	carre/doubld.F
DIMENSIONS_MODULE	carre/frtier.F
DIMENSIONS_MODULE	carre/insert.F
DIMENSIONS_MODULE	carre/limfnd.f
DIMENSIONS_MODULE	carre/mailcn.F
DIMENSIONS_MODULE	carre/maille.F
DIMENSIONS_MODULE	carre/mailrg.F
DIMENSIONS_MODULE	carre/marche.F
DIMENSIONS_MODULE	carre/nulort.F
DIMENSIONS_MODULE	carre/rappel.F
DIMENSIONS_MODULE	carre/selptx.F
DIMENSIONS_MODULE	carre/sortie.F
DIMENSIONS_MODULE	carre/sptris.F
DIMENSIONS_MODULE	carre/warnings.F
DIMENSIONS_MODULE	cntour/cntour.F
DIMENSIONS_MODULE	cntour/struct.F

continued on next page

continued from previous page	
name	used in
DIMENSIONS_MODULE	cntour/trace2.F
DIMENSIONS_MODULE	cntour/trace3.F
DIMENSIONS_MODULE	cntour/trace.F
DIMENSIONS_MODULE	dummy/bidon.F
DIMENSIONS_MODULE	include/carre_dimensions.F
DIMENSIONS_MODULE	trans/b2agfz.F
DIMENSIONS_MODULE	trans/ecrim5.F
DIMENSIONS_MODULE	trans/tradui.F
EXTRA_GRAPHS	carre/carre.F
GFORTRAN	carre/change.F
GFORTRAN	carre/rdfrin.F
GFORTRAN	carre/rdfrre.F
MODIF	carre/carre.F
MODIF	carre/maille.F
MODIF	carre/mailrg.F
READOPT	carre/rdfrch.F
READOPT	carre/rdfrin.F
READOPT	carre/rdfrre.F
SOLPS_CPP	include/carre_dimensions.F
SOLPS_CPP	trans2agfz.F

Table K.1: Defines for Carre and where they are used.

K.4.3 DivGeo

We list below where the `DivGeo` compilation switches are used.

name	used in
AL_MAJOR_VERSION	equtrn/src/dg2ids.f
AL_MAJOR_VERSION	equtrn/src/rdids.f
AL_MAJOR_VERSION	equtrn/src/wrids.f
AL_MICRO_VERSION	equtrn/src/rdids.f
AL_MINOR_VERSION	equtrn/src/rdids.f
CONSTANTS_PROVIDED	equtrn/src/eqdim.f90
GGD_MAJOR_VERSION	equtrn/src/wrids.f
GGD_MICRO_VERSION	equtrn/src/wrids.f
GGD_MINOR_VERSION	equtrn/src/wrids.f
IMAS	equtrn/src/eqdim.f90
IMAS_MAJOR_VERSION	equtrn/src/dg2ids.f
IMAS_MAJOR_VERSION	equtrn/src/eqdim.f90
IMAS_MAJOR_VERSION	equtrn/src/ids2dg.f
IMAS_MAJOR_VERSION	equtrn/src/rdids.f
IMAS_MAJOR_VERSION	equtrn/src/wrids.f
IMAS_MICRO_VERSION	equtrn/src/wrids.f
IMAS_MINOR_VERSION	equtrn/src/eqdim.f90
IMAS_MINOR_VERSION	equtrn/src/rdids.f
IMAS_MINOR_VERSION	equtrn/src/wrids.f
continued on next page	

continued from previous page	
name	used in
NO_GETENV	equtrn/src/dg2ids.f
NO_GETENV	equtrn/src/ids2dg.f
NO_GETENV	equtrn/src/wrids.f
USE_PXFGETENV	equtrn/src/dg2ids.f
USE_PXFGETENV	equtrn/src/ids2dg.f
USE_PXFGETENV	equtrn/src/wrids.f

Table K.2: Defines for DivGeo and where they are used.

K.4.4 B2.5-Eirene

We list here the possible compiler options that can be used when compiling the coupled version of `B2.5-Eirene`. These options are shared across both the `B2.5` and `Eirene` source codes. The options below must therefore, if desired, be declared as part of the `SOLPS_CPP` environment variable, which is defined in the `${SOLPSTOP}/SETUP/config.${HOST_NAME}.${COMPILER}(.local)` files.

ALLOCATE_AND_NAMELIST To be used if the compiler allows dynamically allocated namelist arrays

B25_EIRENE Set internally by the `Makefile` and used to identify differences between source code for standalone `B2.5` and coupled `B2.5-Eirene` applications

MPI_MOD Checks whether the MPI internal variables are brought in *via* the `mpi.mod` module or the `mpif.h` include file (set internally by the `Makefile`)

MPI_VERSION Index of MPI version (set internally by the `Makefile`)

NO_B2_CHEM_SPUT Used to indicate that chemical sputtering is specified through `Eirene` arrays instead of `B2.5`

NO_SAVE Turning off static save of some common blocks

SOLPS4_3 Set internally by the `Makefile` to activate code specific to the `SOLPS4-5 b2sxdr` converter

USE_EXT_OPENMP For use if `B2.5-Eirene` or its components are called from within an OpenMP parallelized region inside a larger workflow (Cannot be used in conjunction with `USE_OPENMP`)

USE_IMPGYRO Used when coupling to IMPGYRO

USE_OPENMP For OpenMP specific code related to internal `Eirene` parallelization (Cannot be used in conjunction with `USE_EXT_OPENMP`). Will be set by default when requesting an OpenMP executable.

K.4.5 B2.5

We list here the additional compiler options available that only affect compilation of `B2.5`. The options below must therefore, if desired, be declared as part of either the `SOLPS_CPP` or the `B25_DEFINES` environment variable, which are defined in the `${SOLPSTOP}/SETUP/config.${HOST_NAME}.${COMPILER}(.local)` files.

AMNS Links to AMNS database in IMAS environment

ASTRA To be used if one wishes to run in coupled mode with the core transport code `ASTRA`

CDF_MOVIE To be used if `CDF` movie output is desired (turned on by default if the NetCDF library is present)

CLASSICAL_CURRENT Turns on classical current drift term

COMPLICATED Uses a more ”complicated” model for feedback schemes

EASY Uses the ”easy” model for chord integration

GOT_AMNSPROTO Links to AMNS database in ITM environment

JET Enables some additional output for coupling to the JET post-processor

NO_21 Do not use 21-moment model for thermal and viscous forces

NO_IEEE Provides replacements to IEEE non-standard arithmetic routines

NO_F01AAF No `NAG` routine `F01AAF` — uses its replacement instead (needed if linking with Version 17 or later of `NAG`)

NO_OPENMP_* Turning off OpenMP parallelization

NO_OPT No compiler optimization

_OPENMP For OpenMP specific code

PAR_OPT Runs the code in a special interpretive parameter optimization mode

PERFMON Enable performance monitoring

PLPLOT Use `PLPLOT` instead of `NCAR` for plotting

USE_CDF_B2STIME Specific for **SOLPS 4.x** `CDF` tallies

WANT_PROCESS Enables the code for the **SOLPS 5.x** to **SOLPS 4.x** converter

WOS Uses the `WOS` libraries for accessing experimental data at IPP

If you want to activate any of these options, you should add `-DOPTION` (where `OPTION` might, for example, be `PERFMON`) to either the `SOLPS_CPP` or the `B25_DEFINES` environment variable with

```
setenv B25_DEFINES '-DOPTION'
```

and then recompile all of the routines affected (if unsure which routines are affected, recompile them all with

```
gmake clean
gmake depend
gmake
```

). If `B25_DEFINES` already had a value, you would want to do

```
setenv B25_DEFINES "$B25_DEFINES -DOPTION"
```

or

```
B25_DEFINES += -DOPTION
```

so that the new option is added to the list. (Unsupported `ksh` or `bash` users would use something like

```
export B25_DEFINES="$B25_DEFINES -DOPTION"
```

). If choosing to use `SOLPS_CPP` instead, the procedure is the same as in the previous section.

We list below where the `B2.5` compilation switches are used.

name	used in
AL_MAJOR_VERSION	b2plot/b2pawpwr.F
AL_MAJOR_VERSION	ids/b2_ual_rewrite.F90
AL_MAJOR_VERSION	ids/b2_ual_write.F90
AL_MAJOR_VERSION	ids/b2mod_ual.F90
AL_MINOR_VERSION	ids/b2mod_ual.F90

continued on next page

continued from previous page

name	used in
AL_MAJOR_VERSION	ids/b2mod_ual_io.F90
AL_MAJOR_VERSION	ids/b2mod_ual_io_grid.F90
ALLOCATE_AND_NAMELIST	b2plot/init_wall.F
ALLOCATE_AND_NAMELIST	modules/b2mod_boundary_namelist.F
ALLOCATE_AND_NAMELIST	modules/b2mod_input_profile.F
ALLOCATE_AND_NAMELIST	modules/b2mod_layer.F
ALLOCATE_AND_NAMELIST	modules/b2mod_ld_namelist.F
ALLOCATE_AND_NAMELIST	modules/b2mod_neutrals_namelist.F
ALLOCATE_AND_NAMELIST	modules/b2mod_numerics_namelist.F
ALLOCATE_AND_NAMELIST	modules/b2mod_sputter.F
ALLOCATE_AND_NAMELIST	modules/b2mod_transport_models.F
ALLOCATE_AND_NAMELIST	modules/b2mod_transport_namelist.F
ALLOCATE_AND_NAMELIST	modules/b2mod_user_namelist.F
ALLOCATE_AND_NAMELIST	modules/b2mod_wall.F
ALLOCATE_AND_NAMELIST	postprocessing/b2yt_wall.F
ALLOCATE_AND_NAMELIST	sources/b2stbr.F
AMNS	preprocessing/b2ardr.F
AMNS	utility/ratadas.F
ASTRA	modules/b2mod_astra_to_b2.F
ASTRA	sources/b2stbc_phys.F
ASTRA	sources/b2stbr.F
ASTRA	sources/b2stcx.F
ASTRA	sources/b2stel.F
ASTRA	transport/b2trcl.F
B25_EIRENE	b2plot/b2pabrmv.F
B25_EIRENE	b2plot/b2paux.F
B25_EIRENE	b2plot/b2pblmtr.F
B25_EIRENE	b2plot/b2phel.F
B25_EIRENE	b2plot/b2phortn.F
B25_EIRENE	b2plot/b2pinp.F
B25_EIRENE	b2plot/b2plot.F
B25_EIRENE	b2plot/b2pnwl.F
B25_EIRENE	b2plot/b2prdl.F
B25_EIRENE	b2plot/b2ptrdl.F
B25_EIRENE	b2plot/b2ptrgl.F
B25_EIRENE	b2plot/b2pwinm.F
B25_EIRENE	b2plot/b2pwldw.F
B25_EIRENE	b2plot/b2pwlprp.F
B25_EIRENE	b2plot/b2pwrdld.F
B25_EIRENE	b2plot/b2pwsort.F
B25_EIRENE	b2plot/b2pzavrv.F
B25_EIRENE	b2plot/clvp.F
B25_EIRENE	b2plot/copy.F
B25_EIRENE	b2plot/get_from_movie_cdf.F
B25_EIRENE	b2plot/map_eirene_chords.F
B25_EIRENE	b2plot/ngread.F
B25_EIRENE	b2plot/ntread.F
B25_EIRENE	b2plot/pltoutline.F
B25_EIRENE	b2plot/plttriang.F
B25_EIRENE	b2plot/process_eirene_chords_boxcar.F

continued on next page

continued from previous page

name	used in
B25_EIRENE	b2plot/process_eirene_chords.F
B25_EIRENE	b2plot/process_eirene_chord_slice.F
B25_EIRENE	b2plot/psta.F
B25_EIRENE	b2plot/pstv.F
B25_EIRENE	b2plot/triangle_surf.F
B25_EIRENE	b2plot/trigmesh.F
B25_EIRENE	convert/b2mds.F
B25_EIRENE	convert/b2yt_ngread.F
B25_EIRENE	convert/b2yt_ntread.F
B25_EIRENE	driver/b2co.F
B25_EIRENE	driver/b2mod_driver.F
B25_EIRENE	driver/b2mod_main.F
B25_EIRENE	ids/b2mod_ual_io.F90
B25_EIRENE	modules/b2mod_b2_to_astra.F
B25_EIRENE	modules/b2mod_mwti.F90
B25_EIRENE	modules/b2mod_running_average.F
B25_EIRENE	modules/b2mod_neutrals_namelist.F
B25_EIRENE	output/b2wdat.F
B25_EIRENE	postprocessing/b2ytdr.F
B25_EIRENE	sources/b2mod_eirene.F
B25_EIRENE	sources/b2stbc_fb.F
B25_EIRENE	sources/b2stbc_phys.F
B25_EIRENE	sources/b2stbm.F
B25_EIRENE	sources/b2stbr.F
B25_EIRENE	sources/eirene_f30f31.F
B25_EIRENE	sources/eirene_mc_init.F
B25_EIRENE	transport/b2tinnt.F
B25_EIRENE	user/b2mod_blnc.F
B25_EIRENE	user/b2mod_blne.F
B25_EIRENE	user/b2mod_blns.F
B25_EIRENE	user/b2mod_trace.F
B25_EIRENE	user/b2mod_usrtrc.F
B25_EIRENE	user/b2mod_wrint.F
B25_EIRENE	user/b2mod_wrsep.F
B25_EIRENE	user/b2mod_wrsrt.F
B25_EIRENE	user/cdfmovie.F
B25_EIRENE	user/eirflux_map.F
B25_EIRENE	user/test_couple.F90
B25_EIRENE	utility/gfsub2.F
B25_EIRENE	utility/rdneutrs.F
B25_EIRENE	utility/xertst.F
BUILDING_CARRE	ids/b2mod_connectivity.F90
CATALYST	driver/b2mod_driver.F
CDF_MOVIE	b2plot/caltraj.F
CDF_MOVIE	b2plot/process_chords.F
CLASSICAL_CURRENT	transport/b2tdia.F
COMPLICATED	sources/b2stbc_fb.F
CONSTANTS_PROVIDED	common/emissivity.constants
CONSTANTS_PROVIDED	ids/b2mod_constants.F90
CRAY	b2plot/b2plot.F

continued on next page

continued from previous page

name	used in
CRAY	driver/b2mnastra.F
CRAY	driver/b2mod_driver.F
CRAY	driver/b2obeq.F
CRAY	driver/b2obfn.F
CRAY	driver/b2oblm.F
CRAY	modules/b2mod_openmp.F
CRAY	postprocessing/b2ytdr.F
CRAY	solvers/b2ux11p.F
CRAY	solvers/b2ux5p.F
CRAY	solvers/b2ux7p.F
CRAY	solvers/b2ux9p.F
CRAY	utility/b2mod_illtern_7.F
CRAY	utility/b2mod_ilutern_11.F
CRAY	utility/b2mod_ilutern_7.F
CRAY	utility/b2mod_ilutern_9.F
DBG	b2plot/b2ptrdl.F
DBG	b2plot/b2ptrgl.F
DBG	b2plot/b2pwrdld.F
DBG	b2plot/fill.F
DBG	b2plot/init_wall.F
DBG	b2plot/ngread.F
DBG	b2plot/plotcontour.F
DBG	b2plot/rlcomp.F
DBG	equations/b2news_.F
DBG	modules/b2mod_input_profile.F
DBG	modules/b2mod_mwti.F
DBG	modules/b2mod_neutrals_namelist.F
DBG	sources/b2stbr_bas.F
DBG	sources/b2stbr_phys.F
DBG	user/b2mod_file.F
DBG	user/b2mod_usrtrc.F
DBG	user/b2mod_wrsrt.F
DBG	utility/ma28.F
DBG	utility/ma28copy.F
DBG	utility/ma28copy3.F
DBG	utility/ma28copy4.F
DBG	utility/my_out.F
DBG	utility/ysmp_d.F
DEF_MAXCHORDS	b2plot/process_chords.F
DEF_MAXCHORDS	b2plot/process_eirene_chords.F
EASY	b2plot/chord.F
EASY	b2plot/chord_triangle.F
F77	user/b2file.F
GFORTRAN	modules/b2mod_openmp.F
GFORTRAN	utility/b2mod_ilutern_7.F
GFORTRAN	utility/b2mod_ilutern_9.F
GFORTRAN	utility/b2mod_ilutern_11.F
GFORTRAN	utility/ilutern_solps.F
GFORTRAN	utility/ma28.F
GFORTRAN	utility/ma28copy.F

continued on next page

continued from previous page

name	used in
GFORTRAN	utility/ma28copy3.F
GFORTRAN	utility/ma28copy4.F
GFORTRAN	utility/mds_routines.F
G95	utility/mds_routines.F
GGD_MAJOR_VERSION	ids/b2mod_ual_io.F90
GGD_MAJOR_VERSION	ids/b2mod_ual_io_data.F90
GGD_MAJOR_VERSION	ids/b2mod_ual_io_grid.F90
GGD_MICRO_VERSION	ids/b2mod_ual_io.F90
GGD_MICRO_VERSION	ids/b2mod_ual_io_grid.F90
GGD_MINOR_VERSION	ids/b2mod_ual_io.F90
GGD_MINOR_VERSION	ids/b2mod_ual_io_grid.F90
GOT_AMNSPROTO	preprocessing/b2ardr.F
GOT_AMNSPROTO	utility/ratadas.F
HITS_PER_CHORD	b2plot/process_chords.F
HITS_PER_CHORD	b2plot/process_eirene_chords.F
HPUX	modules/b2mod_eirbra.F
IBMaix	b2plot/b2plot.F
IBMaix	postprocessing/b2ytdr.F
IBMaix	user/b2file.F
IMAS	b2plot/b2paspwr.F
IMAS	driver/b2mnds.F
IMAS	ids/b2mod_interp.F90
IMAS	ids/b2mod_ual.F90
IMAS	ids/b2mod_ual_io_data.F90
IMAS	ids/b2mod_ual_io_grid.F90
IMAS	ids/b2mod_ual_io.F90
IMAS	ids/b2_ual_write.F90
IMAS	modules/b2mod_b2cmrc.F
IMAS	modules/b2mod_ppout.F
IMAS	preprocessing/b2ardr.F
IMAS	utility/ratadas.F
IMAS_MAJOR_VERSION	b2plot/b2paspwr.F
IMAS_MAJOR_VERSION	driver/b2mod_driver.F
IMAS_MAJOR_VERSION	ids/b2mod_interp.F90
IMAS_MAJOR_VERSION	ids/b2mod_ual.F90
IMAS_MAJOR_VERSION	ids/b2mod_ual_io.F90
IMAS_MAJOR_VERSION	ids/b2mod_ual_io_data.F90
IMAS_MAJOR_VERSION	ids/b2mod_ual_io_grid.F90
IMAS_MAJOR_VERSION	ids/b2_ual_rewrite.F90
IMAS_MAJOR_VERSION	ids/b2_ual_write.F90
IMAS_MAJOR_VERSION	ids/b2_ual_write_b2mod.F90
IMAS_MICRO_VERSION	driver/b2mod_driver.F
IMAS_MICRO_VERSION	ids/b2mod_ual_io.F90
IMAS_MICRO_VERSION	ids/b2_ual_rewrite.F90
IMAS_MICRO_VERSION	ids/b2_ual_write.F90
IMAS_MICRO_VERSION	ids/b2_ual_write_b2mod.F90
IMAS_MINOR_VERSION	driver/b2mod_driver.F
IMAS_MINOR_VERSION	ids/b2mod_interp.F90
IMAS_MINOR_VERSION	ids/b2mod_ual.F90
IMAS_MINOR_VERSION	ids/b2mod_ual_io.F90

continued on next page

continued from previous page

name	used in
IMAS_MINOR_VERSION	ids/b2mod_ual_io_data.F90
IMAS_MINOR_VERSION	ids/b2mod_ual_io_grid.F90
IMAS_MINOR_VERSION	ids/b2_ual_rewrite.F90
IMAS_MINOR_VERSION	ids/b2_ual_write.F90
IMAS_MINOR_VERSION	ids/b2_ual_write_b2mod.F90
ITM	driver/b2mod_driver.F
ITM	preprocessing/b2ardr.F
ITM	utility/ratadas.F
ITM_ENVIRONMENT_LOADED	ids/b2mod_ual.F90
ITM_ENVIRONMENT_LOADED	ids/b2mod_ual_io_data.F90
ITM_ENVIRONMENT_LOADED	ids/b2mod_ual_io_grid.F90
ITM_ENVIRONMENT_LOADED	ids/b2mod_ual_io.F90
JET	driver/b2mnastra.F
JET	driver/b2mnds.F
JET	driver/b2mn_opt.F
JET	driver/b2mod_driver.F
JET	driver/b2mod_main.F
JET	ids/b2_ual_write.F90
JET	modules/b2mod_ppout.F
JET	preprocessing/b2agfs.F
JET	preprocessing/b2ardr.F
LEGACYCOMP	convert/b2cdftallies2mds.F
LEGACYCOMP	modules/b2mod_mdsplus.F
LEGACYCOMP	modules/b2mod_running_average.F
LEGACYCOMP	postprocessing/b2ye.F
LEGACYCOMP	sources/b2sifr_.F
LEGACYCOMP	sources/b2sifrtf.F
LEGACYCOMP	test/check_b2_output.F90
LEGACYCOMP	user/b2mod_batch.F
LEGACYCOMP	utility/cfopen.F
LEGACYCOMP	utility/mds_routines.F
LEGACYCOMP	utility/smax.F
LEGACYCOMP	utility/smin.F
MDSPLUS	convert/b2cdf2mds.F
MDSPLUS	convert/b2cdfmovies2mds.F
MDSPLUS	convert/b2cdftallies2mds.F
MDSPLUS	convert/b2mddr.F
MDSPLUS	convert/b2mds.F
MDSPLUS	driver/b2rd.F
MDSPLUS	modules/b2mod_mdsplus.F
MDSPLUS	utility/mds_routines.F
MPI_MOD	driver/b2mn_opt.F
MPI_MOD	driver/b2mnastra.F
MPI_MOD	driver/b2mod_main.F
MPI_MOD	modules/b2mod_impgyro.F
MPI_MOD	utility/xertst.F
NAG22	b2plot/caltraj.F
NAGF95	b2plot/b2plot.F
NAGF95	b2plot/dhdata.F
NAGF95	b2plot/initialize_plotting.F

continued on next page

continued from previous page

name	used in
NAGF95	b2plot/spect.F
NAGF95	convert/b2mds.F
NAGF95	convert/b2plasmastate_inspect.F
NAGF95	driver/b2co.F
NAGF95	modules/b2mod_ppout.F
NAGF95	modules/b2mod_solpstop.F
NAGF95	output/b2wgpj_gnuplot.F
NAGF95	preprocessing/b2agfs.F
NAGF95	utility/open_file.F
NAGF95	utility/usrnam.F
NAGFOR	b2plot/dhdata.F
NAGFOR	b2plot/set_extralabel.F
NAGFOR	convert/b2plasmastate_inspect.F
NAGFOR	driver/b2mod_driver.F
NAGFOR	driver/b2obeq.F
NAGFOR	driver/b2obfn.F
NAGFOR	driver/b2oblm.F
NAGFOR	modules/b2mod_openmp.F
NAGFOR	modules/b2mod_types.F
NAGFOR	output/b2wgpj_gnuplot.F
NAGFOR	transport/b2trca.F
NAGFOR	transport/b2trcv.F
NAGFOR	utility/calc_atomic_data.F
NAGFOR	utility/smax.F
NAGFOR	utility/smin.F
NCAR4	b2plot/end_page.F
NCAR4	b2plot/initialize_plotting.F
NCAR4	b2plot/terminate_plotting.F
NO_21	transport/b2trcl.F
NO_B2_CHEM_SPUT	modules/b2mod_neutrals_namelist.F
NO_B2_CHEM_SPUT	postprocessing/b2ytdr.F
NO_B2_CHEM_SPUT	sources/b2mod_eirene.F
NO_CDF	b2plot/b2plot.F
NO_CDF	b2plot/caltraj.F
NO_CDF	b2plot/get_from_movie_cdf.F
NO_CDF	b2plot/process_chords.F
NO_CDF	convert/b2cdf2mds.F
NO_CDF	convert/b2cdfmovies2mds.F
NO_CDF	convert/b2cdftallies2mds.F
NO_CDF	convert/b2co_movies_cdf.F
NO_CDF	convert/b2co_tally_cdf.F
NO_CDF	convert/b2co_time_cdf.F
NO_CDF	convert/b2co_wall_cdf.F
NO_CDF	convert/process.F
NO_CDF	driver/b2mod_driver.F
NO_CDF	modules/b2mod_balance.F
NO_CDF	modules/b2mod_movies.F
NO_CDF	modules/b2mod_mwti.F
NO_CDF	modules/b2mod_tallies.F
NO_CDF	modules/b2mod_wall.F

continued on next page

continued from previous page

name	used in
NO_CDF	output/tallies.F
NO_CDF	sources/b2stbr.F
NO_CDF	user/cdfmovie.F
NO_CDF	utility/cdf_routines.F
NO_F01AAF	transport/b2tr21.F
NO_GETENV	b2plot/b2paspwr.F
NO_GETENV	b2plot/b2plot.F
NO_GETENV	b2plot/spect.F
NO_GETENV	convert/b2co_movies_cdf.F
NO_GETENV	convert/b2co_tally_cdf.F
NO_GETENV	convert/b2co_time_cdf.F
NO_GETENV	convert/b2co_wall_cdf.F
NO_GETENV	convert/b2mds.F
NO_GETENV	driver/b2co.F
NO_GETENV	driver/b2mod_driver.F
NO_GETENV	ids/b2_ual_rewrite.F90
NO_GETENV	ids/b2_ual_write.F90
NO_GETENV	ids/b2_ual_write_b2mod.F90
NO_GETENV	ids/b2mod_ual_io.F90
NO_GETENV	modules/b2mod_ppout.F
NO_GETENV	modules/b2mod_solpstop.F
NO_GETENV	preprocessing/b2agfs.F
NO_GETENV	utility/open_file.F
NO_GETENV	utility/usrnam.F
NO_IEEE	utility/nancheck.F
NO_LONG_LINES	output/tallies.F
NO_NAG	b2plot/caltraj.F
NO_NAG	sources/heatdiff1D.F
NO_NAG	transport/b2tr21.F
NO_NAG	utility/nagsubst.F
NO_OPENMP_B2NEWS_LOOP1	equations/b2news.F
NO_OPENMP_B2NEWS_LOOP2	equations/b2news.F
NO_OPENMP_B2NEWS_LOOP3	equations/b2news.F
NO_OPENMP_B2NEWS_LOOP4	equations/b2news.F
NO_OPENMP_B2NEWS_UNDERSCORE_LOOP1	equations/b2news_.F
NO_OPENMP_B2NEWS_UNDERSCORE_LOOP1	equations/b2news_m.F
NO_OPENMP_B2NEWS_UNDERSCORE_LOOP2	equations/b2news_.F
NO_OPENMP_B2NEWS_UNDERSCORE_LOOP2	equations/b2news_m.F
NO_OPENMP_B2NEWS_UNDERSCORE_LOOP3	equations/b2news_.F
NO_OPENMP_B2NEWS_UNDERSCORE_LOOP3	equations/b2news_m.F
NO_OPENMP_B2NEWS_UNDERSCORE_LOOP4	equations/b2news_.F
NO_OPENMP_B2NEWS_UNDERSCORE_LOOP4	equations/b2news_m.F
NO_OPENMP_B2NPMO	equations/b2npmo.F
NO_OPENMP_B2SIFRTF_LOOP1	sources/b2sifrtf.F
NO_OPENMP_B2SIFRTF_LOOP2	sources/b2sifrtf.F
NO_OPENMP_B2SIHS	sources/b2sihs.F
NO_OPENMP_B2SPCX	sources/b2spcx.F
NO_OPENMP_B2SPEL	sources/b2spel.F
NO_OPENMP_B2SQCX	sources/b2sqcx.F
NO_OPENMP_B2SQEL	sources/b2sqel.F

continued on next page

continued from previous page

name	used in
NO_OPENMP_B2SRDT	sources/b2scdt.F
NO_OPENMP_B2SRDT	sources/b2shdt.F
NO_OPENMP_B2SRDT	sources/b2smdt.F
NO_OPENMP_B2SRDT	sources/b2srdt.F
NO_OPENMP_B2SRST	sources/b2srst.F
NO_OPENMP_B2STBR	sources/b2stbr_phys.F
NO_OPENMP_B2STCX	sources/b2stcx.F
NO_OPENMP_B2STEL	sources/b2stel.F
NO_OPENMP_B2XPFE	b2aux/b2xpfe.F
NO_OPENMP_MYBLAS	utility/myblas.F
NO_OPENMP_SFILL	utility/sfill.F
NO_OPT	b2plot/b2plot.F
NO_OPT	ids/b2mod_ual_io.F90
NO_OPT	postprocessing/b2ytdr.F
NO_SAVE	include/BRAEIR.H
NO_SAVE	include/EIRBRA.H
NO_SAVE	include/EIRDIAG.H
_OPENMP	b2aux/b2xpfe.F
_OPENMP	catalyst/fortranAdaptor.F90
_OPENMP	driver/b2mn_opt.F
_OPENMP	driver/b2mnastra.F
_OPENMP	driver/b2mod_driver.F
_OPENMP	driver/b2obeq.F
_OPENMP	driver/b2obfn.F
_OPENMP	driver/b2oblm.F
_OPENMP	equations/b2xpfe.F
_OPENMP	equations/b2news.F
_OPENMP	equations/b2news_.F
_OPENMP	equations/b2news_m.F
_OPENMP	equations/b2npc11.F
_OPENMP	equations/b2npc7.F
_OPENMP	equations/b2npc9.F
_OPENMP	equations/b2npco.F
_OPENMP	equations/b2npmo.F
_OPENMP	equations/b2nxdv.F
_OPENMP	equations/b2nxfc.F
_OPENMP	equations/b2nxfv.F
_OPENMP	equations/b2nxfv_zh.F
_OPENMP	equations/b2nxst.F
_OPENMP	modules/b2mod_heatdiff_common.F
_OPENMP	modules/b2mod_ma28_for_11diag.F
_OPENMP	modules/b2mod_ma28_for_7diag.F
_OPENMP	modules/b2mod_ma28_for_9diag.F
_OPENMP	modules/b2mod_openmp.F
_OPENMP	modules/b2mod_sputter.F
_OPENMP	modules/b2mod_ysmp_sdrv.F
_OPENMP	solvers/b2unavspr7a.F
_OPENMP	solvers/b2upco.F
_OPENMP	solvers/b2urmo.F
_OPENMP	solvers/b2ursc.F

continued on next page

continued from previous page

name	used in
_OPENMP	solvers/b2usc11.F
_OPENMP	solvers/b2usc7.F
_OPENMP	solvers/b2usc9.F
_OPENMP	solvers/b2usco.F
_OPENMP	solvers/b2ush9.F
_OPENMP	solvers/b2usht.F
_OPENMP	solvers/b2usmo.F
_OPENMP	solvers/b2uspo.F
_OPENMP	solvers/b2ux11p.F
_OPENMP	solvers/b2ux5p.F
_OPENMP	solvers/b2ux7p.F
_OPENMP	solvers/b2ux9p.F
_OPENMP	solvers/b2uxm11.F
_OPENMP	solvers/b2uxm9.F
_OPENMP	sources/b2mod_eirene.F
_OPENMP	sources/b2scdt.F
_OPENMP	sources/b2shdt.F
_OPENMP	sources/b2sian.F
_OPENMP	sources/b2siav.F
_OPENMP	sources/b2siav_zh.F
_OPENMP	sources/b2sicf.F
_OPENMP	sources/b2sifr.F
_OPENMP	sources/b2sifr_.F
_OPENMP	sources/b2sifrtf.F
_OPENMP	sources/b2sigp.F
_OPENMP	sources/b2smdt.F
_OPENMP	sources/b2spcx.F
_OPENMP	sources/b2spel.F
_OPENMP	sources/b2sqcx.F
_OPENMP	sources/b2sqel.F
_OPENMP	sources/b2srdt.F
_OPENMP	sources/b2srst.F
_OPENMP	sources/b2stbr.F
_OPENMP	sources/b2stbr_phys.F
_OPENMP	sources/b2stcx.F
_OPENMP	sources/b2stel.F
_OPENMP	sources/heatdiff1D.F
_OPENMP	transport/b2tbfl.F
_OPENMP	transport/b2tfnb.F
_OPENMP	transport/b2tlnl.F
_OPENMP	transport/b2txvsprco.F
_OPENMP	utility/b2mod_illtern_7.F
_OPENMP	utility/b2mod_ilutern_11.F
_OPENMP	utility/b2mod_ilutern_7.F
_OPENMP	utility/b2mod_ilutern_9.F
_OPENMP	utility/b2mod_ipmain.F
_OPENMP	utility/b2opd_subsys.F
_OPENMP	utility/ilutern_solps.F
_OPENMP	utility/ma28.F
_OPENMP	utility/ma28copy.F

continued on next page

continued from previous page

name	used in
_OPENMP	utility/ma28copy3.F
_OPENMP	utility/ma28copy4.F
_OPENMP	utility/myblas.F
_OPENMP	utility/my_out.F
_OPENMP	utility/sfill.F
PAR_OPT	sources/b2stbc_phys.F
PAR_OPT	transport/b2tqna.F
PERFMON	utility/prgend.F
PERFMON	utility/prgini.F
PERFMON	utility/subsys.F
PGF90	modules/b2mod_openmp.F
PGF90	modules/b2mod_types.F
PLPLOT	b2plot/b2plot.F
PLPLOT	b2plot/color.F
PLPLOT	b2plot/end_page.F
PLPLOT	b2plot/font.F
PLPLOT	b2plot/fx.F
PLPLOT	b2plot/fy.F
PLPLOT	b2plot/gslwsc_d.F
PLPLOT	b2plot/initialize_plotting.F
PLPLOT	b2plot/init_page.F
PLPLOT	b2plot/lined_d.F
PLPLOT	b2plot/mesh.F
PLPLOT	b2plot/meshit.F
PLPLOT	b2plot/plotpatch.F
PLPLOT	b2plot/plottarget.F
PLPLOT	b2plot/plottripatch.F
PLPLOT	b2plot/plot_wall_elements.F
PLPLOT	b2plot/pltoutline.F
PLPLOT	b2plot/pltsptrix.F
PLPLOT	b2plot/plttarget.F
PLPLOT	b2plot/plttriang.F
PLPLOT	b2plot/pltvessel.F
PLPLOT	b2plot/pltwall.F
PLPLOT	b2plot/set_a4l.F
PLPLOT	b2plot/set_a4p.F
PLPLOT	b2plot/set_extralabel.F
PLPLOT	b2plot/terminate_plotting.F
PLPLOT	b2plot/trgw.F
PLPLOT	b2plot/trgz.F
PLPLOT	b2plot/trigmesh.F
PLPLOT	utility/b2mod_lbfill.F
PLPLOT	utility/ncardummy.F
PLPLOT	utility/pagdef.F
PLPLOT	utility/pagend.F
PLPLOT	utility/pagini.F
PSR	modules/b2mod_b2plot.F
PSR	modules/b2mod_elements.F
PSR	modules/b2mod_sputter.F
SOLPS4_3	modules/b2mod_b2_to_astra.F

continued on next page

continued from previous page

name	used in
SOLPS4_3	modules/b2mod_diag.F
SOLPS4_3	modules/b2mod_mwti.F90
SOLPS4_3	modules/b2mod_neutrals_namelist.F
SOLPS4_3	modules/b2mod_running_average.F
SOLPS4_3	sources/eirene_f30f31.F
USE_CDF_B2STIME	modules/b2mod_comtimpl.F
USE_CDF_B2STIME	common/COUPLE/COMTIMPL.F
USE_EXT_OPENMP	driver/b2mod_driver.F
USE_IMPGYRO	driver/b2mod_driver.F
USE_IMPGYRO	driver/b2mod_main.F
USE_IMPGYRO	modules/b2mod_impgyro.F
USE_MPI	driver/b2mnastra.F
USE_MPI	driver/b2mn_opt.F
USE_MPI	driver/b2mod_driver.F
USE_MPI	driver/b2mod_main.F
USE_MPI	modules/b2mod_boundary_namelist.F
USE_MPI	modules/b2mod_feedback.F
USE_MPI	modules/b2mod_impgyro.F
USE_MPI	modules/b2mod_input_profile.F
USE_MPI	modules/b2mod_neutrals_namelist.F
USE_MPI	modules/b2mod_numerics_namelist.F
USE_MPI	modules/b2mod_sputter.F
USE_MPI	modules/b2mod_transpbgr_namelist.F
USE_MPI	modules/b2mod_transport_models.F
USE_MPI	modules/b2mod_transport_namelist.F
USE_MPI	modules/b2mod_user_namelist.F
USE_MPI	modules/b2mod_wall.F
USE_MPI	utility/xertst.F
USE_OPENMP	driver/b2mod_driver.F
USE_PXFGETENV	b2plot/b2paspwr.F
USE_PXFGETENV	b2plot/b2plot.F
USE_PXFGETENV	b2plot/initialize_plotting.F
USE_PXFGETENV	b2plot/set_extralabel.F
USE_PXFGETENV	b2plot/spect.F
USE_PXFGETENV	convert/b2co_movies_cdf.F
USE_PXFGETENV	convert/b2co_tally_cdf.F
USE_PXFGETENV	convert/b2co_time_cdf.F
USE_PXFGETENV	convert/b2co_wall_cdf.F
USE_PXFGETENV	convert/b2mds.F
USE_PXFGETENV	convert/b2plasmastate_inspect.F
USE_PXFGETENV	driver/b2co.F
USE_PXFGETENV	driver/b2mod_driver.F
USE_PXFGETENV	ids/b2_ual_rewrite.F90
USE_PXFGETENV	ids/b2_ual_write.F90
USE_PXFGETENV	ids/b2mod_ual_io.F90
USE_PXFGETENV	modules/b2mod_ppout.F
USE_PXFGETENV	modules/b2mod_solpstop.F
USE_PXFGETENV	preprocessing/b2agfs.F
USE_PXFGETENV	utility/open_file.F
USE_PXFGETENV	utility/usrnam.F

continued on next page

continued from previous page	
name	used in
WANT_PROCESS	convert/process.F
WINDOWS	b2plot/initialize_plotting.F
WINDOWS	driver/b2obeq.F
WINDOWS	driver/b2obfn.F
WINDOWS	driver/b2oblm.F
WINDOWS	driver/b2op.F
WINDOWS	modules/b2mod_b2plot_values.F
WINDOWS	modules/b2mod_sputter.F
WINDOWS	postprocessing/b2yv.F
WOS	b2plot/additional_elements.F
WOS	b2plot/flux.F

Table K.3: Defines for B2.5 and where they are used.

K.4.6 Eirene

We list here the possible compiler options that can be used when compiling `Eirene`. The options below must therefore, if desired, be declared as part of either the `SOLPS_CPP` or the `EIRENE_DEFINES` environment variable, which are defined in the `${SOLPSTOP}/SETUP/config.${HOST_NAME}.${COMPILER}(.local)` files.

CHECKBIN Turns on debugging output to check the contents of the output binary files

NO_JSON Turns off JSON I/O format support

OPEN_MPI Used to indicate compilation with the OpenMPI library (automatically set by the `Makefile`)

TRACE Turns on tracing output in geometry routines (set internally by the `Makefile` when requesting a `_debug` compilation)

If you want to activate any of these options, you should add `-DOPTION` (where `OPTION` might, for example, be `TRACE`) to either the `SOLPS_CPP` or the `EIRENE_DEFINES` environment variable with

```
setenv EIRENE_DEFINES '-DOPTION'
```

and then recompile all of the routines affected (if unsure which routines are affected, recompile them all with

```
gmake clean
gmake depend
gmake
```

). If `EIRENE_DEFINES` already has a value, you would want to do

```
setenv EIRENE_DEFINES "$EIRENE_DEFINES -DOPTION"
```

or

```
EIRENE_DEFINES += -DOPTION
```

so that the new option is added to the list. (Unsupported `ksh` or `bash` users would use something like

```
export EIRENE_DEFINES="$EIRENE_DEFINES -DOPTION"
```

). If choosing to use `SOLPS_CPP` instead, the procedure is the same as in the previous section.

We list below where the `Eirene` compilation switches are used.

name	used in
ALLOCATE_AND_NAMELIST	startup-routines/find_param.F
B25_EIRENE	interfaces/couple_SOLPS-ITER/couple_param.F
B25_EIRENE	interfaces/couple_SOLPS-ITER/eirmod_extrab25.F90
B25_EIRENE	main-routines/eirmod_eirene.F
B25_EIRENE	output/outflx.F
CHECKBIN	assistant/wrstrt.F
CHECKBIN	file-handling/wrgeom.F
CHECKBIN	file-handling/wrplam_long.F
CHECKBIN	file-handling/wrplam_shrt.F
CHECKBIN	interfaces/couple_B2/infcop.f
CHECKBIN	main-routines/eirmod_mcarlo.F
CHECKBIN	modules/eirmod_comxs.F
COMPILER_MAJOR_VERSION	assistant/exit_own.F
COMPILER_MINOR_VERSION	assistant/exit_own.F
CRAY	broadcast/broad_iestr.F
CRAY	broadcast/broadcast.F
CRAY	broadcast/broadsput.F
CRAY	broadcast/calstr.F
CRAY	broadcast/collect_census.F
CRAY	broadcast/collect_coutau.F
CRAY	interfaces/couple_SOLPS-ITER/eirmod_wneutrals.F
CRAY	modules/eirmod_cadgeo.F
CRAY	modules/eirmod_cai.F
CRAY	modules/eirmod_ccona.F
CRAY	modules/eirmod_cestim.F
CRAY	modules/eirmod_cgeom.F
CRAY	modules/eirmod_cgrid.F
CRAY	modules/eirmod_clgin.F
CRAY	modules/eirmod_clogau.F
CRAY	modules/eirmod_comnnl.F
CRAY	modules/eirmod_comsig.F
CRAY	modules/eirmod_comsou.F
CRAY	modules/eirmod_comspl.F
CRAY	modules/eirmod_comusr.F
CRAY	modules/eirmod_comxs.F
CRAY	modules/eirmod_coutau.F
CRAY	modules/eirmod_cpes.F
CRAY	modules/eirmod_cpolyg.F
CRAY	modules/eirmod_cref.F
CRAY	modules/eirmod_csdvi.F
CRAY	modules/eirmod_cstep.F
CRAY	modules/eirmod_ctetra.F
CRAY	modules/eirmod_ctext.F
CRAY	modules/eirmod_ctrcei.F
CRAY	modules/eirmod_ctrig.F
CRAY	modules/eirmod_czt1.F
CRAY	modules/eirmod_parmmod.F
CRAY	photons/eirmod_photon.F
CRAY	user-routines/user_iter/broad_usr.F
CRAY	user-routines/user_iter/eirmod_refusr.F
DBG	interfaces/couple_SOLPS-ITER/eirmod_infcop.F

continued on next page

continued from previous page

name	used in
DIMENSIONS_MODULE	interfaces/couple_SOLPS-ITER/couple_param.F
DIMENSIONS_MODULE	interfaces/couple_SOLPS-ITER/eirmod_extrab25.F90
GFORTRAN	modules/eirmod_mpi.F90
GFORTRAN	assistant/exit_own.F
GFORTRAN	broadcast/broad_iestr.F
GFORTRAN	broadcast/broadcast.F
GFORTRAN	broadcast/broadsput.F
GFORTRAN	broadcast/calstr.F
GFORTRAN	broadcast/collect_census.F
GFORTRAN	broadcast/collect_coutau.F
GFORTRAN	broadcast/eirmod_calstr_buffered.F90
GFORTRAN	interfaces/couple_SOLPS-ITER/eirmod_wneutrals.F
GFORTRAN	interfaces/couple_SOLPS-ITER/eirsrt.f
GFORTRAN	modules/eirmod_balanced_strategy.F90
GFORTRAN	modules/eirmod_improved_strategy.F90
GFORTRAN	modules/eirmod_cadgeo.F
GFORTRAN	modules/eirmod_cai.F
GFORTRAN	modules/eirmod_ccona.F
GFORTRAN	modules/eirmod_cestim.F
GFORTRAN	modules/eirmod_cgeom.F
GFORTRAN	modules/eirmod_cgrid.F
GFORTRAN	modules/eirmod_cinit.F
GFORTRAN	modules/eirmod_clgin.F
GFORTRAN	modules/eirmod_clogau.F
GFORTRAN	modules/eirmod_comnnl.F
GFORTRAN	modules/eirmod_comsig.F
GFORTRAN	modules/eirmod_comsou.F
GFORTRAN	modules/eirmod_comspl.F
GFORTRAN	modules/eirmod_comusr.F
GFORTRAN	modules/eirmod_comxs.F
GFORTRAN	modules/eirmod_coutau.F
GFORTRAN	modules/eirmod_cpes.F
GFORTRAN	modules/eirmod_cpolyg.F
GFORTRAN	modules/eirmod_cref.F
GFORTRAN	modules/eirmod_csdvi.F
GFORTRAN	modules/eirmod_cstep.F
GFORTRAN	modules/eirmod_ctetra.F
GFORTRAN	modules/eirmod_ctext.F
GFORTRAN	modules/eirmod_ctrcei.F
GFORTRAN	modules/eirmod_ctrig.F
GFORTRAN	modules/eirmod_czt1.F
GFORTRAN	modules/eirmod_parmmod.F
GFORTRAN	photons/eirmod_photon.F
GFORTRAN	user-routines/user_iter/broad_usr.F
GFORTRAN	user-routines/user_iter/eirmod_refusr.F
G95	assistant/exit_own.F
G95	file-handling/eirmod_openfile.F
G95	interfaces/couple_SOLPS-ITER/couple_specific.F
G95	interfaces/couple_SOLPS-ITER/eirmod_infcop.F
G95	main-routines/integrate_tallies.F

continued on next page

continued from previous page

name	used in
G95	main-routines/scal_volav_tallies.F
G95	main-routines/scale_tallies.F
G95	main-routines/sum_average.F
G95	main-routines/sumostra.F
G95	main-routines/tally_sanity_check.F
G95	particle-tracing/eirmod_collide.F
G95	particle-tracing/escape.F
G95	scoring/getscl4.F
G95	scoring/update.F
G95	scoring/update_surface.F
G95	scoring/updphot.F
G95	startup-routines/find_tet_dim.F
G95	startup-routines/find_triang_dim.F
HPUX	modules/eirmod_cinit.F
LEGACYCOMP	assistant/eirmod_algebra.F
LEGACYCOMP	file-handling/eirmod_openfile.F
LEGACYCOMP	file-handling/wrplam_shrt.F
LEGACYCOMP	interfaces/couple_SOLPS-ITER/couple_specific.F
LEGACYCOMP	interfaces/couple_SOLPS-ITER/eirmod_infcop.F
LEGACYCOMP	main-routines/integrate_tallies.F
LEGACYCOMP	main-routines/scal_volav_tallies.F
LEGACYCOMP	main-routines/scale_tallies.F
LEGACYCOMP	main-routines/sum_average.F
LEGACYCOMP	main-routines/sumostra.F
LEGACYCOMP	main-routines/tally_sanity_check.F
LEGACYCOMP	modules/eirmod_comusr.F
LEGACYCOMP	modules/eirmod_json.F
LEGACYCOMP	modules/eirmod_samsrf.F
LEGACYCOMP	particle-tracing/eirmod_collide.F
LEGACYCOMP	particle-tracing/escape.F
LEGACYCOMP	scoring/getscl4.F
LEGACYCOMP	scoring/update.F
LEGACYCOMP	scoring/update_surface.F
LEGACYCOMP	scoring/updphot.F
LEGACYCOMP	startup-routines/find_tet_dim.F
LEGACYCOMP	startup-routines/find_triang_dim.F
LEGACYCOMP	startup-routines/read_fixform.F
MPI_VERSION	modules/eirmod_mpi.F90
NAGFOR	file-handling/eirmod_openfile.F
NO_B2_CHEM_SPUT	interfaces/couple_SOLPS-ITER/eirmod_infcop.F
NO_GETENV	user-routines/user_iter/get_solpstop.F
NO_JSON	interfaces/couple_SOLPS-ITER/eirmod_infcop.F
NO_JSON	interfaces/couple_SOLPS-ITER/if0prm_json.F
NO_JSON	interfaces/couple_SOLPS-ITER/wr0_json.F
NO_JSON	modules/eirmod_comusr.F
NO_JSON	modules/eirmod_json.F
NO_JSON	startup-routines/read_fixform.F
NO_JSON	startup-routines/read_json.F
NO_JSON	startup-routines/read_reaclines.F
NO_JSON	startup-routines/write_json_amdata.f

continued on next page

continued from previous page

name	used in
NO_JSON	startup-routines/write_json_file.F
NO_JSON	user-routines/user_iter/eirmod_iousr.F
OPEN_MPI	broadcast/broad_iestr.F
OPEN_MPI	broadcast/broadcast.F
OPEN_MPI	broadcast/broadsput.F
OPEN_MPI	broadcast/calstr.F
OPEN_MPI	broadcast/collect_census.F
OPEN_MPI	broadcast/collect_coutau.F
OPEN_MPI	broadcast/eirmod_calstr_buffered.F90
OPEN_MPI	interfaces/couple_SOLPS-ITER/eirmod_wneutrals.F
OPEN_MPI	interfaces/couple_SOLPS-ITER/eirsrt.F
OPEN_MPI	modules/eirmod_balanced_strategy.F90
OPEN_MPI	modules/eirmod_improved_strategy.F90
OPEN_MPI	modules/eirmod_cadgeo.F
OPEN_MPI	modules/eirmod_cai.F
OPEN_MPI	modules/eirmod_ccona.F
OPEN_MPI	modules/eirmod_cestim.F
OPEN_MPI	modules/eirmod_cgeom.F
OPEN_MPI	modules/eirmod_cgrid.F
OPEN_MPI	modules/eirmod_cinit.F
OPEN_MPI	modules/eirmod_clgin.F
OPEN_MPI	modules/eirmod_clogau.F
OPEN_MPI	modules/eirmod_comnnl.F
OPEN_MPI	modules/eirmod_comsig.F
OPEN_MPI	modules/eirmod_comsou.F
OPEN_MPI	modules/eirmod_comspl.F
OPEN_MPI	modules/eirmod_comusr.F
OPEN_MPI	modules/eirmod_comxs.F
OPEN_MPI	modules/eirmod_coutau.F
OPEN_MPI	modules/eirmod_cpes.F
OPEN_MPI	modules/eirmod_cpolyg.F
OPEN_MPI	modules/eirmod_cref.F
OPEN_MPI	modules/eirmod_csdvi.F
OPEN_MPI	modules/eirmod_cstep.F
OPEN_MPI	modules/eirmod_ctetra.F
OPEN_MPI	modules/eirmod_ctext.F
OPEN_MPI	modules/eirmod_ctrcei.F
OPEN_MPI	modules/eirmod_ctrig.F
OPEN_MPI	modules/eirmod_czt1.F
OPEN_MPI	modules/eirmod_parmmod.F
OPEN_MPI	photons/eirmod_photon.F
OPEN_MPI	user-routines/user_iter/broad_usr.F
OPEN_MPI	user-routines/user_iter/eirmod_refusr.F
PGF90	assistant/exit_own.F
PGF90	particle-tracing/folion.F
PGF90	particle-tracing/velocs.F
PGF90	surface-processes/emaxw.F
TRACE	geometry/time-routines/checks.F
TRACE	geometry/time-routines/timea.F
TRACE	geometry/time-routines/eirmod_timep.F

continued on next page

continued from previous page

name	used in
TRACE	geometry/time-routines/eirmod_timer.F
TRACE	geometry/time-routines/timet.F
TRACE	modules/eirmod_improved_strategy.F90
USE_EXT_OPENMP	assistant/exit_own.F
USE_EXT_OPENMP	broadcast/eirmod_calstr_buffered.F90
USE_EXT_OPENMP	diagno/eirmod_linint.F
USE_EXT_OPENMP	geometry/eirmod_learc1.F
USE_EXT_OPENMP	geometry/learc2.F
USE_EXT_OPENMP	geometry/time-routines/eirmod_timea.F
USE_EXT_OPENMP	geometry/time-routines/eirmod_timep.F
USE_EXT_OPENMP	geometry/time-routines/eirmod_timer.F
USE_EXT_OPENMP	interfaces/couple_SOLPS-ITER/uptcop.F
USE_EXT_OPENMP	main-routines/eirmod_eirene.F
USE_EXT_OPENMP	main-routines/eirmod_mcarlo.F
USE_EXT_OPENMP	main-routines/integrate_tallies.F
USE_EXT_OPENMP	main-routines/sumostra.F
USE_EXT_OPENMP	mathematics/eirmod_h1rnm.F
USE_EXT_OPENMP	mathematics/eirmod_ranf.F
USE_EXT_OPENMP	modules/eirmod_cestim.F
USE_EXT_OPENMP	modules/eirmod_cfplk.F
USE_EXT_OPENMP	modules/eirmod_clast.F
USE_EXT_OPENMP	modules/eirmod_comprt.F
USE_EXT_OPENMP	modules/eirmod_comsou.F
USE_EXT_OPENMP	modules/eirmod_comspl.F
USE_EXT_OPENMP	modules/eirmod_comusr.F
USE_EXT_OPENMP	modules/eirmod_comxs.F
USE_EXT_OPENMP	modules/eirmod_coutau.F
USE_EXT_OPENMP	modules/eirmod_crand.F
USE_EXT_OPENMP	modules/eirmod_csdvi.F
USE_EXT_OPENMP	modules/eirmod_cspez.F
USE_EXT_OPENMP	modules/eirmod_cupd.F
USE_EXT_OPENMP	modules/eirmod_czt1.F
USE_EXT_OPENMP	modules/eirmod_openmp.F
USE_EXT_OPENMP	particle-tracing/collide.F
USE_EXT_OPENMP	particle-tracing/colphot.F
USE_EXT_OPENMP	particle-tracing/eirmod_addcol.F
USE_EXT_OPENMP	particle-tracing/eirmod_locate.F
USE_EXT_OPENMP	particle-tracing/eirmod_samsrf.F
USE_EXT_OPENMP	particle-tracing/eirmod_samvol.F
USE_EXT_OPENMP	particle-tracing/eirmod_stdcol.F
USE_EXT_OPENMP	particle-tracing/eirmod_switch_partinfo.F
USE_EXT_OPENMP	particle-tracing/eirmod_velocx.F
USE_EXT_OPENMP	particle-tracing/eirmod_veloel.F
USE_EXT_OPENMP	particle-tracing/eirmod_velopi.F
USE_EXT_OPENMP	particle-tracing/escape.F
USE_EXT_OPENMP	particle-tracing/folion.F
USE_EXT_OPENMP	particle-tracing/folneut.F
USE_EXT_OPENMP	particle-tracing/folstat_ion.F
USE_EXT_OPENMP	particle-tracing/folstat_neut.F
USE_EXT_OPENMP	particle-tracing/fpkcol.F

continued on next page

continued from previous page

name	used in
USE_EXT_OPENMP	particle-tracing/spltrr.F
USE_EXT_OPENMP	particle-tracing/timcol.F
USE_EXT_OPENMP	particle-tracing/time2surface.F
USE_EXT_OPENMP	particle-tracing/torcol.F
USE_EXP_OPENMP	plotting/eirmod_plt2d.F
USE_EXT_OPENMP	plotting/eirmod_tstchm.F
USE_EXT_OPENMP	scoring/eirmod_statis.F
USE_EXT_OPENMP	scoring/eirmod_updlin.F
USE_EXT_OPENMP	scoring/eirmod_uptbgk.F
USE_EXT_OPENMP	scoring/update.F
USE_EXT_OPENMP	scoring/update_spectrum.F
USE_EXT_OPENMP	scoring/update_sptflx.F
USE_EXT_OPENMP	scoring/update_surface.F
USE_EXT_OPENMP	scoring/updphot.F
USE_EXT_OPENMP	surface-processes/eirmod_reflec.F
USE_EXT_OPENMP	surface-processes/eirmod_sheath.F
USE_EXT_OPENMP	surface-processes/eirmod_sputer.F
USE_EXT_OPENMP	surface-processes/reflec_photon.F
USE_EXT_OPENMP	tetrahedra/learct.F
USE_EXT_OPENMP	user-routines/user_iter/eirmod_uptusr.F
USE_MPI	assistant/eirmod_second_own.F
USE_MPI	assistant/exit_own.F
USE_MPI	broadcast/broad_iestr.F
USE_MPI	broadcast/broadcast.F
USE_MPI	broadcast/broadsput.F
USE_MPI	broadcast/calstr.F
USE_MPI	broadcast/collect_census.F
USE_MPI	broadcast/collect_coutau.F
USE_MPI	broadcast/eirmod_calstr_buffered.F90
USE_MPI	interfaces/couple_SOLPS-ITER/eirmod_ccoupl.F
USE_MPI	interfaces/couple_SOLPS-ITER/eirmod_eirbra.F
USE_MPI	interfaces/couple_SOLPS-ITER/eirmod_infcop.F
USE_MPI	interfaces/couple_SOLPS-ITER/eirmod_wneutrals.F
USE_MPI	interfaces/couple_SOLPS-ITER/eirsrt.f
USE_MPI	modules/eirmod_balanced_strategy.F90
USE_MPI	modules/eirmod_improved_strategy.F90
USE_MPI	modules/eirmod_cadgeo.F
USE_MPI	modules/eirmod_cai.F
USE_MPI	modules/eirmod_ccona.F
USE_MPI	modules/eirmod_cestim.F
USE_MPI	modules/eirmod_cgeom.F
USE_MPI	modules/eirmod_cgrid.F
USE_MPI	modules/eirmod_cinit.F
USE_MPI	modules/eirmod_clgin.F
USE_MPI	modules/eirmod_clogau.F
USE_MPI	modules/eirmod_comnnl.F
USE_MPI	modules/eirmod_comsig.F
USE_MPI	modules/eirmod_comsou.F
USE_MPI	modules/eirmod_comspl.F
USE_MPI	modules/eirmod_comusr.F

continued on next page

continued from previous page

name	used in
USE_MPI	modules/eirmod_comxs.F
USE_MPI	modules/eirmod_coutau.F
USE_MPI	modules/eirmod_cpes.F
USE_MPI	modules/eirmod_cpolyg.F
USE_MPI	modules/eirmod_cref.F
USE_MPI	modules/eirmod_csdvi.F
USE_MPI	modules/eirmod_cstep.F
USE_MPI	modules/eirmod_ctetra.F
USE_MPI	modules/eirmod_ctext.F
USE_MPI	modules/eirmod_ctrcei.F
USE_MPI	modules/eirmod_ctrig.F
USE_MPI	modules/eirmod_czt1.F
USE_MPI	modules/eirmod_mpi.F90
USE_MPI	modules/eirmod_parmmod.F
USE_MPI	photons/eirmod_photon.F
USE_MPI	startup-routines/input.F
USE_MPI	user-routines/user_iter/broad_usr.F
USE_MPI	user-routines/user_iter/eirmod_refusr.F
USE_OPENMP	assistant/eirmod_second_own.F
USE_OPENMP	assistant/exit_own.F
USE_OPENMP	broadcast/eirmod_calstr_buffered.F90
USE_OPENMP	diagno/eirmod_linint.F
USE_OPENMP	geometry/eirmod_learc1.F
USE_OPENMP	geometry/learc2.F
USE_OPENMP	geometry/time-routines/eirmod_timea.F
USE_OPENMP	geometry/time-routines/eirmod_timep.F
USE_OPENMP	geometry/time-routines/eirmod_timer.F
USE_OPENMP	interfaces/couple_SOLPS-ITER/uptcop.F
USE_OPENMP	main-routines/eirmod_eirene.F
USE_OPENMP	main-routines/eirmod_mcarlo.F
USE_OPENMP	main-routines/integrate_tallies.F
USE_OPENMP	main-routines/sumostra.F
USE_OPENMP	mathematics/eirmod_h1rnm.F
USE_OPENMP	mathematics/eirmod_ranf.F
USE_OPENMP	modules/eirmod_cestim.F
USE_OPENMP	modules/eirmod_cfplk.F
USE_OPENMP	modules/eirmod_clast.F
USE_OPENMP	modules/eirmod_comprt.F
USE_OPENMP	modules/eirmod_comsou.F
USE_OPENMP	modules/eirmod_comspl.F
USE_OPENMP	modules/eirmod_comusr.F
USE_OPENMP	modules/eirmod_comxs.F
USE_OPENMP	modules/eirmod_coutau.F
USE_OPENMP	modules/eirmod_crand.F
USE_OPENMP	modules/eirmod_csdvi.F
USE_OPENMP	modules/eirmod_csepz.F
USE_OPENMP	modules/eirmod_cupd.F
USE_OPENMP	modules/eirmod_czt1.F
USE_OPENMP	modules/eirmod_openmp.F
USE_OPENMP	particle-tracing/collide.F

continued on next page

continued from previous page	
name	used in
USE_OPENMP	particle-tracing/colphot.F
USE_OPENMP	particle-tracing/eirmod_addcol.F
USE_OPENMP	particle-tracing/eirmod_locate.F
USE_OPENMP	particle-tracing/eirmod_samsrf.F
USE_OPENMP	particle-tracing/eirmod_samvol.F
USE_OPENMP	particle-tracing/eirmod_stdcol.F
USE_OPENMP	particle-tracing/eirmod_switch_partinfo.F
USE_OPENMP	particle-tracing/eirmod_velocx.F
USE_OPENMP	particle-tracing/eirmod_veloel.F
USE_OPENMP	particle-tracing/eirmod_velopi.F
USE_OPENMP	particle-tracing/escape.F
USE_OPENMP	particle-tracing/folion.F
USE_OPENMP	particle-tracing/folneut.F
USE_OPENMP	particle-tracing/folstat_ion.F
USE_OPENMP	particle-tracing/folstat_neut.F
USE_OPENMP	particle-tracing/fpkcol.F
USE_OPENMP	particle-tracing/spltrr.F
USE_OPENMP	particle-tracing/timcol.F
USE_OPENMP	particle-tracing/time2surface.F
USE_OPENMP	particle-tracing/torcol.F
USE_OPENMP	plotting/eirmod_plt2d.F
USE_OPENMP	plotting/eirmod_tstchm.F
USE_OPENMP	scoring/eirmod_statis.F
USE_OPENMP	scoring/eirmod_updlin.F
USE_OPENMP	scoring/eirmod_uptbgk.F
USE_OPENMP	scoring/update.F
USE_OPENMP	scoring/update_spectrum.F
USE_OPENMP	scoring/update_sptflx.F
USE_OPENMP	scoring/update_surface.F
USE_OPENMP	scoring/updphot.F
USE_OPENMP	surface-processes/eirmod_reflec.F
USE_OPENMP	surface-processes/eirmod_sheath.F
USE_OPENMP	surface-processes/eirmod_sputer.F
USE_OPENMP	surface-processes/reflec_photon.F
USE_OPENMP	tetrahedra/learct.F
USE_OPENMP	user-routines/user_iter/eirmod_uptusr.F
USE_PXFGETENV	user-routines/user_iter/get_solpstop.F
WINDOWS	main-routines/eirmod_eirene.f
WINDOWS	user-routines/user_iter/ioflush_usr.F

Table K.4: Defines for Eirene and where they are used.

If you want to activate any of these options, you should add `-DOPTION` (where `OPTION` might, for example, be `NO_NAG`) to either the `SOLPS_CPP` or the `EIRENE_DEFINES` environment variable with

```
setenv EIRENE_DEFINES '-DOPTION'
```

and then recompile all of the routines affected (if unsure which routines are affected, recompile them all with

```
gmake clean
gmake depend
```

```
gmake
```

). If EIRENE_DEFINES already had a value, you would want to do

```
setenv EIRENE_DEFINES "$EIRENE_DEFINES -DOPTION"
```

or

```
EIRENE_DEFINES += -DOPTION
```

so that the new option is added to the list. (Unsupported ksh or bash users would use something like

```
export EIRENE_DEFINES="$EIRENE_DEFINES -DOPTION"
```

). If choosing to use SOLPS_CPP instead, the procedure is the same as in the previous section.

K.5 NAG calls in SOLPS

K.5.1 B2.5

The NAG calls from B2.5 are given in table K.5.

caller routine	calling	function
b2mndr.f	X05BAF	returns the amount of processor time used since an unspecified previous time, via the routine name
b2mod_ilutern_7.f	X05BAF	returns the amount of processor time used since an unspecified previous time, via the routine name
b2mod_input_profile.f	E01BEF	computes a monotonicity-preserving piecewise cubic Hermite interpolant to a set of data points
b2mod_input_profile.f	E01BFF	evaluates a piecewise cubic Hermite interpolant, as computed by E01BEF
b2stbr.f	X05BAF	returns the amount of processor time used since an unspecified previous time, via the routine name
b2tr21.f	F01AAF	performs a matrix inversion (OBSOLETE)
b2tr21.f	F06QFF	performs the matrix-copy operation
b2tr21.f	F07ADF	(DGETRF) computes the LU factorization of a real m by n matrix.
b2tr21.f	F07AJF	(DGETRI) computes the inverse of a real matrix A, where A has been factorized by F07ADF (DGETRF).
b2ux5p.f	X05BAF	returns the amount of processor time used since an unspecified previous time, via the routine name
b2ux7p.f	X05BAF	returns the amount of processor time used since an unspecified previous time, via the routine name
caltraj.f	G05CAF	returns a pseudo-random number taken from a uniform distribution between 0 and 1
caltraj.f	G05CBF	sets the seeds used by the generator mechanism
caltraj.f	X05BAF	returns the amount of processor time used since an unspecified previous time, via the routine name
daytim.f	X05AAF	returns the current date and time
daytim.f	X05ABF	converts from a seven-integer format time and date, as returned by X05AAF, into a character string, returned via the routine name
heatdiff1D.f	D03PCF	integrates a system of linear or nonlinear parabolic partial differential equations (PDEs) in one space variable.
macheps.f	X02AJF	returns the machine precision
continued on next page		

continued from previous page		
caller routine	calling	function
machsfr.f	X02AMF	returns the safe range of floating-point arithmetic

Table K.5: NAG calls from B2.5

K.5.2 EIRENE

The NAG calls from `Eirene` are given in table K.6.

erf.f	S15AEF	returns the value of the error function erf, via the routine name.
second_own.F	X05BAF	returns the amount of processor time used since an unspecified previous time, via the routine name

Table K.6: NAG calls from Eirene

K.6 Performance monitoring

This is an experimental feature!

This only works at the moment using the `linux.Fujitsu` or `linux.ifort64` compilers, and then only at Garching!

You need to set

```
setenv PERFMON -DPERFMON
```

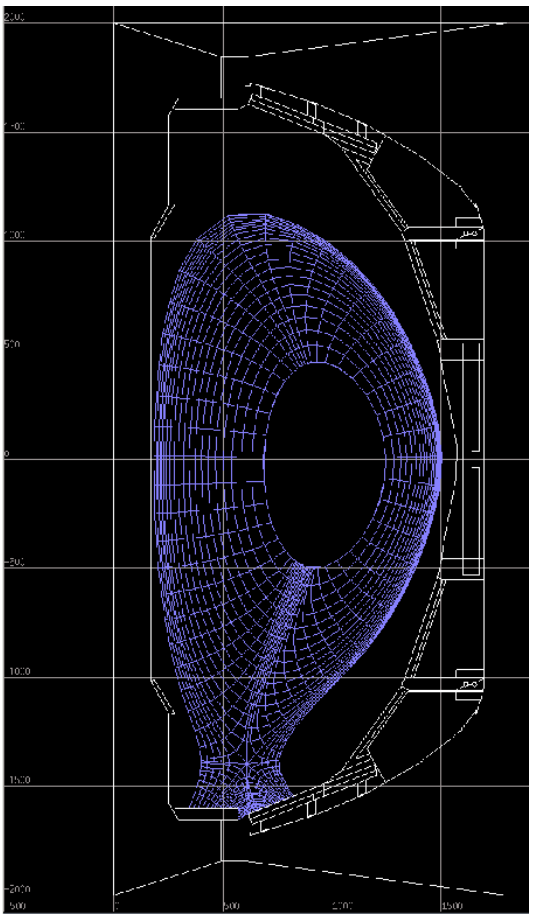
and then do (in `sbb`)

```
rm $OBJECTCODE/prgend.? $OBJECTCODE/prgini.? $OBJECTCODE/subsys.?
gmake
```

Appendix L

DG manual: Version 2.10

A. Kukushkin, K. Kukushkin
ITER Organization
A. Chin, D. Stotler
Princeton Plasma Physics Laboratory



This is the user's manual for the DG geometry definition code. Its primary objective is to simplify the specification of the geometry, and other input data, required for the simulation of the boundary of magnetic fusion devices. The most straightforward application of the code is to divertor tokamak geometries (having one or two X-points), although

in principle other magnetic configurations are possible. The flexible design of DG allows its output to be used with multiple magnetic fusion simulation codes, including B2-Eirene and DEGAS 2. In fact, some of the objects defined in DG and described in this manual may be designed specifically for use by these codes.

L.1 Introduction

This is the user's manual for the DG geometry definition code. Its primary objective is to simplify the specification of the geometry, and other input data, required for the simulation of the boundary of magnetic fusion devices. The most straightforward application of the code is to divertor tokamak geometries (having one or two X-points), although in principle other magnetic configurations are possible. The flexible design of DG allows its output to be used with multiple magnetic fusion simulation codes, including B2-Eirene and DEGAS 2. In fact, some of the objects defined in DG and described in this manual may be designed specifically for use by these codes.

L.1.1 Purpose of DG

DG is the backbone of the **Interactive Graphical Interface** package. It allows user to visualize the model geometry, to modify it, and to specify the geometry-related data necessary for a grid generator and modelling codes. It was originally designed to alleviate transfer of the divertor geometry from the CAD system to the B2-Eirene code package. However, it has proved to become a powerful tool of more general purpose, allowing one, e.g., to specify arbitrary shape of the divertor structures, to check visually the quality of the computational grids produced by a mesh generator, to analyse the error messages from the Monte Carlo routine, to assess easily the compatibility of the plasma equilibrium used in the calculations and the wall geometry of the machine, etc. It can also be very efficient as a sort of very simple drawing program to design the problem geometry from scratch, and not only for B2-Eirene. With use of this code, the most boring and time-consuming part of the preparatory work necessary for the edge plasma modelling in a realistic geometry, becomes really a pleasure. Try it and see for yourself!

L.1.2 DG objects

External objects

The external objects are the files which DG imports or exports. They normally have default extensions by which are used by the code when displaying the lists of files in the I/O control windows. The following table gives a brief description of the files used in DG.

	Description	Extension
Input files:		
DG model	All the data relevant to the model	.dg
Template	Geometry data from CAD drawing or other source	.ogr
Equilibrium	Values of the poloidal magnetic flux on a regular grid	.equ
Mesh	The computational grid to be used by B2-Eirene	
Output files:		
DG model	All the data relevant to the model	.dg
Output file	Specification of the *elements* and some other information - to be used by the interface routines producing the input files for B2-Eirene	.dgo
Targets file	Specification of the *targets* - to be used by the interface routines producing the input files for B2-Eirene and Sonnet or Carre	.trg
Structure file	Specification of the structures - to be used by the interface routines producing the input files for Sonnet or Carre	.str

continued on next page

continued from previous page

	Description	Extension
Configuration file:		
	The file containing information on the DG *layers* and *variables*.	.dgc

Table L.1: DG external objects

Internal objects

The internal objects are those with which you deal when working with DG.

Objects	Description
Drawing objects	
Points	Just points (displayed as red squares) which can be used to create *elements* or for diagnostic purpose.
Irregular points	Every point which is not regular. A regular point connects exactly two *elements* having the same orientation (see *External normals*)
Elements	Straight line segments (displayed in bright white) connecting *points*. Normally, they represent material surfaces in the neutral transport code ("additional surfaces" in Eirene parlance). You can edit elements (create, delete, split, join, change orientation, etc.) and assign *variables* to them.
Surfaces	Contours of constant poloidal flux (displayed in yellow) computed from *equilibrium* data. You can create, move, and delete them. The surfaces allow you to specify the requirements on the radial mesh for the mesh generator (Sonnet or Carre), and also to check the compatibility of your wall structures with the magnetic configuration.
Grid points	Points on the separatrix which represent partition of the magnetic surfaces in different regions for the grid generator (Sonnet or Carre). They are displayed as dashes in magenta, the separatrix being drawn in red.
Separators	Straight line segments (displayed in cyan) connecting the outer nodes of the computational *mesh* with surrounding *structure*. The separators also represent additional surfaces, but you can only re-position their outer edges. Their purpose is separation of *additional cells* - in terms of Eirene. As of V. 1.4 of DG, "separators" have been temporarily disabled.
Sources	Special points which can be grouped together (displayed as white asterisks). They can be used to specify the point sources of neutrals, - and also in calculation of radiation load onto the first wall.
Equilibrium	A simple visualisation of the equilibrium data - red and blue dashes corresponding to negative and positive, respectively, values of the poloidal magnetic flux.
Mesh	Computational mesh, displayed in blue, produced with a grid generator (Sonnet or Carre) and imported into DG. Allows you to check the quality of the mesh and to make the necessary corrections to the input data for the grid generator.
Chords	Straight line segments representing the Eirene and **b2plot** diagnostic chords.
Cells	Regions outside the computational mesh surrounded by the elements, separators, and the mesh edges (*additional cells* - in terms of Eirene).
External normals	Are displayed as magenta dashes and depict the orientation of the elements. They point to the right as you move from the beginning to the end of an element. This is the direction of the external normal in terms of Eirene.
Template	Imported graphics from external source (e.g. from CAD drawing or from old Eirene input file). The template is displayed in cyan, and it can be converted into *elements*.

continued on next page

continued from previous page	
Objects	**Description**
Axes	Coordinate axes which can be displayed on the drawing.
Grid	Coordinate grid which can be displayed on the drawing.
X-point	The X-point of the magnetic equilibrium; it effectively separates the divertor from the core plasma. In double-null configurations, there are two X-points which may or may not be on the same flux surface. The code must be told the topology of the equilibrium (see *topology*) in order to locate the X-points. This must be done before specifying *surfaces* or *grid points*.
Topology	The topology of the magnetic separatrix. A set of commonly used topologies is provided with DG. Once the *equilibrium* and topology have been imported, the code will attempt to locate the X-point(s); if successful, the separatrix (or separatrices) will be drawn in red.
Additional data	
Variables	Typically used to associate various pieces of data with the *elements*. This information can be then passed to other codes via the output files.
Layers	Groups of variables.
Targets	Bound the *surfaces* in the poloidal direction, and thus must be specified for the grid generator. They are thus analogous to the actual divertor targets in a tokamak.
Structure	Bound the *surfaces* in the radial direction. They are thus analogous to the walls of the vacuum vessel in a tokamak.
Target edges	The elements which should be reconnected to the corresponding corners of the calculational mesh in order to avoid mismatches of the mesh and the targets in the Monte Carlo calculations.

Table L.2: DG internal objects

L.2 Using DG

L.2.1 Starting DG

To start DG with a blank window, type:

```
dg &
```

at the UNIX prompt.

To start DG with an existing model (or several models) loaded, type:

```
dg <filename.dg> [...] &
```

After you start DG, the main window will appear.

L.2.2 Main window

After you start DG, the main window appears. It has four areas: menu bar, toolbar, work area and the message line.

Menu bar

The menu bar gives you access to the most of the `DG` functionality. It contains several buttons, which can be used to open the corresponding menus by clicking the left mouse button on it. Every menu can be torn off and left on the screen after opening it. In order to do this, just click at the dashed line at the top of the menu window. In this document, we will show menu items in `typewriter` font.

The following menus are available at present:

File You use this to start a new model, to work with external objects (to save or open the model; to import templates, equilibrium files, topologies, or computational meshes; to produce the output files which are used for generation of the input files for computational codes), or to quit the application.

Edit Allows you to perform some global operations, such as recovering from errors, selecting and de-selecting all elements, creation or removal of groups of objects, and shift or rotation of the picture.

View Here you can control the overall view of the model and specify which objects should be visible on the screen.

Commands Allows you to perform global commands on the model such as converting a *template* to *elements* and renumbering the elements.

Variables You will use this to specify additional geometry-related information for the output files (eg., location of targets, boundary conditions for the Monte Carlo routine, etc.).

Options From this menu you can tailor the main window and operating mode to fit you preferences. It also allows you to customize the DG variables *(only for experienced users!).*

Window Allows you to open more windows with different views of the same model, to close the current window, to open a window showing the statistics for the model, and to open the `Toolbox`.

Help This is the way to get some online help on DG and to identify its current version.

Toolbar

The toolbar allows you to assign different *tools* to each of the mouse buttons. It has 3 option menus-for the three buttons- that contain the list of available tools. If your mouse has less than three buttons, then only the leftmost option menu(s) are activated.

Work area

The work area is the area where the drawing is displayed. Here you can move around the drawing, and you can use *tools* to edit it. The work area can optionally display coordinate axes and coordinate grid to simplify positioning of objects.

Message line

The message line is located below the work area. It is used to display error messages and information on the objects. Error messages are sometimes displayed in an error box instead of the message line, and in this case you must click `OK` to proceed.

L.2.3 Using tools

Tools are used to perform actions in the work area. They allow you to create, move, modify, or remove objects, to query information on the objects and to move around the drawing. In this document, we will indicate tools in `italic typewriter` font.

A tool *must* be assigned to a mouse button in order to be used. You can select the tool to be assigned to each of the mouse buttons either from the toolbar, or from the `ToolBox` (accessible from the `Window` menu). There is one small difference between these two options. Operating with the toolbar, only the left mouse button should be used to select the tools associated with the three mouse buttons, On the other hand, when working with the `Toolbox`, a tool is associated with a given mouse button by clicking that button on the tool.

To use a tool, move the mouse pointer into the work area and depress the mouse button to which the tool has been assigned. Then, you can optionally drag the mouse. To complete the mouse drag, release the mouse button with the mouse pointer still inside the work area. To cancel the action associated with the drag, move the mouse pointer out of the work area and release the mouse button.

Some tools require you to "select" an object. To select an object, move the mouse pointer to the object and depress the mouse button. The object selected will be highlighted in another colour. The *green* colour means OK, while the *red* one indicates that the operation cannot be done. Normally, some explanatory text appears in such cases on the message line.

L.2.4 Using Undo

You can reverse the effect of almost any operation by issuing the `Edit|Undo` menu command. You can also reverse the effect of `Undo` command by `Edit|Redo`. The `DG` keeps track of practically all the operations you did starting from opening the window, and thus you can recover from almost all the errors you could make in the coarse of the model editing. To reverse the effect of all the Undo commands you issued, use `Edit|Redo` all.

L.3 Working with models

L.3.1 Opening an existing model

To open an existing `DG` model from `DG`, use the `File|Open...` menu command. A dialog box will appear prompting you for a filename. You can either select a file from the list or type the filename in the edit field. Then press `Ok`. A new window displaying the selected model will be opened. If you issue the `File|Open...` command immediately after starting `DG`, the file will be loaded in the current window. If an error happens, the error box appears. Press `Ok` to close the error box.

To start `DG` with an existing model loaded, specify the filename on the `DG` command line. If you specify multiple filenames, `DG` will open a separate window for each model.

L.3.2 Creating a new model

To create a new, empty model with `DG`, use the `File|New` menu command. A new blank window will be opened.

When you start `DG` without parameters, the `DG` window is blank.

L.3.3 Saving a model

To save a model, use the `File|Save` menu command. A dialog box will appear prompting you for a filename. If the current model has a filename assigned to it, this filename will appear in the edit field. In order to save the model under a different name, or to save the newly-created file, you should either enter the new name in the edit field, or select a name from the list (to overwrite an existing file). After that, press `OK`. An auto-save feature automatically saves a version of the file with a '$\sim$' appended to the name every 5 minutes (see `Options|Save interval`); of course, a new file must be saved before this can work. If you need to use this file (e.g., in the event that `DG` crashes unexpectedly), you should rename this file to remove the '$\sim$'.

L.3.4 Quitting DG

To quit DG, you must close all windows.

To close a window, issue the Window|Close menu command. You can also use the window manager to close a window. You can also use the window manager to close a window click the "X" at the top right of the menu. With Motif Window Manager, simply double-click the system menu. If you've made changes to the model in this window, a warning box will appear. Select Save to save the model. Select Discard to close the window without saving changes. Select Cancel to leave the window open.

To close all windows, use the File|Exit program menu command.

There is a caveat in the current version of DG that iconified windows containing unsaved work are not de-iconified after these commands. You will have to de-iconify them manually in order to see their warning boxes. All other iconified windows will be closed without problems.

L.4 Working with elements

L.4.1 Converting a template

It is often more convenient to convert a template into a set of elements and then edit them to suit your needs, rather than to define all elements by hand.

First, you should import the template. Choose the File|Template... menu command. A dialog box will appear prompting for a filename, similar to that for opening DG files. After importing the template, choose View|Picture view to adjust the zoom factor in order to make the template visible. Enable View|Axes and View|Grid to display the coordinate system.

DG lets you adjust the template position by moving it by a fixed distance and/or rotating it 90 degrees around (0,0). To adjust the position, choose the Edit|Move/Rotate... menu command. A dialog box will appear. Enable the Template check box. To move the template, enter X- and Y-increments and press Move. To rotate, press Rotate. If the template moves out of sight, again choose View|Picture view. Press Close when done.

To convert the template into a set of elements, choose Commands|Convert template.

Note that after the conversion, the template remains visible. This is convenient for editing elements, because the difference from the original template is visible. You can hide or display the template by disabling or enabling View|Template. To unload the template permanently, choose Edit|Delete|Template.

Elements produced in this manner are sometimes improperly connected or have wrong external normals. Manual editing of the elements may be required to remedy these problems.

L.4.2 Creating elements manually

There are several tools for creating new elements.

Use Add element to create the first element or to attach an element to a pre-defined point. Move the mouse pointer to the starting point and press the mouse button. Then move the pointer to the ending point and release the mouse button.

Use Split element to split an element into two. Select an element. It will split into two. Drag the newly created point to the desired location and release the mouse button.

Use Connect points to create an element between two pre-defined points. Select the first point, move the mouse cursor to the second one and release the mouse button. You cannot create more than one element between the same two points.

You can also use the Edit|Create|Point... menu command to create points that will be later connected using the Connect points tool. A dialog box will appear. Enter the desired coordinates and press Create. The dialog box will remain on the screen, allowing you to create more than one point without re-opening it. To close the dialog box, press Cancel.

L.4.3 Editing elements

There are several tools for modifications of existing elements.

Use Delete to remove an element. Select the element and release the mouse button.

Use Join elements to join two adjanced elements into one (reverse effect of Split element). Select the point between the elements and release the mouse button. You can only join elements around a regular point. There must be exactly two elements attached and their external normals must point to the same direction.

Use Move/Stretch to move a point with all attached elements. Select the point and drag it to the new location. Then release the mouse button.

Use Reposition to attach one end of an element to another point. Select the element near one of its ends, move the mouse pointer to another pre-defined point and release the button.

L.4.4 Changing external normals

Having the external normals of adjacent elements aligned permits them to be easily marked; some operations also require the normals of adjacent elements to be aligned (i.e., for the points involved in the operation to all be regular).

To display or hide external normals, enable or disable View|External normals.

Use the Reverse normals tool to change directions of external normals. Select an element to change its normal. Moving the mouse pointer to adjacent elements will force their normals to point to the same direction as that of the first element. Release the mouse button to end the action.

L.4.5 Renumbering elements

Each element has a unique integer label associated with it. These determine the order in which the elements are written into the output file. This feature is very useful for debugging. DG uses an internal counter to assign these numbers. Each time the user creates an element, the counter is incremented by one and its value is used as the element number. However, when the user deletes an element, the remaining elements are not renumbered and the counter is not decremented. Since many unused numbers consume disk space and may slow down programs that use DG output, these unused element numbers should be periodically eliminated. Renumbering elements is absolutely safe except when you are doing some debugging and need these numbers to identify elements in the output file.

To display or hide element numbers, enable or disable View|Display|Numbers.

To count unused numbers, choose Window|Statistics in order to display information about the current file. The 'unused numbers' entry contains this count.

To eliminate unused element numbers, choose Commands|Renumber elements.

L.5 Controlling how objects are displayed

L.5.1 Changing the zoom factor and location

You can use the `Zoom` tool to move around the drawing and to change the zoom factor. You can also use commands from the `View` menu.

To move around with the `Zoom` tool, press and release the mouse button anywhere in the work area without moving the mouse. The work area will be adjusted so that the location where the button was pressed is moved to the centre. Alternatively, you can press Shift+mouse button, move the mouse pointer and release the button. The work area will be adjusted so that the location where the button was pressed is now shown where the button has been released.

To change the zoom factor with the `Zoom` tool, press the mouse button, move the mouse in order to draw a rectangle around the area to be shown in the work area. Then, release the mouse button. Note that the area actually shown may be enlargened in order to preserve the aspect ratio.

The `View|Zoom In` command increases the zoom factor.

The `View|Zoom Out` command decreases the zoom factor.

The `View|Picture view` command adjusts the zoom factor and location so that all objects fit into the work area. This command ignores objects that are hidden because of settings in the `View` menu.

The `View|Selection view` command adjusts the zoom factor and location so that all marked objects fit into the work area.

L.5.2 Hiding and displaying objects

Toggle buttons in the `View` menu let you hide or display the coordinate system and various kinds of objects.

Note that you cannot select an object that is not displayed. In particular, you can only select points that are displayed (a small square is drawn around the point). The `View|Points` setting controls the display of points. When it is disabled, the `View|Irregular points` setting controls the display of irregular points. It is convenient to display only irregular points while correcting a converted template, because irregularities can be seen very easily.

L.5.3 Rotating the view

The display may now be viewed at any angle of rotation. The user may specify the desired angle by using the Rotate tool, the commands under the `View|Rotate` menu, or the equivalent keyboard shortcuts to increment (”Ctrl+(”), decrement (”Ctrl+)”), or zero the angle (`Rectify`, shortcut ”Ctrl+C”); see Fig. L.1. The rotation does not affect the underlying structures being displayed, but only the drawing procedure. The `Previous view` command will undo rotations. Rotations are saved starting with file format version 115. Files of older formats are still compatible; the default angle will be 0. Note that the `View|Picture view` command zeroes the rotation angle.

L.5.4 Stretching and shrinking the view

The aspect ratio may be ”unlocked” by enabling the `View|Mode|Stretch` button. In `Stretch` view mode, the `View|Stretch/Shrink` menu item becomes active (see Fig. L.2), and the user may use these tools to stretch or shrink the x and y axes. To stretch or shrink the current view by a factor of 2, use the first four buttons of this menu. The keyboard shortcuts are ”Ctrl+Shift+H” to stretch horizontally (in the x direction), ”Ctrl+H” to shrink horizontally, ”Ctrl+Shift+V” to stretch vertically (in the y direction), and ”Ctrl+V” to shrink vertically. Precise stretching or shrinking factors can be specified with the `Stretch ...` dialog. The aspect ratio can be reset with the `Justify` menu item. Additionally, zooming with the highlighted rectangle now zooms to display exactly the contents inside the

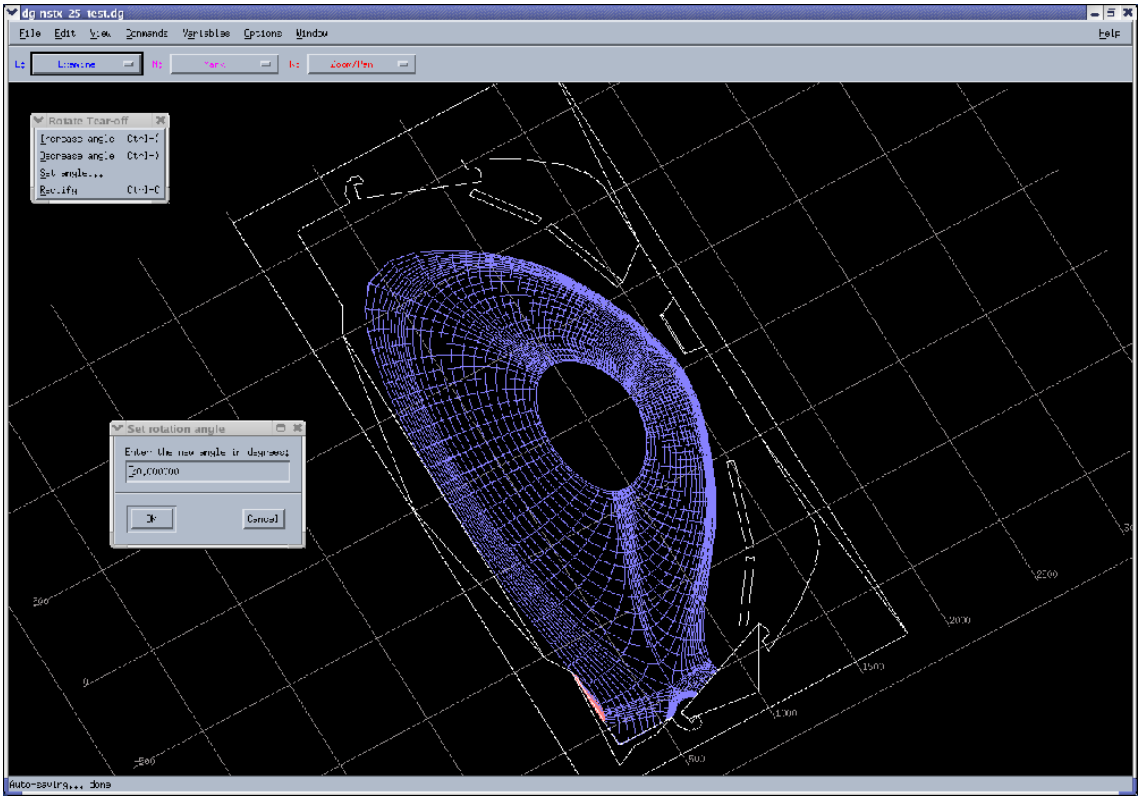


Figure L.1: Demonstration of a rotated view with associated dialog box and tear-off menu.

rectangle, rather than expanding to ensure the proper aspect ratio. Disabling `Stretch` mode redraws the display with proper aspect ratio and preserves the ratio in future changes of view.

The `Stretch` tool works a little differently since it can stretch either horizontally or vertically. If the mouse is within 45° of an imaginary horizontal line through the center of the window, a mouse drag will stretch or shrink the view in the horizontal direction. Likewise, if the mouse is within 45° of the vertical line through the center of the window, a mouse drag stretches or shrinks in the vertical direction. If `Stretch` mode is disabled, the `Stretch` tool does nothing.

Note that the `View|Picture view` command resets the aspect ratio, but does not disable `Stretch` mode.

L.5.5 Top-down view

To facilitate visualization and specification of chords with nonzero toroidal extent (see Sec. L.7), a view from above the torus is now available via the `View|Mode|Top-down view` button. When this button is enabled, the main window depicts the view of the present geometry looking down along the y axis. The elements in the model are assumed to be toroidally symmetric. In this view, the elements are displayed as 48 (arbitrarily chosen) straight line segments. The only other objects currently displayed in this view are chords, although working drawing procedures exist for all markable objects. Since displaying every element in the model would likely result in confusion, the user must select the specific elements to be depicted. The variable `Top View Elements` has been defined for this purpose (`Variables|Top View Elements`). The appearance of chords (with some toroidal extent) can be toggled with either `View|Display|Chords` or `View|Display|3dchords`. As in the normal (poloidal) view, `View|Display|Elements` toggles the appearance of elements in the top view. Since top view was intended to be used only for chords, many tools are disabled in this mode. Likewise, since top view uses the same screen coordinates as the normal view, it may be necessary to re-zoom after switching between the two views. `View|Picture view`

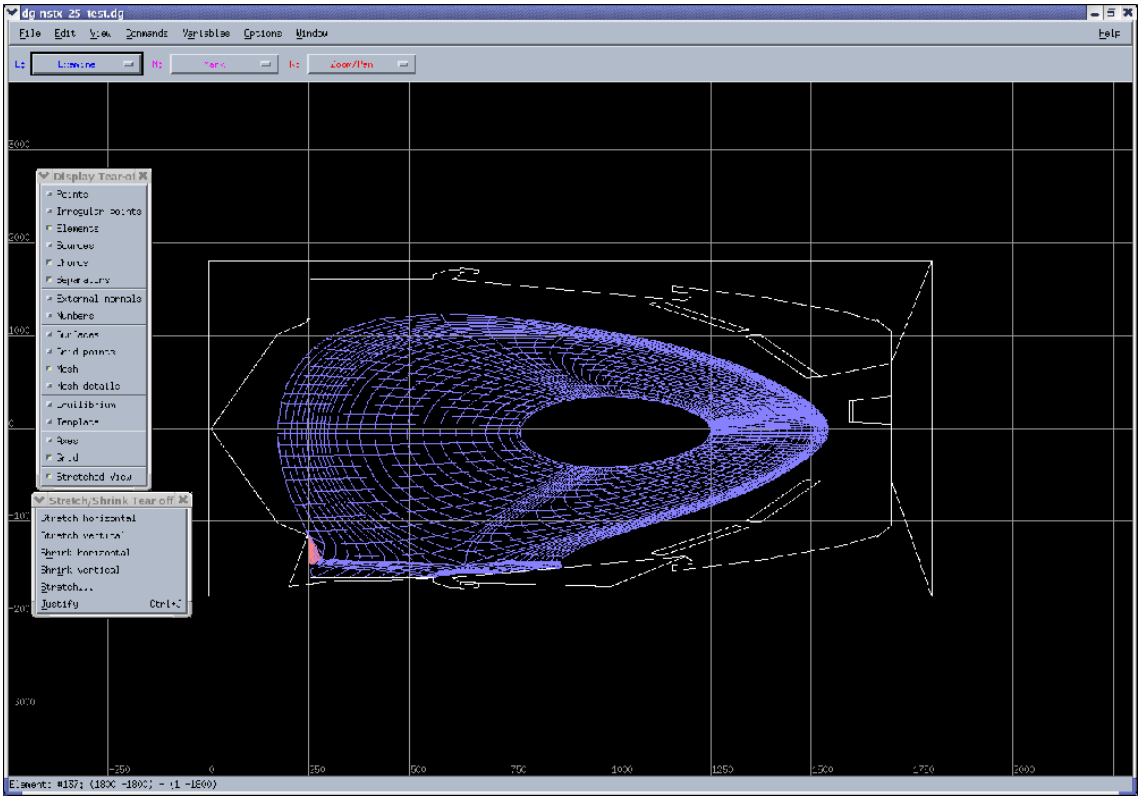


Figure L.2: Demonstration of the variable aspect ratio capability along with the associated tear-off menu.

can be used for this purpose in either mode.

L.6 Working with magnetic surfaces and grid points

L.6.1 Equilibrium conversion and manipulation

The `${SOLPSTOP}/modules/DivGeo/equtrn` directory has some utilities for converting between various widely used formats for magnetic equilibrium files and for making minor modifications to them.

The listings and documentation below may be out of date. An updated documentation of the supported scripts can be found in `${SOLPSTOP}/modules/DivGeo/equtrn/doxygen/refman.pdf`.

d2d

This is a script that invokes the `dg2dg` executable. Its sole function is to double the resolution of a `DG` equilibrium file. To invoke it:

```
d2d <filein>.equ <fileout>.equ
```

d2e

This is a script that invokes the `dg2ef` executable to convert a `DG` equilibrium file to the `EFIT` format (specifically, the `EFIT` "g" file). This is invoked with (the file suffixes are recommended for clarity and are not enforced):

```
d2e <filein>.equ <fileout>.efit
```

Some older DG files lack the magnetic field data needed for this conversion. The default name for the file containing its value is field.snn (i.e., using the above invocation). Alternatively, the name of the file containing the field can be specified as a third argument:

```
d2e <filein>.equ <fileout>.efit <fieldfile>
```

d2v

This is a script that invokes the dg2vr executable to convert a DG equilibrium file to the Carre (or TdeV) equilibrium format. This is invoked with (the file suffixes are recommended for clarity and are not enforced):

```
d2v <filein>.equ <fileout>.v
```

Some older DG files lack the magnetic field data needed for this conversion. The default name for the file containing its value is field.snn (i.e., using the above invocation). Alternatively, the name of the file containing the field can be specified as a third argument:

```
d2v <filein>.equ <fileout>.v <fieldfile>
```

dg_minuspsi_dg

This is a program that flips the sign of a DG equilibrium file. This is invoked with (the file suffixes are recommended for clarity and are not enforced):

```
dg_minuspsi_dg <filein>.equ <fileout>.equ
```

dg_symm_dg

This is a program that symmetrises a DG equilibrium file. It is recommended to be used when dealing with a quasi-CDN equilibrium to be modelled as a true CDN. This is invoked with (the file suffixes are recommended for clarity and are not enforced):

```
dg_symm_dg <filein>.equ <fileout>.equ
```

Some older DG files lack the magnetic field data needed for this conversion. The default name for the file containing its value is field.snn (i.e., using the above invocation). Alternatively, the name of the file containing the field can be specified as a third argument:

```
dg_symm_dg <filein>.equ <fileout>.equ <fieldfile>
```

A fourth optional (real) argument can be added to mix in a fraction of the old equilibrium with the symmetrised one, so as to not fully symmetrise it, but rather change the distance between the two X-points.

e2d

This is a script that invokes the ef2dg executable to convert an EFIT format (specifically, the EFIT "g" file) equilibrium to the format read by DG. This is invoked with (the file suffixes are recommended for clarity and are not enforced):

```
e2d <filein>.efit <fileout>.equ
```

Another program that produce output in an EFIT-like format that is readable by DG using this script is FreeGS[1].

[1] Documentation available at https://freegs.readthedocs.io/en/latest/index.html.

move_equ

This script rigidly moves an equilibrium `.equ` file, in the radial and/or vertical directions. The syntax is:

```
move_equ <filein>.equ <fileout>.equ [<fieldfile>]
```

The third argument `<fieldfile>` is optional and has the same meaning as in the `dg_symm_dg` script above.

The user is then prompted to provide the radial and vertical shifts (in metres) to apply to the equilibrium.

n2d

This is a script that invokes the `nk2dg` executable to convert an equilibrium in the "Naka" format to the format read by `DG`. This is invoked with (the file suffixes are suggested for clarity and are not enforced):

```
n2d <filein>.nk <fileout>.equ
```

rps

This is a script that invokes the `risepsi` executable. Its sole function is to increase the poloidal flux values in the `DG` equilibrium file by the specified amount. To invoke it:

```
rps <filein>.equ <fileout>.equ <increment>
```

v2d

This is a script that invokes the `vr2dg` executable to convert an equilibrium in the `Carre` (or TdeV) format to the format read by `DG`. This is invoked with (the file suffixes are recommended for clarity and are not enforced):

```
v2d <filein>.v <fileout>.equ
```

By default, this script will look for the value of the magnetic field in a file `field.snn`. Alternatively, the name of the file containing the field can be specified as a third argument:

```
v2d <filein>.v <fileout>.equ <fieldfile>
```

cropequ

This is an executable (no corresponding script) that crops a `DG` equilibrium file. The code is invoked with (the file suffixes are recommended for clarity and are not enforced):

```
cropequ <filein>.equ <fileout>.equ
```

The code will print out the dimension and spatial extent of the input equilibrium grid then request the values for the output grid in the order: number of radial grid points, minimum radius, maximum radius, number of vertical grid points, minimum vertical coordinate, maximum vertical coordinate. The code will require the requested spatial extent to lie within that of the input grid. If it does not, the offending parameter is reset to match the corresponding value from the input grid.

The output file will contain a new equilibrium, re-computed with the new specified grid dimensions, with interpolation of the data to the new grid point locations.

dgcut

This is an executable (no corresponding script) that chops a DG equilibrium file. The code is invoked with (the file suffixes are recommended for clarity and are not enforced):

```
dgcut <filein>.equ <fileout>.equ
```

The code will print out the spatial extent of the input equilibrium grid then request the values for the output grid in the order: minimum radius, maximum radius, minimum vertical coordinate, maximum vertical coordinate. The code will require the requested spatial extent to lie within that of the input grid. If it does not, the offending parameter is reset to match the corresponding value from the input grid.

The output file will contain only the data from the original input file that is within the new coordinate bounds being prescribed.

prinequ

This is an executable (no corresponding script) that prints out the contents of a DG equilibrium file in tabular format. The file is read from standard input and the result is sent to standard out. So, the preferred invocation would be (the file suffixes are recommended for clarity and are not enforced):

```
prinequ < <filein>.equ > <fileout>.txt
```

L.6.2 Importing equilibrium

An equilibrium file must be imported before defining any surfaces or grid points.

To import an equilibrium file, choose File|Import|Equilibrium... . A dialog box will appear prompting for a filename. Select a file and press Ok. Select View|Picture view to make sure the equilibrium grid is visible in the work area. Red colour represents positive ψ values, while blue represents the negative ones.

L.6.3 Output mode

The detailed procedures for creating and editing surfaces and grid points differ slightly for the Sonnet and Carre mesh generators. So, before undertaking these tasks, the user should select the appropriate output mode in Options |Output mode. Since only the Carre code is distributed along with this manual, only procedures pertinent to it will be described.

This dialog box also allows the user to determine which objects are examined for consistency when DG's output is requested (via File|Output...). Selecting all of these checks is recommended. Users should override them only if they are aware of the consequences of the object in question being improperly defined.

L.6.4 Creating and editing surfaces

As was noted in Sec. L.1.2, surfaces are drawn between targets and are bounded by structures. The target related variables are single elements set using the variables Variables|Inner edge of inner target, etc., where the first "inner" and "outer" refers to the inner and outer edge of the computational mesh, respectively, and the second to the usual inner and outer divertor target. There is an additional set of targets available for double-null geometries; see: Variables|Add|Extra targets for DN. The structures are set with the variable Variables|Structure. There may be one or more of these.

In Carre mode, the user begins the process of surface creation by manually creating the surface that will represent the innermost core surface via the Add surface tool. Just click it inside core plasma at the desired location (the

tool will not work anywhere else). Its location can be altered via the Move tool. It can only be deleted by executing the Edit|Delete|Surfaces command (this deletes *all* surfaces).

The next step is to open the Create surface(s)... dialog, Edit|Create|Surfaces... (see Fig L.3). In Carre mode, the user can only create multiple surfaces (i.e., the Single option will not work). The Area menu determines in which part (topologically) of the equilibrium the surfaces will be drawn. The number of cells must be input manually. The line graph in the dialog box is a simple depiction of the relative spacing of the surfaces. In Carre mode, only the Delta Law is available. In this case, the two parameters Delta1 and Delta2 specify a quadratic function used to determine this spacing (The deltas specify the relative normalized difference between the levels of the first two, and the last two surfaces).

While these parameters can be input directly, the preferred means of setting them is to click and drag the black line in the graph to achieve the desired spacing variation. When the Create button is clicked, the surfaces will be generated with this spacing and drawn in DG's main window. If a model has an existing set of surfaces that the user wishes to tweak, click on one with the Examine tool and then click the << button in the Create surface(s)... dialog; this will import the parameters describing these surfaces.

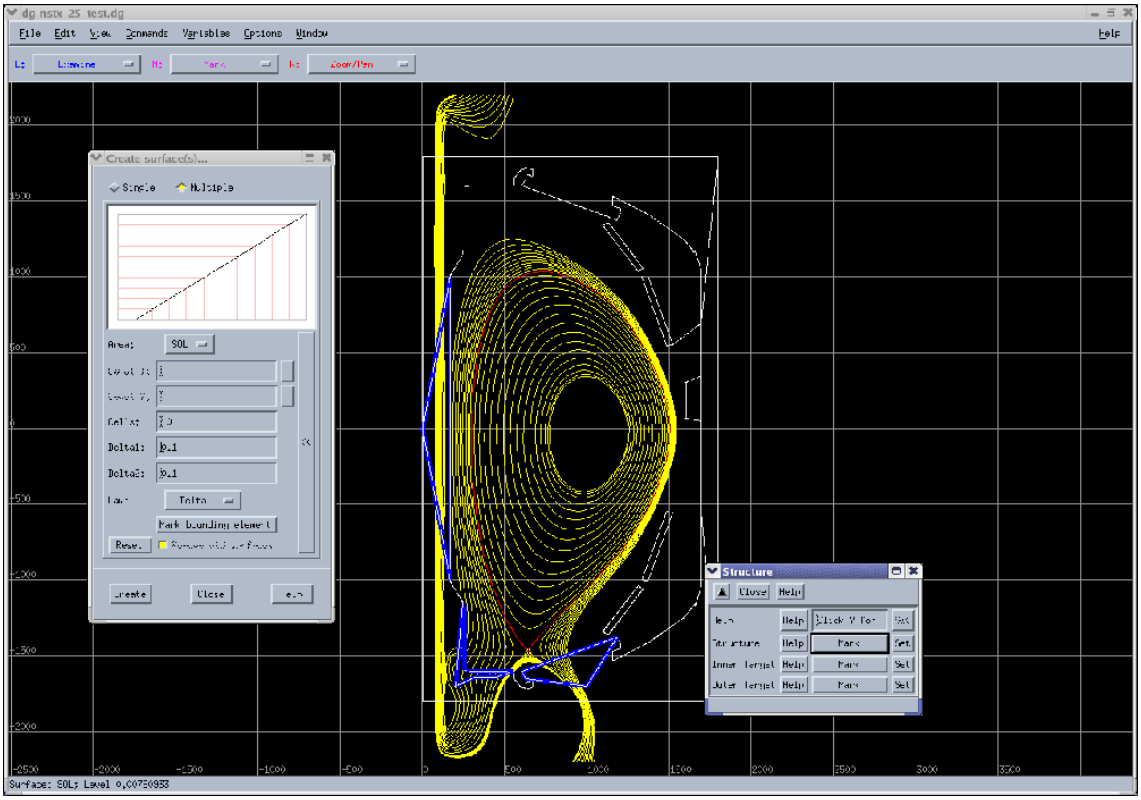


Figure L.3: Create Surface(s)... dialog with highlighted Structures.

Note that the dialog will *not* enforce constraints on the number of cells (e.g., that the number of core and PFR cells must be equal); the user must check to be sure that all such constraints are satisfied. Carre will perform its own set of checks when invoked as well.

DG's algorithm for calculating the outermost flux surface level for unbounded regions (e.g., PFR and SOL) uses the Structure and Inner / Outer targets variables. The only elements that the algorithm is aware of are those specified in the Structure variable, shown in Fig. L.3. Additionally, the algorithm recognizes at least two subsets of the structures, the "targets", which consist of all closed loops in the structure on which elements in the Inner target and Outer target variables lie. All target elements must lie on a closed loop in order that the targets have a well-defined "inside" and "outside". The outermost surface level is then defined as the flux value differing most

greatly from the value on the separatrix, whose corresponding flux surface passes through every target but encounters no other elements in the structure. This value is computed, to as much accuracy as the resolution of the equilibrium allows, as the least of 1) the global maximum on each target and 2) the global minimum on all other elements in the structure, where maximum and minimum are understood to be in terms of absolute deviation from the separatrix flux.

With this procedure, it is clear that the outermost level must be a flux extremum of some element of the structure; we call this element the "bounding element". The capability was added to DG to record this bounding element during surface creation and mark it from within the Create Surface(s) ... dialog. See, e.g., Fig. L.4. Note that the core region has no bounding element. By identifying what DG considers as the bounding element, we may easily adjust the extent of the flux surfaces by moving the bounding element to change the level of its extremum. Note that once the structure is changed, the bounding element might different than before.

In case of difficulty ...

The process of creating surfaces is perhaps the most confusing and difficult aspect of using DG. Moreover, the algorithm it uses for choosing the outermost flux surface level differs in detail from the one used by Carre. If you are having problems getting the surfaces you want, you might benefit from reading the precise description of DG's current algorithm for setting the outermost flux level:

It is to be understood that "maximum" and "minimum" always refer to the absolute flux deviation from the separatrix. First, find the surface zone and the separatrix(ces) of the zone. If there are two separatrices, draw surfaces from one to the other. Else if the zone is marked as "limit by surface", find the farthest surface in the zone and draw surfaces to it. (In practice, only one surface by be added manually in this zone, so that surfaces are always bounded by that one.)

Else, initialize the maximum level to infinity and get the structure variable, which includes the targets. The structure is split into chains of adjacent elements, where each element in a chain may be connected to at most one other element at each endpoint. If there are branching points in any chain, an error is returned. The chains are organized into three groups: the first group consists of chains which contain elements that have been specified as a target. These chains must contain the entire target variable and must form a closed loop, otherwise an error is returned. The second group consists of chains with no target elements that form a closed loop, and the third group consists of chains with no target elements that do not form a closed loop.

For each chain in the first group, for each element in the chain, if neither endpoint is in the surface zone, skip it, else get the maximum and minimum flux levels on the element from the equilibrium. Find the maximum level on the chain, and set the surface level maximum to the innermost of these levels. Also, find the minimum level on the chain, and set the surface level minimum to the outermost of these levels.

For each of the elements in the second or third group, if the first endpoint is not in the surface zone, skip it, else get the minimum flux level on the element. If it is less than the current surface level minimum, skip it, else if it is still less than the current surface level maximum, set the surface level maximum to this level.

You can use the Add surface tool to create surfaces visually. It is also possible to create surfaces with the Edit|Create|Surface menu command. To move or delete surfaces, use Move/Stretch and Delete tools.

To use the Add surface tool, press the mouse button anywhere in the equilibrium. A surface will appear. Move the mouse pointer to adjust the surface position. When ready, release the mouse button.

The Edit|Create|Surface menu command is similar to Edit|Create|Point. A dialog box will appear prompting for coordinates. After you press Create, a magnetic surface will be drawn through the specified location.

To use Move/Stretch and Delete tools, select a surface. Then drag it to the new location and release the mouse button resp. simply release the mouse button to delete it.

To delete all surfaces, use the Edit|Delete|Surfaces menu command.

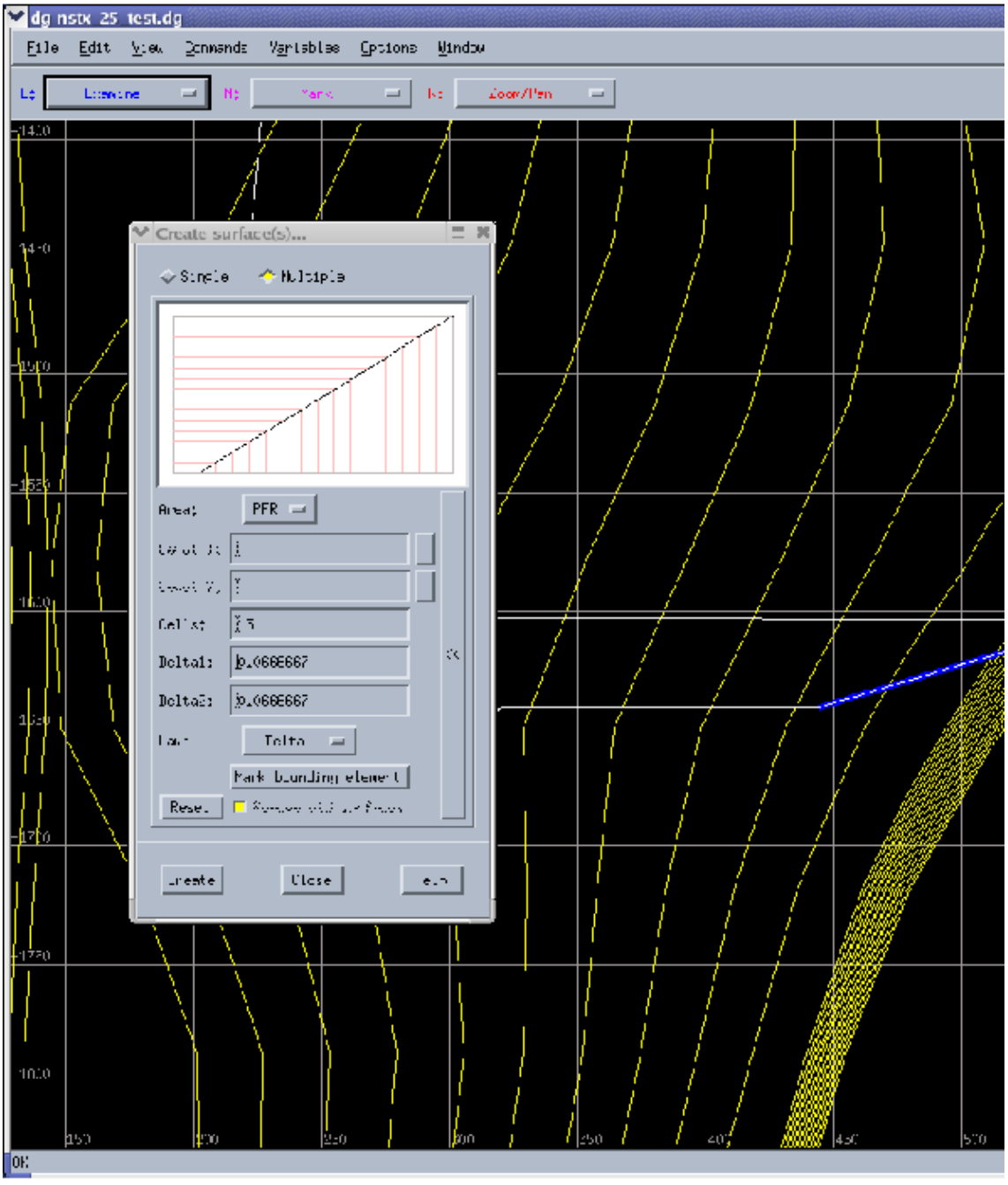


Figure L.4: Zoom of the PFR showing the highlighted ”bounding element”.

L.6.5 Defining the X-point

In some older DG version, you must define the X-point before working with grid points. This is necessary if version of DG used cannot detect the X-point location automatically. By defining it, you give DG a hint how to display grid points and process user input. DG does not write the X-point location into the output file.

For performance reasons, the separatrix is drawn as series of line segments. Because of this, it is not displayed very accurately, especially near the X-point.

First, define the targets so that they are intersected by the separatrix. Then, use the Set X Point tool to define a rectangle around the X-point. Press the mouse button at one corner of the rectangle, move the mouse pointer to the opposite corner and release the mouse button. In case of success, the separatrix will be drawn in the work area. Otherwise, an error message will be displayed in the message line. If you get the Invalid X point location error message, try selecting a bigger rectangle.

Note that the separatrix inside this rectangle is represented by two intersecting line segments that could even intersect with surfaces!

With an X-point defined, you can use the Set X Point tool to redefine it.

L.6.6 Limiter and linear configurations

If you are dealing with a limiter or linear configuration, DG needs to be "tricked", because its internal algorithm requires some quantities that can only be computed in a divertor topology. The implemented solution is to provide two sets of variables, one of which is a dummy only used to satisfy the DG internal requirements, and another set that provides the actual case information.

The procedure to follow is:

1. Import a single-null Equilibrium. The best would be something of a very different size, so that it lies completely outside the geometry you really want to model. Make sure the sign of the single-null equilibrium is the same as that of your limiter configuration (use the dg_minuspsi_dg program to change its sign if necessary, see Section L.6.1). Import the single-null Topology. Create a Surface in the core region of the diverted equilibrium as usual - preferably at about the same depth that you will want for the limiter case grid. Create also two closed polygons to act as targets for this SN equilibrium. Declare them as Inner target and Outer target, and include them as part of the Structure in the usual way, as decribed in Section L.6.4. Also include these target polygons in the Elements not for Eirene list.
2. Add the Limiter and linear configurations set of variables, and cut and paste the name of the limiter configuration equilibrium file into the Real equilibrium there. If treating a linear case, leave the name of Real equilibrium to be its default value of -. Specify the depth of the grid you wish to obtain by setting the Core width (in units of m).
3. Include the limiter or linear device walls in the Structure, plus some grid-limiting parts as needed.
4. The Target specification variables refer to the two halves of the limiter on either side of the tangency point, or the two ends of the linear device, as the case may be. Provide there all relevant material and sputtering properties. For limiter cases, Targets #1 and #2 should be positioned so that you travel clockwise around the plasma when going from Target #1 to Target #2. For linear cases, the poloidal index will increase as you go from Target #1 to Target #2. Use the real elements you want to be reconnected to the grid corners as the SOL edges. For the PFR edges, create two fake elements and specify 0 as the Surface Type in the General Surface Data for these two elements. Although these may be fictitious elemens, they must NOT be included in the Elements not for Eirene list.
5. Create the Surfaces and Grid points as usual. Only the SOL zone for the Grid points and the SOL and Core areas for the Surfaces are meaningful. The Grid points in the Core will provide the

poloidal (or axial) resolution. For linear cases, the radial point distribution will be given by the SOL number of Surfaces and the number of Cells in the Core should be set to 1.

6. For limiter configurations, make sure that the Major radius in Global Eirene data is set to its real positive value (in units of cm). For linear devices, set it to any nonzero positive number.
7. As usual, define all other variables (Surface data, Gas puff, Plasma species, etc...) as needed. When defining the TRIA-EIRENE parameters, define the contour indices as they pertain to the limiter or linear configuration geometry description.

Once you are finished setting up all the case variables, follow the usual procedure: Output the case files, use the lns script to establish the necessary symbolic links (the scripts checks if you are doing a limiter case, and, if so, will have dg.equ link to the file you indicated as the Real equilibrium), then use the carre - script to create the grid, Import the Grid and inspect it. For linear cases, you cannot use Carre and must provide the grid independently. You may have to play with the usual settings for the grid generation, such as the Surfaces and Grid points together with the Carre guard length in the Target specifications, and also with the Core width in the Limiter configuration in order to get the grid which suits you and is acceptable for the plasma modelling code (no pink segments in the grid). If you find it impossible to create the necessary distribution of the surfaces or the grid points because of local extrema appearing on the visualisation curve, then try to adjust the Radial scale or Poloidal scale in the Limiter and linear configurations layer.

L.6.7 Snowflake configurations

Similar to limiter or linear configurations, DG needs to be "tricked", because it is not yet a valid topology in DG.

The procedure to follow is:

1. Import a valid (preferably double-null) Equilibrium and Template for the same machine as the desired snowflake. Declare the structure, Inner target and Outer target, as if preparing the double-null stand-in.
2. Add four targets to the Target specification variables, the edges of which should correspond to the snowflake grid, not the double-null stand in. Provide there all relevant material and sputtering properties.
3. Identify the four contours for the TRIA-EIRENE parameters variable that apply to the snowflake grid, not the double-null stand-in. Export these four surfaces as templates, which must be used to manually generate a tria.in file later (there are also Python routines to do this in the snowflake example run).
4. Create the Surfaces and Grid points for the double-null stand in. This step is done because DG needs surfaces and grid points before outputting.
5. As usual, define all other variables (Surface data, Gas puff, Plasma species, etc...) as needed.

Once you are finished setting up all the case variables, follow the usual procedure: Output the case files, use the lns script to establish the necessary symbolic links. Rather than running carre, simply run uinp with the externally generated snowflake grid file. This should generate an input.eir file, which will house the desired material and strata data. However, the plasma data will be relevant to the double-null grid, not the desired snowflake. As such, it should be edited later to contain correct plasma information. Instructions to do this are located in the snowflake example run directory.

L.6.8 Creating and editing grid points

The Add grid point tool lets you visually define grid points. In addition, the Edit|Create|Grid point(s) menu command lets you define a grid point by specifying its value or fill a zone on the separatrix with a given number of grid points. In Carre mode, the only available method for creating grid points is with the multiple

option of the Create grid point(s)... dialog box. In particular, the Add grid point tool will not work, nor will the Move or Delete tools have any effect on the grid points.

In Carre mode, the operation of this dialog box is directly analogous to that used to create surfaces (Sec. L.6.4). The "zone" refers to the section of the separatrix along which grid points are to be created. The number of cells must be input manually. The line graph in the dialog box is a simple depiction of the relative spacing of the grid points. In Carre mode, only the Delta Law is available. In this case, the two parameters Delta1 and Delta2 specify a quadratic function used to determine this spacing. While these parameters can be input directly, the preferred means of setting them is to click and drag the black line in the graph to achieve the desired spacing variation. When the Create button is clicked, the grid points will be generated with this spacing and drawn in DG's main window. If a model has an existing set of grid points that the user wishes to tweak, click on one of these grid points with the Examine tool and then click the << button in the Create grid point(s)... dialog; this will import the parameters describing these grid points. Note that the dialog will *not* enforce constraints on the number of cells (e.g., that the number of grid points on either side of the PFR must be equal); the user must check to be sure that all such constraints are satisfied. Again, Carre will perform its own series of such checks when invoked.

With the Add grid point tool, move the mouse pointer to the desired location on the separatrix and press the mouse button. A grid point will be created at this location. You can move the mouse pointer to adjust its position. When finished, release the mouse button.

The Edit|Create|Grid point(s) menu command brings up a dialog box prompting you for a separatrix zone and alternatively, number of grid points to create or a grid point value. Two radio buttons let you choose between filling the zone with grid points and creating a grid point with an exact value. After entering values, press Create. Filling a zone with grid points will remove all previous grid points from that zone.

Use the Move/Stretch tool to move a grid point or to "stretch" grid points of one zone. To move a single grid point, select it, then drag it to the new location and release the mouse button. To "stretch" grid points, move the mouse pointer to one of them, press Shift+mouse button and drag it to the new location. All other grid points in that zone will be "stretched".

Use the Delete tool to remove grid points. Select the grid point to delete and release the mouse button.

Use the Edit|Delete|Grid points menu command to delete all grid points.

L.7 Working with chords

Chords are presently the only objects in DG that are not confined to the poloidal plane (i.e., that may be non-axisymmetric). For purposes of defining chords in three dimensions, two coordinate modes have been established. In "cartesian" mode, Z is the vertical axis in the "top-down view" (Sec. L.5.5); the horizontal axis in the poloidal plane is X and the vertical axis is Y (note: this is a left-handed coordinate system). In "cylindrical" mode, Z is the vertical coordinate in the poloidal plane, identical to Y in "cartesian" mode. In the "top-down view", the horizontal axis is designated as R and the toroidal angle as φ (counterclockwise relative to the horizontal axis) is entered in degrees.

Chords can be displayed with either View|Display|Chords or View|Display|3dchords. For chords confined to the poloidal plane, these are identical, and chords will not be visible in the "top-down view". For chords with a toroidal extent, selecting View|Display|Chords in the poloidal plane results in straight line segments connecting the X and Y "cartesian" coordinates of the chord end points. Equivalently in the "cylindrical" mode, these are the R and Z coordinates at the end points. In contrast, View|Display|3dchords shows the projection of the toroidal path of the chords on the $Z = 0$ ("cartesian") or $\varphi = 0$ ("cylindrical") plane. In the "top-down view", both View|Display|Chords and View|Display|3dchords show the straight line paths of the chords in the X-Z ("cartesian") or $R\cos\varphi$-$R\sin\varphi$ ("cylindrical") plane.

Starting with file format version 115, chords with X, Y, and Z components may be saved and loaded.

L.7.1 Creating chords

The Add Chord tool allows you to create chords graphically. To create a chord, move the mouse pointer to the desired location of its beginning and press the mouse button the tool is assigned to. Without releasing the mouse button, move the mouse pointer to the desired location of the other end of the chord and release the mouse button. Chords can also be created from marked Elements via the Commands|Convert|Elements to chords command. Duplicate chords are not produced and the original elements are deleted.

The third method for creating chords is with the Edit|Create|Chord... dialog.

L.7.2 Modifying chords

The Move tool allows chords to be manipulated in either the poloidal plane or in "top-down view". The user can manipulate either the Chords object (displayed by View|Display|Chords) or the 3dchords object (displayed by View|Display|3dchords. The Adjust chord tool moves an end point of the chord while keeping its radial coordinate constant. Holding down the "shift" key while using the Move tool replicates this behavior, but with slightly different graphics that conform to the rest of the Move tool.

Chords have an orientation, just like elements; this orientation is displayed with View|Display|External normals. The Reverse normals tool can be used to change this orientation. Note that when this tool is dragged, it will only operate on objects of the same type as the first object encountered, either chords or elements.

The Extend chord tool linearly extends (i.e., replaces) the second endpoint of a chord to lie on the nearest element, or the nearest marked element if elements have been marked. The Edit|Extend chords commands applies does this for all marked chords. The Edit|Mark all chords command can be used to mark all chords in preparation for extension.

Note: as of August 2007, the "mathematically correct" procedure for extending chords has not been debugged. However, two macros defined in chords.c may be used to select a different preferred implementation of the procedure. EXTCHORD_2D selects the 2-D procedure, which does not support chords with nonzero toroidal extent, and EXTCHORD_OLD selects the partially correct, stable 3-D procedure, which does not work for horizontal chords and ignores horizontal elements. (This is enabled in the current release of DG). If neither macro is defined, the code defaults to the incomplete procedure, which may show unpredictable behavior during use.

The Edit|Delete|Chords command deletes all chords.

L.8 Working with sources

L.8.1 Creating and editing sources

DG allows the user to define sources and to attach variables to them.

To create sources, use the Add source tool. Move the mouse pointer to the desired location in the work area and press the mouse button. A source will be created at this location. You can move the mouse pointer to adjust its position. When finished, release the mouse button.

To create a source by entering its coordinates, select Edit|Create|Source... . A dialog box will appear prompting for coordinates. Enter values and press Create. A source will be created. The dialog box will remain on the screen, allowing you to define multiple sources without re-opening it. Press Cancel to close the dialog box.

To move a source to a new location, use the Move/Stretch tool. Select the source to move, drag it to the new location and release the mouse button.

To delete a source, use the Delete tool. Select the source to delete and release the mouse button.

To delete all sources, issue the Edit|Delete|Sources menu command.

L.9 Working with meshes

L.9.1 Importing a SONNET format mesh

DG allows you to import a mesh in the Sonnet format. Carre's output mesh can be converted into this format, as can meshes produced by other codes. At present, it can only read the ASCII version of the grid file. The grid can then be displayed in the working area.

To import the mesh, use the File|Import|Mesh... menu command. A dialog box will appear prompting you for a filename. Select a file from the list or type the filename in the edit field. Then press Load. The mesh will be imported and displayed in the work area. If nothing is displayed, select View|Picture view to enlarge the area displayed. Also, make sure that View|Display|Mesh is enabled.

To hide or display the mesh, toggle View|Display|Mesh. To unload it, select Edit|Delete|Mesh.

L.9.2 Modifying a mesh

DG's graphical interface permits limited modification of a mesh that has been imported. The altered mesh can be written out to a text file (in the Sonnet format) for use by other codes. The primary tool for this purpose is the Move mesh point tool. The actions of this tool are governed by the settings specified in the Options|Mesh editing... dialog. They only affect one-dimensional movements of mesh points.

Slide along specifies how movements of mesh points are confined. When Magnetic surfaces is in effect, the movement is confined to a magnetic surface passing through the original location of the mesh point. This method should work fine except for the vicinity of the X-point, where magnetic surfaces drawn by DG become distorted. When Splines is in effect, the movement is confined to a spline passing through a row of existing mesh points. This method may be useful in the vicinity of the X-point.

Margin specifies the minimum distance between adjacent mesh points. If this limit is about to get exceeded, adjacent points start moving in order to "make room". The Double border setting, when enabled, instructs DG to treat outermost mesh cells as a double border. They would be resized together with adjacent "regular" cells.

A modified mesh can be written out with the File|Export|Mesh... dialog.

L.9.3 Repairing mesh cells using rotated view and variable aspect ratio

Mesh cells highlighted in pink in DG's main window are either concave (have an interior angle greater than 180 degrees) or consist of overlap with a neighboring cell. These problems typically arise where the mesh generator (Carre or otherwise) is forcing the mesh to conform to the specified divertor target shape and usually involve cells that are stretched significantly in one direction.

The ability to rotate and stretch the view allows the structure of the problem cells to be discerned and repaired. As an example, we consider a certain mesh generated for the NSTX experiment which contains twisted cells near the inner divertor, Fig. L.5. By zooming, stretching, and rotating the view using the tools described in Secs. L.5.3 and L.5.4, we arrive at a much clearer depiction of the problem, Fig. L.6. We see that the edges of the cells crossed the enforced boundary, twisting the mesh cells in the process. At this point, the user can straightforwardly use the Move mesh point tool to manually repair the cells.

L.9.4 Defining cells

With DG, you can define cells between the Sonnet grid and the structure (both objects must be present). You define cells by creating separators between them. You create separators with the Edit|Create|Separators menu

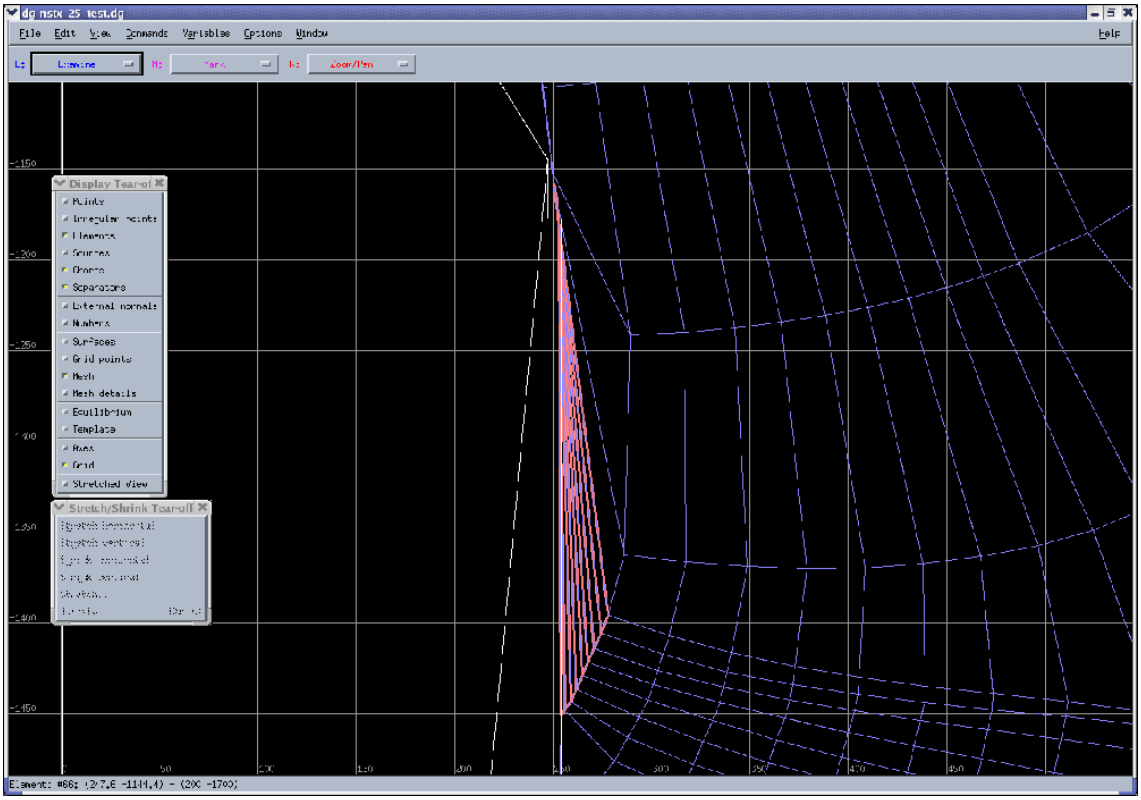


Figure L.5: Zoom of the twisted mesh cells near the inner divertor target of an NSTX geometry.

command. To adjust individual separators, use the `Reposition` tool. The `Edit|Delete|Separators` menu command removes all separators. You cannot add or remove individual separators with `DG`.

To work with separators, you must define a suitable structure and import a `Sonnet` grid.

Use the `Edit|Create|Separators` menu command to create separators between the structure and the `Sonnet` grid.

Use the `Reposition` tool to attach a separator to a different structure point. Select the separator and move the mouse cursor a structure point. Then release the mouse button.

Use the `Edit|Delete|Separators` menu command to delete all separators.

L.10 Working with variables

L.10.1 Basics

With `DG`, you can use `Variables` to specify information that cannot be edited visually in the work area. Variables are written into the output file and can be processed by other programs. Some variables (targets and structure) also have important functions in `DG`.

Variables must be set up by a power user before using them. This procedure is described in **Customizing DG** (section L.12). Each variable has following attributes:

Name is written into the output file and is otherwise invisible.

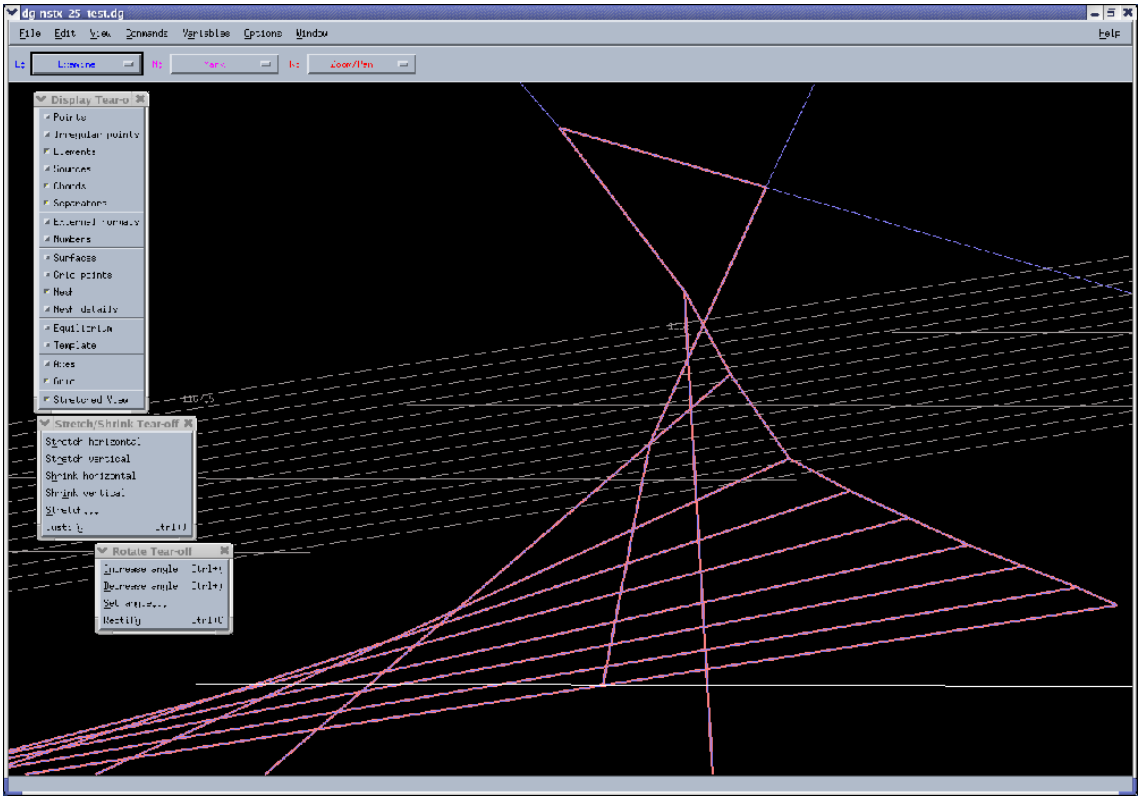


Figure L.6: After stretching and rotation, the overlapping of the cells in the upper part of the region cells depicted in Fig. L.5 becomes clear.

Description is the name that is shown to the user.

Type defines type of values the variable can hold.

Scope specifies what objects the variable applies for.

Export/No export determines whether values are written into the output file.

Default value is the default value for newly-created objects.

Help text is displayed to the user when s/he asks for it.

These attributes are normally defined by a power user while setting up variables. A normal user should have some idea of the type and the scope in order to be able to set correct values for correct objects.

Each variable has a `type`. A variable can hold either a text value (`integer`, `real` or `text`) or a set of elements (`element`, `set of elements`, `target`, `structure`, `part of structure`). `DG` prevents the user from entering values of a wrong type.

The `scope` determines objects the variable is attached to. There are 4 possible scopes:

Layers means that the variable has one value for the whole layer.

Elements means the variable has a separate value for each element.

Sources means the variable has a separate value for each source.

Chords means the variable has a separate value for each chord.

As noted above, variables are organized into `layers` with `DG` displaying each layer in a separate dialog box.

The power user defines `layer types` and the minimum and the maximum number of layers of each type.

L.10.2 Working with layers

Layers may have one or more instantiations in a particular model. For some layers, a single instance is all that is needed to specify the desired information; `DG` will not permit the creation of additional instances of this same layer. In the latter case, `DG` will assign an integer count as the instances are created. Some layer instances are predefined (e.g., `Structure`) and cannot be created or destroyed by the user, power or otherwise.

To create an instance of a specific layer, select the layer from the `Variables|Add` sub-menu. A dialog box for editing the variables in this layer instance will appear. If the maximum number of instances of a specific layer already exist, this layer will not be displayed in the `Add` menu. If this menu becomes empty, the `Variables|Add` cascade button will be grayed.

To destroy an instance of a layer, select its name in the `Variables|Remove` sub-menu. If there are multiple instances of a layer, there will be a sub-menu associated with that layer, listing the integer indices of those instances. If only the minimum number of instances of a specific layer already exist, layers of this type will not be displayed in this menu. If this menu becomes empty, the `Variables|Remove` cascade button will be grayed.

To display the dialog box for an existing layer instance, select its name and instance index (if applicable) under the `Variables` menu.

To hide the dialog box for a layer, press the `Close` button in that dialog box.

L.10.3 Setting variables

To set variables in a layer, you need to display the dialog box for that layer first.

The dialog box for each instance of a layer displays its variables, allowing them to be set, viewed, and modified. The title of this dialog box is the name of the layer; if there are multiple instances, the integer index of the particular index is also displayed here. At the top of the dialog box, there is a row of buttons that affect the entire box. Below this are fields for the individual variables. Finally, there is a status line at the bottom of the dialog box.

The "△" button on top of the dialog box allows you to "roll it up" so that the box occupies less screen space. In order to restore the dialog box to its full size, press this button again.

The dialog box can be resized using the window manager. If it is reduced in size, scrollbars appear, allowing you to scroll the content. The functionality of the dialog box is NOT affected by resizing it or rolling it up. In particular, the `Set all` button also affects variables that are invisible.

For each variable, the dialog box contains its name, a `Help` button, a text field for entering the new value (for text-like variables) or a `Mark` button (for `set of objects` variables), and a `Set` button. The `Help` button, when pressed, displays help about the variable. The text field is used to enter new values for text variables. When the text field has keyboard focus, the status line displays the current value of the variable. For text variables that apply to elements, chords or sources, only variables that apply to marked objects are accessed. `Filename` variables differ from `Text` in that the user can select the file interactively (press the `Enter` key on your keyboard in the text field to bring up the file selection dialog); the variable is not valid unless the file exists and is readable at the time of the variable check.

The `Mark` button, available only for `Set of objects` variables, marks the objects contained in the variable. The `Set` button sets the variable to the new value. For text variables, the new value must be entered in the text field. For variables storing sets of objects, the objects must be marked.

The `Set all` button in the top row, when pressed, has exactly the same effect as pressing the `Set` button of each variable except that empty new values for variables that apply to elements, sources or chords are ignored. This button is only available if the layer consists of text variables only.

The Reset all button, when pressed, resets all text fields to present values of corresponding variables. It is only available in layers consisting of text variables only. The text field is cleared for variables that apply to elements, sources or chords when marked objects have different values.

The Close button closes the dialog box.

Pressing the right mouse button over a text field or a button associated with a variable brings up a pop-up menu. If the mouse has less than 3 buttons, the menu can be displayed by pressing "Ctrl+left" button. The Reset command sets the text field to the current value of the variable, similar to the Reset all button but affecting only one variable. For variables storing sets of objects, this command acts exactly like the Mark button. The Display values command displays values of variables that apply to elements, chords or sources next to corresponding objects. In order to remove these "labels", highlight any object or use the View|Remove labels command from the main menu. Commands in the Compare sub-menu compare values of the variable to the value entered in the edit field and mark matching objects. Both the Display values command and the Compare sub-menu are only available for variables that apply to elements, sources or chords.

To set a value for a variable that contains a set of elements, mark the desired elements and press its > button.

To set a value for a text variable that applies to the layer, enter the value in the New Value text field and press the > button.

To set a value for a text variable that applies to elements or sources, first mark elements or sources the value should be assigned to. To set a new value, enter it in the New Value text field and press the > button.

Current values for variables are always displayed in corresponding Old Value text boxes. For variables that apply to elements or sources, only values for marked objects are considered. If all values are the same, a value will be displayed, otherwise the word Different.

To copy the current value of a variable into its New Value text field, press its < button. For a variable containing a set of elements, the elements belonging to the variable will be marked.

The Accept button has exactly the same effect as pressing all > buttons except that empty new values are ignored.

The Reset button has exactly the same effect as pressing all < buttons.

The Hold toggle button controls how New Value fields of variables that apply to elements or sources are updated. With Hold disabled, these fields are updated every time objects are marked and unmarked. When it is enabled, no such updates happen.

You can easily copy values between different elements or sources. Disable Hold and mark objects containing the desired value. Then enable Hold and mark objects the values should be assigned to. Press Accept to set new values.

L.11 Creating output files

L.11.1 Types of output files

DG produces 3 output files:

Structure file contains the structure, with .str extension

Targets file contains targets, surfaces and grid points, with .trg extension

Output file contains elements, sources, separators, and the values of the variables associated with all of the layer instances for the current model, with .dgo extension

The current version of DG requires structure, targets, surfaces, and grid points to be defined. Also, all integer and real variables must be set to non-empty values.

L.11.2 Producing output files

DG generates names of output files by attaching different extensions to the name of the DG file. If the current file has no name yet, save it with the File|Save command before producing the output.

To produce the output files, choose the File|Output menu command. A dialog box will appear asking for confirmation. Press Ok to create the files.

L.12 Customizing DG

L.12.1 Installing DG

To install DG, you need to:

- Compile and link its sources
- Make sure that the help file exists and has the right name
- Make sure that the configuration file exists and has the right name

DG is an X Windows application. The correct DISPLAY setting is necessary.

Some useful infos on installing are contained in the **Customizing DG** section (L.12).

L.12.2 Files needed by DG

dg Executable file

dg.dgh Help file

dg.dgc Configuration file

L.12.3 Specifying directory masks

You can customize directory masks DG displays in file selection dialog boxes by setting the following environment variables:

DG_OPEN_MASK Directory mask for File|Open (defaults to the current directory)

DG_SAVE_MASK Directory mask for File|Save (defaults to the current directory)

DG_EQUIL_MASK Directory mask for File|Equilibrium (defaults to the current directory)

DG_TEMPLATE_MASK Directory mask for File|Template (defaults to the templates/ subdirectory, if present, or to the current directory)

DGSONNETMASK Directory mask for File|Sonnet grid (defaults to ${SOLPSTOP}/modules/DivGeo/device/${DEVICE})

L.12.4 The configuration file

There is a special DG file, the **configuration file**. Its filename is normally produced by appending .dgc to the name of the executable. The configuration file has two functions:

1. When the File|New command is executed, the configuration file is loaded into the new window.

2. When a DG file is being opened, the configuration file is used to update its definitions of variables and layers.

DG normally writes these definitions in DG files. When a DG file is opened, they are normally ignored and definitions from the configuration file are used instead. This ensures that all files loaded in DG have identical definitions of layers and variables.

Default values are not updated in the current version of DG.

Specifying the configuration file

Sometimes it is necessary to use a different configuration file or not to use any at all.

To use a different configuration file, start DG with the following command:

```
dg -cfg <filename-with-extension>
```

To use DG without loading a configuration file, type

```
dg -nocfg
```

NOTE: There are some bugs in DG that appear only when it is started without any configuration and sometimes cause an exception.

L.12.5 Creating or modifying the configuration file

Use the Options | Setup menu command to create/modify the configuration file.

The Setup dialog box and all dialog boxes you open during setup are modal. For this reason, you can work only with the topmost dialog box.

Choose the Variables... sub-menu in order to edit definitions of variables and layers in the current file. The Layer types dialog box will appear.

Press Save as configuration to save the current file as a configuration file. A dialog box will appear prompting for a filename. Normally, it already contains the right name. Select Ok to save the file in the current window as configuration.

Press Close to close the dialog box.

Adding, modifying and removing layer types

To define a layer type, choose Add in the Layer types dialog box. The new Layer type options dialog box will appear.

Enter the name of the layer type to create in the Name text field. The name should contain only alphanumeric characters. This name will be written into the output file.

Enter the description in the Description text field. In the MinInstances/MaxInstances text field, enter the minimum and the maximum number of layers of this type. You will have to use 0 for MinInstances because there are no layers of this type yet. You can change MinInstances later after creating the minimum number of layers.

Press the Ok button to create the layer type.

To remove a layer type, select it in the list and press Delete.

To modify a layer type, select it in the list and press Modify. The Layer type options dialog box will appear. You can change name, description, Min/MaxInstances here. Note that if you change the name, layers contained in DG files under the old name will be ignored.

To close the Layer types dialog box, press Close.

Setting up variables

To define the variables that will constitute a layer, first display its Layer type options dialog box (Modify). This dialog box contains a rectangular area with buttons. These buttons represent row/column positions of variables in dialog boxes. Buttons containing the string Empty mean that no variable is defined at this position.

To create a variable at a specific position, press the Empty button at that position. The New variable dialog box will appear.

Enter the name of the variable to be created in the Name field. The name must contain only alphanumeric character.

Select the type and the scope in corresponding option menus. Check the Do not export toggle button to prevent the variable from being written into the output file. Structure and targets are normally not exported because they are written into separate files.

Press Ok to create the variable.

To modify a variable definition, press its button in the Layer type options dialog box. The Variable options dialog box will appear. After making changes, press Ok. Note that if you change the name, variables contained in DG files under the old name will be ignored.

To remove a variable, press its button in the Layer type options dialog box. The Variable options dialog box will appear. Press Remove to remove the variable.

Adding help text to variables

DG lets you add help text to variables. The user can display the help text by pressing the Help button located near the name of the variable in the Variables dialog box.

To edit a help text for a variable, display its Variable options dialog box and press Edit help. The Help for variable editor dialog box with a text editing field will appear. Edit the help text in the editing field and press Ok.

Setting default values

DG lets you define default values for variables. When a new layer, element or source is created, the default values are used for the newly created object.

It is suggested that you define default values for all variables because DG will not generate output files when some variables have empty values. A search for empty values can be very difficult and cost a lot of time.

To define default values for variables in a layer type, display its Layer type options dialog box and choose the variable for which you wish to set a default. In its Variables: dialog box there is a Default value field for this purpose.

Default values from the configuration file are used only to create new DG files. When opening an existing file, default values from the configuration are ignored.

L.13 File formats

L.13.1 Simple template file format

Besides the `Neutral File` exported from `PATRAN`, `DG` can also import templates in a very simple format where the geometry is specified as a set of polygons (in general, open). In this case, each line of the input file either contains a pair of the co-ordinates of the next corner, or is empty.

L.13.2 Structure file format

The first line in the structure file contains the number of closed chains in the structure.

Then, for each chain, starting with the outermost, the following block appears:

```
<number of points in the chain>
<x1> , <y1>
...
```

For the outermost chain, the bounding rectangle is added.

Points are output in counterclockwise order when the external normals of the elements in the chain point outwards. Otherwise, they are output in clockwise order.

L.13.3 Targets file format

The targets file (for a single-null model) has the following format:

```
target 1
line
  <x> , <y>
  ...
target 2
line
  <x> , <y>
  ...
region 1
levels
  <level>
  ...
region 2
levels
  <level>
  ...
region 3
levels
  <level>
  ...
zone 0
points
  <value>
  ...
zone 1
points
```

```
  <value>
  ...
zone 2
points
  <value>
  ...

<eof>
```

The `target1` and `target2` blocks contain point coordinates for targets.

The `region 1...3` blocks contain levels of surfaces. Regions have following meaning:

region 1 Main Plasma

region 2 SOL

region 3 Private Flux

The `zone 1...3` blocks contain values of grid points (between 0 and 1). Zones have following meaning:

zone 1 Main chamber

zone 2 Inner divertor

zone 3 Outer divertor

L.13.4 Output file format

The output file has the following format:

```
p1
  <x> , <y> , <z>
  ...
p2
  <x> , <y> , <z>
misselem
  <number>
  ...
sprtrs
  <number>
  ...
sources
  <x> , <y>
cells <cellnum>
  <number>
  ...
...
<layer name> <layer number>
  <variable>
  ...
...
finish
```

userdata

`p1` and `p2` sections containt coordinates of origins and ends of elements and/or separators, sorted after element number. For unused numbers, dummy coordinates are output. The Z coordinate is always 0.

The misselem section contains unused element numbers. Elements with these numbers **must** be ignored.

The sprtrs section lists numbers of separators in p1 and p2 arrays.

The sources section lists sources.

If separators are defined, there is a cells section for each cell. Each section lists numbers of elements and separators included in its cell.

For each layer, a <layer name> <layer number> section is present. Layer name is the name of the layer type. Layer number is used to distinguish between layers of the same type. Each section contains a <variable> entry for each variable in the layer.

The <variable> block has following formats:

- Text variables that apply to the layer:

  ```
  <variable name> <value>
  ```

- Text variables that apply to elements or sources:

  ```
  <variable name>
    <value>
    ...
  ```

- For variables that apply to elements, a value for each element number is included. Dummy values are used for unused numbers. For variables that apply to sources, a value for each source is included. Values are written in the same order as sources in the sources section.

- ”Set of elements” variables that apply to the layer:

  ```
  <variable name>
    <element number>
    ...
  ```

Appendix M

Some information from the SOLPS 2002 Course Notes

The following information is extracted from the SOLPS Course Notes given at JET in 2002.

M.1 Breakdown of B2.5 programs

`B2.5` consists of a number of programs

b2ag prepares the geometry and magnetic field (see 3.1)

b2ah prepares the default physics parameters (see 3.2)

b2ai prepares the initial plasma state (see 3.4)

b2ar prepares a table of atomic rate coefficients (see 3.3)

b2co one option when running the main program is to write out a short restart file periodically; this program converts such a restart file into the normal version (see 3.15)

b2fu the opposite of **b2uf**; it converts a formatted plasma file `b2fplasmf` to the normal unformatted `b2fplasma`

b2md saves the **SOLPS 5.x** output to the `MDSplus` server (see 6)

b2mn the main program (see 3.5)

b2pl the plot program (see 3.6 and Appendix J)

b2rd retrieves the **SOLPS 5.x** output from the `MDSplus` server (see 6)

b2ts an example post-processing program; it reads the basic output files produced by **b2mn** and prints some simple geometric quantities

b2uf the normal `b2fplasma` file (used for plotting) is written using FORTRAN unformatted writes; this program reads such a file and writes it out as a formatted file, `b2fplasmf`. Typically used for moving the `b2fplasma` to a different computer architecture.

b2xd was used in the transition stage from **SOLPS 4.x** to **SOLPS 5.x** — it converts from the **SOLPS 5.x** plasma state file to that used by **SOLPS 4.x**

b2ydm post processes the heat load profiles on the dome and evaluates outlet coolant temperature (see 3.20)

b2ye computes an error assessment of a coupled `B2.5-Eirene` run

b2yg plots the geometry and magnetic field

b2yh prints the table of physics parameters

b2yi plots the plasma state

b2yi_gnuplot a version of **b2yi** using `gnuplot` instead of `NCAR`

b2ym claims to produce a movie

b2ymb post processes the heat load profiles on the targets and creates input files for the monoblock heat transfer model (see 3.18)

b2yn plots the progress of the inner `B2.5` iterations

b2yp displays the plasma state

b2yq claims to provide a quick display of the evolution data

b2yr plots the atomic rate coefficients

b2yrp post processes the heat load profiles on the reflector plates and creates input files for the RPs heat transfer model (see 3.19)

b2yt transfers the plasma from one mesh to another mesh of a different size (see 3.14)

b2_ual_write writes out the plasma state in IDS format (see 3.21)

M.2 B2.5 ".dat" and ".prt" files

filename	formatted?	read by
b2ag.dat	formatted	b2ag b2ai
b2ah.dat	formatted	b2ah
b2ai.dat	formatted	b2ai
b2ar.dat	formatted	b2ar
b2md.dat	formatted	b2md
b2mn.dat	formatted	b2md b2mn b2pl b2yi b2yi_gnuplot b2yn b2yp b2yt b2_ual_write
b2ydm.dat	formatted	b2ydm
b2yg.dat	formatted	b2yg
b2yh.dat	formatted	b2yh
b2yi.dat	formatted	b2yi
b2ym.dat	formatted	b2ym
b2ymb.dat	formatted	b2ymb
b2yn.dat	formatted	b2yn
b2yp.dat	formatted	b2yp
b2yq.dat	formatted	b2yq
b2yr.dat	formatted	b2yr
b2yrp.dat	formatted	b2yrp
b2yt.dat	formatted	b2yt
stra.dat	formatted	b2ar
weis.dat	formatted	b2ar

filename	formatted?	written by
b2ag.prt	formatted	b2ag
b2ah.prt	formatted	b2ah
b2ai.prt	formatted	b2ai
b2ar.prt	formatted	b2ar
b2co.prt	formatted	b2co
b2fu.prt	formatted	b2fu
b2md.prt	formatted	b2md
b2mn.prt	formatted	b2mn
b2rd.prt	formatted	b2rd
b2uf.prt	formatted	b2uf
b2ydm.prt	formatted	b2ydm
b2ye.prt	formatted	b2ye
b2yg.prt	formatted	b2yg
b2yh.prt	formatted	b2yh
b2yi.prt	formatted	b2yi
b2ym.prt	formatted	b2ym
b2ymb.prt	formatted	b2ymb
b2yn.prt	formatted	b2yn
b2yp.prt	formatted	b2yp
b2yq.prt	formatted	b2yq
b2yr.prt	formatted	b2yr
b2yrp.prt	formatted	b2yrp
b2yt.prt	formatted	b2yt

M.3 B2.5 communication files

filename	formatted?	written by	read by	contents
b2fgmtry	un*formatted	b2ag	b2ah b2ai b2co b2fu b2md b2mn b2pl b2ts b2uf b2xd b2yg b2yh b2yp b2yt b2_ual_write	geometry
b2fmovie	un*formatted	b2mn	b2ym	movie of plasma evolution
b2fparam	un*formatted	b2mn	b2md b2fu b2pl b2ts b2uf b2xd b2yh b2yp b2_ual_write	run parameters used for plotting
b2fpardf	un*formatted	b2ah	b2co b2mn b2yh b2yt	transport parameters
b2fplasma	unformatted	b2fu b2mn	b2md b2pl b2ts b2uf b2xd b2_ual_write	plasma state used for plotting
b2fplasmf	formatted	b2uf	b2fu	plasma state used for plotting
b2frates	un*formatted	b2ar	b2md b2mn b2pl b2yn b2yp b2yr b2yt b2_ual_write	atomic physics rates
b2fstate	un*formatted	b2co b2mn b2rd	b2fu b2md b2pl b2ts b2uf b2xd b2yi b2yi_gnuplot b2yp b2yr b2yt b2_ual_write	final plasma state
b2fstati	un*formatted	b2ai	b2co b2mn b2yi b2yi_gnuplot	initial plasma state
b2fstatt	un*formatted	b2yt		plasma state on a new grid
b2fgmtrt	un*formatted	b2rd		recovered geometry file
b2ftrace	un*formatted	b2mn	b2yq	residuals
b2ftrack	un*formatted	b2mn	b2yn	residuals
b2specp	formatted	b2yp		sources
b2faveri	formatted		b2mn	initial averaged plasma solution
b2favere	formatted	b2mn	b2ye	final averaged plasma solution
tran	formatted	b2mn		plasma state used for archiving at JET

Appendix N

Subversion repository logs

N.1 SOLPS-ITER

```
a586a83 Pointer to B2.5 submodule with bug fix for OpenMP Zhdanov treatment + updated
    documentation
```

N.2 Changes in output for the AUG_16151_D test case

Whenever the code is changed, small changes can appear in the results of a simulation. In this section we look at some changes that have appeared as a result of the changes between **SOLPS 5.0** revisions r1360 and r1463.

The base case was stored as revision 1360 at 2008-01-30 13:39:40 GMT and saved in MDSplus as 21393. This forms the **REF** case for the following discussion.

The first variant, **000** looked at used the r1463 for all files except for `b2mod_constants.F`, where the r1403 version was used. (r1404 introduced a new more accurate set of physics constants.)

The second variant, **001**, used the r1463 revision for all files, but the run was performed with the previously generated `b2frates`, `b2fgmtry` and `b2fpardf` files.

The third variant, **002**, used the r1463 revision for all files, with the run performed with newly generated `b2frates`, `b2fgmtry` and `b2fpardf` files.

Two subvariants of **002** were then performed: **002a** with old `b2frates` but newly calculated `b2fgmtry` and `b2fpardf`; and **002b** with old `b2fgmtry` but newly calculated `b2frates` and `b2fpardf`.

The following observations were made (see figure N.1):

000 same as **REF**

001 small kick in the residuals (< 0.01)

002 larger kick in the residuals (between 100 and 1000)

002a very similar to **002**

002b very similar to **001**

Comparing (figure N.2) the converged values of T_e at the outer target, **000**, **001** and **002b** gave the same answer as **REF**. **001** and **002b** gave the same answer, and the maximum relative change was 0.997408. (For n_e at the outer target, the maximum relative error was 1.00227.)

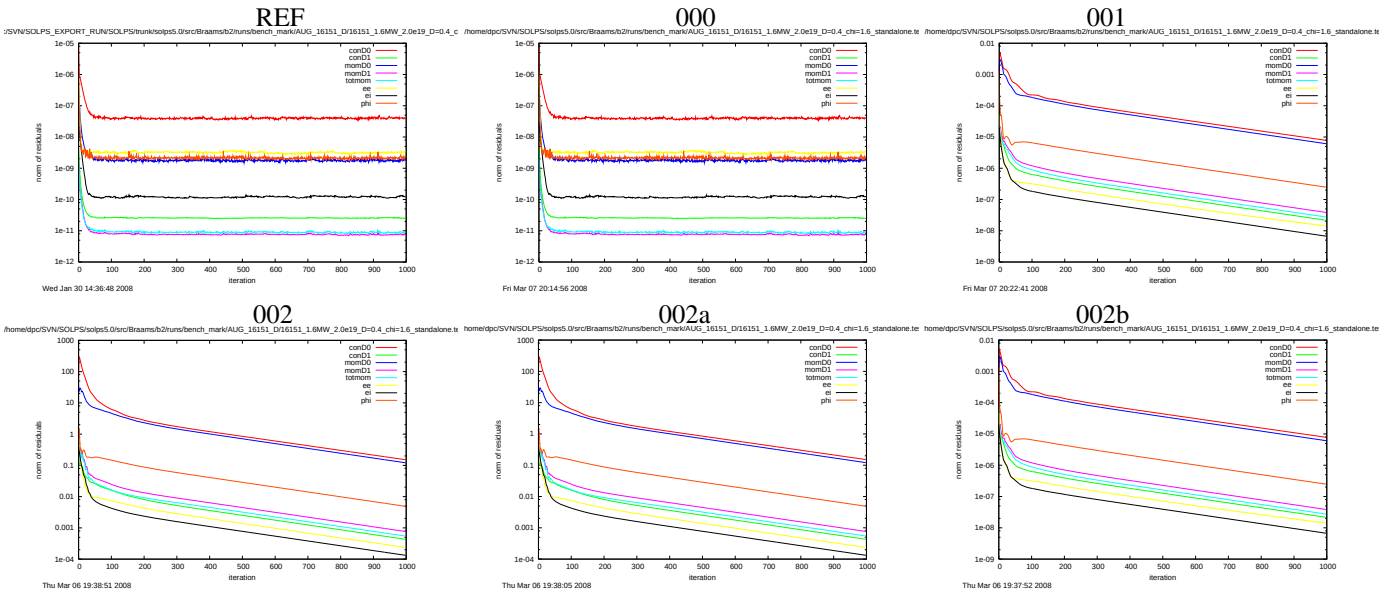


Figure N.1: Residuals of the AUG_16151_D standard case for different code versions and options.

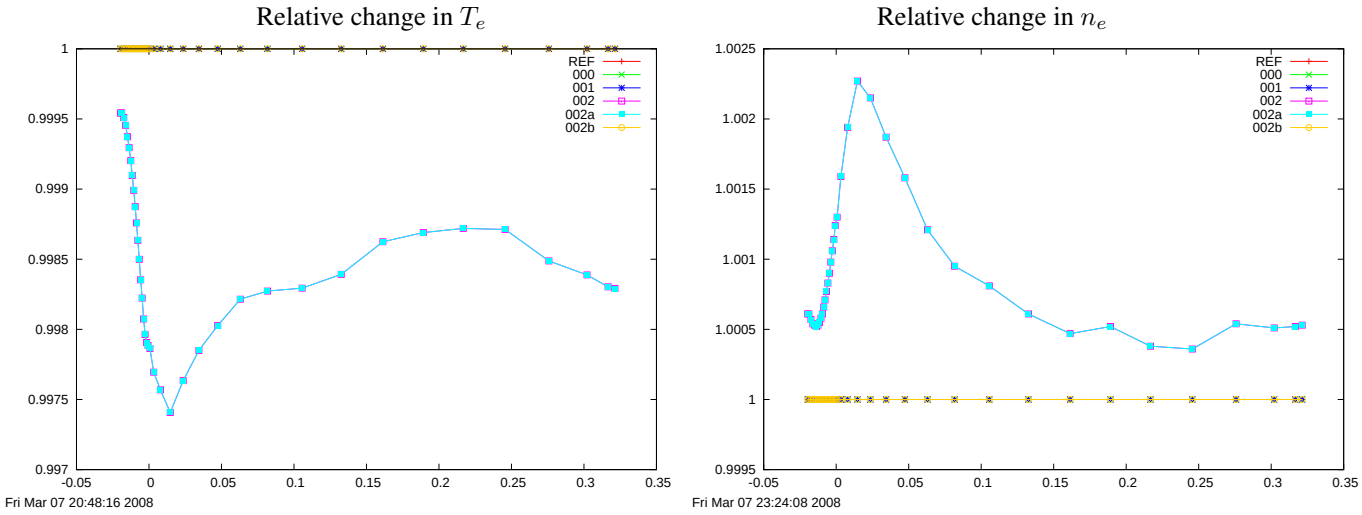


Figure N.2: Relative changes for the considered options.

The conclusion: the largest effect seems to have come from the correction in the cell volumes (causing a change in `b2fgmtry`), but this caused a small final change. The change of physics constants introduced a very small effect, seen only as a jump in the residuals.

Appendix O

License

O.1 European Union Public Licence v. 1.1

EUPL ©the European Community 2007

This European Union Public Licence (the "EUPL") applies to the Work or Software (as defined below) which is provided under the terms of this Licence. Any use of the Work, other than as authorised under this Licence is prohibited (to the extent such use is covered by a right of the copyright holder of the Work). The Original Work is provided under the terms of this Licence when the Licensor (as defined below) has placed the following notice immediately following the copyright notice for the Original Work:

Licensed under the EUPL v.1.1

or has expressed by any other mean his willingness to license under the EUPL.

O.1.1 1. Definitions

In this Licence, the following terms have the following meaning: - *The Licence*: this Licence. - *The Original Work or the Software*: the software distributed and/or communicated by the Licensor under this Licence, available as Source Code and also as Executable Code as the case may be. - *Derivative Works*: the works or software that could be created by the Licensee, based upon the Original Work or modifications thereof. This Licence does not define the extent of modification or dependence on the Original Work required in order to classify a work as a Derivative Work; this extent is determined by copyright law applicable in the country mentioned in Article 15. - *The Work*: the Original Work and/or its Derivative Works. - *The Source Code*: the human-readable form of the Work which is the most convenient for people to study and modify. - *The Executable Code*: any code which has generally been compiled and which is meant to be interpreted by a computer as a program. - *The Licensor*: the natural or legal person that distributes and/or communicates the Work under the Licence. - *Contributor(s)*: any natural or legal person who modifies the Work under the Licence, or otherwise contributes to the creation of a Derivative Work. - *The Licensee* or "*You*": any natural or legal person who makes any usage of the Software under the terms of the Licence. - *Distribution and/or Communication*: any act of selling, giving, lending, renting, distributing, communicating, transmitting, or otherwise making available, on-line or off-line, copies of the Work or providing access to its essential functionalities at the disposal of any other natural or legal person.

O.1.2 2. Scope of the rights granted by the Licence

The Licensor hereby grants You a world-wide, royalty-free, non-exclusive, sub- licensable licence to do the following, for the duration of copyright vested in the Original Work: - use the Work in any circumstance and for all usage, -

reproduce the Work, - modify the Original Work, and make Derivative Works based upon the Work, - communicate to the public, including the right to make available or display the Work or copies thereof to the public and perform publicly, as the case may be, the Work, - distribute the Work or copies thereof, - lend and rent the Work or copies thereof, - sub-license rights in the Work or copies thereof.

Those rights can be exercised on any media, supports and formats, whether now known or later invented, as far as the applicable law permits so.

In the countries where moral rights apply, the Licensor waives his right to exercise his moral right to the extent allowed by law in order to make effective the licence of the economic rights here above listed.

The Licensor grants to the Licensee royalty-free, non exclusive usage rights to any patents held by the Licensor, to the extent necessary to make use of the rights granted on the Work under this Licence.

O.1.3 3. Communication of the Source Code

The Licensor may provide the Work either in its Source Code form, or as Executable Code. If the Work is provided as Executable Code, the Licensor provides in addition a machine-readable copy of the Source Code of the Work along with each copy of the Work that the Licensor distributes or indicates, in a notice following the copyright notice attached to the Work, a repository where the Source Code is easily and freely accessible for as long as the Licensor continues to distribute and/or communicate the Work.

O.1.4 4. Limitations on copyright

Nothing in this Licence is intended to deprive the Licensee of the benefits from any exception or limitation to the exclusive rights of the rights owners in the Original Work or Software, of the exhaustion of those rights or of other applicable limitations thereto.

O.1.5 5. Obligations of the Licensee

The grant of the rights mentioned above is subject to some restrictions and obligations imposed on the Licensee. Those obligations are the following:

Attribution right: the Licensee shall keep intact all copyright, patent or trademarks notices and all notices that refer to the Licence and to the disclaimer of warranties. The Licensee must include a copy of such notices and a copy of the Licence with every copy of the Work he/she distributes and/or communicates. The Licensee must cause any Derivative Work to carry prominent notices stating that the Work has been modified and the date of modification.

Copyleft clause: If the Licensee distributes and/or communicates copies of the Original Works or Derivative Works based upon the Original Work, this Distribution and/or Communication will be done under the terms of this Licence or of a later version of this Licence unless the Original Work is expressly distributed only under this version of the Licence. The Licensee (becoming Licensor) cannot offer or impose any additional terms or conditions on the Work or Derivative Work that alter or restrict the terms of the Licence.

Compatibility clause: If the Licensee Distributes and/or Communicates Derivative Works or copies thereof based upon both the Original Work and another work licensed under a Compatible Licence, this Distribution and/or Communication can be done under the terms of this Compatible Licence. For the sake of this clause, “Compatible Licence” refers to the licences listed in the appendix attached to this Licence. Should the Licensee’s obligations under the Compatible Licence conflict with his/her obligations under this Licence, the obligations of the Compatible Licence shall prevail.

Provision of Source Code: When distributing and/or communicating copies of the Work, the Licensee will provide a machine-readable copy of the Source Code or indicate a repository where this Source will be easily and freely available for as long as the Licensee continues to distribute and/or communicate the Work.

Legal Protection: This Licence does not grant permission to use the trade names, trademarks, service marks, or names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the copyright notice.

O.1.6 6. Chain of Authorship

The original Licensor warrants that the copyright in the Original Work granted hereunder is owned by him/her or licensed to him/her and that he/she has the power and authority to grant the Licence.

Each Contributor warrants that the copyright in the modifications he/she brings to the Work are owned by him/her or licensed to him/her and that he/she has the power and authority to grant the Licence.

Each time You accept the Licence, the original Licensor and subsequent Contributors grant You a licence to their contributions to the Work, under the terms of this Licence.

O.1.7 7. Disclaimer of Warranty

The Work is a work in progress, which is continuously improved by numerous contributors. It is not a finished work and may therefore contain defects or "bugs" inherent to this type of software development.

For the above reason, the Work is provided under the Licence on an "as is" basis and without warranties of any kind concerning the Work, including without limitation merchantability, fitness for a particular purpose, absence of defects or errors, accuracy, non-infringement of intellectual property rights other than copyright as stated in Article 6 of this Licence.

This disclaimer of warranty is an essential part of the Licence and a condition for the grant of any rights to the Work.

O.1.8 8. Disclaimer of Liability

Except in the cases of wilful misconduct or damages directly caused to natural persons, the Licensor will in no event be liable for any direct or indirect, material or moral, damages of any kind, arising out of the Licence or of the use of the Work, including without limitation, damages for loss of goodwill, work stoppage, computer failure or malfunction, loss of data or any commercial damage, even if the Licensor has been advised of the possibility of such damage. However, the Licensor will be liable under statutory product liability laws as far such laws apply to the Work.

O.1.9 9. Additional agreements

While distributing the Original Work or Derivative Works, You may choose to conclude an additional agreement to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or services consistent with this Licence. However, in accepting such obligations, You may act only on your own behalf and on your sole responsibility, not on behalf of the original Licensor or any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against such Contributor by the fact You have accepted any such warranty or additional liability.

O.1.10 10. Acceptance of the Licence

The provisions of this Licence can be accepted by clicking on an icon "I agree" placed under the bottom of a window displaying the text of this Licence or by affirming consent in any other similar way, in accordance with the rules of applicable law. Clicking on that icon indicates your clear and irrevocable acceptance of this Licence and all of its terms and conditions.

Similarly, you irrevocably accept this Licence and all of its terms and conditions by exercising any rights granted to You by Article 2 of this Licence, such as the use of the Work, the creation by You of a Derivative Work or the Distribution and/or Communication by You of the Work or copies thereof.

O.1.11 11. Information to the public

In case of any Distribution and/or Communication of the Work by means of electronic communication by You (for example, by offering to download the Work from a remote location) the distribution channel or media (for example, a website) must at least provide to the public the information requested by the applicable law regarding the Licensor, the Licence and the way it may be accessible, concluded, stored and reproduced by the Licensee.

O.1.12 12. Termination of the Licence

The Licence and the rights granted hereunder will terminate automatically upon any breach by the Licensee of the terms of the Licence.

Such a termination will not terminate the licences of any person who has received the Work from the Licensee under the Licence, provided such persons remain in full compliance with the Licence.

O.1.13 13. Miscellaneous

Without prejudice of Article 9 above, the Licence represents the complete agreement between the Parties as to the Work licensed hereunder.

If any provision of the Licence is invalid or unenforceable under applicable law, this will not affect the validity or enforceability of the Licence as a whole. Such provision will be construed and/or reformed so as necessary to make it valid and enforceable.

The European Commission may publish other linguistic versions and/or new versions of this Licence, so far this is required and reasonable, without reducing the scope of the rights granted by the Licence. New versions of the Licence will be published with a unique version number.

All linguistic versions of this Licence, approved by the European Commission, have identical value. Parties can take advantage of the linguistic version of their choice.

O.1.14 14. Jurisdiction

Any litigation resulting from the interpretation of this License, arising between the European Commission, as a Licensor, and any Licensee, will be subject to the jurisdiction of the Court of Justice of the European Communities, as laid down in article 238 of the Treaty establishing the European Community.

Any litigation arising between Parties, other than the European Commission, and resulting from the interpretation of this License, will be subject to the exclusive jurisdiction of the competent court where the Licensor resides or conducts its primary business.

O.1.15 15. Applicable Law

This Licence shall be governed by the law of the European Union country where the Licensor resides or has his registered office.

This licence shall be governed by the Belgian law if:

- a litigation arises between the European Commission, as a Licensor, and any Licensee;
- the Licensor, other than the European Commission, has no residence or registered office inside a European Union country.

O.1.16 Appendix

"Compatible Licences" according to article 5 EUPL are: - *GNU General Public License (GNU GPL) v. 2 - Open Software License (OSL) v. 2.1, v. 3.0 - Common Public License v. 1.0 - Eclipse Public License v. 1.0 - Cecill v. 2.0*

Appendix P

Contributing

The SOLPS-ITER code suite is a complex package with many cross-relations within its separate components and this needs to be kept in mind when developers are involved in any coding improvements. The list below provides a non-exhaustive set of coding requirements that are requested of developers, in order to maintain the integrity of the code package and a consistent coding style, as well as facilitate further development by others and code usage by the community. One must always stay mindful that the code is meant to be used on a wide variety of OS platforms, compilers, and physics use cases, so any change that would compromise the portability of the code is absolutely to be avoided, unless it has been specifically authorized by the ITER code Responsible Officer (RO) and presented to the user community for comment before incorporation into the main code branches. Of course, usual coding recommendations about including clear comments, explicit variable and routine names, and respecting indentations also apply throughout.

All code change suggestions should be made through GIT pull requests (PR) to the appropriate main distributions branches (currently `develop`, `release/3.1.1`, and `feature/wg-release` for the `3.0.9`, `3.1.1`, and `3.2.0` versions respectively). If a code change may apply to more than one version, then the pull request need only be done for one of the branches, but structured in such a way that cherry-picking it to add to the other branches can be easily achieved, and including in the PR description for which other branches it is meant. The cherry-picking will then be the responsibility of the ITER RO. The proposed side branch to be pulled in should already be positioned at the top of its destination branch, since only fast-forward merges are allowed. If a pull request involves branches in the main SOLPS-ITER repository and one or more of its submodules, the pull request at the SOLPS-ITER top level need not include the changes in the pointers to the updated submodules. These are the responsibility of the ITER RO, as the requester may be unaware of other developments in the submodules that would change the actual pointers to be included.

Contributors to the SOLPS-ITER code base are expected to follow **ALL** the rules below. Failure to abide by them will lead to their pull requests being delayed and/or summarily declined.

1. When working on a new code feature or bug fix, it is important to work in a branch forked from a development branch from the official distribution. These are the `develop`, `release/3.1.1`, and `feature/wg-release` branches. It is necessary to keep that forked branch abreast of any changes occurring in the reference branch, either by rebasing, merging, or cherry-picking the appropriate commits, before submitting any pull requests. Doing so regularly decreases the amount of work required to inspect the pull request and ensures all relevant code updates can be introduced with minimal risk of errors.
2. Code that only works for a particular set of cases, and is not general, should be avoided whenever possible and must always be identified as such. The code should then include safeties and/or error/warning messages to prevent its unintentional use by an unsuspecting user.
3. If modifying the code equations or introducing a new physics term, this shall be reflected in the physics model description chapter of the SOLPS-ITER manual, within the same commit or pull request.
4. Any pull request introducing changes to the physics results of the code that would affect some of the provided reference examples must contain pointers to newly converged occurrences of these affected examples.

5. When introducing a new switch or extending its functionality, a description of this switch must be added to the documentation files, within the same commit or pull request. For the **B2.5** code, the list of input file switches is given in the XML file `$SOLPSTOP/modules/B2.5/src/documentation/b2input.xml`. When editing the `b2input.xml` file, lines should begin with tabs, as these are used by the Python script that builds `b2cdci.F` and `b2cdcn.F`. The latter two files are not to be changed, since they will be automatically regenerated. If the value of the switch depends on the numbering/ordering/bundling of species or grid size, a conversion rule must also be provided within the **b2yt** program.
6. In addition, changes to the default values of switches need to be explicitly mentioned in the commit message.
7. Same when introducing or extending a namelist variable, the new variable and/or functionality must be added in `b2input.xml`, as well as in the relevant section of the SOLPS-ITER manual (`$SOLPSTOP/doc/solps/solps.tex`). Depending on the namelist involved and the type of variable (array depending on the numbering of species or grid size, in particular), conversion rules must also be provided within the **b2yt** and **b2sxdr** programs, as appropriate. It may also be necessary to provide a recommended default setting within **Uinp** or **b2us**. Again, all of this within the same commit or pull request as the new variable or functionality being introduced.
8. If a new input/output file is introduced, the `Makefile`, run submission, and relevant file manipulation scripts must be adapted to its presence. If the file is optional, the relevant run `Makefile(s)` must operate without any errors (no matter how mild) whether the file is present or not. A description of the new file and its usage must also be added in the documentation. Again, all of this within the same commit or pull request where the new file is introduced.
9. Importantly, code changes that imply a change of the physics result when code defaults are being used must be identified as such in the commit message. Such commit messages should then start with "Changes default behaviour" (ALL CAPS allowed if it is a big/important change!).
10. If a new module is introduced, its location must be added to the `src/modules/.new_modules` file, for compilation purposes, within the same commit, so that the `Makefile` is aware of the need to rebuild the `dependencies` file.
11. If the contents of one of the standard SOLPS-ITER I/O files is changed, it may be necessary to change the "version number" of that file, to ensure full backward compatibility. This backward compatibility will then need to be checked. If a version number change for a **B2.5** file is necessary, it must be included and described in `b2mod_version.F`, within the same commit. Any relevant conversion rules must also be added to the **b2yt**, **b2us**, and **b2sxdr** programs, as appropriate. **Such changes cannot occur without explicit consultation and approval by the ITER RO**.
12. If a new variable is added to the MDSplus tree database, or the definition of an existing variable is changed, the value of the variable `dbversion` in `b2mds.F` must be incremented and a description of the change given in the comments above the setting of `dbversion`. The MDSplus tree file descriptions in `$SOLPSTOP/doc/solps/MDSPLUS` may also need to be updated.
13. If a new **B2.5** routine is added, and the routine name is `b2*`, then a short description of the routine must be added to the `b2cdcr.F` file, within the same commit. In general, the name of any new routine and its subroutine statement must be on the same line, so that the `tags` application in `emacs` will be able to find the routine when requested. In the same vein, to facilitate searches through the code base, please keep calling statements on the same line, i.e. `call` and the name of the subroutine being called should be kept together.
14. If a new main variable is added, its description and use shall be included in the `b2cdcv.F` file, within the same commit.
15. If a new script is added, it shall be listed and described in the SOLPS-ITER manual, within the same commit.
16. If a new **b2plot** function is added, it must be added to the list in `b2mod_b2plot_help.F` and described in the SOLPS-ITER manual, within the same pull request.
17. When adding a `use module` statement, and if the module is to be obtained from an outside library, an `! IGNORE` comment must be added at the end of the line so it is not included when building the `dependencies` files.
18. A commit should not be made unless it has been verified that, at least, the code compiles with the new changes included. Creating a pull request will trigger a series of Continuous Integration (CI) tests, and these must also be passed for the pull request to be merged in, unless specifically authorized by the ITER RO. If the changes involve some pre-processor switches, the compilation test must be done with the pre-processor switches both on and off. The compilation should result in no warnings and no errors, for all tested compilers. Compilation

should be tested, ideally, on more than one compiler. For general distribution, SOLPS-ITER is to be made to compile on the Intel, GNU, Portland Group (NVIDIA), NAG, and Cray compilers. Only Fortran standard should be used, as not all extensions are supported by all compilers, or alternative code must be provided with appropriate pre-processor bracketing. The `-DLEGACYCOMP` pre-processor pragma is available to allow for legacy Fortran compilers that do not support the 2003 standard extensions. Other compiler-specific pragmas include `-DGFORTRAN`, `-DPGF90`, `-DNAGFOR`, or `-DCRAY`. A full list of the currently used pre-processor bracketing variables is available in the SOLPS-ITER manual, and should be updated, within the same pull request, whenever a pragma is being used within a source code file.

19. To avoid memory leaks, which may become problematic when using SOLPS-ITER as a recurring actor in an IMAS workflow, every array allocation statement shall be twinned with a corresponding array deallocation, preferably within the same subroutine or module.
20. Double-precision real numerical values inside **B2.5** and **Carre** should be written as `0.0_R8`, NOT `0.`, `0.0`, `0.0e0`, or `0.0d0`. In **Eirene**, they should be written as `0.0_DP`. Single precision reals use the `_R4` (in **B2.5** and **Carre**) and `_SP` (in **Eirene**) extensions. Quadruple precision variables use `_R16`. Long integers use `_I8`.
21. Do not use hash characters (`#`) as continuation characters in Fortran 77 files, as these are misinterpreted by the pre-processor.
22. If the real result of a ratio of two integers is needed, use the `rratio(m,n)` function provided within the **B2.5** code.
23. Care must be taken NOT to import special DOS carriage return control characters (`^M`) before committing files. The GIT repository is configured to automatically convert such files. Files must also be checked to have proper read/write/execute permissions.
24. In source code, trailing whitespace characters and indentation by means of "tab" characters are to be avoided. Tagging individual lines with comments indicating author and date is not mandatory, since proper use of the version control tools is sufficient to keep traceability of the code source. Progressive indentation into loops and `IF` statements should be kept consistent within the same file to maintain good code readability.
25. Within modules, individual subroutines and functions must be terminated with either an `END SUBROUTINE` or `END FUNCTION` statement, respectively. Some compilers will fail if only `END` is used. Preferably, also include the name of the ending subroutine or function in that statement, and apply the same rule also to individual subroutine files.
26. The use of `ENTRY` statements must be avoided. Instead, it is recommended to create a module that contains the entries as separate subroutines.
27. The use of `GOTO` statements should be avoided and substituted by `SELECT CASE` or `IF` statements whenever possible.
28. Statement functions should be avoided and substituted by external or contained functions.
29. The use of `stop` statements within **B2.5** or **Eirene** is not MPI-compliant and therefore not allowed. Within **B2.5**, use the `xerrab` or `xertst` functions to cause the code to stop upon some error. In **Eirene**, the relevant routine to call is `EIRENE_EXIT_OWN`.
30. Avoid special characters (accents, Greek letters, etc...) in the GIT commit messages as the GIT log is appended to the manual and the LaTeX builders may not be able to digest such entries.
31. When introducing a new variable in **Eirene**, and if its use is not confined to the SOLPS-ITER interface and user routines, the code must be modified in such a way that ALL interfaces will still function, so that the new **Eirene** code can still pass all the CI tests managed on the **Eirene** JuGit repository maintained by FZJ. This may mean providing a default value for the new variable and a way to recover old behaviour when running **Eirene** coupled to other codes or in stand-alone mode.
32. When adding a new integer or logical variable to the **Eirene** input file, if it is added to an existing input line, this must be done in a backward-compatible and compiler-safe fashion through the use of `fix_integer_input` and `fix_logical_input` calls respectively.
33. When modifying the format of the **Eirene** input file, the converter program **b2yt** and input file builder program **Uinp** must also be modified to reflect this change, within the same pull request.
34. When adding or modifying a variable in the `fort.44` or `fort.46` output files from **Eirene**, a conversion or transcription rule must be provided in `b2yt_ngread` or `b2yt_ntread`, respectively, in the same pull request, as well as a description of the changed or added variable in the SOLPS-ITER manual.
35. In comments, avoid the use of apostrophes, as they can be mis-interpreted as unbalanced quote characters by

certain compilers.

36. When using a mathematical or physical constant, always apply the value provided by the `b2mod_constants` module.

Bibliography

[1] WIESEN, S. et al., J. Nucl. Mater. **463** (2015) 480, 21st PSI, Kanazawa, Japan, 2014.

[2] BONNIN, X. et al., Plasma and Fusion Research **11** (2016) 1403102, PET-15 conference, 2015, Nara, Japan.

[3] REITER, D. et al., Fus. Sci. Technol. **47** (2005) 172.

[4] HECHT, F., J. Numer. Math. **20** (2012) 251.

[5] SUMMERS, H. P., Atomic data and analysis structure users manual, JET-IR 06 (Abingdon: JET Joint Undertaking) (1994).

[6] SUMMERS, H. P. et al., Plasma Physics and Controlled Fusion **48** (2006) 263.

[7] HULSE, R. A., Nucl. Technol./Fusion **3** (1983) 259.

[8] BEHRINGER, K., Description of the impurity code STRAHL, Technical Report JET-R(87)08, JET Joint Undertaking, Culham, 1987.

[9] WEISHEIT, J. C., J. Phys. B (Atom. Molec. Phys.) **8** (1975) 2556.

[10] JANEV, R. K. et al., J. Nuclear Mater. **121** (1984) 10.

[11] BONNIN, X. et al., Journal of Nuclear Materials **415** (2011) S488.

[12] GHOOS, K. et al., Contrib. Plasma Physics **56** (2016) 616.

[13] GHOOS, K. et al., J. Comp. Phys. **322** (2016) 162.

[14] GHOOS, K. et al., M&C 2017, Jeju, Korea, April 16-20, 2017 .

[15] MARCHAND, R. et al., Computer Phys. Comm. **96** (1996) 232.

[16] SCHNEIDER, R. et al., Contrib. Plasma Phys. **46** (2006) 3, DOI 10.1002/ctpp.200610001.

[17] ROGNLIEN, T. et al., Contrib. Plasma Physics **38** (1998) 152.

[18] ROGNLIEN, T. et al., Czech. Journ. Phys. **48** (1998) 201.

[19] ROGNLIEN, T. et al., Phys. Plasmas **6** (1999) 1851.

[20] HINTON, F. et al., Nucl. Fusion **34** (1994) 899.

[21] MIHAILOVSKII, A. et al., Beitr. Plasmaphys. **24** (1984) 335.

[22] HIRSHMAN, S. P. et al., Nucl. Fusion **21** (1981) 1079.

[23] ROZHANSKY, V. et al., Nucl. Fusion **39** (1999) 613.

[24] RADFORD, G. J. et al., Contrib. Plasma Physics **36** (1996) 187, 5th Workshop on Plasma Edge Theory, December 1995, Asilomar, USA.

[25] MCDANIEL, E. et al., *The mobility and diffusion of ions in gases*, Wiley, New York, 1973.

[26] ROZHANSKY, V. et al., *Collisional transport in partially ionised plasma*, Energoatomizdat, Moscow, 1988.

[27] BAELMANS, M., *Code Improvements and Applications of a Two-dimensional Edge Plasma Model for Toroidal Devices*, PhD thesis, Universiteit Leuven, Belgium, 1993, published as Report No. Jül-2891 and as Report LPP-ERM/KMS No. 100 in 1994.

[28] ROSENBLUTH, M. et al., Phys. Fluids **15** (1972) 116.

[29] SCHNEIDER, R. et al., Contrib. Plasma Physics **46** (2006) 3.

[30] BRAAMS, B. et al., Contrib. Plasma Phys. **36** (1996) 276.

[31] BOOK, D. L., *NRL Plasma Formulary*, Office of Naval Research, Naval Research Laboratory, Washington, D.C. 20375, 1980 (Revised).

[32] ZAGORSKI, R. et al., Contrib. Plasma Physics **38** (1998) 61.

[33] STANGEBY, P. et al., Phys. Plasmas **2** (1995) 707.

[34] FENG, Y. et al., Journal of Nuclear Materials **266–269** (1999) 812.

[35] CHANKIN, A. V. et al., Plasma Physics and Controlled Fusion **36** (1994) 1485, Preprint in JET-P(94)18, 1994.

[36] NEUHAUSER, J. et al., Nucl. Fusion **26** (1986) 1679.

[37] CHODURA, R., Plasma flow in the sheath and presheath of a scrape-off layer, in *Physics of Plasma Wall Interaction in Controlled Fusion (Proc. NATO Workshop Val-Morin, Quebec, Canada)*, edited by POST, D. E. et al., pages 99–134, New York, 1984, Plenum Press.

[38] LLE 48 (1991), 178.

[39] KRASHENINIKOV, S. et al., Contrib. Plasma Physics **32** (1992) 255, 3rd Workshop on Plasma Edge Theory, June 1992, Bad Honnef, FRG.

[40] IGITKHANOV, Y. et al., Contrib. Plasma Physics **32** (1992) 308, 3rd Workshop on Plasma Edge Theory, June 1992, Bad Honnef, FRG.

[41] ABOU-ASSALEH, Z. et al., Contrib. Plasma Phys. **32** (1992) 268.

[42] PRASAD, M. et al., Phys. Fluids B **3** (1991) 3087.

[43] COSTER, D. et al., Physica Scripta **T108** (2004) 7, Hydrogen in Condensed Matter 2003 Workshop, Helsinki.

[44] WARRIER, M. et al., Computer Physics Communications **160** (2004) 46.

[45] UMANSKY, M. et al., Journal of Nuclear Materials **313–316** (2003) 559, 15th PSI, Gifu, Japan, 27.–31. May 2002.

[46] KUKUSHKIN, A. S. et al., Plasma Phys. Controlled Fusion **44** (2002) 931.

[47] BRAGINSKII, S. I., Transport processes in a plasma, in *Reviews of Plasma Physics*, edited by LEONTOVICH, M. A., volume 1, pages 205–311, Consultants Bureau, New York, 1965.

[48] BALESCU, R., *Transport processes in plasmas I*, North Holland, New York, 1998.

[49] ZHDANOV, V., *Transport Processes in Multicomponent Plasma*, Taylor and Francis, New York and London, 2002.

[50] MAKAROV, S. O. et al., Physics of Plasmas **28** (2021) 062308.

[51] ROZHANSKY, V. et al., Nuclear Fusion **49** (2009) 025007.

[52] SAUTER, O. et al., Computer Physics Communications **184** (2013) 293.

[53] RHIE, C. et al., AIAA J. **21** (1983) 1525.

[54] NAULIN, V., Phys. Rev. E **71** (2005) 015402(R).

[55] KORZUEVA, V. et al., Nuclear Materials and Energy **42** (2025) 101847.

[56] MAKAROV, S. O., *Development and Implementation of a Generalized Multi-Ion Transport Model in Plasma Edge Fluid Codes*, PhD thesis, Technische Universität München, 2024.

[57] MAKAROV, S. et al., Nuclear Fusion **63** (2023) 026014.

[58] STANGEBY, P., Phys. Plasmas **2** (1995) 702.

[59] RIEMANN, K.-U., Journal of Physics D: Applied Physics **24** (1991) 493.

[60] RIEMANN, K.-U., Phys. Plasmas **4** (1997) 4158.

[61] DAUBE, T. et al., Phys. Plasmas **6** (1999) 2409.

[62] ROGNLIEN, T. D. et al., J. Nucl. Mater. **266–269** (1999) 654.

[63] SALA, M. et al., Plasma Phys. Control. Fusion **62** (2020) 055005.

[64] BONNIN, X. et al., J. Nucl. Mater. **290–293** (2001) 829, PSI-14 conference, 2000, Rosenheim.

[65] COWLEY, C. et al., Plasma Phys. Control. Fusion **65** (2023) 035011.

Index

[rranareg_per_rsanareg.py, 60
_OPENMP, 620
2_ual_write, 676
2d, 51
2d, tesepm, 103
2d.summarize, 52
2d_plots, 52
2d_plots_av, 52
2d_profiles, 52
2d_profiles_extended, 52
2da, 51
2dt, 51
2dt_av, 52, 470
2dt_tallies.py, 51
2dvi, 52

ADAS, 40, 43, 76, 571, 580
adas, 43
ADAS, bundling, 76
add_b2plot_to_series, 55
add_central_device, 32, 55
ADD_TE_CORR_TO_PO, 93
addspec, 96
ADPAK, 76
AL_MAJOR_VERSION, 615
AL_MICRO_VERSION, 615
AL_MINOR_VERSION, 615
ALLOCATE_AND_NAMELIST, 619
AMDS, 43
AMJUEL, 42
AMNS, 76, 619
an3da, 468
an3di, 468
an3dl, 468
an3dr, 468
an3dtl, 469
an3dtr, 469
ASTRA, 619
ave.awk, 55
Averaging, 177
awk.derivative, 60
awk.first+even, 55
awk.integrate, 55
awk.mesh, 55
awk.notfirst, 55
awk.plot_mesh, 55
awk.reverse, 55
awk.transpose, 55

b2, 56
B2.5, 37
B2.5, builds, 37
B2.5, code documentation, 219
B2.5, communication files, 678
B2.5, compiling, 27, 590
B2.5, compiling options, 610
B2.5, equations, 372
B2.5, input/output files, 677
B2.5, list of namelists, 321
B2.5, list of routines, 222
B2.5, list of switches, 266
B2.5, list of variables, 239
B2.5, Makefile, 590
B2.5, model, 372
B2.5, modifying the source, 37
B2.5, obtaining, 19
B2.5, programs, 675
B2.5, recover an archived run, 137–139, 141
B2.5, run recovery, 135
B2.5, runs, 44
B2.5, src, 37
B2.5, Updating, 19
B2.5,local, 37
b2.atomic_physics_rescale.parameters, 102
b2.boundary.parameters, 81, 132
b2.boundary.parameters, D+C+He example, 163
b2.boundary.parameters.2, 132
b2.neutrals.parameters, 102, 132
b2.neutrals.parameters, D+C+He example, 164
b2.neutrals.parameters.2, 132
b2.numerics.parameters, 92
b2.numerics.parameters, D+C+He example, 164
b2.sources.profile, 97
b2.transport.inputfile, 94, 530
b2.transport.parameters, 90, 94, 132, 377, 384, 388, 392, 395, 402
b2.transport.parameters, D+C+He example, 164
b2.transport.parameters.2, 132
b2.transport.profile, output by b2plot, 571
b2.user.parameters, 102

b2.wall_save.parameters, 132
b2.wall_save.parameters.2, 132
B25_DEFINES, 34, 620
B25_EIRENE, 619
b2_ual_rewrite, 57, 143
b2_ual_write, 142
b2ag, 67, 675
b2ag.dat, 67, 677
b2ag.dat, D+C+He example, 152
b2ag.prt, 677
b2ah, 71, 123, 377, 378, 384, 388, 392, 395, 402, 675
b2ah.dat, 94, 132, 677
b2ah.dat, D example, 71
b2ah.dat, D+C+He example, 153
b2ah.dat.2, 132
b2ah.prt, 677
b2ai, 78, 124, 675
b2ai.dat, 677
b2ai.dat, D example, 78
b2ai.dat, D+C+He example, 152
b2ai.prt, 677
b2ar, 40, 76, 675
b2ar.dat, 677
b2ar.dat, D example, 76
b2ar.dat, D+C+He example, 153
b2ar.prt, 677
b2bremreg, 475
b2cdca.F, 219
b2cdci.F, 266
b2cdcn.F, 321
b2cdcr.F, 222
b2cdcv.F, 239
b2cmpa, 74
b2cmpb, 74
b2cmpt, 75
b2co, 45, 58, 60, 64, 135, 675
b2co.prt, 677
b2divua, 475
b2divue, 475
b2exba, 475
b2exbe, 475
b2favere, 177, 178, 678
b2favere.2, 132
b2favere_OK, 52
b2faveri, 177, 678
b2fgmtrt, 137–139, 141, 678
b2fgmtry, 67, 121, 132, 137–139, 141, 678
b2fgmtry.old, 132
b2fmovie, 80, 128, 678
b2fparam, 80, 123, 678
b2fpardf, 71, 123, 132, 678
b2fplasma, 80, 136, 678
b2fplasmf, 136, 678
b2fraa, 475
b2frates, 76, 127, 132, 678
b2fstate, 45, 64, 80, 124, 135, 137–139, 141, 177, 678
b2fstate_OK, 52
b2fstate_OK_bool, 52, 66
b2fstati, 78, 124, 177, 678
b2fstatt, 132, 678
b2ftrace, 80, 130, 678
b2ftrack, 80, 129, 678
b2fu, 136, 675
b2fu.prt, 677
b2joule, 475
b2md, 179, 675
b2md.dat, 677
b2md.prt, 677
b2mn, 80, 124, 377, 378, 384, 388, 392, 395, 402, 675
b2mn.dat, 94, 132, 677
b2mn.dat, D example, 80
b2mn.dat, D+C+He example, 161
b2mn.dat.2, 132
b2mn.prt, 677
b2mod_dimensions.F, 26, 30, 53, 54, 588
b2movies.nc, 135
b2movies.nc and eirenemovies.nc, quantities stored in, 472
b2movies.nc, dab2, 473
b2movies.nc, dmb2, 473
b2movies.nc, eirene_eael_she, 473
b2movies.nc, eirene_eapl_shi, 473
b2movies.nc, eirene_emel_she, 473
b2movies.nc, eirene_empl_shi, 473
b2movies.nc, eirene_epel_she, 473
b2movies.nc, eirene_eppl_shi, 473
b2movies.nc, eirene_papl_sna, 473
b2movies.nc, eirene_pmpl_sna, 473
b2movies.nc, eirene_pppl_sna, 473
b2movies.nc, eirenemovies.nc, 472
b2movies.nc, emolrad, 473
b2movies.nc, emolradsum, 473
b2movies.nc, eneutrad, 473
b2movies.nc, eneutradsum, 473
b2movies.nc, fchx, 472
b2movies.nc, fchy, 472
b2movies.nc, fhex, 472
b2movies.nc, fhey, 472
b2movies.nc, fhix, 472
b2movies.nc, fhiy, 472
b2movies.nc, fnax, 472
b2movies.nc, fnay, 472
b2movies.nc, na, 472
b2movies.nc, ne, 472
b2movies.nc, pefluxa, 473
b2movies.nc, pefluxm, 473
b2movies.nc, pfluxa, 473

b2movies.nc, pfluxm, 473
b2movies.nc, po, 472
b2movies.nc, rcxhi, 473
b2movies.nc, rcxhisum, 473
b2movies.nc, rcxna, 473
b2movies.nc, refluxa, 473
b2movies.nc, refluxm, 473
b2movies.nc, rfluxa, 473
b2movies.nc, rfluxm, 473
b2movies.nc, rqahe, 473
b2movies.nc, rqahesum, 473
b2movies.nc, rqrad, 473
b2movies.nc, rrahi, 472
b2movies.nc, rrahisum, 472
b2movies.nc, rrana, 472
b2movies.nc, rsahi, 472
b2movies.nc, rsahisum, 472
b2movies.nc, rsana, 472
b2movies.nc, tab2, 473
b2movies.nc, te, 472
b2movies.nc, ti, 472
b2movies.nc, tmb2, 473
b2p, 56
b2pc, 56
b2pl, 57, 104, 675
b2plot, 45, 104
b2plot, a0, 518
b2plot, a4l, 515, 556
b2plot, a4p, 515, 556
b2plot, abs, 513, 536
b2plot, acol, 515, 563
b2plot, adas, 519, 580
b2plot, additional commands, 530
b2plot, ahal, 505
b2plot, alf, 505
b2plot, alph, 505
b2plot, amju, 519, 580
b2plot, anjuel, 519
b2plot, arithmetic commands, 542
b2plot, arow, 515, 562
b2plot, arox, 513, 562, 569
b2plot, aroy, 513, 562, 569
b2plot, arww, 515, 562
b2plot, aspc, 515, 556
b2plot, ave, 513, 572
b2plot, avogr, 529
b2plot, b, 505
b2plot, b0, 518
b2plot, b2br, 505
b2plot, b2eatcr, 505
b2plot, b2espcr, 505
b2plot, b2im, 505
b2plot, b2in, 505
b2plot, b2io, 505
b2plot, b2ne, 505
b2plot, b2ni, 505
b2plot, b2nm, 505
b2plot, b2nn, 505
b2plot, b2ra, 505
b2plot, backscae, 523
b2plot, backscat, 523
b2plot, bala, 513, 573
b2plot, bb, 505
b2plot, bbdyemis, 523
b2plot, bc, 505
b2plot, bcch, 505
b2plot, bche, 505
b2plot, bchi, 505
b2plot, bcmo, 505
b2plot, bcna, 505
b2plot, bcol, 515, 562
b2plot, bix, 505
b2plot, biy, 505
b2plot, bold, 515, 558
b2plot, both, 515, 556
b2plot, bpol, 506
b2plot, brad, 506
b2plot, brch, 505
b2plot, brhe, 506
b2plot, brhi, 506
b2plot, brmo, 506
b2plot, brna, 506
b2plot, btor, 506
b2plot, bvis, 506
b2plot, bw, 516, 558
b2plot, bx, 506
b2plot, by, 506
b2plot, bz, 506
b2plot, c, 529
b2plot, c_cp_b1, 523
b2plot, c_cp_b2, 523
b2plot, c_cp_b3, 523
b2plot, c_cp_b4, 523
b2plot, cam, 519, 579
b2plot, cd, 506
b2plot, ceqp, 506
b2plot, changes from SOLPS4.0 to SOLPS5.0, 120
b2plot, character commands, 519
b2plot, chbx, 519, 578
b2plot, chhichii, 506
b2plot, chineo, 506
b2plot, chmp, 519, 579
b2plot, chor, 519, 575
b2plot, chord plots, 575, 576, 582
b2plot, chpl, 519, 576
b2plot, chs, 513, 536

b2plot, chvl, 520, 578
b2plot, ckappa_a, 523
b2plot, ckappa_b, 524
b2plot, ckappaa1, 523
b2plot, ckappaa2, 523
b2plot, ckappaa3, 523
b2plot, ckappaa4, 523
b2plot, ckappaa5, 523
b2plot, ckapppa1, 524
b2plot, ckapppa2, 524
b2plot, ckapppa3, 524
b2plot, ckapppa4, 524
b2plot, ckapppa5, 524
b2plot, cltb, 520, 575
b2plot, clvd, 516, 565
b2plot, clvl, 516, 567
b2plot, clvn, 516, 565
b2plot, clvp, 516, 566
b2plot, clvs, 516, 566
b2plot, cmd, 520, 577
b2plot, cmdcdf, 520, 577
b2plot, coathick, 524
b2plot, col, 516, 558
b2plot, colour table, 561
b2plot, comp, 516, 556
b2plot, coni, 506
b2plot, conn, 506
b2plot, constants, 529
b2plot, cont, 516, 532
b2plot, contour overlays, 561
b2plot, contour parameters, 565
b2plot, cooltemp, 524
b2plot, cora, 506
b2plot, coro, 530
b2plot, cp, 527
b2plot, cp_b1, 524
b2plot, cp_b2, 524
b2plot, cp_b3, 524
b2plot, cp_b4, 524
b2plot, cpa, 506
b2plot, cpo, 506
b2plot, cr, 506
b2plot, crx, 506
b2plot, cry, 506
b2plot, cte, 506
b2plot, ctgdns, 524
b2plot, cti, 506
b2plot, cua, 506
b2plot, cut, 506
b2plot, cvsx, 506
b2plot, cvsy, 506
b2plot, cvv, 506
b2plot, cz, 506
b2plot, d, 506
b2plot, dab2, 506, 569
b2plot, dase, 513, 569
b2plot, data manipulation, 536
b2plot, data plotting commands, 532
b2plot, dcol, 516, 561
b2plot, dd, 506
b2plot, ddw, 526, 536
b2plot, ddx, 513, 536
b2plot, ddy, 514, 536
b2plot, debu, 530
b2plot, debug, 530
b2plot, deltab, 506
b2plot, deltal, 506
b2plot, deltapar, 506
b2plot, deltaw, 506
b2plot, deltawn, 506
b2plot, deltawt, 506
b2plot, depo_ml, 524
b2plot, deposit, 524
b2plot, dfhe, 506
b2plot, dfhet, 506
b2plot, dfhi, 506
b2plot, dfhit, 506
b2plot, dfhj, 506
b2plot, dfhm, 506
b2plot, dfhp, 506
b2plot, dfmo, 506
b2plot, dfna, 506
b2plot, dhet, 506
b2plot, dhit, 506
b2plot, dib2, 507
b2plot, dkappadt, 527
b2plot, dmb2, 507
b2plot, dnneo, 507
b2plot, dp, 507
b2plot, drop, 514, 550
b2plot, dup, 514, 550
b2plot, dvua, 507
b2plot, dvue, 507
b2plot, dx, 507
b2plot, dy, 507
b2plot, eabs, 522, 536
b2plot, earow, 522, 562
b2plot, earox, 522, 562
b2plot, earoy, 522, 562
b2plot, east, 524
b2plot, eavg, 522, 570
b2plot, ebal, 507
b2plot, ebld, 507
b2plot, ebls, 507
b2plot, ecam, 520, 579
b2plot, echbx, 520, 579

b2plot, echmp, 520, 579
b2plot, echor, 520, 576
b2plot, echs, 522, 536
b2plot, echvl, 520, 579
b2plot, eclv, 516, 567
b2plot, eclvd, 522, 566
b2plot, eclvn, 522, 566
b2plot, eclvp, 522, 566
b2plot, eclvs, 522, 566
b2plot, ecol, 516, 563
b2plot, edena, 521
b2plot, edeni, 521
b2plot, edenm, 521
b2plot, edims, 522, 554
b2plot, edissml, 507
b2plot, edrop, 522, 554
b2plot, edup, 522, 554
b2plot, eexp, 522, 541
b2plot, ehx, 507
b2plot, ehy, 507
b2plot, ei, 524
b2plot, eirc, 514, 569
b2plot, Eirene quantities, 521
b2plot, Eirene stack commands, 522
b2plot, elbl, 522, 559
b2plot, elist, 522, 554
b2plot, eln, 522, 541
b2plot, elog, 522, 541
b2plot, em*, 522, 542
b2plot, em+, 522, 542
b2plot, em-, 522, 542
b2plot, em/, 522, 543
b2plot, emax, 518
b2plot, emeq, 522, 546
b2plot, emge, 522, 547
b2plot, emgt, 522, 547
b2plot, emin, 518
b2plot, emle, 522, 547
b2plot, emlt, 522, 547
b2plot, emmax, 522, 543
b2plot, emmin, 522, 543
b2plot, emne, 522, 547
b2plot, end, 530, 574
b2plot, eof, 530, 574
b2plot, eoutp, 522, 570
b2plot, eover, 523, 554
b2plot, epos, 523, 547
b2plot, eprt, 523, 570
b2plot, eps0, 530
b2plot, erel, 524
b2plot, ero_ml, 524
b2plot, erosion, 524
b2plot, esel, 523, 539
b2plot, esqrt, 523, 541
b2plot, esumz, 523, 539
b2plot, esurf, 523, 534
b2plot, eswap, 523, 554
b2plot, etal, 523, 569
b2plot, etos, 523, 554
b2plot, ev, 530
b2plot, evac, 523, 569
b2plot, ewid, 516, 563
b2plot, exba, 507
b2plot, exbe, 507
b2plot, exch, 97, 507
b2plot, exhe, 97, 507
b2plot, exhi, 97, 507
b2plot, exit, 530, 574
b2plot, exmo, 97, 507
b2plot, exna, 97, 507
b2plot, exne, 97
b2plot, exp, 541
b2plot, expf, 520, 563
b2plot, expl, 516, 563
b2plot, exps, 516, 563
b2plot, extl, 520, 574
b2plot, ezrng, 523, 539
b2plot, ezsel, 523, 540
b2plot, f.lx, 514, 533
b2plot, f.ly, 514, 533
b2plot, f.x, 514, 532
b2plot, f.y, 514, 533
b2plot, f30, 530, 571
b2plot, f31, 530, 571
b2plot, fac_exb, 507
b2plot, fac_vis, 507
b2plot, facdrift, 507
b2plot, fax_ext, 507
b2plot, fay_ext, 507
b2plot, fchp, 507
b2plot, fchx, 507
b2plot, fchx32, 507
b2plot, fchx52, 507
b2plot, fchxn, 507
b2plot, fchxvspa, 507
b2plot, fchxvspe, 507
b2plot, fchy, 507
b2plot, fchy32, 507
b2plot, fchy52, 507
b2plot, fchyn, 507
b2plot, fchyvspa, 507
b2plot, fchyvspe, 507
b2plot, fclv, 516, 567
b2plot, fcol, 516, 562
b2plot, fetx, 465, 467, 507, 513
b2plot, fety, 507, 513

b2plot, ffbz, 507
b2plot, fhetx, 507
b2plot, fhety, 507
b2plot, fhex, 507
b2plot, fhey, 507
b2plot, fhitx, 507
b2plot, fhity, 507
b2plot, fhix, 507
b2plot, fhiy, 507
b2plot, fhjx, 508
b2plot, fhjy, 508
b2plot, fhmx, 508
b2plot, fhmy, 508
b2plot, fhpx, 508
b2plot, fhpy, 508
b2plot, fhtx, 508, 513
b2plot, fhty, 508, 513
b2plot, filo, 516, 568
b2plot, flal, 508
b2plot, flalc, 508
b2plot, fle, 508
b2plot, fli, 508
b2plot, flke, 508
b2plot, flki, 508
b2plot, fluca, 508
b2plot, fluce, 508
b2plot, fluci, 508
b2plot, flucs, 508
b2plot, flucv, 508
b2plot, flux, 520, 579
b2plot, flvisc, 508
b2plot, fmax, 518
b2plot, fmin, 518
b2plot, fmox, 508
b2plot, fmoy, 508
b2plot, fmx, 508
b2plot, fmy, 508
b2plot, fnax, 508, 569
b2plot, fnax32, 508
b2plot, fnax52, 508
b2plot, fnaxnodr, 508
b2plot, fnay, 508, 569
b2plot, fnay32, 508
b2plot, fnay52, 508
b2plot, fnaynodr, 508
b2plot, fnex, 508
b2plot, fnex32, 508
b2plot, fnex52, 508
b2plot, fney, 508
b2plot, fney32, 508
b2plot, fney52, 508
b2plot, fnix32, 508
b2plot, fnix52, 508
b2plot, fniy32, 508
b2plot, fniy52, 508
b2plot, font, 516, 558
b2plot, fpsi, 514, 557
b2plot, fraa, 508
b2plot, gcol, 516, 562
b2plot, gdcl, 508
b2plot, glbl, 520, 559
b2plot, glob, 516, 560
b2plot, glsz, 516, 560
b2plot, graph appearance commands, 558
b2plot, graph format commands, 556
b2plot, graphformat, 556
b2plot, grid, 516, 564
b2plot, hcol, 516, 562
b2plot, help, 530
b2plot, hx, 508
b2plot, hy, 508
b2plot, hy1, 508
b2plot, i*, 516, 547
b2plot, i**, 516, 547
b2plot, i+, 516, 547
b2plot, i-, 516, 547
b2plot, i/, 516, 547
b2plot, ibnd, 524
b2plot, idrop, 516, 551
b2plot, idup, 516, 551
b2plot, iix, 508
b2plot, iiy, 508
b2plot, imapx, 524
b2plot, imapy, 524
b2plot, imp, 516, 572
b2plot, inttrgw, 528, 537
b2plot, inttrgw*, 528, 537
b2plot, inttrgz, 528, 537
b2plot, intw, 526, 537
b2plot, intw*, 526, 537
b2plot, intx, 514, 537
b2plot, inty, 514, 537
b2plot, ionrad, 508
b2plot, iover, 516, 552
b2plot, iprt, 516, 552
b2plot, iseq, 509
b2plot, isge, 509
b2plot, isgt, 509
b2plot, isle, 509
b2plot, islt, 509
b2plot, ismain, 530
b2plot, isne, 509
b2plot, iswap, 516, 552
b2plot, isweq, 524
b2plot, iswgt, 524
b2plot, iswle, 524

b2plot, iswlt, 524
b2plot, iswne, 524
b2plot, itoe, 516, 552
b2plot, itom, 516, 552
b2plot, itor, 516, 552
b2plot, itostr, 516, 552
b2plot, itot, 516, 552
b2plot, itow, 516, 552
b2plot, itrgeq, 527
b2plot, itrgge, 527
b2plot, itrggt, 527
b2plot, itrgle, 527
b2plot, itrglt, 527
b2plot, itrgne, 527
b2plot, iweq, 524
b2plot, iwge, 524
b2plot, iwgt, 524
b2plot, iwle, 524
b2plot, iwlt, 524
b2plot, iwne, 524
b2plot, ix, 509
b2plot, ixeq, 509
b2plot, ixge, 509
b2plot, ixgt, 509
b2plot, ixle, 509
b2plot, ixlt, 509
b2plot, ixne, 509
b2plot, ixref, 530
b2plot, iy, 509
b2plot, iyeq, 509
b2plot, iyge, 509
b2plot, iygt, 509
b2plot, iyle, 509
b2plot, iylt, 509
b2plot, iyne, 509
b2plot, iyref, 530
b2plot, joul, 509
b2plot, jsep, 530
b2plot, jxa, 530
b2plot, jxi, 530
b2plot, kappa, 527
b2plot, kappa_a, 524
b2plot, kappa_a1, 524
b2plot, kappa_a2, 524
b2plot, kappa_a3, 524
b2plot, kappa_a4, 524
b2plot, kappa_a5, 524
b2plot, kappa_ab, 527
b2plot, kappa_b, 524
b2plot, kappap, 527
b2plot, kappapa1, 524
b2plot, kappapa2, 524
b2plot, kappapa3, 524
b2plot, kappapa4, 524
b2plot, kappapa5, 524
b2plot, kappapdt, 527
b2plot, kbolt, 530
b2plot, ke, 509
b2plot, kye, 509
b2plot, kyi, 509
b2plot, kyi0, 509
b2plot, lacl, 516, 567
b2plot, lbof, 516, 568
b2plot, lbon, 516, 567
b2plot, lbsa, 517, 568
b2plot, lbsz, 517, 560
b2plot, lcp, 524
b2plot, ldepth, 524
b2plot, lemis, 524
b2plot, lfsz, 517, 560
b2plot, lhor, 517, 567
b2plot, line, 514, 533
b2plot, linf, 517, 556
b2plot, lint, 517, 534
b2plot, lintrg, 528, 557
b2plot, linw, 526, 565
b2plot, linx, 517, 557
b2plot, linxtrg, 528, 557
b2plot, linxw, 526, 565
b2plot, list, 514, 550
b2plot, lix, 509
b2plot, liy, 509
b2plot, lkappa, 525
b2plot, ln, 514, 540
b2plot, lnlam, 509
b2plot, lnn, 509
b2plot, log, 514, 540
b2plot, logf, 517, 556
b2plot, logt, 517, 534
b2plot, logtrg, 528, 557
b2plot, logw, 526, 565
b2plot, logx, 517, 557
b2plot, logxtrg, 528, 557
b2plot, logxw, 526, 565
b2plot, lrho, 525
b2plot, lthick, 525
b2plot, ltt, 509
b2plot, lwid, 517, 560
b2plot, m*, 514, 542
b2plot, m+, 514, 542
b2plot, m-, 514, 542
b2plot, m/, 514, 542
b2plot, mach, 509
b2plot, mach4, 509
b2plot, massa, 521
b2plot, massi, 521

b2plot, massm, 521
b2plot, mch2, 509
b2plot, mcol, 517, 564
b2plot, mdims, 514, 550
b2plot, me, 530
b2plot, meq, 514, 546
b2plot, mesh, 517, 533
b2plot, mesh plot, 580
b2plot, mge, 514, 546
b2plot, mgt, 514, 546
b2plot, mhal, 509
b2plot, mi, 509
b2plot, mi_ext, 509
b2plot, mle, 514, 546
b2plot, mlt, 514, 546
b2plot, mmax, 514, 542
b2plot, mmid, 514, 557
b2plot, mmin, 514, 542
b2plot, mne, 514, 546
b2plot, molrad, 509
b2plot, movi, 520, 579
b2plot, movie, 520, 579
b2plot, mp, 530
b2plot, mprt, 520, 569
b2plot, mpsi, 514, 558
b2plot, mtab, 520, 570
b2plot, mu0, 530
b2plot, mvsk, 517, 571
b2plot, na, 509, 569
b2plot, na0, 509
b2plot, na_ext, 509
b2plot, natm, 530
b2plot, nbnd, 530
b2plot, ndec, 517, 557
b2plot, ndepth, 530
b2plot, ne, 509
b2plot, ne2, 509
b2plot, ne_ext, 509
b2plot, nei_ic, 568
b2plot, neo_ic, 517
b2plot, nese, 514, 571
b2plot, neutrad, 509
b2plot, new, 517, 560
b2plot, nflfhix, 509
b2plot, nflfhiy, 509
b2plot, nflfnax, 509
b2plot, nflfnay, 509
b2plot, nh, 509
b2plot, ni, 509, 510
b2plot, ni0, 510
b2plot, nice, 517, 560
b2plot, nion, 530
b2plot, niso, 510
b2plot, nmol, 530
b2plot, nmovie, 520
b2plot, noax, 517, 560
b2plot, nocp, 517, 565
b2plot, north, 525
b2plot, nreg, 510
b2plot, ns, 530
b2plot, nseg, 525
b2plot, nspe, 514, 571
b2plot, nspec, 514
b2plot, nspecies, 530
b2plot, nua, 510
b2plot, nuab, 510
b2plot, nustar, 510
b2plot, nx, 530
b2plot, nxtl, 530
b2plot, nxtr, 530
b2plot, ny, 530
b2plot, ocol, 517, 564
b2plot, omax, 518
b2plot, omega, 510
b2plot, omin, 518
b2plot, omovie, 520, 579
b2plot, omp, 517, 572
b2plot, oson, 520, 580
b2plot, outl, 517, 564
b2plot, outp, 520, 570
b2plot, outptrg, 528, 571
b2plot, outpw, 526, 571
b2plot, over, 514, 550
b2plot, owid, 517, 564
b2plot, p*, 510
b2plot, page, 517, 556
b2plot, page format commands, 556
b2plot, part, 517, 574
b2plot, pbal, 510
b2plot, pdena, 521
b2plot, pdeni, 521
b2plot, pdenm, 521
b2plot, pdev, 520, 572
b2plot, pefa, 510
b2plot, pefm, 510
b2plot, pfla, 510, 569
b2plot, pflm, 510
b2plot, phi, 510
b2plot, phys, 517, 556
b2plot, pi, 530
b2plot, pit, 510
b2plot, pitch, 510
b2plot, plasma quantities, 505
b2plot, plathick, 525
b2plot, plot commands, 515
b2plot, plotting plasma quantities, 581

b2plot, plotting target quantities, 583
b2plot, plotting wall quantities, 582
b2plot, plwl, 526, 533
b2plot, pne, 514, 537
b2plot, po, 510
b2plot, po0, 510
b2plot, pos, 514, 546
b2plot, pot, 510
b2plot, pot*, 510
b2plot, poti, 510
b2plot, pr, 510
b2plot, psi, 510
b2plot, psta, 523, 538
b2plot, pstv, 523, 538
b2plot, pswa, 526, 538
b2plot, psx, 514, 538
b2plot, psxperp, 514, 538
b2plot, psy, 514, 538
b2plot, pux, 521
b2plot, puy, 521
b2plot, pvol, 514, 538
b2plot, pvx, 521
b2plot, pvy, 521
b2plot, pwne, 526, 538
b2plot, pz, 510
b2plot, q, 510
b2plot, qc, 510
b2plot, qe, 530
b2plot, qevap, 525
b2plot, qrad, 525
b2plot, quit, 530, 574
b2plot, qz0, 510
b2plot, qz1, 510
b2plot, r*, 517, 543
b2plot, r**, 517, 543
b2plot, r+, 517, 543
b2plot, r-, 517, 543
b2plot, r/, 517, 543
b2plot, r0, 519
b2plot, r1, 525
b2plot, r2, 525
b2plot, radiation plots, 584
b2plot, rc, 510
b2plot, rco, 510
b2plot, rcxi, 510
b2plot, rcxm, 510
b2plot, rcxn, 510
b2plot, rdi, 510
b2plot, rdi1, 510
b2plot, rdrop, 517, 550
b2plot, rdup, 517, 551
b2plot, real variables, 518
b2plot, ree, 510
b2plot, refa, 510
b2plot, refm, 510
b2plot, rei, 510
b2plot, rem*, 523, 544
b2plot, rem**, 523, 544
b2plot, rem+, 523, 544
b2plot, rem-, 523, 544
b2plot, rem/, 523, 544
b2plot, remma, 523, 544
b2plot, remmi, 523, 544
b2plot, reso, 517, 571
b2plot, rfla, 510, 569
b2plot, rflm, 510
b2plot, rhol, 510
b2plot, rigz, 514, 517, 540
b2plot, rix, 510
b2plot, riy, 510
b2plot, rlcl, 510
b2plot, rm*, 514, 543
b2plot, rm**, 514, 544
b2plot, rm+, 514, 543
b2plot, rm-, 515, 543
b2plot, rm/, 515, 544
b2plot, rmax, 519
b2plot, rmin, 519
b2plot, rmma, 515, 544
b2plot, rmmax, 515
b2plot, rmmi, 515, 544
b2plot, rmmin, 515
b2plot, rmo, 510
b2plot, rmt, 510
b2plot, ro, 510, 569
b2plot, rover, 517, 551
b2plot, rpo, 510
b2plot, rprt, 517, 551
b2plot, rpsi, 515, 558
b2plot, rqae, 510
b2plot, rrai, 511
b2plot, rram, 511
b2plot, rran, 511
b2plot, rsai, 511
b2plot, rsam, 511
b2plot, rsan, 511
b2plot, rshi, 519
b2plot, rswap, 517, 551
b2plot, rtoe, 517, 551
b2plot, rtom, 517, 551
b2plot, rtostr, 517, 551
b2plot, rtot, 517, 551
b2plot, rtow, 517, 551
b2plot, rtrgm*, 528, 545
b2plot, rtrgm**, 528, 545
b2plot, rtrgm+, 528, 545

b2plot, rtrgm-, 528, 545
b2plot, rtrgm/, 528, 545
b2plot, rtrgmma, 528, 545
b2plot, rtrgmmi, 528, 545
b2plot, rud, 511
b2plot, rudc, 511
b2plot, rvd, 511
b2plot, rwm*, 526, 545
b2plot, rwm**, 526, 546
b2plot, rwm+, 526, 545
b2plot, rwm-, 526, 545
b2plot, rwm/, 526, 545
b2plot, rwmma, 526, 546
b2plot, rwmmi, 526, 546
b2plot, s, 511
b2plot, same, 517, 560
b2plot, saw, 520, 553
b2plot, saws, 520, 553
b2plot, sawx, 520, 553
b2plot, scat, 520, 552
b2plot, sch, 511
b2plot, scol, 517, 563
b2plot, sdrop, 520, 552
b2plot, sdup, 520, 553
b2plot, sep, 517, 563
b2plot, sepw, 518, 564
b2plot, shco, 518, 573
b2plot, she, 511
b2plot, shi, 511
b2plot, sig, 511
b2plot, slbl, 520, 559
b2plot, smaf, 511
b2plot, smag, 511
b2plot, smav, 511
b2plot, smax, 519
b2plot, smfr, 511
b2plot, smin, 519
b2plot, smo, 511
b2plot, smp, 511
b2plot, smpr, 511
b2plot, smpt, 511
b2plot, smq, 511
b2plot, smth, 511
b2plot, sna, 511
b2plot, sonn, 520, 579
b2plot, south, 525
b2plot, sover, 520, 553
b2plot, spec, 515, 571
b2plot, spln, 518, 568
b2plot, sprt, 520, 553
b2plot, sput, 525
b2plot, sput2, 525
b2plot, sputchem, 525
b2plot, sputfrac, 525
b2plot, sputres, 525
b2plot, sq, 518, 556
b2plot, sqrt, 515, 540
b2plot, srcml, 511
b2plot, sswap, 520, 553
b2plot, stack commands, 513
b2plot, stack manipulation, 550
b2plot, stacks, 504
b2plot, stckcoef, 525
b2plot, stdl, 518, 559
b2plot, stoe, 515, 553
b2plot, stow, 515, 553
b2plot, str, 511
b2plot, sublheat, 525
b2plot, sumt, 523, 538
b2plot, sumtrgw, 528, 538
b2plot, sumtrgz, 528, 538
b2plot, sumw, 526, 539
b2plot, sumx, 515, 539
b2plot, sumy, 515, 539
b2plot, sumz, 515, 539
b2plot, surf, 518, 533
b2plot, swap, 515, 550
b2plot, swid, 518, 572
b2plot, sx, 511
b2plot, sxperp, 511
b2plot, sy, 511
b2plot, sz, 511
b2plot, t_#p, 515, 534
b2plot, t_pt, 519
b2plot, t_r+, 518, 574
b2plot, t_r-, 518, 574
b2plot, ta_ext, 511
b2plot, tab2, 511
b2plot, target, 518, 565
b2plot, target commands, 528
b2plot, target quantities, 527
b2plot, tclvd, 528, 566
b2plot, tclvn, 528, 566
b2plot, tclvp, 528, 567
b2plot, tclvs, 528, 566
b2plot, tcol, 518, 562
b2plot, te, 511
b2plot, te0, 511
b2plot, tese, 515, 572
b2plot, tgdns, 525
b2plot, thermev, 525
b2plot, ti, 511
b2plot, ti0, 511
b2plot, tib2, 511
b2plot, time, 525
b2plot, tindex, 521

b2plot, tix, 511
b2plot, tiy, 511
b2plot, tlbl, 520, 559
b2plot, tmb2, 511
b2plot, traj, 515, 534
b2plot, trajectories plot, 534
b2plot, tranprof, 530, 571
b2plot, trg.w, 529, 533
b2plot, trg.z, 529, 533
b2plot, trgabs, 529, 536
b2plot, trgblbl, 520, 559
b2plot, trgchs, 529, 536
b2plot, trgddp, 529, 537
b2plot, trgddw, 529, 537
b2plot, trgddz, 529, 537
b2plot, trgdepth, 527
b2plot, trgdrop, 529, 554
b2plot, trgdup, 529, 554
b2plot, trgeast, 527
b2plot, trgeq, 529, 548
b2plot, trgexp, 529, 541
b2plot, trgfmax, 519
b2plot, trgfmin, 519
b2plot, trgge, 529, 548
b2plot, trggt, 529, 548
b2plot, trgibnd, 527
b2plot, trgiz, 527
b2plot, trgle, 529, 548
b2plot, trglist, 529, 555
b2plot, trgln, 529, 541
b2plot, trglog, 529, 541
b2plot, trglt, 529, 548
b2plot, trgm*, 529, 548
b2plot, trgm+, 529, 548
b2plot, trgm-, 529, 548
b2plot, trgm/, 529, 549
b2plot, trgmmax, 529, 548
b2plot, trgmmin, 529, 548
b2plot, trgne, 529, 548
b2plot, trgnorth, 527
b2plot, trgoff, 518, 565
b2plot, trgomax, 519
b2plot, trgomin, 519
b2plot, trgover, 529, 554
b2plot, trgpos, 529, 549
b2plot, trgprt, 520, 570
b2plot, trgr1, 527
b2plot, trgr2, 527
b2plot, trgrmax, 519
b2plot, trgrmin, 519
b2plot, trgseg, 527
b2plot, trgsouth, 527
b2plot, trgsqrt, 529, 541
b2plot, trgsurf, 529, 534
b2plot, trgswap, 529, 554
b2plot, trgtab, 520, 570
b2plot, trgtemp, 527
b2plot, trgtow, 529, 555
b2plot, trgvol, 527
b2plot, trgwest, 527
b2plot, trgwidth, 527
b2plot, trgwsel, 529, 540
b2plot, trgxmax, 519
b2plot, trgxmin, 519
b2plot, trgymax, 519
b2plot, trgymin, 519
b2plot, trgz1, 528
b2plot, trgz2, 528
b2plot, trgzmax, 519
b2plot, trgzmin, 519
b2plot, trgzsel, 529, 540
b2plot, triarea, 521
b2plot, triavol, 521
b2plot, trig, 518, 564
b2plot, trigmesh, 518, 533
b2plot, triple plot, 583
b2plot, troubleshooting, 531
b2plot, tt0, 519
b2plot, ua, 511, 513, 569
b2plot, ua0, 511
b2plot, ua_ext, 511
b2plot, ud, 511
b2plot, ue, 511, 513
b2plot, uedgegrd, 520, 580
b2plot, up, 511, 569
b2plot, user, 530, 572
b2plot, uu, 511
b2plot, uuse, 515, 534, 569
b2plot, vadia.x, 512
b2plot, vadia.y, 512
b2plot, valfven, 512
b2plot, vapprs, 525
b2plot, vapprs_a, 525
b2plot, vapprs_b, 525
b2plot, vaprtemp, 525
b2plot, vcol, 518, 564
b2plot, vcon, 512
b2plot, vd, 512
b2plot, vedia.x, 512
b2plot, vedia.y, 512
b2plot, veexb.x, 512
b2plot, veexb.y, 512
b2plot, vesl, 518, 564
b2plot, vexb.x, 512
b2plot, vexb.y, 512
b2plot, visa, 512

b2plot, visc, 512
b2plot, vlax, 512
b2plot, vlay, 512
b2plot, vnneo, 512
b2plot, vol, 512
b2plot, vth, 512
b2plot, vthe, 512
b2plot, vthi, 512
b2plot, vv, 512, 569
b2plot, vv0, 512
b2plot, vvse, 515, 534, 569
b2plot, vwid, 518, 564
b2plot, vxa, 521
b2plot, vxdena, 521
b2plot, vxdeni, 521
b2plot, vxdenm, 521
b2plot, vxi, 521
b2plot, vxm, 521
b2plot, vya, 521
b2plot, vydena, 521
b2plot, vydeni, 521
b2plot, vydenm, 521
b2plot, vyi, 521
b2plot, vym, 521
b2plot, vza, 521
b2plot, vzdena, 521
b2plot, vzdeni, 522
b2plot, vzdenm, 522
b2plot, vzi, 522
b2plot, vzm, 522
b2plot, wabs, 526, 536
b2plot, wadia.x, 512
b2plot, wadia.y, 512
b2plot, wall, 518, 526, 564
b2plot, wall commands, 525
b2plot, wall quantities, 523
b2plot, wall_ank, 526, 574
b2plot, wallarea, 525
b2plot, wallblbl, 520, 560
b2plot, walldist, 525
b2plot, wallf, 526, 563
b2plot, wallflux, 525
b2plot, wallhflx, 525
b2plot, wallngth, 525
b2plot, walltemp, 525
b2plot, wb, 512
b2plot, wce, 512
b2plot, wchs, 526, 536
b2plot, wci, 512
b2plot, wclvd, 526, 567
b2plot, wclvn, 526, 567
b2plot, wclvp, 526, 567
b2plot, wclvs, 526, 567
b2plot, wcol, 518, 562
b2plot, wdrop, 526, 555
b2plot, wdup, 526, 555
b2plot, west, 525
b2plot, wexp, 526, 542
b2plot, wlbl, 520, 559
b2plot, wlist, 526, 555
b2plot, wlld, 530
b2plot, wln, 526, 541
b2plot, wlog, 526, 541
b2plot, wm*, 526, 549
b2plot, wm+, 526, 549
b2plot, wm-, 526, 549
b2plot, wm/, 526, 549
b2plot, wmax, 519
b2plot, wmeq, 526, 549
b2plot, wmge, 526, 549
b2plot, wmgt, 526, 550
b2plot, wmin, 519
b2plot, wmle, 526, 550
b2plot, wmlt, 526, 550
b2plot, wmmax, 526, 549
b2plot, wmmin, 526, 549
b2plot, wmne, 526, 549
b2plot, woff, 526, 565
b2plot, wover, 527, 555
b2plot, wpe, 512
b2plot, wpos, 527, 550
b2plot, wprt, 521, 527, 570
b2plot, writ, 518, 569
b2plot, write, 518
b2plot, wsaw, 527, 553
b2plot, wsel, 527, 539
b2plot, wslbl, 521, 559
b2plot, wsqrt, 527, 542
b2plot, wsumz, 527, 539
b2plot, wswap, 527, 555
b2plot, wtab, 521, 570
b2plot, wtos, 515, 527, 555
b2plot, wtot, 527, 555
b2plot, ww, 512
b2plot, wwid, 527, 565
b2plot, wzrng, 527, 539
b2plot, wzsel, 527, 540
b2plot, xcol, 518, 562
b2plot, xlab, 521, 575
b2plot, xmax, 519
b2plot, xmin, 519
b2plot, xpar, 518, 557
b2plot, xpol, 518, 557
b2plot, xref, 519
b2plot, xreg, 512
b2plot, xsel, 515, 540

b2plot, xtow, 515, 555
b2plot, xymap, 512
b2plot, xyrz, 518, 572
b2plot, ylab, 521, 575
b2plot, ymax, 519
b2plot, ymin, 519
b2plot, yref, 518, 572
b2plot, yreg, 512
b2plot, ysel, 515, 540
b2plot, ytow, 515, 555
b2plot, ytra, 518, 563
b2plot, z0, 519
b2plot, z1, 525
b2plot, z2, 525
b2plot, za, 512
b2plot, za2_ext, 512
b2plot, za_ext, 512
b2plot, zcol, 518, 561
b2plot, zeff, 512
b2plot, zero, 518, 561
b2plot, zi, 512
b2plot, zmax, 519
b2plot, zmin, 519
b2plot, zn, 512
b2plot, zn_ext, 512
b2plot, zrng, 515, 539
b2plot, zsel, 515, 540
b2plot, zshi, 519
b2plot, ztrgeq, 528
b2plot, ztrgge, 528
b2plot, ztrggt, 528
b2plot, ztrgle, 528
b2plot, ztrglt, 528
b2plot, ztrgne, 528
b2plot, zwid, 518, 561
b2plot,tr, 521
b2plot,tz, 521
B2PLOT_DEV, 505
b2pp, 57
b2pr, 57
b2pt, 57
b2qie, 475
b2radreg, 475
b2rd, 137, 179, 675
b2rd.prt, 677
b2run, 45, 63, 67, 71, 76, 78, 80, 121, 123, 124, 127–130, 132, 135–137, 178
b2run_hgw_clus, 45
b2sext_sch_reg, 476
b2sext_sch_reg_region, 48
b2sext_she_reg, 476
b2sext_she_reg_region, 48
b2sext_shi_reg, 476
b2sext_shi_reg_region, 48
b2sext_smo_reg, 47, 476
b2sext_smo_reg_region, 48
b2sext_sna_reg, 47, 476
b2sext_sna_reg_region, 48
b2sext_sne_reg, 476
b2sext_sne_reg_region, 48
b2she, 475
b2she0, 475
b2shi, 475
b2shi0, 475
b2specp, 124, 678
b2sral, 97
b2sral_inputfile, 97
b2stbc, 81
b2stbc_boundary_namelist, 81
b2stbc_sch_reg, 475
b2stbc_sch_reg_region, 48
b2stbc_she_reg, 475
b2stbc_she_reg_region, 48
b2stbc_shi_reg, 475
b2stbc_shi_reg_region, 48
b2stbc_smo_reg, 47, 475
b2stbc_smo_reg_region, 48
b2stbc_sna_reg, 47, 475
b2stbc_sna_reg_region, 48
b2stbc_sne_reg, 475
b2stbc_sne_reg_region, 48
b2stbm_sch_reg, 476
b2stbm_sch_reg_region, 48
b2stbm_she_reg, 476
b2stbm_she_reg_region, 48
b2stbm_shi_reg, 476
b2stbm_shi_reg_region, 48
b2stbm_smo_reg, 47, 476
b2stbm_smo_reg_region, 48
b2stbm_sna_reg, 47, 476
b2stbm_sna_reg_region, 48
b2stbm_sne_reg, 476
b2stbm_sne_reg_region, 48
b2stbr_sch_reg, 475
b2stbr_sch_reg_region, 48
b2stbr_she_reg, 475
b2stbr_she_reg_region, 48
b2stbr_shi_reg, 476
b2stbr_shi_reg_region, 48
b2stbr_smo_reg, 47, 476
b2stbr_smo_reg_region, 48
b2stbr_sna_reg, 47, 476
b2stbr_sna_reg_region, 48
b2stbr_sne_reg, 476
b2stbr_sne_reg_region, 48
b2str, 476

b2sxdr, 19, 45
b2tallies.nc, 135, 475
b2tallies.nc, b2bremreg, 475
b2tallies.nc, b2divua, 475
b2tallies.nc, b2divue, 475
b2tallies.nc, b2exba, 475
b2tallies.nc, b2exbe, 475
b2tallies.nc, b2fraa, 475
b2tallies.nc, b2joule, 475
b2tallies.nc, b2qie, 475
b2tallies.nc, b2radreg, 475
b2tallies.nc, b2sext_sch_reg, 476
b2tallies.nc, b2sext_she_reg, 476
b2tallies.nc, b2sext_shi_reg, 476
b2tallies.nc, b2sext_smo_reg, 476
b2tallies.nc, b2sext_sna_reg, 476
b2tallies.nc, b2sext_sne_reg, 476
b2tallies.nc, b2she, 475
b2tallies.nc, b2she0, 475
b2tallies.nc, b2shi, 475
b2tallies.nc, b2shi0, 475
b2tallies.nc, b2stbc_sch_reg, 475
b2tallies.nc, b2stbc_she_reg, 475
b2tallies.nc, b2stbc_shi_reg, 475
b2tallies.nc, b2stbc_smo_reg, 475
b2tallies.nc, b2stbc_sna_reg, 475
b2tallies.nc, b2stbc_sne_reg, 475
b2tallies.nc, b2stbm_sch_reg, 476
b2tallies.nc, b2stbm_she_reg, 476
b2tallies.nc, b2stbm_shi_reg, 476
b2tallies.nc, b2stbm_smo_reg, 476
b2tallies.nc, b2stbm_sna_reg, 476
b2tallies.nc, b2stbm_sne_reg, 476
b2tallies.nc, b2stbr_sch_reg, 475
b2tallies.nc, b2stbr_she_reg, 475
b2tallies.nc, b2stbr_shi_reg, 476
b2tallies.nc, b2stbr_smo_reg, 476
b2tallies.nc, b2stbr_sna_reg, 476
b2tallies.nc, b2stbr_sne_reg, 476
b2tallies.nc, b2str, 476
b2tallies.nc, b2visa, 476
b2tallies.nc, b2wrong1, 476
b2tallies.nc, b2wrong2, 476
b2tallies.nc, b2wrong3, 476
b2tallies.nc, fchxreg, 476
b2tallies.nc, fchyreg, 476
b2tallies.nc, fhexreg, 476
b2tallies.nc, fheyreg, 476
b2tallies.nc, fhixreg, 476
b2tallies.nc, fhiyreg, 476
b2tallies.nc, fhjxreg, 476
b2tallies.nc, fhjyreg, 476
b2tallies.nc, fhmxreg, 476
b2tallies.nc, fhmyreg, 476
b2tallies.nc, fhpxreg, 476
b2tallies.nc, fhpyreg, 476
b2tallies.nc, fhtxreg, 476
b2tallies.nc, fhtyreg, 476
b2tallies.nc, fnaxreg, 476
b2tallies.nc, fnayreg, 476
b2tallies.nc, nareg, 476
b2tallies.nc, ne2reg, 476
b2tallies.nc, nereg, 476
b2tallies.nc, nireg, 476
b2tallies.nc, poreg, 476
b2tallies.nc, qconvexreg, 476
b2tallies.nc, qconveyreg, 476
b2tallies.nc, qconvixreg, 476
b2tallies.nc, qconviyreg, 476
b2tallies.nc, quantities stored in, 475
b2tallies.nc, rcxhireg, 476
b2tallies.nc, rcxmoreg, 476
b2tallies.nc, rcxnareg, 477
b2tallies.nc, rdneureg, 477
b2tallies.nc, resco_reg, 477
b2tallies.nc, reshe_reg, 477
b2tallies.nc, reshi_reg, 477
b2tallies.nc, resmo_reg, 477
b2tallies.nc, resmt_reg, 477
b2tallies.nc, respo_reg, 477
b2tallies.nc, rqahereg, 477
b2tallies.nc, rqbrmreg, 477
b2tallies.nc, rqradreg, 477
b2tallies.nc, rrahireg, 477
b2tallies.nc, rramoreg, 477
b2tallies.nc, rranareg, 477
b2tallies.nc, rsahireg, 477
b2tallies.nc, rsamoreg, 477
b2tallies.nc, rsanareg, 477
b2tallies.nc, tereg, 477
b2tallies.nc, tireg, 477
b2tallies.nc, volreg, 477
b2time.nc, 135, 178, 465
b2time.nc, an3da, 468
b2time.nc, an3di, 468
b2time.nc, an3dl, 468
b2time.nc, an3dr, 468
b2time.nc, an3dtl, 469
b2time.nc, an3dtr, 469
b2time.nc, batchsa, 470
b2time.nc, dn3da, 468
b2time.nc, dn3di, 468
b2time.nc, dnsepm, 466
b2time.nc, dp3da, 468
b2time.nc, dp3di, 468
b2time.nc, dpsepm, 466

b2time.nc, fc3dl, 467
b2time.nc, fc3dr, 468
b2time.nc, fc3dtl, 469
b2time.nc, fc3dtr, 469
b2time.nc, fch2d, 470
b2time.nc, fchsap, 467
b2time.nc, fchsapp, 467
b2time.nc, fchsip, 467
b2time.nc, fchsipp, 467
b2time.nc, fchxap, 466
b2time.nc, fchxip, 465
b2time.nc, fchyap, 466
b2time.nc, fchyip, 466
b2time.nc, fe3dl, 467
b2time.nc, fe3dr, 468
b2time.nc, fe3dtl, 469
b2time.nc, fe3dtr, 469
b2time.nc, feesap, 467
b2time.nc, feesapp, 467
b2time.nc, feesip, 467
b2time.nc, feesipp, 467
b2time.nc, feexap, 465
b2time.nc, feexip, 465
b2time.nc, feeyap, 466
b2time.nc, feeyip, 466
b2time.nc, feisap, 467
b2time.nc, feisapp, 467
b2time.nc, feisip, 467
b2time.nc, feisipp, 467
b2time.nc, feixap, 466
b2time.nc, feixip, 465
b2time.nc, feiyap, 466
b2time.nc, feiyip, 466
b2time.nc, fetsap, 467
b2time.nc, fetsapp, 467
b2time.nc, fetsip, 467
b2time.nc, fetsipp, 467
b2time.nc, fetxap, 466
b2time.nc, fetxip, 465
b2time.nc, fetyap, 466
b2time.nc, fetyip, 466
b2time.nc, fhe2d, 470
b2time.nc, fhi2d, 470
b2time.nc, fi3dl, 467
b2time.nc, fi3dr, 468
b2time.nc, fi3dtl, 469
b2time.nc, fi3dtr, 469
b2time.nc, fl3dl, 467
b2time.nc, fl3dr, 468
b2time.nc, fl3dtl, 469
b2time.nc, fl3dtr, 469
b2time.nc, fn3dl, 467
b2time.nc, fn3dr, 468
b2time.nc, fn3dtl, 469
b2time.nc, fn3dtr, 469
b2time.nc, fna2d, 470
b2time.nc, fnisap, 467
b2time.nc, fnisapp, 467
b2time.nc, fnisip, 467
b2time.nc, fnisipp, 467
b2time.nc, fnixap, 465
b2time.nc, fnixip, 465
b2time.nc, fniyap, 466
b2time.nc, fniyip, 466
b2time.nc, fo3dl, 467
b2time.nc, fo3dr, 468
b2time.nc, fo3dtl, 469
b2time.nc, fo3dtr, 469
b2time.nc, ft3dl, 467
b2time.nc, ft3dr, 468
b2time.nc, ft3dtl, 469
b2time.nc, ft3dtr, 469
b2time.nc, ke3da, 468
b2time.nc, ke3di, 468
b2time.nc, kesepm, 466
b2time.nc, ki3da, 468
b2time.nc, ki3di, 468
b2time.nc, kin2d, 469
b2time.nc, kisepm, 466
b2time.nc, mn3da, 468
b2time.nc, mn3di, 468
b2time.nc, mn3dl, 468
b2time.nc, mn3dr, 468
b2time.nc, mn3dtl, 469
b2time.nc, mn3dtr, 469
b2time.nc, nastep, 470
b2time.nc, ne2d, 469
b2time.nc, ne3da, 468
b2time.nc, ne3di, 468
b2time.nc, ne3dl, 467
b2time.nc, ne3dr, 468
b2time.nc, ne3dtl, 469
b2time.nc, ne3dtr, 469
b2time.nc, nemxap, 466
b2time.nc, nemxap_av, 470
b2time.nc, nemxap_std, 471
b2time.nc, nemxip, 466
b2time.nc, nemxip_av, 470
b2time.nc, nemxip_std, 471
b2time.nc, nesepa, 466
b2time.nc, nesepa_av, 470
b2time.nc, nesepa_std, 471
b2time.nc, nesepi, 466
b2time.nc, nesepi_av, 470
b2time.nc, nesepi_std, 470
b2time.nc, nesepm, 466

b2time.nc, nesepm_av, 470
b2time.nc, nesepm_std, 470
b2time.nc, ntstep, 465
b2time.nc, po2d, 469
b2time.nc, po3da, 468
b2time.nc, po3di, 468
b2time.nc, po3dl, 468
b2time.nc, po3dr, 468
b2time.nc, po3dtl, 469
b2time.nc, po3dtr, 469
b2time.nc, pomxap, 466
b2time.nc, pomxap_av, 470
b2time.nc, pomxap_std, 471
b2time.nc, pomxip, 466
b2time.nc, pomxip_av, 470
b2time.nc, pomxip_std, 471
b2time.nc, posepa, 466
b2time.nc, posepa_av, 470
b2time.nc, posepa_std, 471
b2time.nc, posepi, 466
b2time.nc, posepi_av, 470
b2time.nc, posepi_std, 470
b2time.nc, posepm, 466
b2time.nc, posepm_av, 470
b2time.nc, posepm_std, 471
b2time.nc, pwmxap, 466
b2time.nc, pwmxip, 466
b2time.nc, quantities stored in, 465
b2time.nc, rcxhi2d, 470
b2time.nc, rcxna2d, 470
b2time.nc, rqahe2d, 470
b2time.nc, rqrad2d, 470
b2time.nc, rrahi2d, 470
b2time.nc, rrana2d, 470
b2time.nc, rsahi2d, 470
b2time.nc, rsana2d, 470
b2time.nc, te2d, 469
b2time.nc, te3da, 468
b2time.nc, te3di, 468
b2time.nc, te3dl, 467
b2time.nc, te3dr, 468
b2time.nc, te3dtl, 469
b2time.nc, te3dtr, 469
b2time.nc, temxap, 466
b2time.nc, temxap_av, 470
b2time.nc, temxap_std, 471
b2time.nc, temxip, 466
b2time.nc, temxip_av, 470
b2time.nc, temxip_std, 471
b2time.nc, tesepa, 466
b2time.nc, tesepa_av, 470
b2time.nc, tesepa_std, 471
b2time.nc, tesepi, 466
b2time.nc, tesepi_av, 470
b2time.nc, tesepi_std, 470
b2time.nc, tesepm, 466
b2time.nc, tesepm_av, 470
b2time.nc, tesepm_std, 471
b2time.nc, ti2d, 469
b2time.nc, ti3da, 468
b2time.nc, ti3di, 468
b2time.nc, ti3dl, 468
b2time.nc, ti3dr, 468
b2time.nc, ti3dtl, 469
b2time.nc, ti3dtr, 469
b2time.nc, timesa, 465
b2time.nc, timxap, 466
b2time.nc, timxap_av, 470
b2time.nc, timxap_std, 471
b2time.nc, timxip, 466
b2time.nc, timxip_av, 470
b2time.nc, timxip_std, 471
b2time.nc, tisepa, 466
b2time.nc, tisepa_av, 470
b2time.nc, tisepa_std, 471
b2time.nc, tisepi, 466
b2time.nc, tisepi_av, 470
b2time.nc, tisepi_std, 470
b2time.nc, tisepm, 466
b2time.nc, tisepm_av, 470
b2time.nc, tisepm_std, 471
b2time.nc, tmhacore, 466
b2time.nc, tmhadiv, 467
b2time.nc, tmhasol, 467
b2time.nc, tmne, 466
b2time.nc, tmte, 466
b2time.nc, tmti, 466
b2time.nc, tp3dl, 468
b2time.nc, tp3dr, 468
b2time.nc, tp3dtl, 469
b2time.nc, tpmxap, 466
b2time.nc, tpmxip, 466
b2time.nc, tpsepa, 466
b2time.nc, tpsepi, 466
b2time.nc, vs3da, 468
b2time.nc, vs3di, 468
b2time.nc, vssepm, 466
b2time.nc, vx3da, 468
b2time.nc, vx3di, 468
b2time.nc, vxsepm, 466
b2time.nc, vy3da, 468
b2time.nc, vy3di, 468
b2time.nc, vysepm, 466
b2tqna, 90, 94
b2tqna_inputfile, 94
b2tqna_transport_namelist, 90

b2ts, 675
b2uf, 136, 675
b2uf.prt, 677
b2us, 57
b2visa, 476
b2wall.nc, 135
b2wrong1, 476
b2wrong2, 476
b2wrong3, 476
b2xd, 19, 675
b2ydm, 141, 675
b2ydm.dat, 141, 677
b2ydm.prt, 677
b2ye, 178, 675
b2ye.prt, 677
b2yg, 121, 676
b2yg.dat, 677
b2yg.prt, 677
b2yh, 123, 676
b2yh.dat, 677
b2yh.prt, 677
b2yi, 124, 676
b2yi.dat, 677
b2yi.prt, 677
b2yi_gnuplot, 676
b2ym, 128, 676
b2ym.dat, 677
b2ym.prt, 677
b2ymb, 138, 676
b2ymb.dat, 138, 677
b2ymb.prt, 677
b2yn, 129, 676
b2yn.dat, 677
b2yn.prt, 677
b2yp, 124, 676
b2yp.dat, 677
b2yp.prt, 677
b2yq, 130, 676
b2yq.dat, 677
b2yq.prt, 677
b2yr, 127, 676
b2yr.dat, 127, 677
b2yr.prt, 677
b2yrp, 139, 676
b2yrp.dat, 139, 677
b2yrp.prt, 677
b2yt, 19, 57, 132, 676
b2yt.dat, 677
b2yt.prt, 677
balance, 61
Balescu, 369, 390
baserun, 40
basic parameters, 74
batchsa, 470
BCCHAR, 81
BCCON, 82
BCEND, 82
BCENE, 85
BCENI, 86
BCMOM, 84
BCPOS, 82
BCPOT, 88
BCSTART, 82
blank, 55
boundary conditions, 74, 81, 455
boundary conditions, drifts, 90
boundary conditions, drifts and currents, 361
BOUNDARY_FILENAME, 89
BOUNDARY_TIME_MOD, 89
BOUNDARY_TIME_SWITCH, 89
Braginskii, 369
bremsstrahlung, 52
BUILDING_CARRE, 617
builds, 37
Bulk_properties, 40

calculate_sputter_inner_target.py, 60
calculate_sputter_outer_target.py, 60
canonical, 55
Carre, 57
carre, 57
Carre, compiling, 27, 211
Carre, obtaining, 19
Carre, running, 211
Carre, Updating, 19
CATALYST, 615
CDF_MOVIE, 619
cell corner indices, 513
CFLAB, 92
CFLAL, 91
CFLME, 91
CFLME_CORE, 92
CFLME_SOL, 92
CFLME_TANH_A, 92
CFLME_TANH_B, 92
CFLMI, 91
CFLMI_CORE, 92
CFLMI_SOL, 92
CFLMI_TANH_A, 92
CFLMI_TANH_B, 92
CFLMV, 91
CFLMV_CORE, 92
CFLMV_SOL, 92
CFLMV_TANH_A, 92
CFLMV_TANH_B, 92
check_b2_output, 52
CHECKBIN, 632

chop_pwd, 55
Classical transport, 369
CLASSICAL_CURRENT, 619
climb_and_look_for, 57
cnvtria, 60
COCOS, 372
commands, 61
compare_2_solps.py, 60
COMPILER_MAJOR_VERSION, 615
COMPILER_MINOR_VERSION, 615
Compiling the code, 27
COMPLICATED, 620
CON_FN, 89
config, 37
CONPAR, 88
CONSTANTS_PROVIDED, 615
convert_from_solps4.3, 45
Converting from SOLPS4 to SOLPS5, 18
Converting from SOLPS5 to SOLPS4, 18
copy_tracing_files, 57
CORE_DT_FACTOR, 93
CORE_DT_SUPPRESSION, 93
CORR_CORE_DN, 93
CORR_CORE_DT, 93
correct_b2yt_timestamps, 57, 134
correct_baserun_timestamps, 57, 134
count, 55
Coupling to other codes via MDSplus, 205
cpu, 52, 102
CRAY, 616
cropequ, 655
csd3submit, 45, 64
cumulussubmit, 45, 64
currents.py, 52
cview, 193

D+_fluid_fluxes, 52
D0_fluid_fluxes, 52
d2d, 57, 653
d2e, 57, 653
d2v, 57, 654
D_coupled_conservation, 52
D_coupled_conservation_core, 52
D_fluid_conservation, 52
D_fluid_conservation_core, 52
D_fluid_fluxes, 52
dab2, 473
darwinsubmit, 45, 64
data, 62
database, 202
database query via IDL, 202
database query via python, 203
datestamp, 55
DBG, 616
DBX, 34
DEF_ISOEXTRA, 588
DEF_MAXCHORDS, 588
DEF_NADD, 588
DEF_NATM, 588
DEF_NBC, 588
DEF_NCHEN, 588
DEF_NCUT, 588
DEF_NFL, 588
DEF_NGSTAL, 588
DEF_NION, 588
DEF_NISO, 588
DEF_NLIM, 589
DEF_NMOL, 589
DEF_NPHID, 589
DEF_NPLS, 589
DEF_NPRNL, 589
DEF_NRCX, 589
DEF_NREAC, 589
DEF_NREC, 589
DEF_NREI, 589
DEF_NREL, 589
DEF_NSD, 589
DEF_NSPCSRFG, 589
DEF_NSRFS, 589
DEF_NSTRA, 589
DEF_NSTS, 589
DEF_NTOR, 589
DEF_NXD, 589
DEF_NYD, 589
define_compilers, 55
DEFINES, used in compiling B2.5, 619
DEFINES, used in compiling B2.5-Eirene, 619
DEFINES, used in compiling Carre, 617
DEFINES, used in compiling DivGeo, 618
DEFINES, used in compiling Eirene, 632
DEFINES, used throughout SOLPS-ITER, 615
delta, 52
deltp1, 212
deltpn, 212
deltr1, 212, 213
deltrn, 212, 213
density, 52
determine_SOLPSTOP, 57
DEVICE, 34, 209
DG, adding elements, 649
DG, adding grid points, 661
DG, adding surfaces, 658
DG, change sign of equilibrium, 654
DG, changing external normals, 650
DG, chop equilibrium, 656
DG, chords, 662
DG, configuration file, 669

DG, connecting points, 649
DG, convert to Carre equilibrium, 654
DG, convert to EFIT equilibrium, 653
DG, converting a template, 649
DG, converting from Carre equilibrium, 655
DG, converting from EFIT equilibrium, 654
DG, converting from Naka equilibrium, 655
DG, creating a source, 663
DG, creating chords, 663
DG, creating output files, 668
DG, crop equilibrium, 655
DG, customizing, 669
DG, defining cells, 664
DG, deleting a source, 663
DG, deleting chords, 663
DG, deleting elements, 650
DG, deleting grid points, 662
DG, deleting surfaces, 658
DG, diplaying numbers, 650
DG, directory masks, 669
DG, displaying objects, 651
DG, doubling resolution, 653
DG, eliminating unused numbers, 650
DG, equilibrium, 656
DG, extending chords, 663
DG, file formats, 672
DG, files needed by, 669
DG, grid points, 661
DG, hiding numbers, 650
DG, hiding objects, 651
DG, importing a SONNET grid, 664
DG, importing an equilibrium, 656
DG, installing, 669
DG, joining elements, 650
DG, limiter configurations, 660
DG, linear configurations, 660
DG, menu bar, 647
DG, meshes, 664
DG, message line, 647
DG, modifying a mesh, 664
DG, modifying chords, 663
DG, moving a source, 663
DG, moving an equilibrium, 655
DG, moving elements, 650
DG, moving grid points, 662
DG, moving surfaces, 658
DG, new, 648
DG, obtaining, 19
dg, obtaining, 19
DG, open, 648
DG, output mode, 656
DG, print equilibrium, 656
DG, quit, 649
DG, renumbering elements, 650
DG, repairing cells, 664
DG, repositioning elements, 650
DG, rotating view, 651
DG, running, 209
DG, save, 648
DG, shifting flux values, 655
DG, Snowflake configurations, 661
DG, sources, 663
DG, splitting elements, 649
DG, starting, 646
DG, stretching and shrinking view, 651
DG, surfaces, 656
DG, symmetrise equilibrium, 654
DG, template, 649
DG, toolbar, 647
DG, top-down view, 652
DG, undo, 648
DG, updating, 19
dg, Updating, 19
DG, using tools, 647
DG, variable attributes, 665
DG, variables, 665
DG, variables, default value, 666
DG, variables, export/no export, 666
DG, variables, help text, 666
DG, variables, name, 665, 666
DG, variables, scope, 666
DG, variables, setting, 667
DG, variables, type, 666
DG, work area, 647
DG, X-point, 660
DG, zoom, 651
dg2dg, 653
dg2ef, 653
dg2vr, 654
dg_minuspsi_dg, 654
dg_symm_dg, 654
dgcut, 656
DIMENSIONS_MODULE, 615
display_tallies, 52
DivGeo, compiling, 27, 210
DivGeo, obtaining, 19
DivGeo, Updating, 19
divheat, 97
dn3da, 468
dn3di, 468
dnsepm, 466
dp3da, 468
dp3di, 468
dpmax, 212
dpmin, 212
dpsepm, 466

drifts and currents, boundary conditions, 361
drifts, boundary conditions, 90
drmax, 212, 213
drmin, 212, 213
DTCO, 93
DTCO_SOL, 93
DTEE, 93
DTEE_SOL, 93
DTEI, 93
DTEI_SOL, 93
DTMO, 93
DTMO_SOL, 93
Dynamic mode, 171

e2d, 57, 372, 654
EASY, 620
easybuild, 26
edena, 474
edenm, 474
ef2dg, 654
efgwmake, 45
Eirene, 41, 145
Eirene, compilation, 42
Eirene, compiling, 27
Eirene, obtaining, 19
Eirene, Updating, 19
EIRENE_DEFINES, 34, 632, 640
eirene_eael_she, 473
eirene_eapl_shi, 473
eirene_emel_she, 473
eirene_empl_shi, 473
eirene_epel_she, 473
eirene_eppl_shi, 473
eirene_papl_sna, 473
eirene_pmpl_sna, 473
eirene_pppl_sna, 473
eirenemovies.nc, 135
eirenemovies.nc, edena, 474
eirenemovies.nc, edenm, 474
eirenemovies.nc, pdena, 474
eirenemovies.nc, pdenm, 474
eirenemovies.nc, vxdena, 474
eirenemovies.nc, vxdenm, 474
eirenemovies.nc, vydena, 474
eirenemovies.nc, vydenm, 474
eirenemovies.nc, vzdena, 474
eirenemovies.nc, vzdenm, 474
eirobj, 45
eirobjx, 45
emolrad, 473
emolradsum, 473
ENE_FN, 89
ENEPAR, 89
energy_analysis, 52, 103
energy_analysis.py, 60, 103
energy_analysis_fht, 53
energy_analysis_total, 53
energy_balance, 53
energy_balance_core, 53
energy_balance_core_extended, 53
energy_balance_coupled, 53
energy_balance_extended, 53
eneutrad, 473
eneutradsum, 473
ENI_FN, 89
ENIPAR, 89
env_var_length, 55
env_var_prune, 55
equations, 372
equtrn, 653
error assessment, 178
EXTRA_GRAPHS, 617

F77, 616
fc3dl, 467
fc3dr, 468
fc3dtl, 469
fc3dtr, 469
fch2d, 470
fchsap, 467
fchsapp, 467
fchsip, 467
fchsipp, 467
fchx, 472
fchxap, 466
fchxip, 465
fchxreg, 53, 476
fchy, 472
fchyap, 466
fchyip, 466
fchyreg, 53, 476
fe3dl, 467
fe3dr, 468
fe3dtl, 469
fe3dtr, 469
feesap, 467
feesapp, 467
feesip, 467
feesipp, 467
feexap, 465
feexip, 465
feeyap, 466
feeyip, 466
feisap, 467
feisapp, 467
feisip, 467
feisipp, 467
feixap, 466

feixip, 465
feiyap, 466
feiyip, 466
fetsap, 467
fetsapp, 467
fetsip, 467
fetsipp, 467
fetxap, 466
fetxip, 465
fetyap, 466
fetyip, 466
FFLAGSEXTRA, 34
fhe2d, 470
fhex, 472
fhexreg, 53, 476
fhey, 472
fheyreg, 53, 476
fhi2d, 470
fhix, 472
fhixreg, 53, 476
fhiy, 472
fhiyreg, 53, 476
fhjxreg, 53, 476
fhjyreg, 53, 476
fhmxreg, 53, 476
fhmyreg, 53, 476
fhpxreg, 53, 476
fhpyreg, 53, 476
fhtxreg, 53, 476
fhtyreg, 53, 476
fi3dl, 467
fi3dr, 468
fi3dtl, 469
fi3dtr, 469
first_setup, 57
fixbb, 55
fl3dl, 467
fl3dr, 468
fl3dtl, 469
fl3dtr, 469
FLAG_ALF, 91
FLAG_DNA, 90
FLAG_DPA, 90
FLAG_HCE, 91
FLAG_HCI, 90
FLAG_SIG, 91
FLAG_VLA, 90
FLAG_VSA, 90
fluid_core_fluxes.py, 60
fluid_fluxes, 53
fluid_fluxes.py, 60
flux limits, 362
fluxt, 53
fn3dl, 467
fn3dr, 468
fn3dtl, 469
fn3dtr, 469
fna2d, 470
fnax, 472
fnaxreg, 476
fnay, 472
fnayreg, 476
fnisap, 467
fnisapp, 467
fnisip, 467
fnisipp, 467
fnixap, 465
fnixip, 465
fniyap, 466
fniyip, 466
fo3dl, 467
fo3dr, 468
fo3dtl, 469
fo3dtr, 469
fort.30, 67
fort.44, 165
fort.44_aver, 165
fort.46, 165
fort.46_aver, 165
FreeGS, 654
ft3dl, 467
ft3dr, 468
ft3dtl, 469
ft3dtr, 469

G95, 616
GAMMAE, 89
GAMMAI, 89
Geometry for linear device, 68
get_git_email, 57
get_git_name, 57
get_parm.bpl, 57
get_series, 55
get_series.py, 60
get_solps_fluids, 53
GFORTRAN, 616
GGD_MAJOR_VERSION, 616
GGD_MICRO_VERSION, 616
GGD_MINOR_VERSION, 616
GIT repository log, SOLPS-ITER, 679
GKS_ROOT, 34
GLI_HOME, 34
GNUPLOT_CMD, 49
GNUPLOT_COUNT, 49
GNUPLOT_DATA, 49
GNUPLOT_DELAY, 49
GNUPLOT_EXTRA, 49

GNUPLOT_LINEWIDTH, 49
GNUPLOT_MAX_POINTS, 50
GNUPLOT_MAX_POINTS_FACTOR, 50
GNUPLOT_NOTIMESTAMP, 50
GNUPLOT_POINTSIZE, 50
GNUPLOT_POSTSCRIPT_OPTIONS, 50
GNUPLOT_TERM, 50
GNUPLOT_TMP, 50
GOT_AMNSPROTO, 620
GR_ROOT, 34
grf, 55
gtdr, 55
gtfl, 55

H2VIBR, 42
H5DIR, 34
He_coupled_conservation_core, 52
He_fluid_conservation, 52
He_fluid_fluxes, 52
HITS_PER_CHORD, 589
Hostnames, 23
HPUX, 616
HYDHEL, 42

IBMaix, 616
idl, 194
IDS, Interface Data Structues, 142
IDS_yaml_replace, 57
IMAS, 23, 26, 34, 616
IMAS_MAJOR_VERSION, 616
IMAS_MICRO_VERSION, 616
IMAS_MINOR_VERSION, 616
IMASDIR, 142
import_files_from_convert, 57, 134
import_files_from_convert_wg, 57
incatm, 40
include, 36
Initial set-up, 23
input.dat, 57, 132
input.dat.2, 132
input.dat_to_mesh.extra, 57
inspect_plasmastate, 55, 135
install_dependencies, 55
install_miniforge, 56
Installation on local machines, 29
internal_energy_balance, 53
internal_energy_balance_coupled, 53
ionization_potentials, 39
ippsubmit, 45, 64
ISOEXTRA, 588
ITER_divertor_HTM, 45
itermake, 46
itersubmit, 46, 64
ITM, 616
ITM_ENVIRONMENT_LOADED, 616
itmsubmit, 46, 64
ixref_profile, 97
IY_SOL_DTXX, 93
iyref_profile, 97

JET, 620
jetdsp, 193
jetsubmit, 46, 64
jfrssubmit, 46, 64
joule_heating, 53
jxa, 465
jxi, 465

ke3da, 468
ke3di, 468
kesepm, 466
ki3da, 468
ki3di, 468
kin2d, 469
kind_coeff, 95
kisepm, 466

last10, 57
last_2d, 56
last_2d.py, 60
LBNDUSR, 89
LCBS, 89
LEGACYCOMP, 616
LFEEDBACK, 89
lg, 212
lln, 56
lndr, 56
lns, 210, 661
localsubmit, 46, 64
lxlyplot , 51
lxlyplot2, 51
lxlyplot3, 51
lxlyplot4, 51
lxlyplot5, 51
lxlyplot6, 51
lxlyplot7, 51
lxlyplot8, 51
lxlyplot9, 51
lxyplot , 51
lxyplot2, 51
lxyplot3, 51
lxyplot4, 51
lxyplot5, 51
lxyplot6, 51
lxyplot7, 51
lxyplot8, 51
lxyplot9, 51

Makefile, 27
marconisubmit, 46, 64
Matlab, 61
matlab, 195
mds_id, 57
MDSPLUS, 179, 616
MDSplus SOLPS tree structure, 180
MDSPLUS_DIR, 34
mesh.extra, 57
METHANE, 43
min_ave_max.awk, 60
mn3da, 468
mn3di, 468
mn3dl, 468
mn3dr, 468
mn3dtl, 469
mn3dtr, 469
model, 372
MODIF, 617
modifying the source of B2.5, 37
modules, 36
MOM_FN, 89
MOMPAR, 89
move_equ, 655
move_optional_files, 58
move_save_files, 58
movie frames, b2movies.nc, eirenemovies.nc, 472
movie, python, 198
mpcdfsubmit, 46, 64
MPI_MOD, 619
MPI_VERSION, 619
MSCL_ROOT, 34
mtv_3d, 53
mtv_3d_f, 53
mtv_3d_nf, 53
mtv_3d_t, 53
MySQL, 202

n2d, 58, 655
na, 472
NADD, 588
NAG, 22
NAG calls in B2.5, 641
NAG calls in EIRENE, 642
NAG calls in SOLPS, 641
NAG22, 616
NAGDIR, 34
NAGF95, 616
nareg, 54, 476
nareg.py, 60
nareg_species, 54
nareg_species.py, 60
nastep, 470
NATM, 588
NBC, 81, 588
nc2text, 23, 28, 58, 61
nc_reduce, 28, 58, 61
ncar.cfg, 121, 124, 127–129, 212
NCAR4, 616
NCARG_ROOT, 34
NCDIR, 34
NCHEN, 588
NCUT, 588
ndata, 96
ne, 472
ne2d, 469
ne2reg, 476
ne3da, 468
ne3di, 468
ne3dl, 467
ne3dr, 468
ne3dtl, 469
ne3dtr, 469
nemxap, 466
nemxap_av, 470
nemxap_std, 471
nemxip, 466
nemxip_av, 470
nemxip_std, 471
neoart, 568
neoclassical, 568
nereg, 54, 476
nereg.py, 60
nesepa, 466
nesepa_av, 470
nesepa_std, 471
nesepi, 466
nesepi_av, 470
nesepi_std, 470
nesepm, 466
nesepm_av, 470
nesepm_feedback, 54
nesepm_feedback_by_core, 54
nesepm_feedback_by_pfr, 54
nesepm_feedback_fluid, 54
nesepm_std, 470
netcdf, 22
neutral sources, coupling, 366
neutrals, first-flight model, 366
neutrals, fluid model, 364
neutrals, flux limits, 365
neutrals, transport coefficients, 365
NFL, 588
NGSTAL, 588
NIISO, 89
NION, 588
nireg, 54, 476

NISO, 588
nk2dg, 655
NLIM, 589
NMOL, 589
nn2d, 58, 655
NNISO, 89
NO_21, 620
NO_B2_CHEM_SPUT, 619
NO_CDF, 616
no_div, 96
NO_F01AAF, 620
NO_GETENV, 616
NO_IEEE, 620
NO_JSON, 632
NO_LONG_LINES, 616
NO_MANUAL, 23
NO_NAG, 616
NO_OPENMP, 620
NO_OPT, 620
no_pflux, 96
NO_SAVE, 619
noquote_pwd, 56
NPHID, 589
NPLS, 589
npr, 212, 213
NPRNL, 589
nptseg, 212
NRCX, 589
NREAC, 589
NREC, 589
NREG, 49, 54, 92, 491
NREI, 589
NREL, 589
nrelax, 213
NSD, 589
nsdata, 97
NSPCSRFG, 589
NSRFS, 589
NSTRA, 589
NSTS, 589
NTOR, 589
ntstep, 465
NXD, 589
nxdata, 97
NXISO1, 89
NXISO2, 89
NYD, 589
NYISO1, 89
NYISO2, 89

obtain_tracing_files, 58
Obtaining the code, 19
OPEN_MPI, 632

palettes, 61
PAR_OPT, 620
parallel viscosity, 360
PARM_ALF, 91
PARM_DNA, 91
PARM_DPA, 91
PARM_HCE, 91
PARM_HCI, 91
PARM_SIG, 91
PARM_VLA, 91
PARM_VSA, 91
parse_b2pw, 58
particle_conservation_coupled, 54
pasmin, 212, 213
pdena, 474
pdenm, 474
pefluxa, 473
pefluxm, 473
PERFMON, 34, 620, 642
performance monitoring, 642
pfluxa, 473
pfluxm, 473
PGF90, 616
PHIISO, 89
pipe_b2.log, 58
pipe_run.log, 58
pitagorasubmit, 46, 64
plasmastate, 58, 135
Platforms, 25
plot, 49
plot_bc_conpar13, 60
plot_cps, 50
plot_it, 50
plot_mesh, 50
plot_png, 50
plot_ps, 50
plot_rzpsi.dat, 50
plot_series, 56
plot_set, 50
plot_trc, 56
plot_trc.py, 60
plot_trg, 56
plot_trouble, 50
plotdg, 61
PLOTPOINTS, 50
PLPLOT, 620
pntrat, 213
po, 472
po2d, 469
po3da, 468
po3di, 468
po3dl, 468
po3dr, 468

po3dtl, 469
po3dtr, 469
pomxap, 466
pomxap_av, 470
pomxap_std, 471
pomxip, 466
pomxip_av, 470
pomxip_std, 471
poreg, 476
posepa, 466
posepa_av, 470
posepa_std, 471
posepi, 466
posepi_av, 470
posepi_std, 470
posepm, 466
posepm_av, 470
posepm_std, 471
postprocessing, 61
POT_FN, 89
POTPAR, 89
preserve_scratch_to_work, 58
prinequ, 656
PRINT_PLOT, 50
print_species, 54
print_versions, 58
ps_GV, 58
ps_mac, 58
PSR, 616
pwmxap, 466
pwmxip, 466
python, 195
Python scripts, 60
python, database qyery, 203
python, movie, 198
python, plots to compare profiles, 195

qconvexreg, 476
qconveyreg, 476
qconvixreg, 476
qconviyreg, 476
QSUB.postprocess, 58
QSUB.postprocess_coupled, 58
querying database, 202
quote_pwd, 56

ratadas.filelist, 40
rcxhi, 473
rcxhi2d, 470
rcxhireg, 47, 476
rcxhireg_region, 48
rcxhisum, 473
rcxmoreg, 47, 476
rcxmoreg_region, 48
rcxna, 473
rcxna2d, 470
rcxnareg, 47, 477
rcxnareg_region, 48
rdneureg, 47, 477
READOPT, 617
recover_initial_files, 58, 135
reflection, 54
reflection_energy, 54
refluxa, 473
refluxm, 473
regions, 94, 491
regions, 1-D cases, 492
regions, CDN cases, 496
regions, DDN cases, 498
regions, limiter cases, 493
regions, periodic BC stellarator island cases, 500
regions, SF cases, 502
regions, SN cases, 494
regression_test, 58
regxy_sum_over_species, 58
relax, 213
repart, 212
Requirements, 22
res_av, 58, 177
res_D+C, 46
res_D+C+Ar+Ne+He, 46
res_D+C+He, 46
res_D+O, 46
resall_D, 46, 103
resall_D+_only, 46
resall_D+D, 46
resall_D+T, 46
resall_H, 46
resave_mds, 46
RESCALE_BR, 102
RESCALE_CX, 102
RESCALE_QA, 102
RESCALE_RA, 102
RESCALE_RD, 102
RESCALE_SA, 102
resco, 46
resco_D+C, 46
resco_D+C+He, 47
resco_D+O, 46
resco_reg, 46, 477
resco_reg_region, 46
reshe_reg, 47, 477
reshi_reg, 47, 477
resmo, 47
resmo_D+C, 47
resmo_D+C+He, 47
resmo_D+O, 47

resmo_reg, 47, 477
resmo_reg_region, 47
resmod, 47
resmt_reg, 47, 477
resopt, 47
respo, 47
respo_num, 47
respo_reg, 47, 477
resrest, 46
reviewplus, 194
rfluxa, 473
rfluxm, 473
risepsi, 655
rlcept, 213
rlndr, 56
rps, 58, 655
rqahe, 473
rqahe2d, 470
rqahereg, 47, 477
rqahereg_region, 48
rqahesum, 473
rqbrmreg, 47, 477
rqbrmreg_region, 48
rqrad, 473
rqrad2d, 470
rqradreg, 47, 477
rqradreg.py, 60
rqradreg_region, 48
rrahi, 472
rrahi2d, 470
rrahireg, 47, 477
rrahireg_region, 48
rrahisum, 472
rramoreg, 47, 477
rramoreg_region, 48
rrana, 472
rrana2d, 470
rranareg, 48, 477
rranareg.py, 60
rranareg_region, 49
rsahi, 472
rsahi2d, 470
rsahireg, 48, 477
rsahireg_region, 49
rsahisum, 472
rsamoreg, 48, 477
rsamoreg_region, 49
rsana, 472
rsana2d, 470
rsanareg, 48, 477
rsanareg.py, 60
rsanareg_region, 49
Run set-up, 30
run_av, 58, 177

save_initial_files, 58
save_mds, 46
saveb2, 56
sb2, 37
sbr, 44
scr, 45
sdata, 97
sei, 42
sellan, 213
setup.csh, 30, 32, 34
setup_baserun_B25_links, 58
setup_baserun_eirene_json_links, 59, 152
setup_baserun_eirene_links, 58, 152
sgesubmit, 46, 64
shotnumber.history, 137–139, 141, 142
sign conventions, 372
slib, 44
SLURM.efgw_compile, 59
SLURM.iter_compile, 59
SLURM.postprocess, 59
SLURM.postprocess_coupled, 59
SOLPS review, 353
SOLPS tree structure in MDSplus, 180
SOLPS-GUI, 19
SOLPS-ITER, compiling, 27
SOLPS-ITER, obtaining, 19
SOLPS-ITER, Updating, 19
SOLPS-ITER_CI_env, 59
SOLPS-ITER_CI_test, 59
SOLPS-ITER_debug_CI_test, 59
SOLPS-ITER_MPI_CI_test, 59
SOLPS-ITER_OpenMP_CI_test, 59
solps-iter_update, 59
solps-iter_update_3.0.9_release, 59
solps-iter_update_3.1.0_release, 59
solps-iter_update_3.1.1_release, 59
solps-iter_update_develop, 59
solps-iter_update_wg_release, 59
solps-iter_update_wg_workflow, 59
SOLPS4_3, 619
SOLPS4to5, 59
SOLPS_CPP, 34, 616, 620, 632, 640
SOLPS_FLUIDS, 49, 54, 132
SOLPSTOP, 505
SOLPSTOP file, 64
SOLPSTOP_FORCE, 57, 505
SOLVECO, 93
SOLVEEE, 93
SOLVEEI, 93
SOLVEET, 93
SOLVEMO, 93
SOLVEMT, 93

SOLVEPO, 93
sources, 97
sources_filename, 97
sources_time_mod, 98
sources_time_switch, 98
spbstusubmit, 64
species, 54
SPUTER, 43
sputter, 54
sputter_chemical, 54
sputter_chemical_energy, 54
sputter_energy, 54
sputter_physical, 54
sputter_physical_energy, 54
sputter_RES, 54
sputter_RES_energy, 54
src, 37
src.local, 37
ssb, 37
STIME, 56
stop, 35
str2ogr, 59
stra.dat, 40, 76, 677
STRAHL, 40, 76
subversion repository logs, 679
sum_trg.py, 60
summarize_run, 59
Surface_properties, 40
switch_to_master, 59

tab2, 473
tdata, 96
te, 472
te2d, 469
te3da, 468
te3di, 468
te3dl, 467
te3dr, 468
te3dtl, 469
te3dtr, 469
TEISO, 89
temxap, 466
temxap_av, 470
temxap_std, 471
temxip, 466
temxip_av, 470
temxip_std, 471
tereg, 54, 477
tereg.py, 60
tesepa, 466
tesepa_av, 470
tesepa_std, 471
tesepi, 466
tesepi_av, 470
tesepi_std, 470
tesepm, 466
tesepm_av, 470
tesepm_std, 471
tflux, 54
thermal_evaporation, 54
thermal_evaporation_energy, 54
ti, 472
ti2d, 469
ti3da, 468
ti3di, 468
ti3dl, 468
ti3dr, 468
ti3dtl, 469
ti3dtr, 469
TIISO, 89
time-dependent quantities, b2tallies.nc, 475
time-dependent quantities, b2time.nc, 465
time_dep, 54
TIME_FACTOR_REQUIRED, 93
timesa, 465
timxap, 466
timxap_av, 470
timxap_std, 471
timxip, 466
timxip_av, 470
timxip_std, 471
tireg, 54, 477
tireg.py, 60
tisepa, 466
tisepa_av, 470
tisepa_std, 471
tisepi, 466
tisepi_av, 470
tisepi_std, 470
tisepm, 466
tisepm_av, 470
tisepm_std, 471
tmhacore, 466
tmhadiv, 467
tmhasol, 467
tmne, 466
tmte, 466
tmti, 466
tp3dl, 468
tp3dr, 468
tp3dtl, 469
tpmxap, 466
tpmxip, 466
tpsepa, 466
tpsepi, 466
TRACE, 632
tracing, 135

tran, 678
transfer_solps-iter_runs, 59
transport coefficients, 75, 90, 94
TRANSPORT_FILENAME, 92
TRANSPORT_TIME_MOD, 92
TRANSPORT_TIME_SWITCH, 92
tree structure in MDSplus, 180
triang, 59, 134
triang.conf, 59
TRIM, 43
trim.data, 40
truncate_tracing_files, 60

UAL, 34
Updating the code, 19
USE_CDF_B2STIME, 620
USE_EXT_OPENMP, 619
USE_IMPGYRO, 619
USE_MPI, 616
USE_OPENMP, 619
USE_PXFGETENV, 616
ustcsubmit, 46, 64

v2d, 60, 655
viscous_heating, 54
volreg, 477
vr2dg, 655
vreg, 491
vs3da, 468
vs3di, 468
vssepm, 466
vx3da, 468
vx3di, 468
vxdena, 474
vxdenm, 474
vxsepm, 466
vy3da, 468
vy3di, 468
vydena, 474
vydenm, 474
vysepm, 466
vzdena, 474
vzdenm, 474

wall power loading, 531
wall segments, 582
WANT_PROCESS, 620
wavelength_to_energy, 56
weis.dat, 40, 76, 677
Weisheit, 76
width, 212, 213
WINDOWS, 616
Windows, 26
WOS, 620

xdata, 97
xlyplot , 51
xlyplot2, 51
xlyplot3, 51
xlyplot4, 51
xlyplot5, 51
xlyplot6, 51
xlyplot7, 51
xlyplot8, 51
xlyplot9, 51
xreg, 491
xyplot , 50
xyplot2, 50
xyplot3, 50
xyplot4, 50
xyplot5, 50
xyplot6, 50
xyplot7, 50
xyplot8, 50
xyplot9, 50

yreg, 491

zeffreg, 54
zeffreg.py, 60
Zhdanov, 369, 390