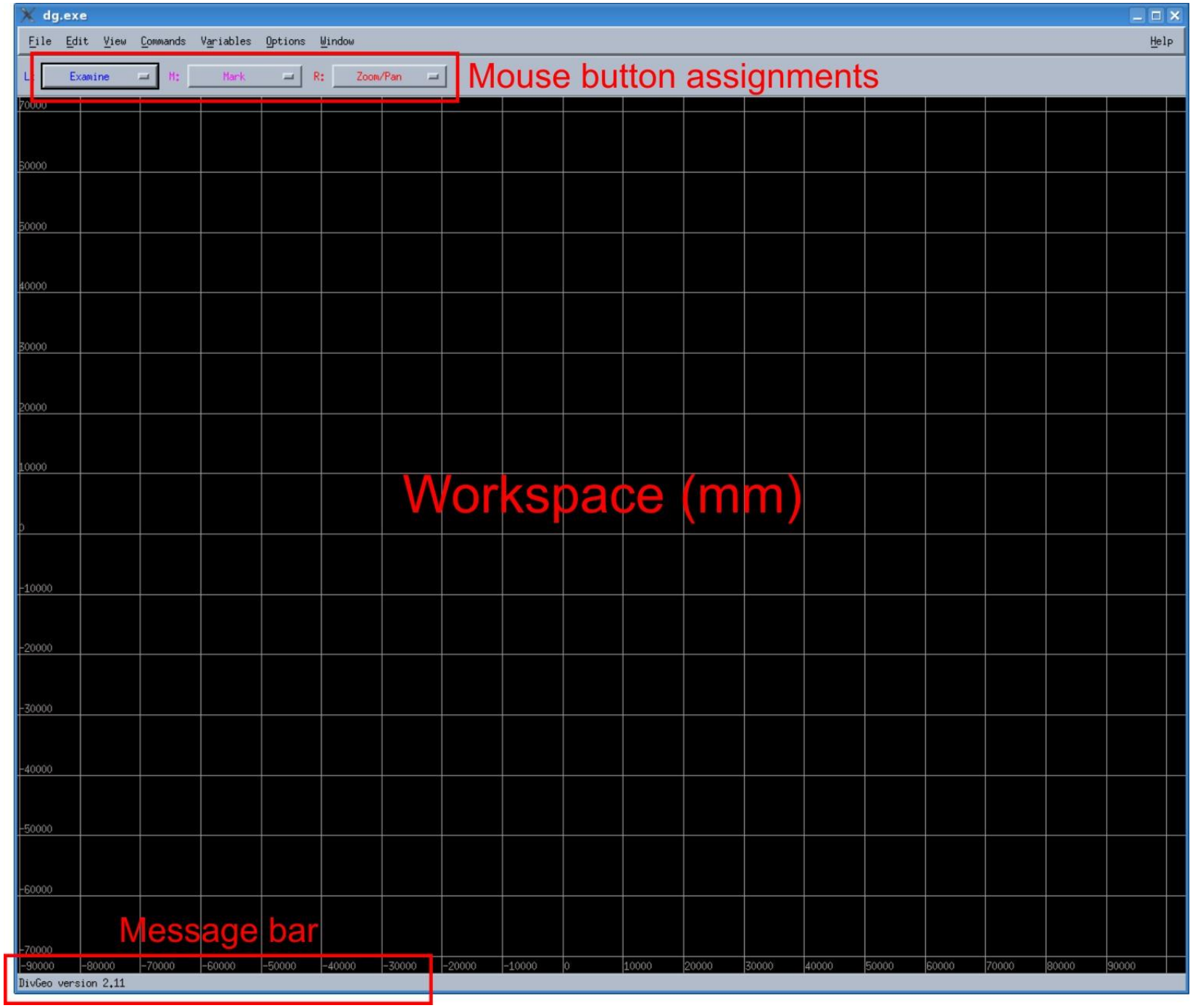
Instruction	Description

Basic setup of the run directory.
 	
`$ stop`
`$ cd runs`
`$ mkdir <your test case directory name>`
`$ cd < your test case directory name>`
`$ mkdir baserun`
`$ cd baserun`
	This might be a bit different for the usual setup used by some current SOLPS users, and is part of the general reorganization of the code with the SOLPS-ITER release. The idea is to try and have everything located in one working directory called `baserun`.

Copy the EFIT equilibrium file `g990429019.00940` into `baserun`.	This is a C-Mod lower single-null equilibrium for shot 990429029 at 940 ms into the discharge.
$ setenv DEVICE cmod	Tell the SOLPS environment that you are running a C-Mod case
Convert the EFIT equilibrium format to the DG equilibrium format. $ e2d `g990429019.00940 g990429019.00940.equ`	
Increase the resolution of the equilibrium data, which allows improved (smoother) contouring in the grid generator. $ d2d `g990429019.00940.equ g990429019.00940.x2.equ`	Using a higher resolution equilibrium can avoid some problems when generating the fluid grid, but it also causes everything to run more slowly. It is sometimes necessary to run `d2d` more than once – the convention is to use `.x4.`, `.x8.`, etc. file names.
Copy the wall geometry file `wall_geometry_990429019.ogr` into `baserun`.	A list of R,Z points in machine coordinates that describes the layout of the plasma facing components. The points are in **mm** and must form a closed polygon, i.e. the first and last points must be the same.

Start the DG, or DivGeo, SOLPS setup tool; see the graphic on the next page. `$ dg &`	General comments: • CTRL+P fits the workspace to the geometry elements, and is one of your best friends since it is an easy way to un-zoom and re-center the view. • The mouse button assignments are very important, but if you change them often then you can make a lot of mistakes. It is suggested that the left mouse button always be set to Zoom/Pan and the right button to Mark (which lets you select geometry elements), which are two of the most commonly used operations. This leaves the middle button free for other functions, of which there are many. • CTRL+U un-selects all marked objects. • SHIFT + Mark selects all objects within a particular region, and is very useful. • Helpful messages appear at the bottom of the window. • DG is a very powerful tool, but it takes some time and effort to learn. • DG can crash, especially when run remotely over a slow connection, so save often. • The manual is here.


dg.exe
File Edit View Commands Variables Options Window Help
Examine
M: Mark
R: Zoom/Pan
Mouse button assignments
70000
60000
50000
40000
30000
20000
10000
0
-10000
-20000
-30000
-40000
-50000
-60000
-70000
Workspace (mm)
Message bar
-90000 -80000 -70000 -60000 -50000 -40000 -30000 -20000 -10000 0 10000 20000 30000 40000 50000 60000 70000 80000 90000
DivGeo version 2.11

Import the vessel wall description

The first step when setting up a new case is to import the R,Z data that describes the wall geometry.

`File > Import > Template`

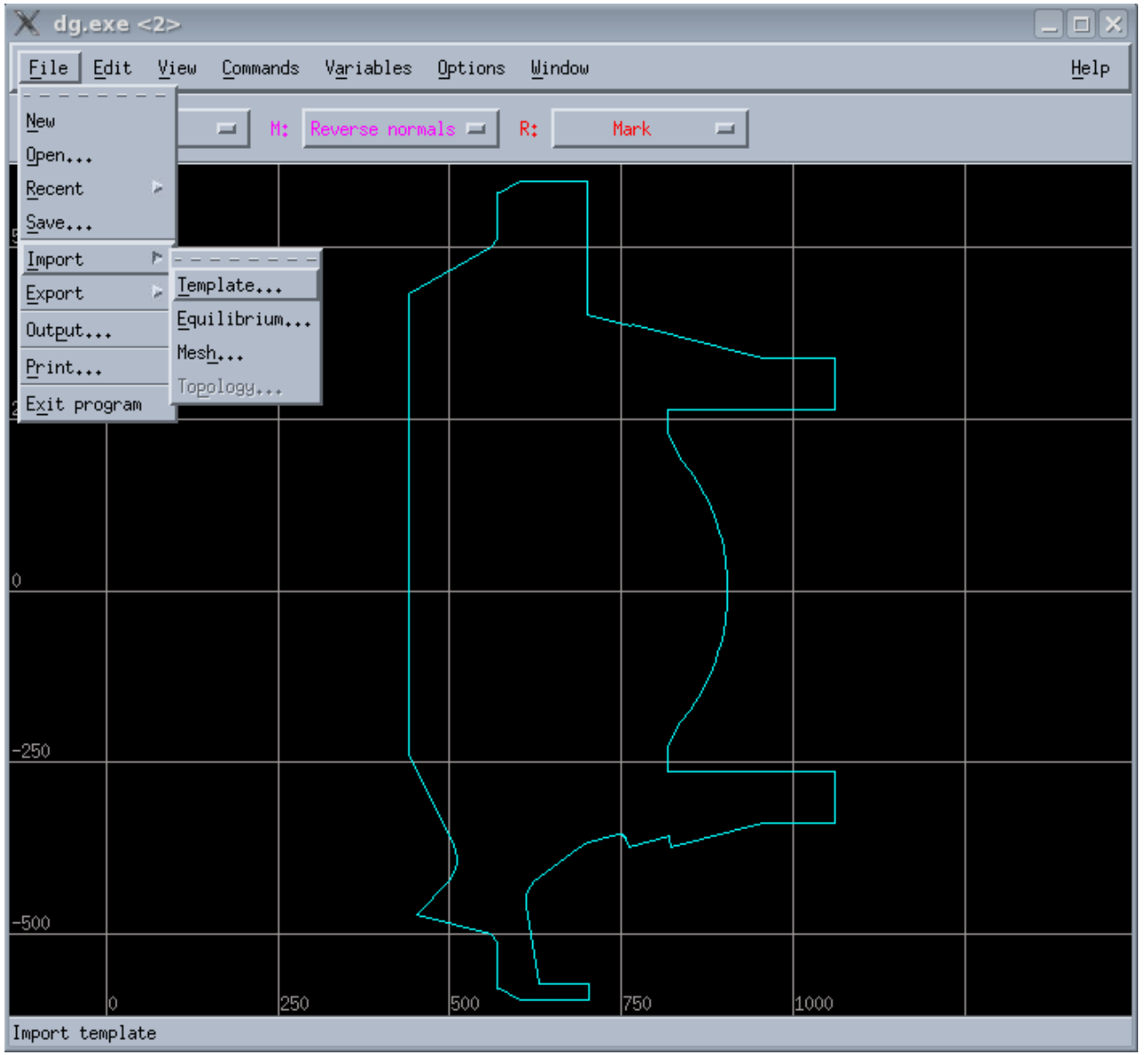
and load `wall_geometry_990429019.ogr` (that was copied into the `baserun` directory), which should appear in the dialogue box.

Press CTRL+P to fit the wall data to the workspace.

If you cannot see the wall, then

`View > Display`

and make sure that the Template radio button is pressed



Import the magnetic equilibrium data

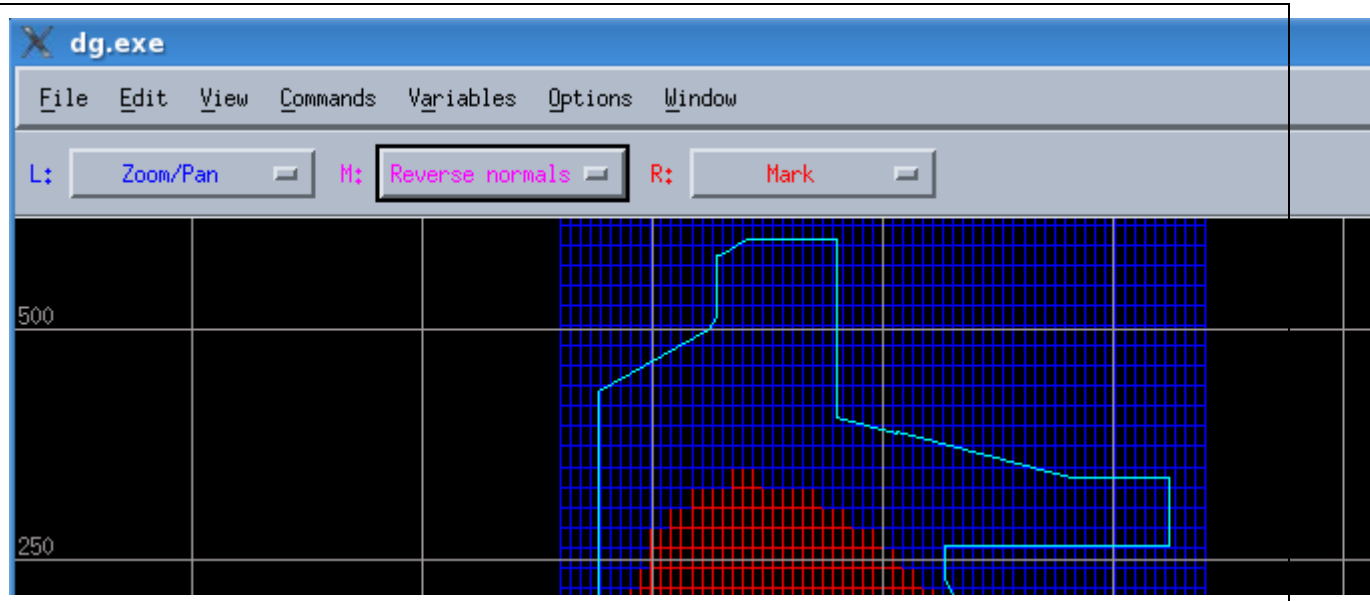
`File > Import > Equilibrium`

and load `g990429019.00940.x2.equ`, which was copied into `baserun`.

If you cannot see the equilibrium displayed as blue (SOL) and red (core and PFR) rectangles after it is loaded, then

`View > Display`

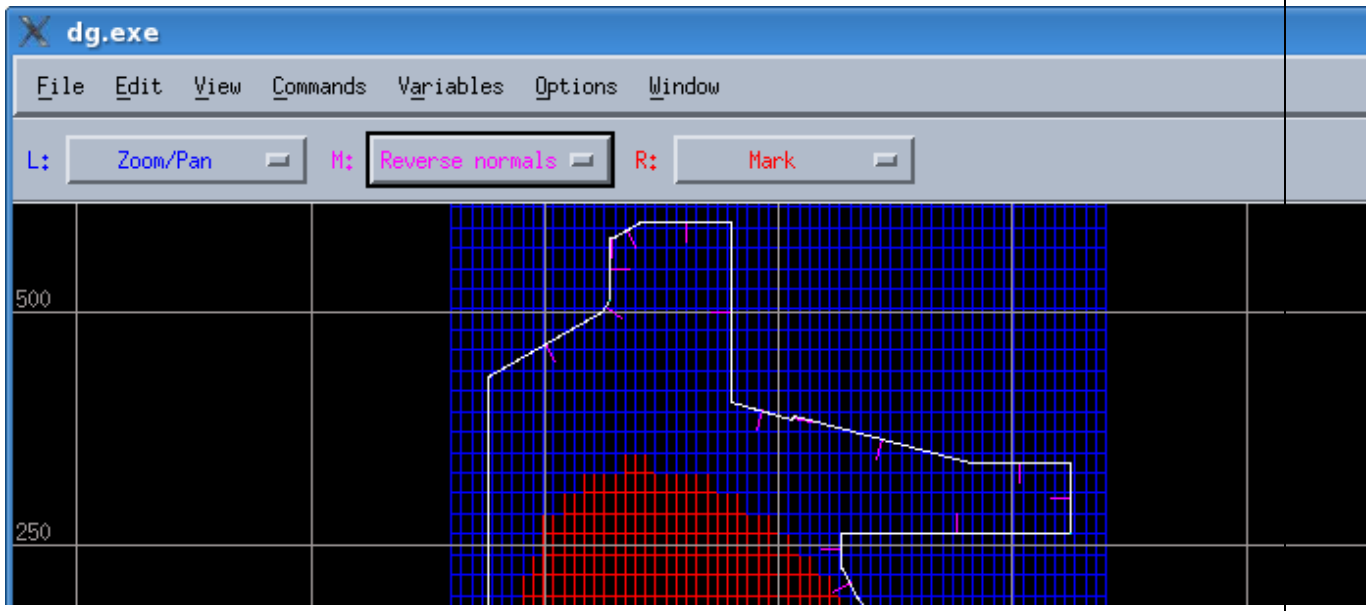
and make sure that the Equilibrium button is pressed.



Convert the wall segments to geometry elements

`Command > Convert > Template to elements`

The short pink lines indicate the surface normals.

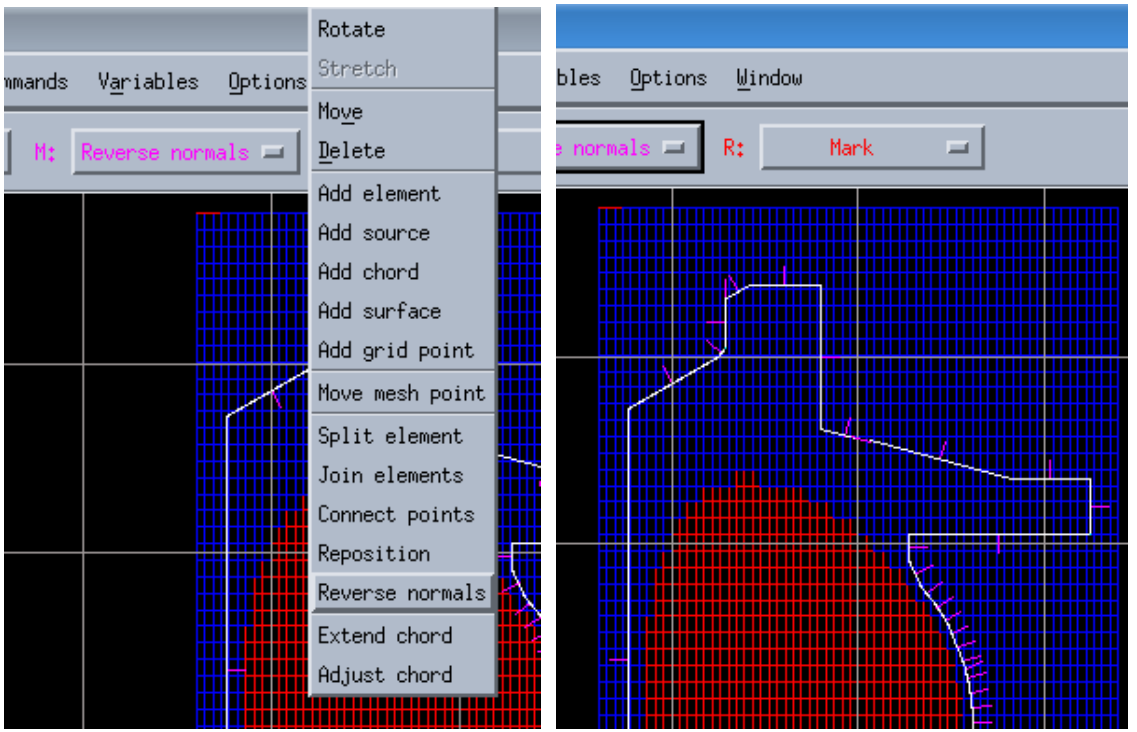


Make sure that all surface normals are pointing away from the plasma

This convention is required by later steps in the grid triangulation and input build-up process.

Set the middle mouse button to “Reverse normals”.

Then click SHIFT+Reverse normals (middle button) somewhere on the vessel wall and all of the normals should flip.



Setting the magnetic topology

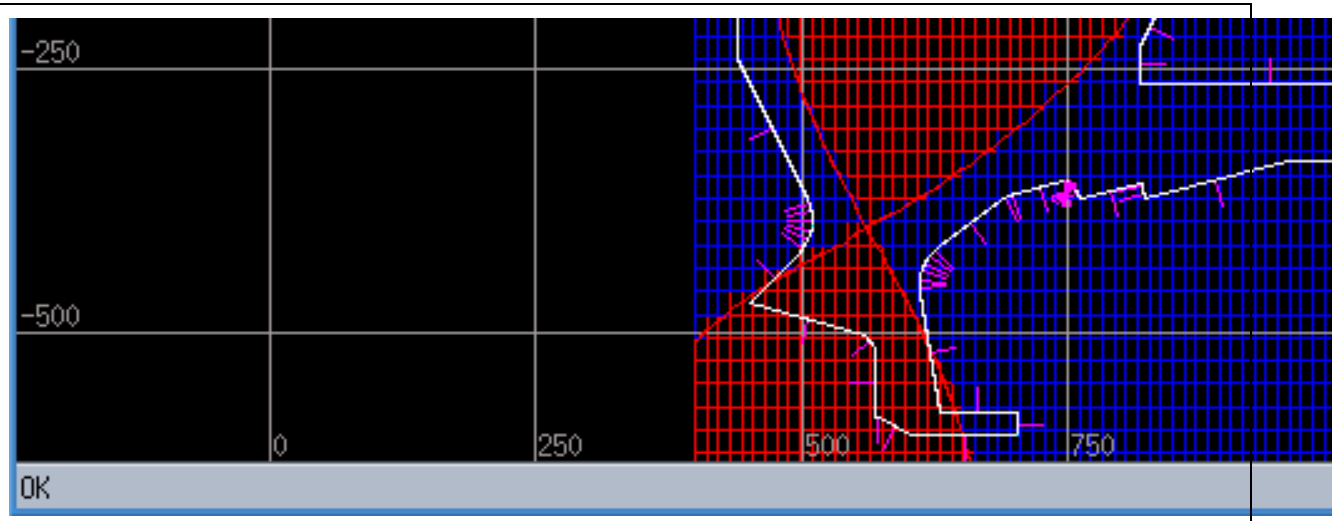
Help DG and tell it which magnetic topology you want the modelling grid to have.

`File > Import > Topology`

and choose one of:

SN	lower single-null
SN-up	upper single-null
DDN	disconnected double-null, lower primary x-point
CDN	connected double-null
DDN-up	disconnected double-null, upper primary x-point

Select SN for this C-Mod test case. The separatrix will be marked in by a red line if the topology is applied correctly.



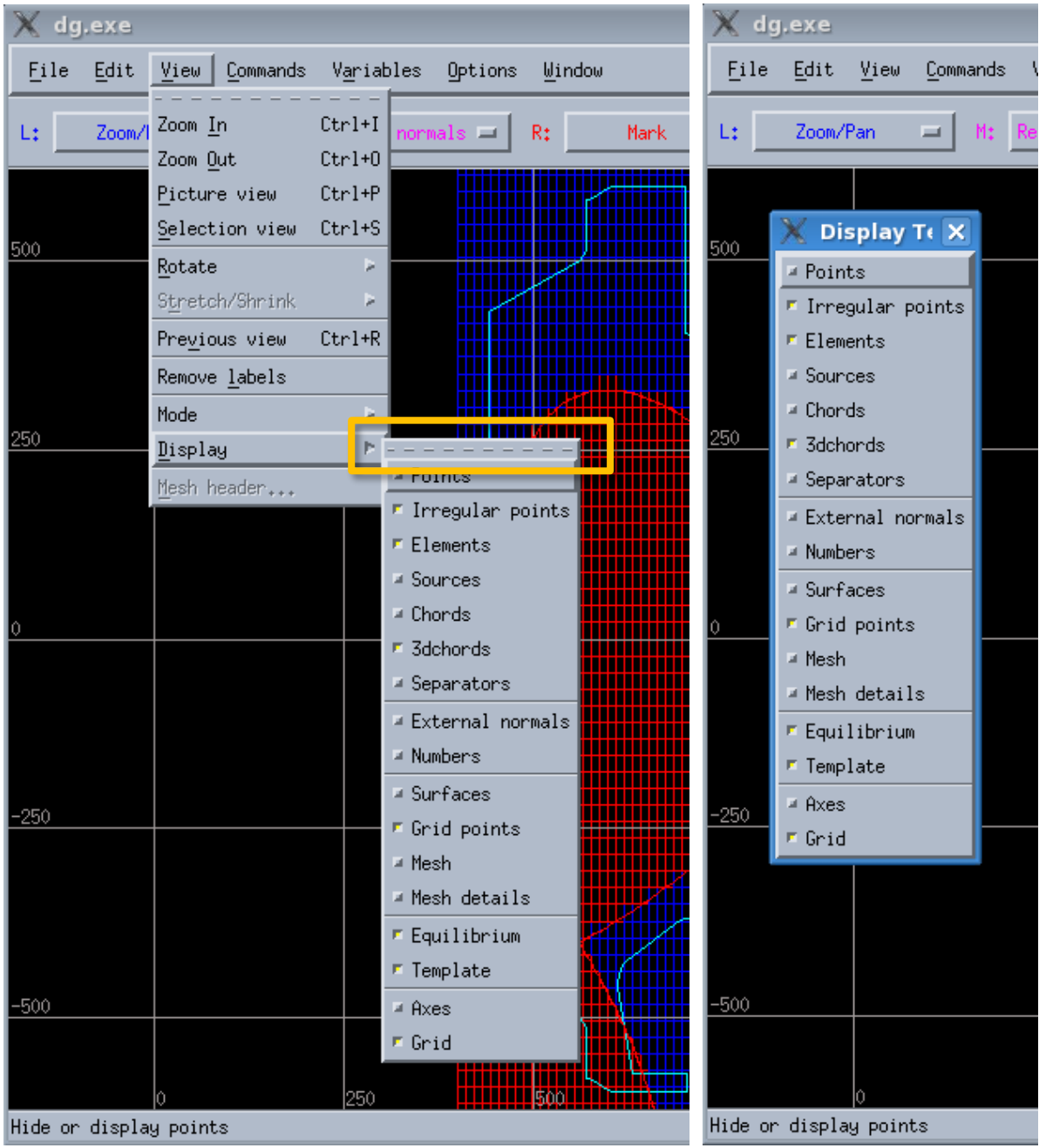
“Floating” the Display dialogue box

The workspace can become very “busy” at times, and so it is useful to “float” the Display list so that it is easy to access.

`View > Display`

and then left-click on the dashed line at the top of the list; see the yellow box on the image to the right. Drag the Display dialogue box to a convenient location on your desktop. It does not have to stay inside your DG workspace.

You can float any menu list that has a dashed line at the top.



Defining the extent of the targets

The target definitions need to satisfy the following rules:

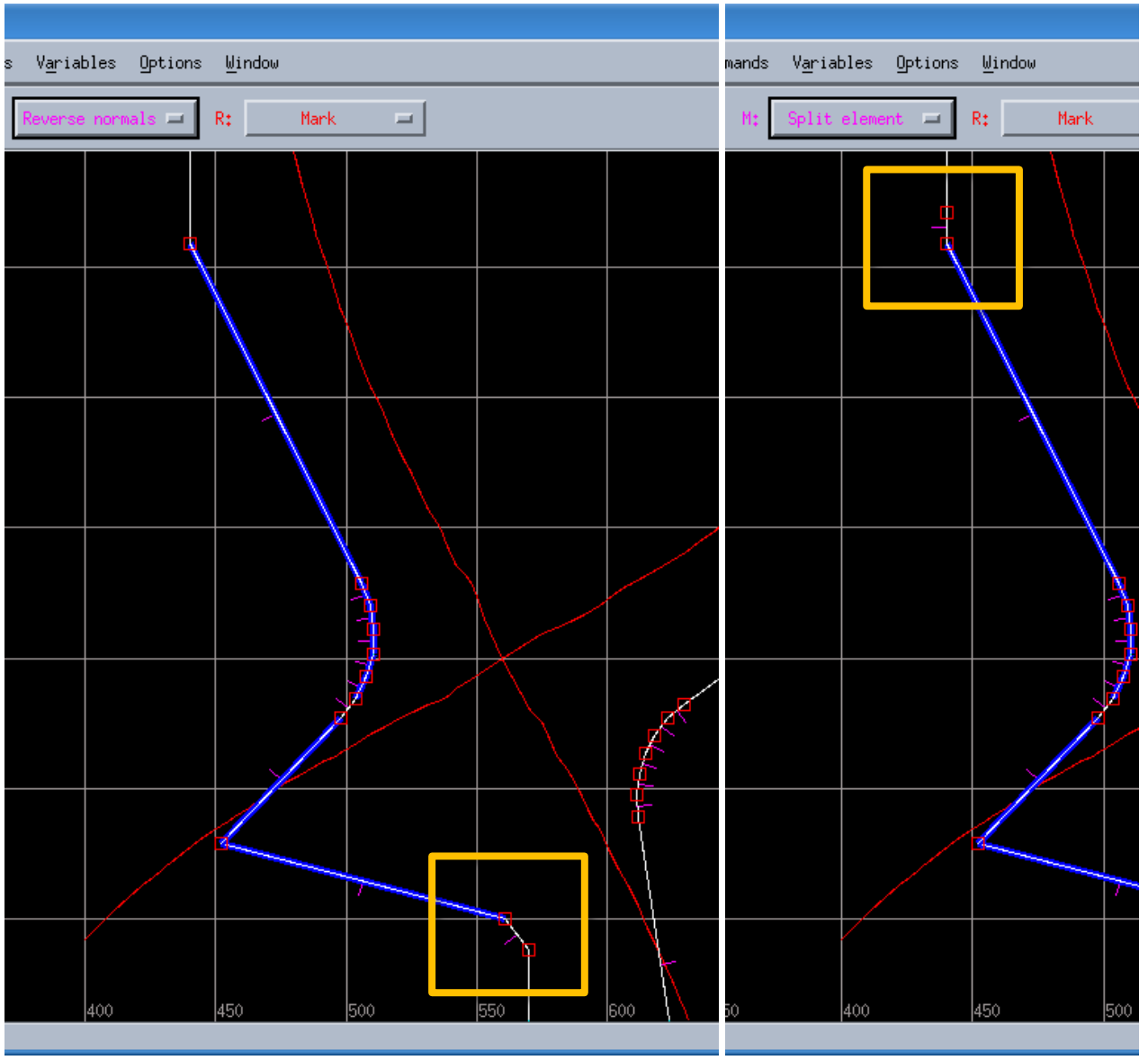
1) Each target segment needs to have a short wall element at each end, which will not be part of the target, but will be used to separate the target from the main wall in a subsequent setup step.
2) The targets must be closed polygons.
3) The surfaces normals of the closed polygon must all point inward.
4) The plasma-wetted part of the target must consist of at least two (2) wall elements.

In the C-Mod example, rule (1) is only satisfied at one end of the inner target; see the yellow box in the first image on the right (the wall segments that are going to be associated with the inner target are highlighted in blue). Therefore, a short segment needs to be added:

Change the assignment of the middle mouse button to "Split element", and then click on the vertical segment just above the target, which adds a point on the wall and creates a new segment.

Avoid making very short segments, which can cause problems for the triangle grid generator.

The result is show in the second image, and rule (1) is now satisfied.



Defining the targets

For creating the closed polygon required by (2), it is necessary to add a point behind the target and then connect it to the existing points at the ends of the target.

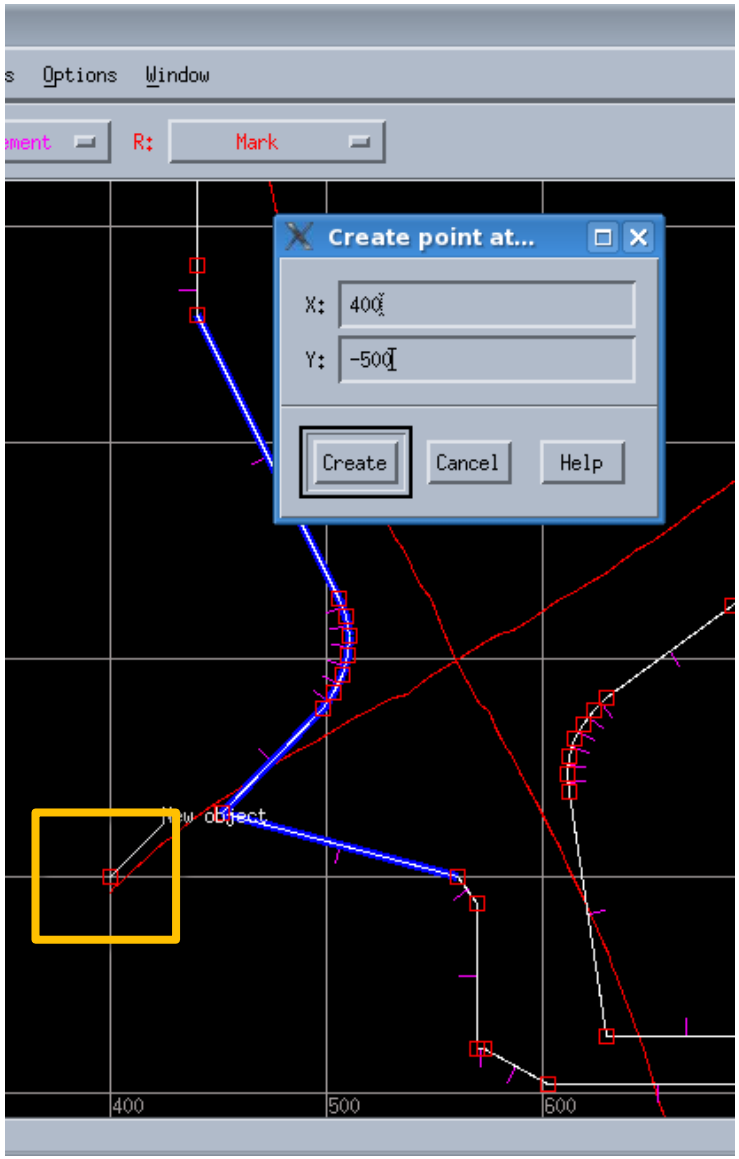
`Edit > Create > Point…`

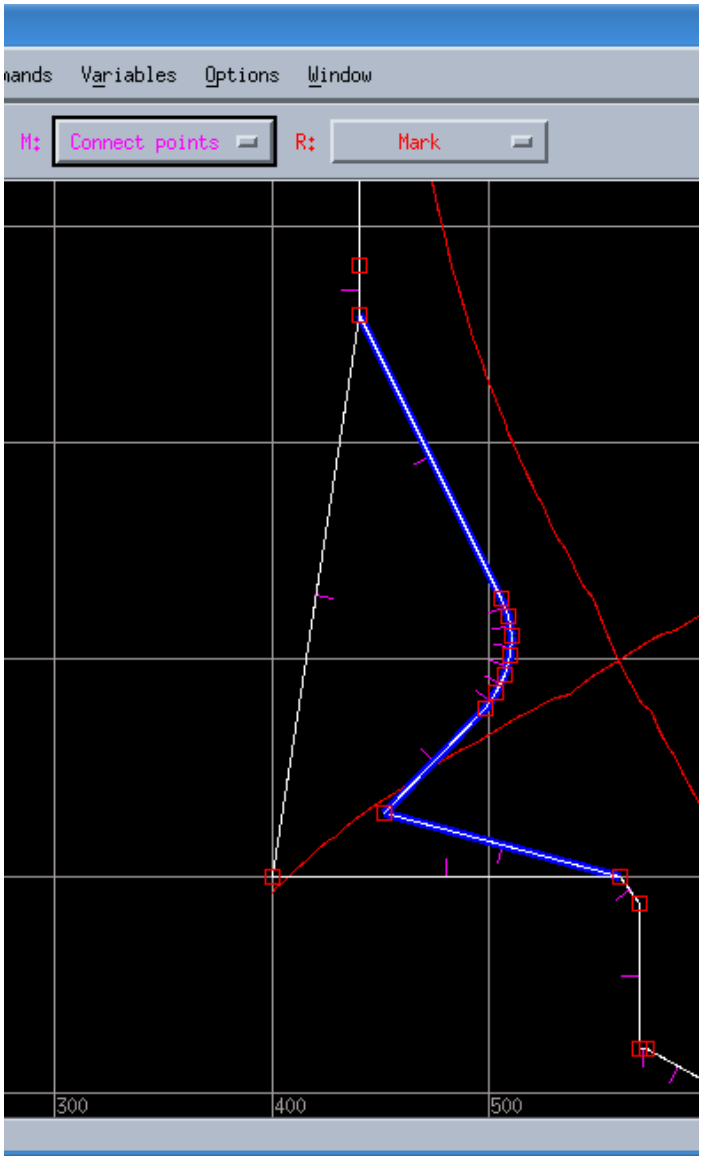
and add a point at (400,-500).

Then, change the middle mouse button to “Connect points”, and middle-click on the new point, and drag the cursor to one end of the target and release the mouse button.

Do the same for the point at the other end of the target.

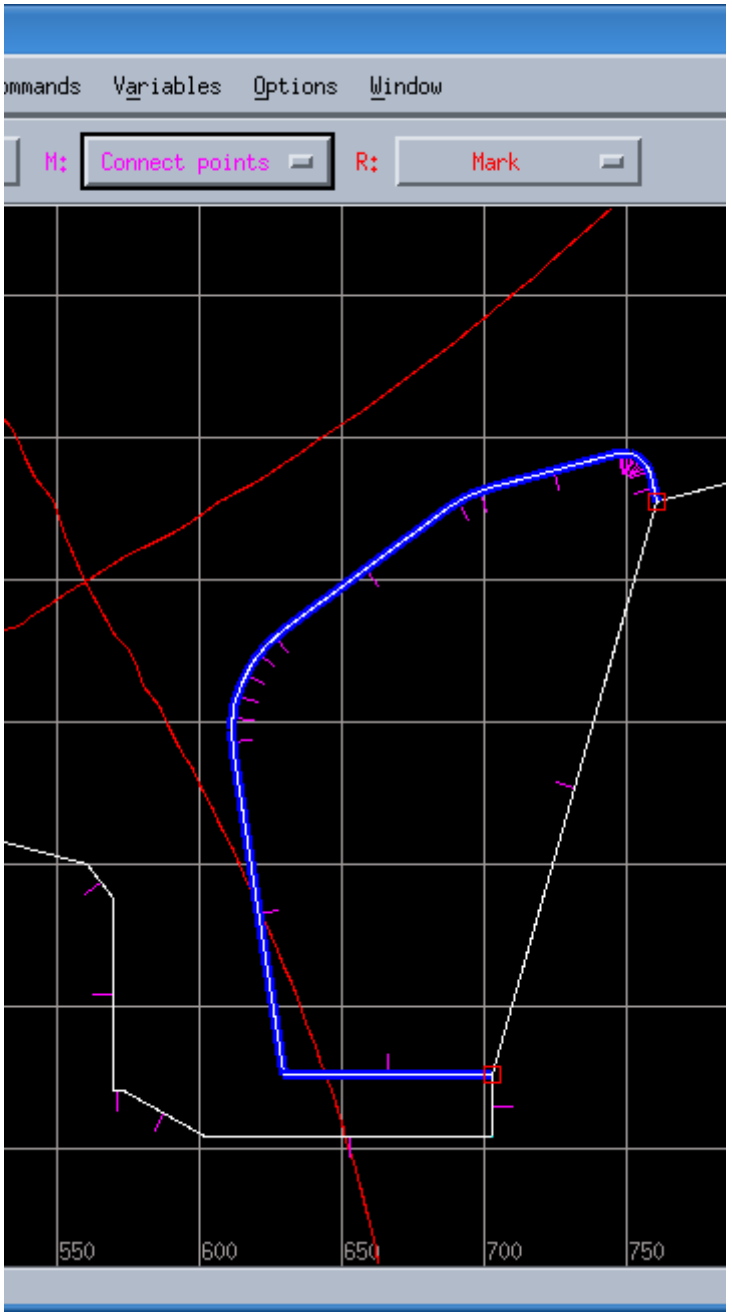
For rule (3), check that the surface normals of the closed polygon are all facing inward, and if not, assign the middle button to “Reverse normals and flip them.





Defining the targets, cont.

The same thing, but for the outer target. Rules (1) and (2) are easy to satisfy in this case, since it was not necessary to split a wall segment or to add an extra point.



Setting the “Structure” variable for “Structure”

This is the primary definition for the vessel wall.

`Variables > Structure`

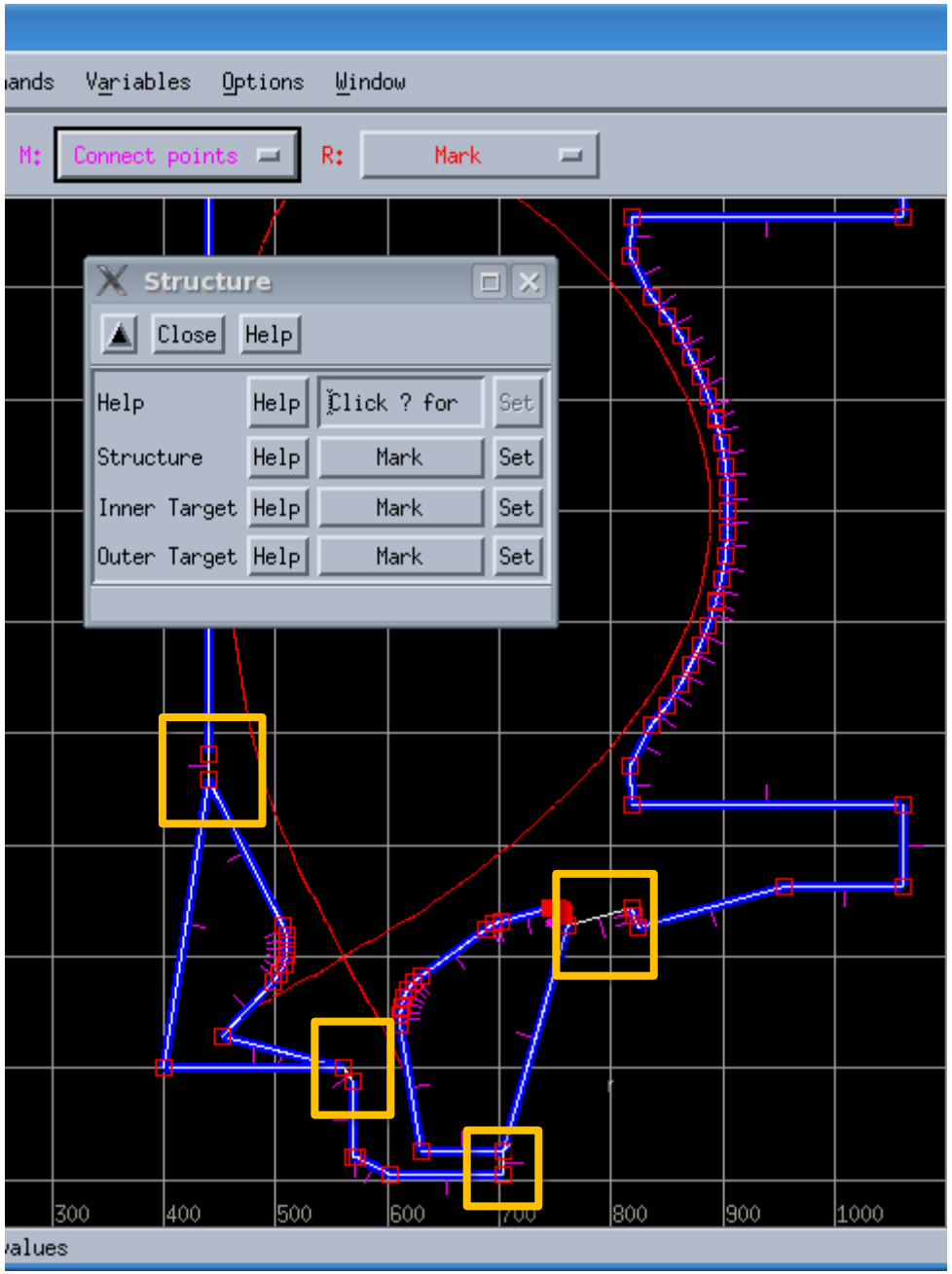
(If “Structure” does not appear in the list, then try `Variables > Add > Structure`)

Use the right mouse button (assigned to Mark) to select all segments except the short segments that are just outside the targets. Using SHIFT+right click will help a lot. Right clicking on a selected segment will un-select it.

When the highlighting is complete, left-click on “Set” in the “Structure” dialogue box, at the end of the line marked “Structure”.

CTRL+U to unmark everything.

To check the assignment, left-click on the “Mark” button in the “Structure” dialogue box, and only the desired segments should be highlighted.



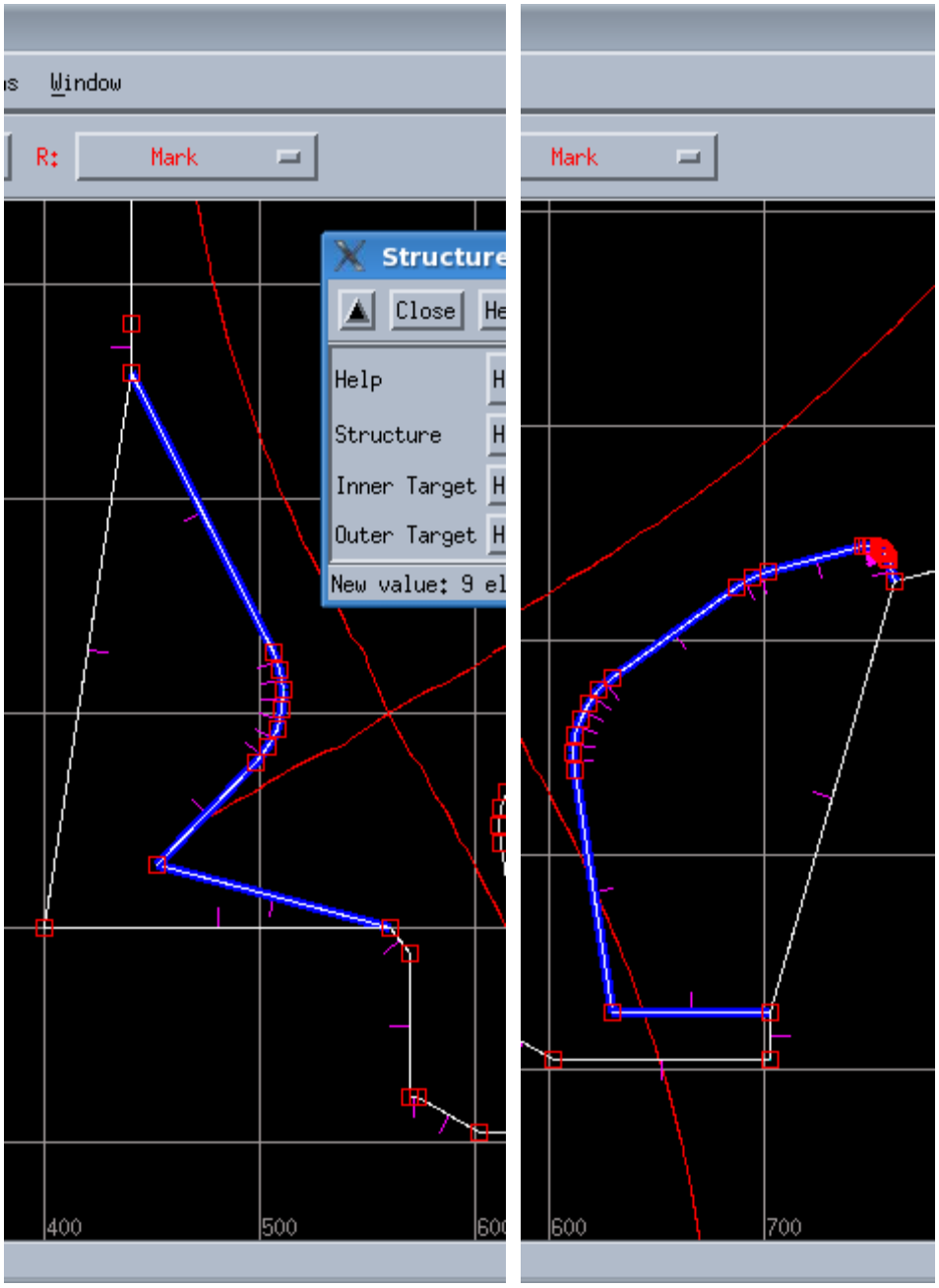
Setting the "Structure" variable for the targets

CTRL+U to unmark everything.

Mark all of the segments for the inner target and then click on "Set". Do not include the segments that are behind the target. (Good to click "Mark" after pressing "Set" to confirm that the assignment was done properly.)

CTRL+U

Assign the outer target.

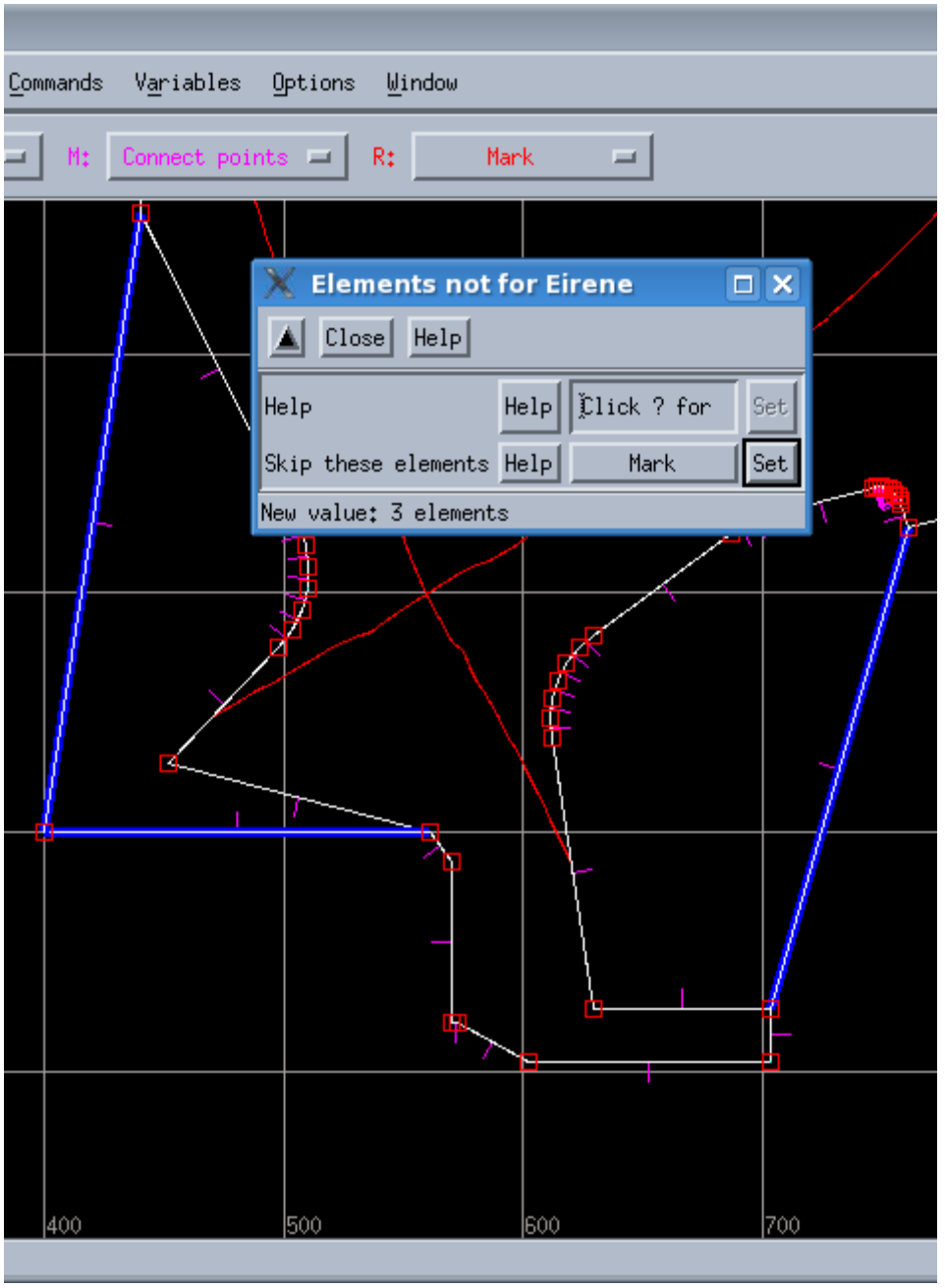


Setting elements that are to be ignored by EIRENE

`Variables > Add > Elements not for Eirene`

CTRL+U

Mark the elements behind the targets and click “Set”.



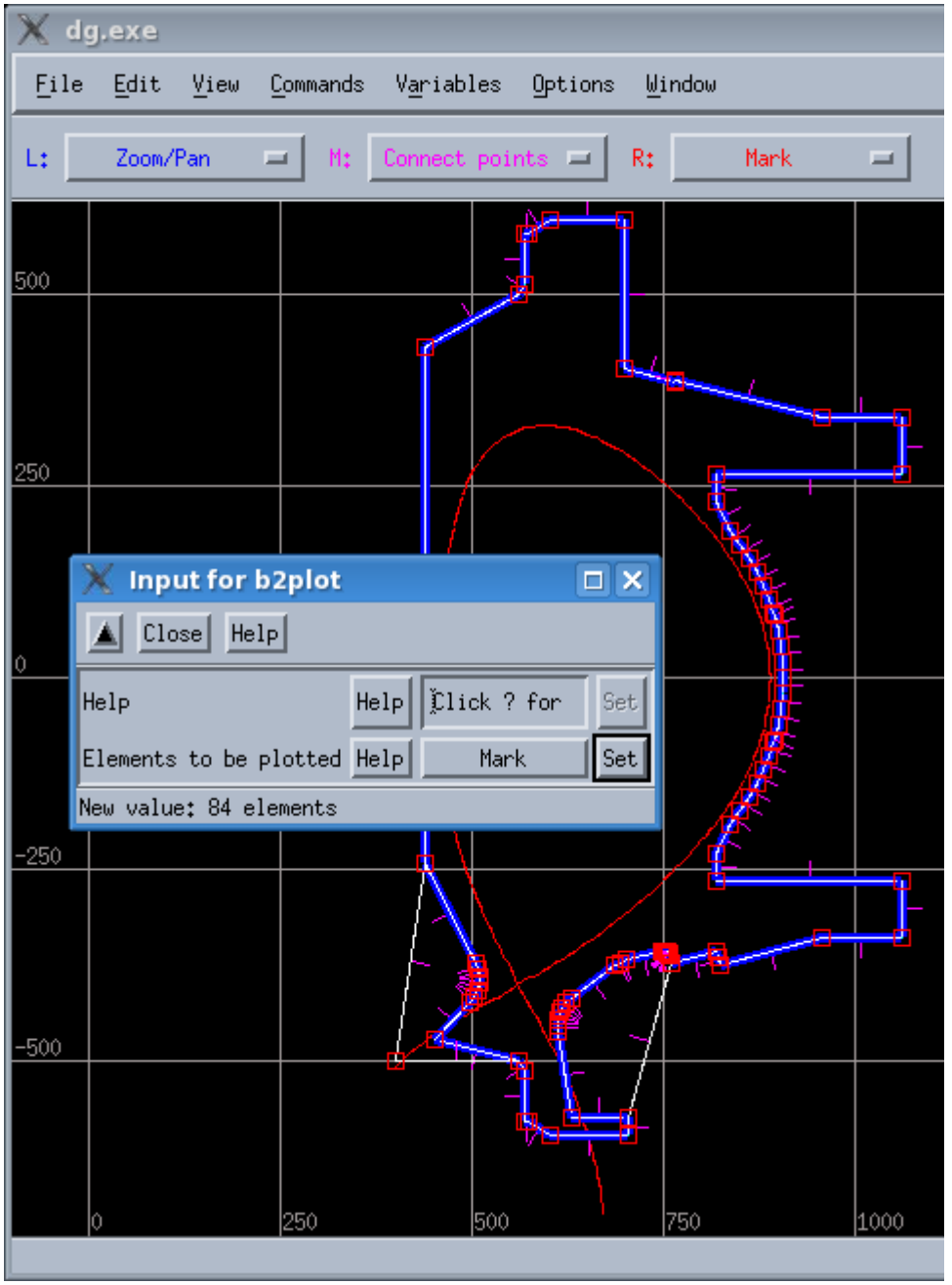
Setting the elements that are used by B2plot

Defining the wall surfaces that will be written to the mesh.extra file that defines the wall specification in B2plot.

`Variables > Add > Input to b2plot`

CTRL+U

Mark everything except the segments behind the targets.



Setting the target specifications

`Variables > Target specification > #1`

#1 is the inner target for this case. The target index as a function of magnetic topology is as follows:

Topology	Target #1	Target #2	Target #3	Target #4
SN-down	HFS	LFS		
SN-up	LFS	HFS		
DN	HFS-down	HFS-up	LFS-up	LFS-down

The target numbering must follow the order of the B2.5 grid indices, which increase as one goes around the core plasma in a clockwise direction.

CTRL+U

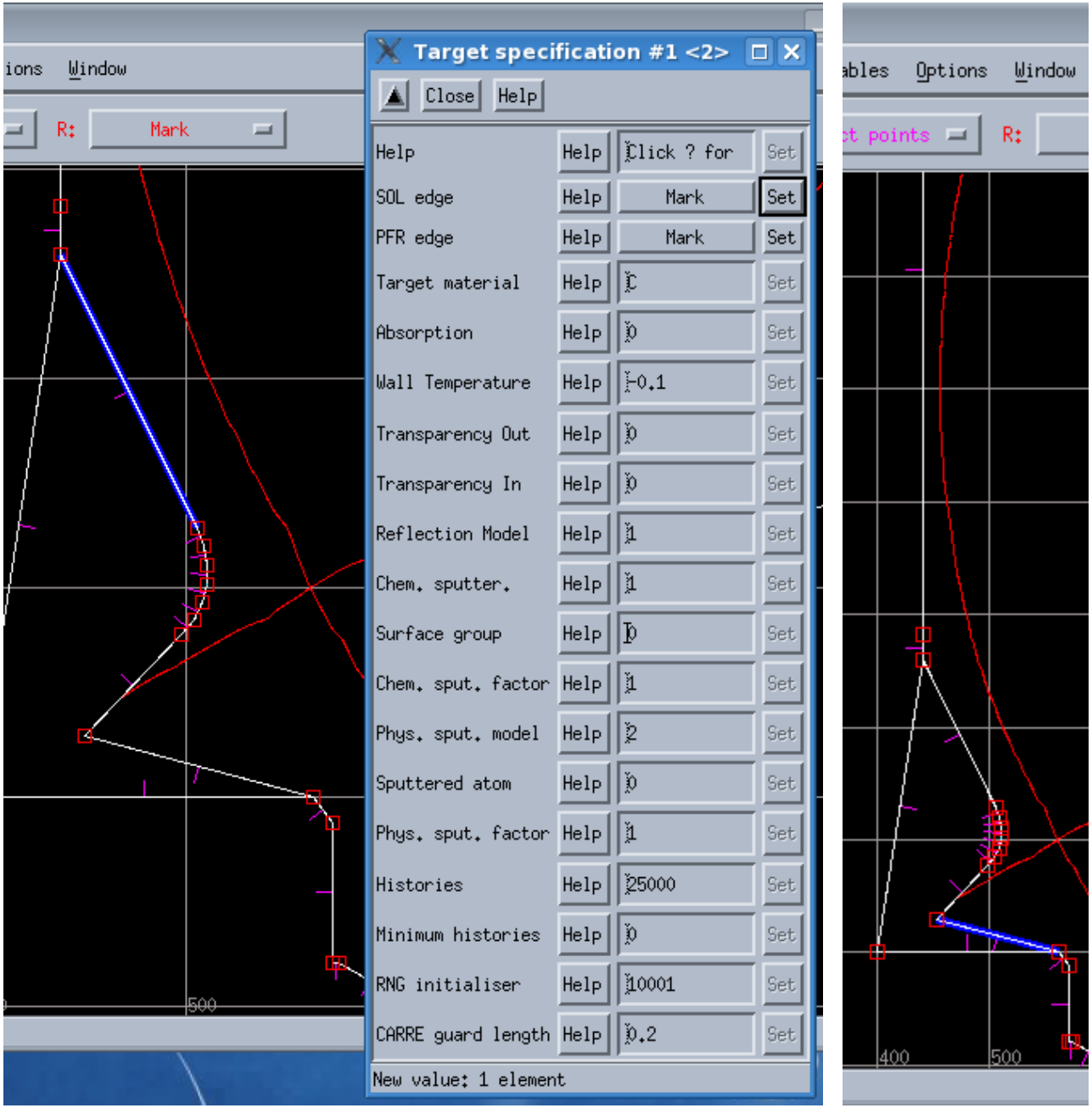
Mark the upper-most segment on the inner target, which is in the Scrape-Off Layer, and click “Set” for “SOL edge”; as shown on right.

CTRL+U (or right-click on the SOL edge segment to unmark, if tired of pressing CTRL+U all the time…)

Mark the segment that defines the lower extent of the target (show in right-most figure), and click “Set” for “PFR edge”.

Change ”Target material” to Mo for this case, to match the C-Mod target material, and click “Set”.

Assign “Chem. sput.” (chemical sputtering) to 0, and click “Set”.



Setting the target specifications, cont

Repeat for the outer target:

`Variables > Target specification > #2`

The outer target is less straightforward and then inner target for this case. The “edge” settings for the targets have to intersect the outer radial boundary of the grid, but it’s not obvious where the SOL radial boundary edge will be at this stage.

Trick!

Set the middle button to “Add surface”.

The outer radial extent of the SOL is set by the first intersection between a flux surface and the main chamber wall (“tangency point”), which will be near the inner midplane for this case.

Hold the middle button down at the inner midplane and the flux coutour will appear, and you can see where it maps to at the outer target; see the figure on the right. Set “SOL edge” to this segment. (Note: You can accidentally add a surface in the core if you’re not careful with your clicking. Press CTRL+Z for undo if this happens.)

For “PFR edge”, the segment indicated in the right-most figure is not the same as the lower-, outer-most segment included when the Structure variable was set for the outer target – this is OK. The “PFR edge” is only set based on where the radial boundary of the grid will be, as determined by intersection with the divertor knee.

Assign the Mo as the material and turn off chemical sputtering.

Poloidal grid points

The poloidal distribution of cells on the Carre grid are set by the "poloidal grid points" in DG. These are defined separately for the divertor legs and the SOL.

`Edit > Create > Grid points`

Set "Zone" to "Inner divertor" and "Cells" to 18.

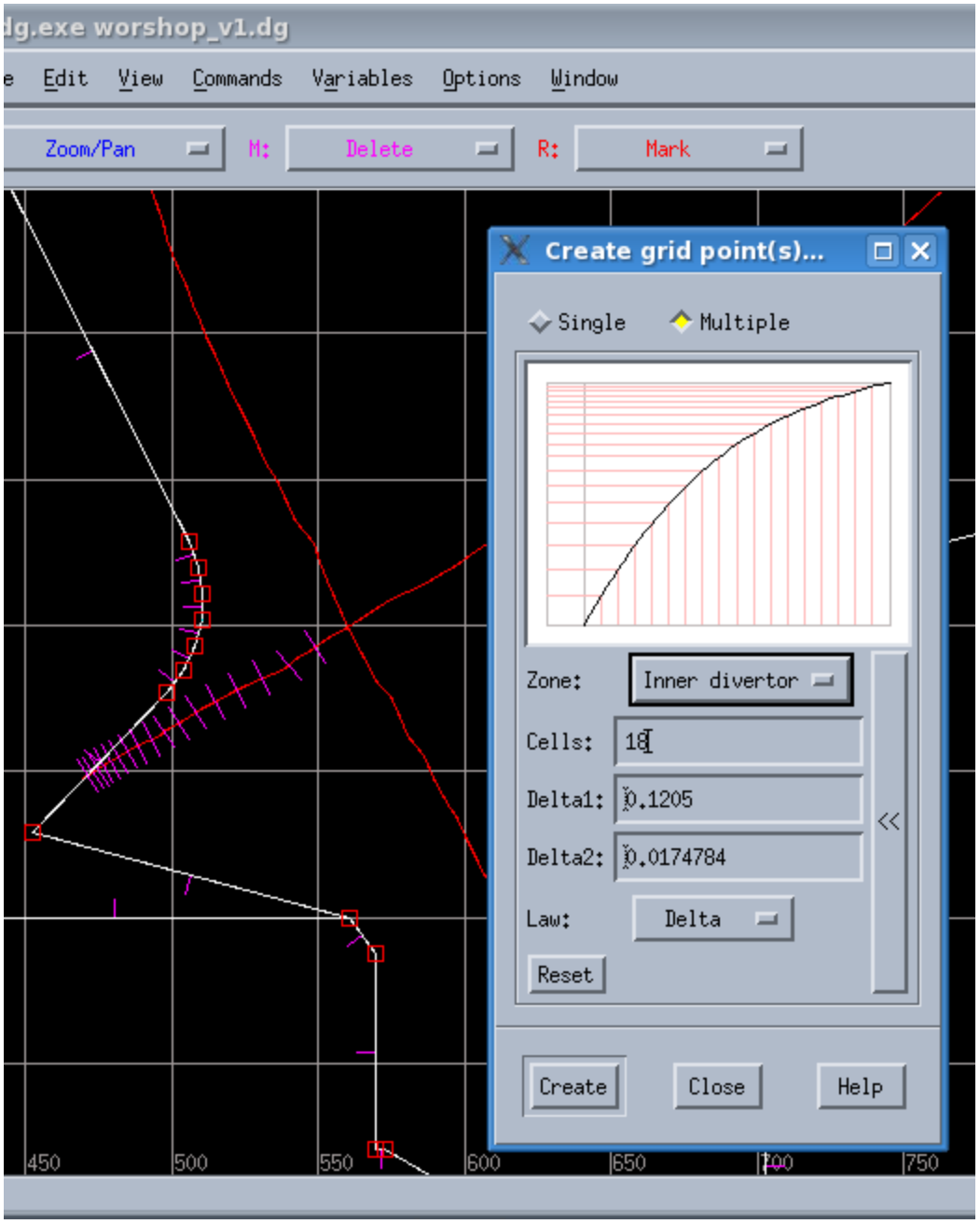
The distribution of the cells can be adjusted by left-clicking and draggin the black line on the plot. Click "Create" to update the workspace.

Generally speaking, good practice is to have higher spatial resolution near the targets, where gradients may be large.

Repeat for the outer divertor.

Set 48 points in the SOL, with a roughly uniform distribution of points (a straight line on the grid point distribution plot).

The spacing of the grid points around the x-point should be symmetric.



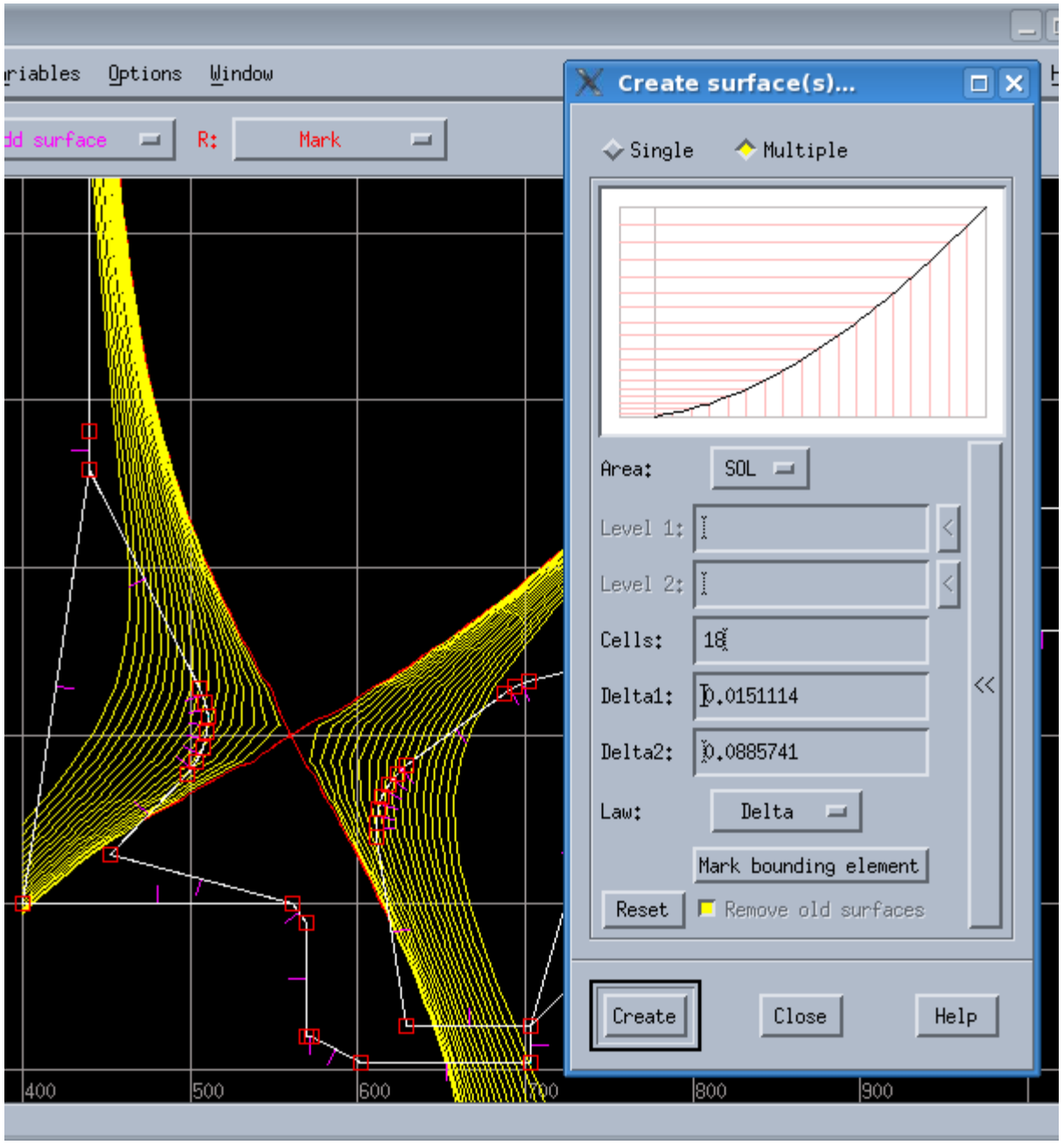
Radial surfaces

The radial surfaces in DG define the boundaries between rings on the Carre grid.

`Edit > Create > Surfaces…`

Set 18 surfaces in the SOL.

Adjust the radial distribution to give higher spatial resolution near the separatrix.



Radial surfaces, cont.

There is an issue in the PFR for this particular case: the flux surface which is tangent to the “knee” will miss the bottom of the outer target and cross the entrance to the “plenum” in the sub-divertor.

Trick!

Create a virtual structure in the PFR volume, which will cause DG to reduce the radial extent of the grid.

Set the middle mouse button to “Add surface” and identify the flux surface which intersect the bottom of the outer target.

For this C-Mod case, add three points: (550,-450), (550,-460), and (525,-460). (`Edit > Create > Point…`)

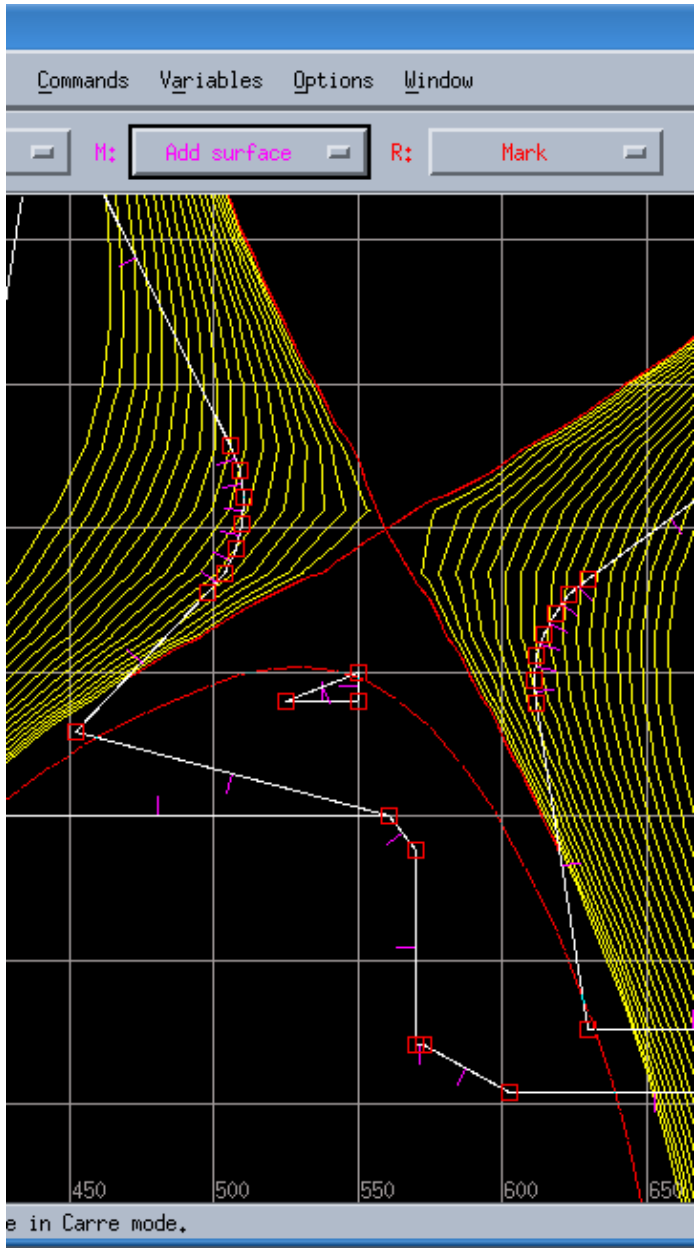
Create surface elements between the points using “Connect points” the middle mouse button.

Make sure the surface normals point inward, using “Reverse normals” with the middle button, if required.

Add the new surfaces to the “Elements not for Eirene” variable – there should now be 6 segments in the list (the PFR triangle and the three surfaces that are behind the targets).

Add the new surfaces to the “Structure”variable – there should now be 84 segments in the list.

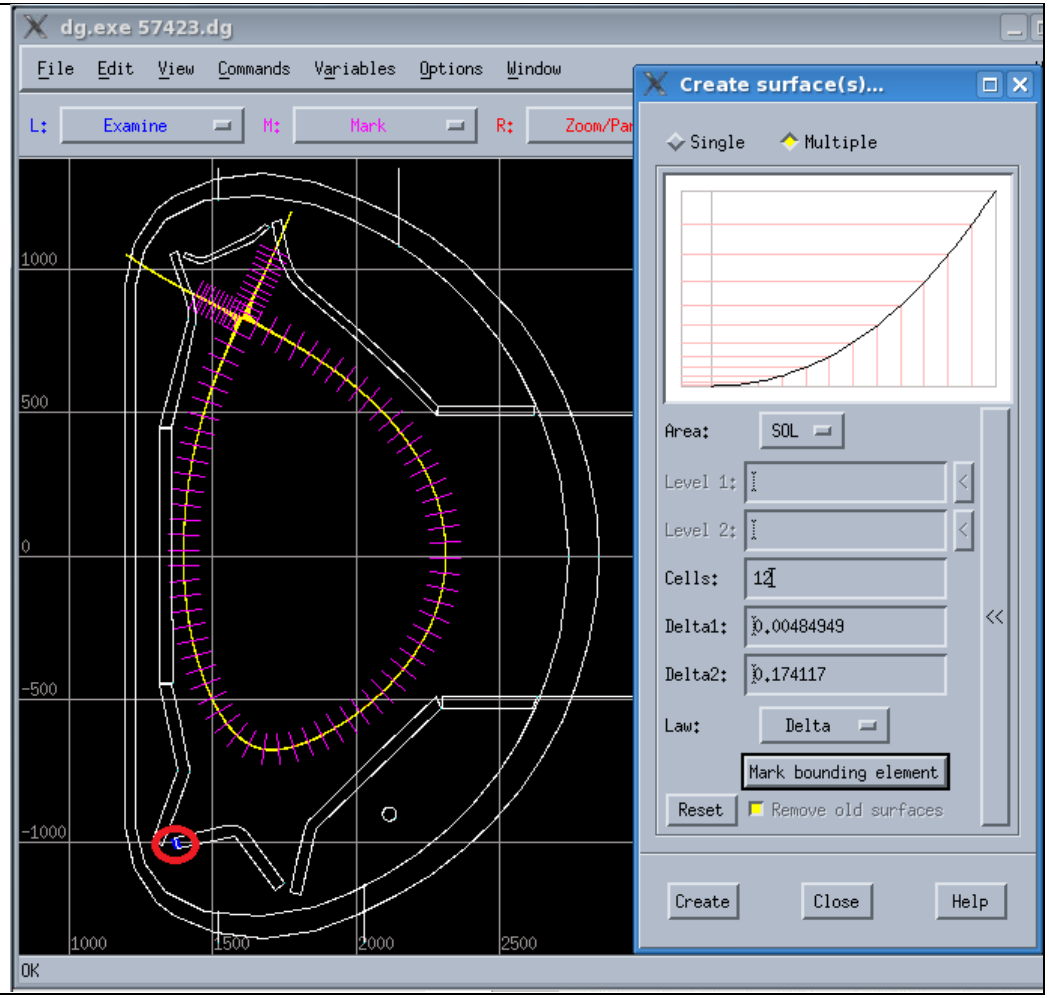
Set 18 surfaces in the PFR.



Radial surfaces, cont.

When asking for radial surfaces, you may find that DG only cover a very small area as opposed to what you expected. See an example on the right.

The way to understand this is by clicking the `Mark bounding element` box. The element marked (highlighted in blue and circled in red) is clearly not in contact with the SOL field lines, despite the fact that it should be the limiting element tangent to the outermost SOL flux surface.



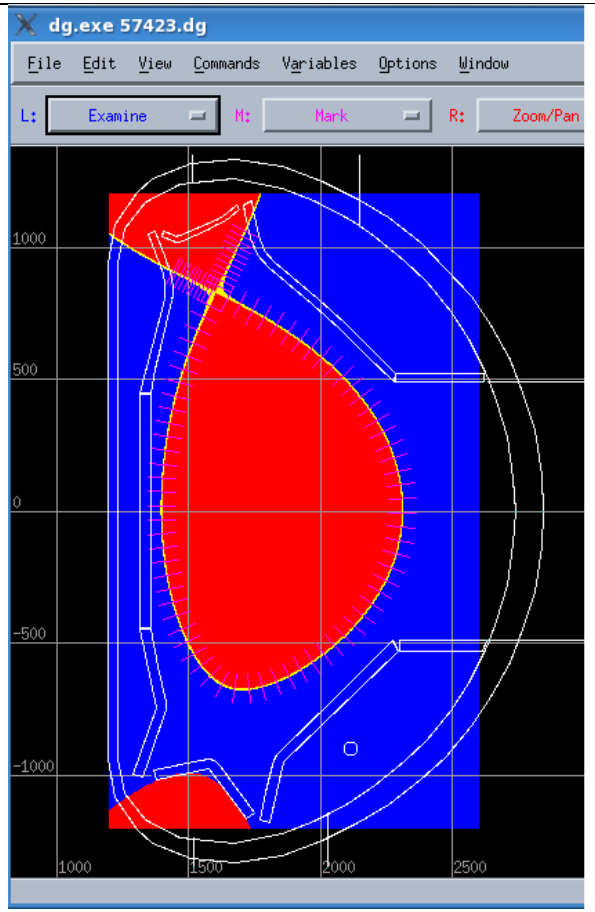
Radial surfaces, cont.

A second look at the equilibrium informs us.

The equilibrium is actually a largely disconnected double-null, and the bounding element is lying very close to the boundary of the bottom lobe of the equilibrium , so at psi values very close to those of the separatrix.

What must be done in such cases is crop the equilibrium so the bottom lobe in not part of the computational domain. This is done by means of the `cropequ` utility, which allows one to only keep a subset of the current equilibrium file.

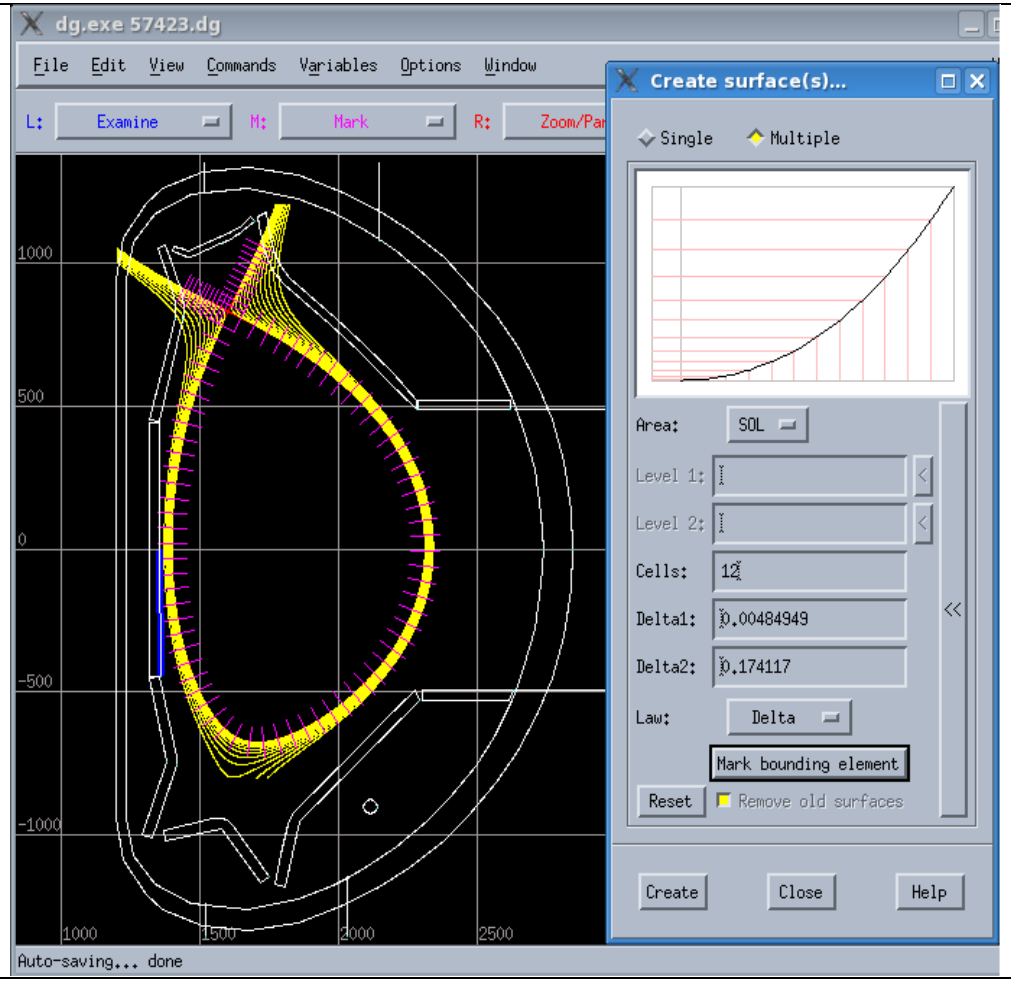
```
[bonninx@hpc-login4 baserun]$ cropequ g057423.005001.x32.equ
g057423.005001.x32_crop.equ
 rdeqdg: before rdeqlh
 rdeqdg: after rdeqlh. nr,nz,btf,rtf=        1025        1025  -
2.3429489135742188        1.7999999523162842
 reading rgr...
 reading zgr...
 reading pfm...

 Source grid:
    nr,rmin,rmax=        1025   1.2000000000000000        2.5999998999999998
    nz,zmin,zmax=        1025  -1.1999998999999999        1.2000000000000000
 Input new values (list-directed format) =>
1025 1.2 2.6 1025 -0.8 1.2
```



Radial surfaces, cont.

We then load the cropped equilibrium, re-import the `SN-up` topology, and ask for new surfaces, and obtain the surfaces to the right.

Note that some of the outermost surfaces are still pointing towards the outer LFS divertor because of the double-null equilibrium topology. These surfaces can be removed by adding a new structure that intercepts these structure so they are no longer included.



Radial surfaces, cont.

Now, back to the core region:

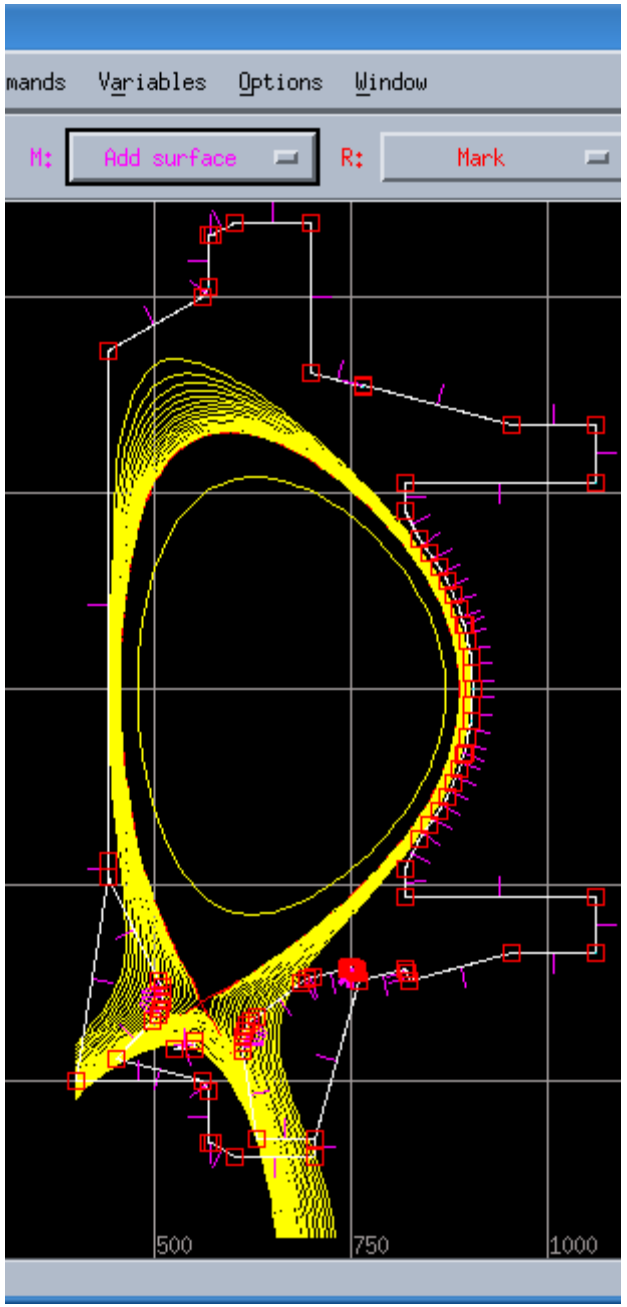
It is necessary to add a surface which will define the extent to which the grid penetrates into the core.

Assign “Add surface” to the middle mouse button.

Middle-click and hold somewhere in the core, and release the mouse button when happy with the location of the inner radial boundary.

The number of radial surfaces in the core must be the same as for the PFR, i.e. 18 in this case, using

`Edit > Create > Surfaces…`



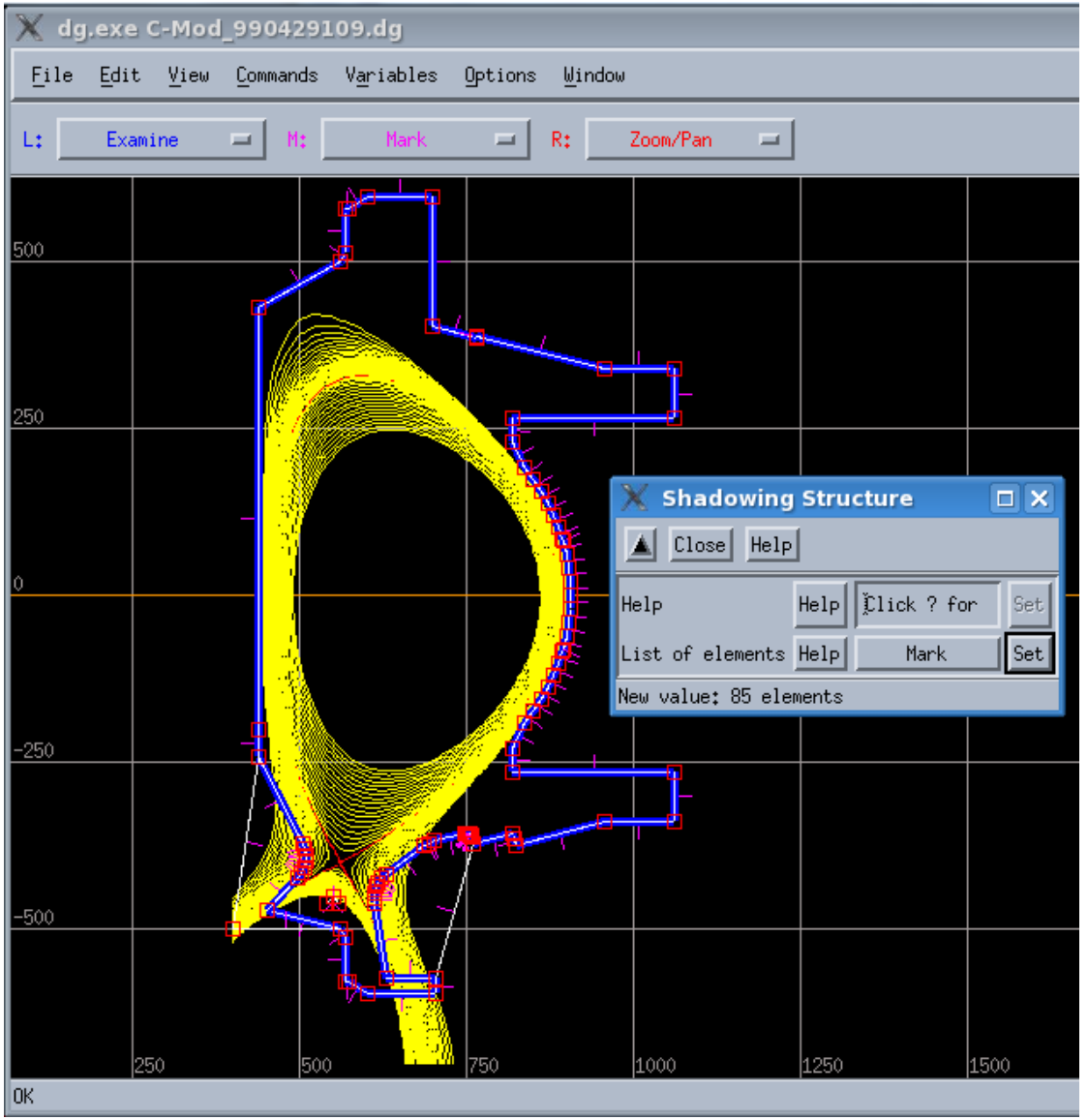
Setting the shadowing structure

The shadowing structure is the set of physical wall elements that can receive light from the plasma (or its reflections). It is used to compute the radiative contribution to the wall heat loads.

`Variables > Add > Shadowing structure`

CTRL+U

Mark all the segments likely to receive light from the plasma. The shadowing structure must be continuous and closed.



Adding some core radiation

To include core radiation in the wall heat loads, one needs to specify the amount of core radiation (in MW):

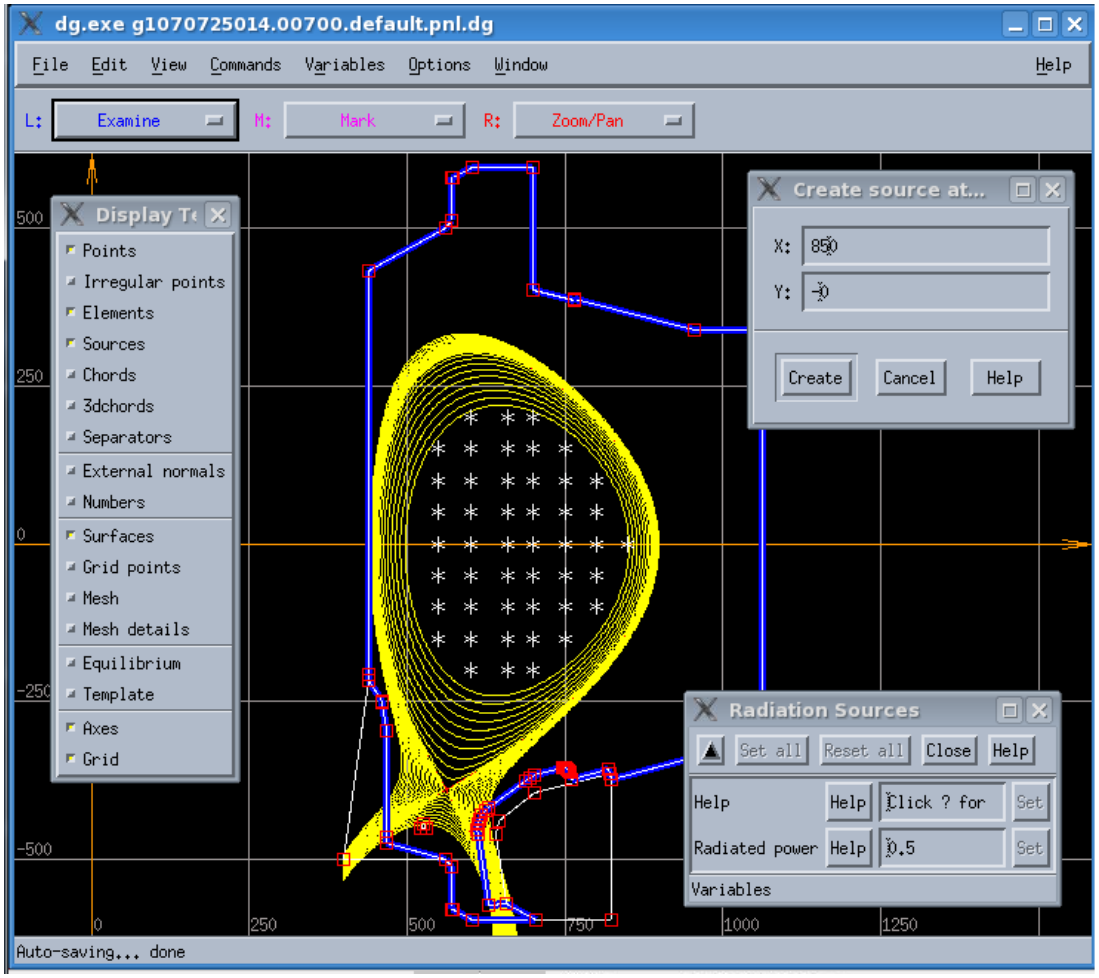
`Variables > Add > Radiation sources`

Enter in the “Radiated Power” field the amount of core radiation (in MW) that you want to consider.

You then need to specify the location from where this core radiation is emitted. This is done by providing a set of point sources. The radiated power will be spread evenly among these point sources. You create them by:

`Edit > Create > Source`

And specify the X and Y coordinates (in mm) of the point source location. You may input as few or as many point sources as you’d like. The point sources (if you choose to display them) are shown as white asterisks in the DG model.



Defining “plot zones”

A “plot zone” is a set of walls on which the power load (including contributions from the plasma particles, Eirene neutrals, and radiation) can be computed by b2plot.

`Variables > Add > Plot zone`

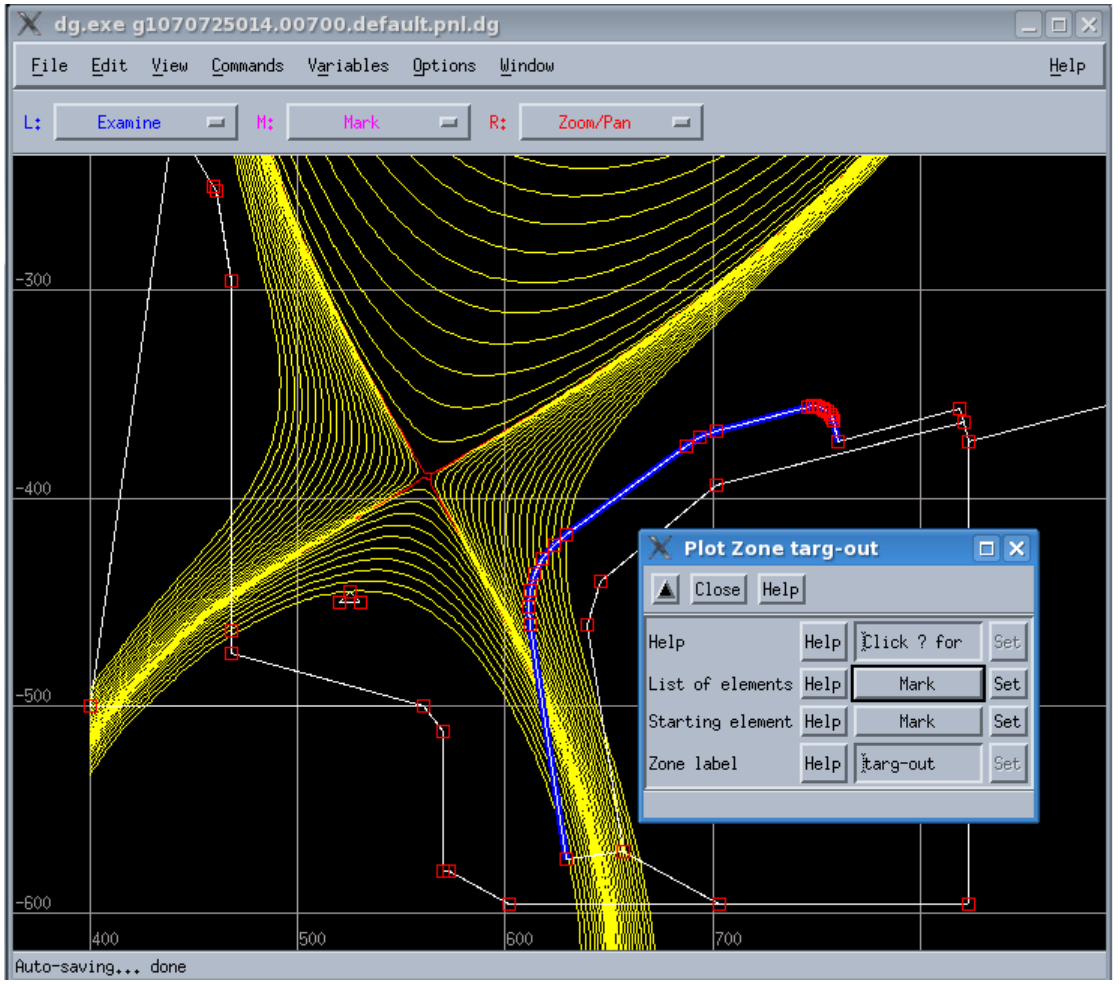
CTRL+U

Mark the set of elements that you wish to include in your plot zone. This set should be continuous.

CTRL+U

Mark the “Starting element”, i.e. the first element of the plot zone set, such that, as you travel along the plot zone set, the plasma is to your LEFT.

Give the zone a label (Zone-label) that will be used in the files created by b2plot (8 characters maximum, no spaces, stars or ellipses).



Configuring the plasma species to be included in the simulations

`Variables > Plasma species D`

Impurity species can be added using:

`Variables > Add > Plasma species`

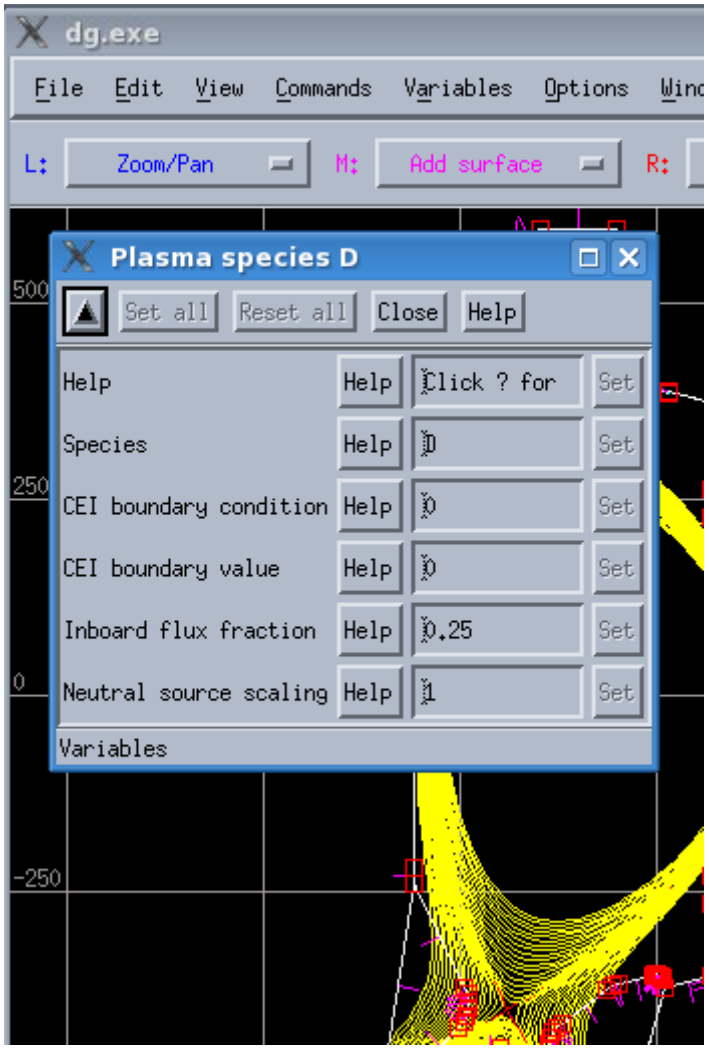
DG recognizes a few “generic” species: H, D, T, He, Be, C, N, Ne, and Ar, for which a full consistent default set of reactions will be provided by Uinp. For all other elements, Uinp will look for the corresponding ADAS ionization and recombination rates, and, if present in your database, will include them as part of your model.

If one wishes to use a different reaction set than the default, one can instead choose to load the reactions from an AMDS file, using:

`Variables > Add > Reference to AMDS`

and giving the name of the AMDS file requested. The number of AMDS files to be loaded is not limited. These files are to be found in the $SOLPSTOP/modules/AMDS directory.

One can also specify a simple boundary condition of either flux or value for the density of the highest ionization charge state of that species along the core boundaries.



EIRENE setup of the “void” regions outside the Carre grid

EIRENE will use a triangle grid in regions that are outside the fluid grid, and this variable defines the zones for the triangle mesh generator.

Mark all of the main chamber elements, including the “SOL edge” segments for the targets.

`Variables > Add > TRIA-EIRENE parameters`

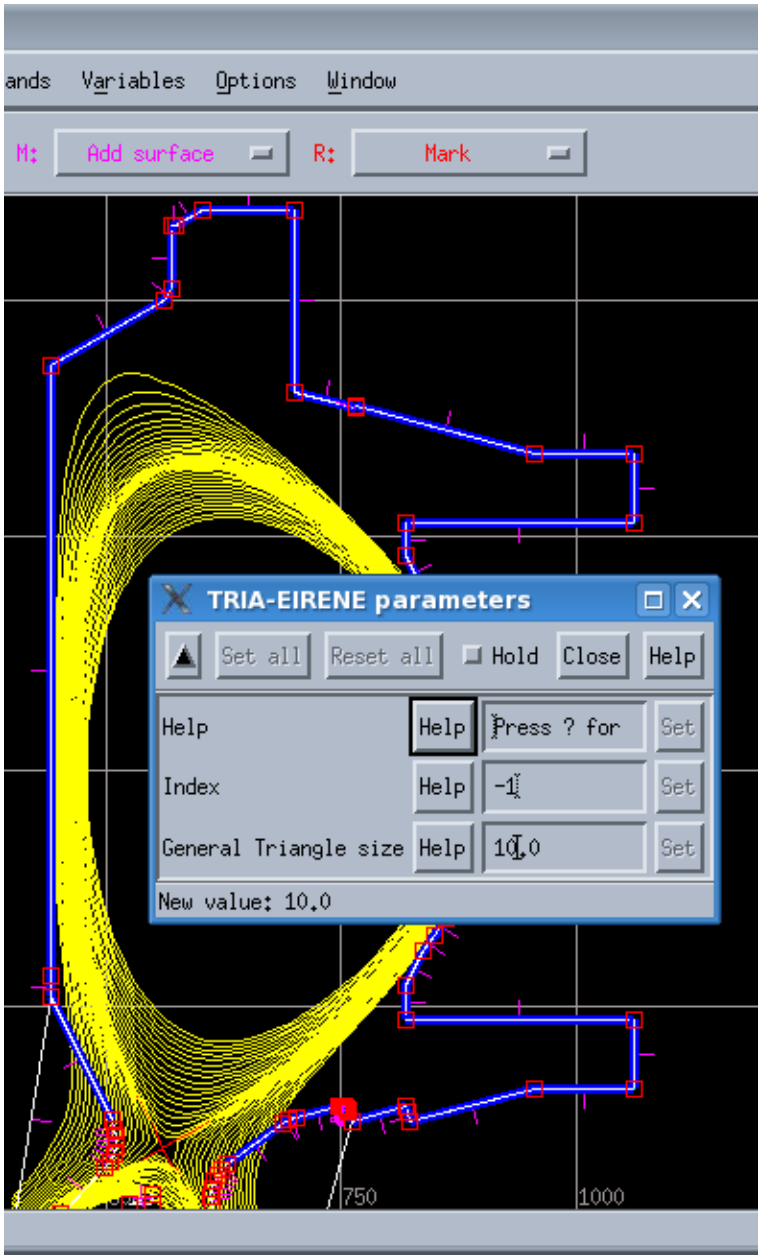
Set index to -1 in the dialogue box, and “General Triangle size” to 10.0, which will generate large triangles. The negative index indicates that a mesh should be generated inside the marked region, and a positive value means the opposite.

CTRL+U

Similarly, mark the wall segments in the PFR, including the “PFR edge” elements.

Set index to -2 in the dialogue box, and “General Triangle size” to 10.0.

Trick: Right-click in the `Index` box and you will be able to display the values currently assigned.



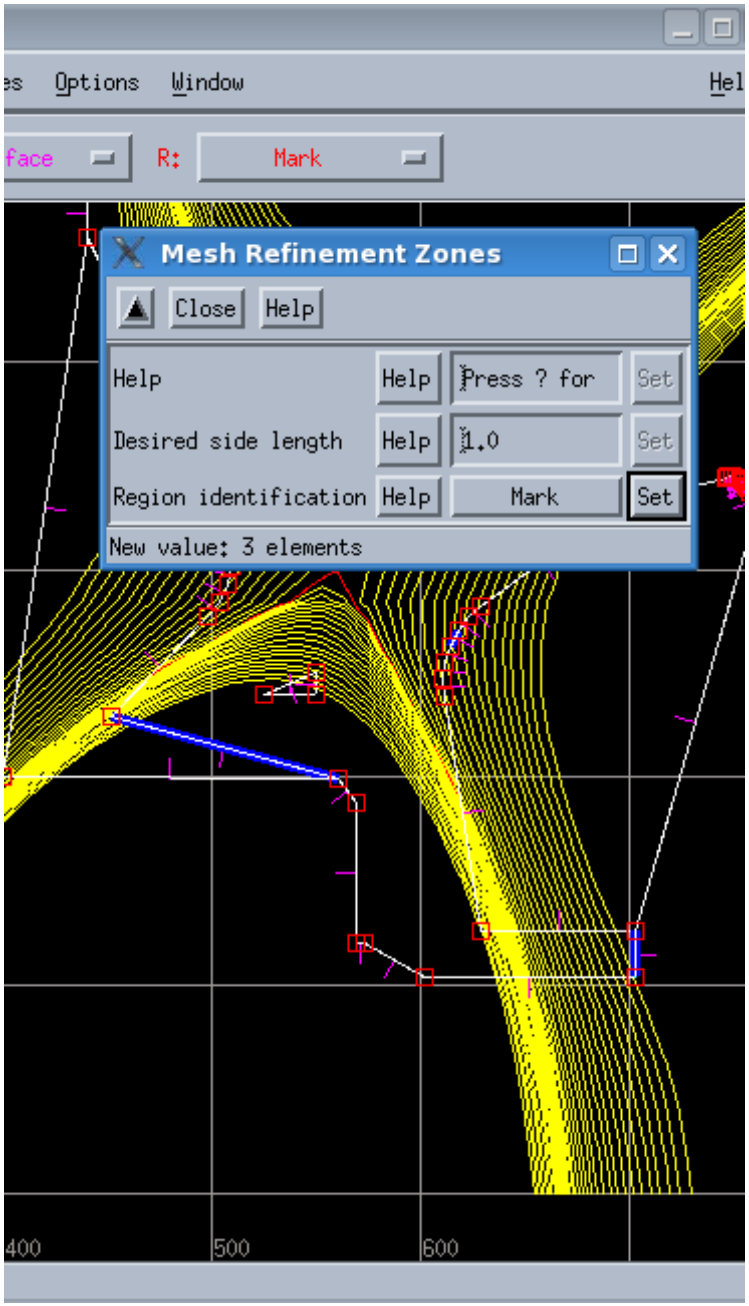
Local refinement of the EIRENE triangle grid

It's possible to increase the spatial resolution on sub-regions of the triangle mesh, i.e. the PFR.

`Variables > Add > Mesh Refinement Zones`

Select wall elements that bound the region of interest (left-right, top-bottom), as shown on the right, and click "Set" for "Region identification".

Set "Desired side length" to the desired characteristic scale size of the triangles in this region.



Choose the toroidal approximation `Variables > Global Eirene Data` Set the “Major Radius”to a negative (real) number, such as -1.0, if you want to use the toroidal approximation instead of the cylindrical approximation.	

Write the output data files that are needed by later steps
 	
`Commands > Check variables`
 	
Hopefully you see this at the bottom of the screen:
 
 	
. If not, call Xavier…
 	
`Commands > Rebuild Carre objects`
 	
`File > Save`
 	
`File > Output`
 	
The latter command will create three files:
<DG_model_name>.dgo, the DG “output” file
<DG_model_name>.str, the “structure” file (used by Carre)
<DG_model_name>.trg, the “targets” file (used by Carre)	

Prepare the links of the DG output files for later programs Because the other SOLPS-ITER programs expect to find the DG output files in a “standard”place, a set of symbolic links is produced to fulfill this requirement, by means of the command: `lns <DG_model_name>` Note that you should not include the “.dg” extension in the DG model name.	
Launch the mesh building script We now proceed with the creation of a plasma gridm using the Carre grid generator. `carre -`	[bonninx@hpc-login4 baserun]$ carre - Store directory: /home/ITER/bonninx/SOLPS-ITER_3/modules/Carre/meshes/cmod Grid directory: /home/ITER/bonninx/SOLPS-ITER_3/modules/Carre/meshes/cmod Input directory: . Standard linking - the input files are assumed to exist in "." Help, Prepare, Grid, Save, Convert, sTore, Next, Remove, Input, Output, Quit ? (p)
Preparation step `p` (or `<Enter>`) This steps reads the DG files and translates them into the format needed by Carre.	class = cmod griddir = /home/ITER/bonninx/SOLPS-ITER_3/modules/Carre/meshes/cmod indir = . grid_stem = g1070725014.00700.default.pnl rdeqdg: before rdeqlh rdeqdg: after rdeqlh. nr,nz,btf,rtf= 257 257 5.40751075744629 0.660000026226044 reading rgr... reading zgr... reading pfm... Help, Prepare, Grid, Save, Convert, sTore, Next, Remove, Input, Output, Quit ? (g)

Gridding step `g` (or `<Enter>`) The first question you must answer is whether the X- and O- points identified by Carre are correct (they usually are). If they are not, then, you refuse the selection and indicate yourself which of the extrema are X- and O- points. `y`	Starting newpag OK cpsets ok cprect ok cpcldr ok Pre-selected points are identified. O-point: 6.8030E-01 -8.0087E-03 X-point: 5.6218E-01 -3.8994E-01 Do you accept the selection (y/n)?

Grid parameter selection step

Carre then provides a table of parameters. Please refer to Chapter 8 of the SOLPS-ITER manual for a full description. Carre attempts to provide a grid that must satisfy three criteria simulatenously:

1. The grid cells must be as locally orthogonal as possible
2. The grid cells must align with the targets in their vicinity
3. The size of neighbouring grid cells must not vary too quickly.

The specific definitions of these criteria and their respective weights can be found in the original Carre paper, which you will find at: `$SOLPSTOP/modules/Carre/doc/Carre_Paper.pdf`.
In some instances, it is relatively easy to find a satisfactory solution meeting these three criteria, but this is not always the case, especially in geometries where the targets are almost parallel to the flux surfaces, putting criteria 1 and 2 at odds with one another.

Quickly, the data provided in the Carre table indicates the poloidal spacing of the grid points along the separatrix segments (check the Carre paper for their numbering convention), the radial spacing of the successive flux surfaces as one steps away from the separatrix (again, the numbering convention is given in the Carre paper), the penetration depth of the grid (`pntrat`), the guard lengths (i.e. the vicinity over which criterion 2 above is applied, in meters), and some numerical parameters used for the optimization algorithm that attempts to build the mesh.

The poloidal spacings are deduced from the distribution of grid points chosen in DG. The guard lengths are given by the “CARRE guard length” parameters in the DG “Target specifications”. The `pntrat` value is obtained from the innermost DG surface chosen in the core.

The radial extent of the grid (and therefore the radial grid spacings) is determined by the first tangency points (in the PFR and main chamber vessel) between the flux surfaces and the “Structure” defined in DG. However, the algorithm is not identical to DG’s, so the values found may differ.

```
      SEPARATRIX SEGMENTS:

 #  nptseg(i)  lg(i)        deltp1(i)      deltpn(i)       dpmin        dpmax
================================================================================
 1     21     1.1724E-01   9.1116E-03  1.8834E-04  1.8834E-04  9.1116E-03
 2     21     1.1053E-01   8.8644E-03  1.8323E-04  1.8323E-04  8.8644E-03
 3     41     1.7723E+00  1.1810E-02  1.5681E-02  1.1810E-02  6.0815E-02

      RADIAL DISTRIBUTIONS FOR EACH REGION:
        Distribution in psi: repart=    2

 region npr(i)    width     deltr1(i)       deltrn(i)       drmin         drmax
================================================================================
 1     21     2.7276E-02  9.6815E-05   2.4586E-03  9.6815E-05   2.4586E-03
 2     11    -1.8005E-02 -2.1524E-04 -3.1739E-03 -3.1739E-03 -2.1524E-04

      CENTRAL REGION: region i=  3     pntrat max.= 0.39977943

 pntrat npr(i)    width       deltr1(i)       deltrn(i)       drmin          drmax
================================================================================
  0.161  11    -1.4957E-01 -1.7755E-03 -2.6181E-02 -2.6181E-02 -1.7755E-03

      GUARD LENGTH FOR EACH DIVERTOR PLATE:

 tgarde(1)    tgarde(2)
================================================================================
  0.20000       0.20000

      RELAXATION PARAMETERS USED TO CONSTRUCT THE MESH:

================================================================================
nrelax       relax        pasmin        rlcept
 5000        0.200        1.000E-03     1.000E-06
```

Carre criterion checks Often, the first pass at the Carre grid parameters will not pass internal muster. Carre checks for a few minimal requirements: - The poloidal grid spacing must be above a minimal threshold, defined by the `pasmin` parameter. This is not enforced by DG and is the most common correction you'll have to make. - The radial and poloidal grid spacings in each region must have the same sign. The spacings are constrained by the first and last values of the region to grid and the total interval length. The code will not proceed until these requirements are met and will show messages like this:	Invalid data for segment 1! The numbers dpmin and dpmax must be larger than pasmin in absolute value. Modify deltp1, deltpn or pasmin accordingly. dpmin,dpmax,pasmin = 1.8834E-04 9.1116E-03 1.0000E-03 Invalid data for segment 2! The numbers dpmin and dpmax must be larger than pasmin in absolute value. Modify deltp1, deltpn or pasmin accordingly. dpmin,dpmax,pasmin = 1.8323E-04 8.8644E-03 1.0000E-03 Type the name of the variable to be changed followed by '=', and its numerical value. For example: nptseg(2)=32 (return) Type "end" to stop.

Modifying the Carre parameters (I)
 	
Follow the instructions! Usually changing the value of `pasmin` is a good start.
 	
`pasmin=0.99e-3`
`end`
 	
In this case, this is not sufficient…
	Invalid data for segment 1!
	The numbers dpmin and dpmax must be larger than pasmin
	in absolute value.
	Modify deltp1, deltpn or pasmin accordingly.
	dpmin,dpmax,pasmin = 1.8834E-04 9.1116E-03 9.9000E-04
	
	Invalid data for segment 2!
	
	The numbers dpmin and dpmax must be larger than pasmin
	in absolute value.
	Modify deltp1, deltpn or pasmin accordingly.
	dpmin,dpmax,pasmin = 1.8323E-04 8.8644E-03 9.9000E-04

Modifying the Carre parameters (II)

We now modify the poloidal grid spacings. The 1st spacing is the one touching the X-point, and the last spacing is at the targets.

```
deltpn(1)=1.0e-3
deltpn(2)=1.0e-3
end
```

Now the minimal criteria are met!

```
y
```

SEPARATRIX SEGMENTS:

#	nptseg(i)	lg(i)	deltp1(i)	deltpn(i)	dpmin	dpmax
1	21	1.1724E-01	9.1116E-03	1.0000E-03	1.0000E-03	9.1116E-03
2	21	1.1053E-01	8.8644E-03	1.0000E-03	1.0000E-03	8.8644E-03
3	41	1.7723E+00	1.1810E-02	1.5681E-02	1.1810E-02	6.0815E-02

…

RELAXATION PARAMETERS USED TO CONSTRUCT THE MESH:

nrelax	relax	pasmin	rlcept
5000	0.200	9.900E-04	1.000E-06

Do you wish to accept these values (y/n)?

Producing the grid Carre then proceeds, one region at a time (`ireg` index), to build the flux surfaces (`ir` index) in order, stepping from the separatrix outward. Normal output looks like the one of the right. However, this is not always the case. If the Carre algorithm fails to converge while building the flux surfaces, you will get a (non-fatal) error message that allows you to continue, but suggests some possible changes to the gridding parameters that might improve convergence, although it may not improve the "quality" of the final grid. You may also get a fatal error message that indicates that Carre was not able to build the next flux surface. This usually occurs because the radial spacings required are too small compared to the resolution of the magnetic equilibrium provided (fix this by refining the equilibrium a further step, using `d2d`, or by increasing the minimal radial spacings by modifying the `deltr1` and `deltrn` parameters). Sometimes, this is because the limiting structures in the DG structure are so small that they may stepped over as Carre moves from one flux surface to the next, and can be fixed by going back to the DG model and making this limiting structures bigger (a few cm is usually sufficient).	ireg= 1 ir= 2 ir= 3 ir= 4 ir= 5 ir= 6 …. ir= 7 ir= 8 ir= 9 ir= 10 ir= 11
Saving the grid parameters If you wish to remember the settings change you made, choose the `Save` option. The grid parameters are then written in the `carre.dat` file. If you wish to re-use these parameters, skip the `Prepare` step in your next invocation of the `carre` script.	Help, Prepare, Grid, Save, Convert, sTore, Next, Remove, Input, Output, Quit ? (c)s Saved grid settings in carre.dat file

Converting the grid output from Carre `c` (or `<Enter>`) The conversion step takes the `carre.out` file containing the Carre grid and converts it to the Sonnet and B2.5 formats, respectively a *.sno and *.geo file.	Help, Prepare, Grid, Save, Convert, sTore, Next, Remove, Input, Output, Quit ? (c)c Name of the file containing the carre grid carre.out Select the output format 1: standard mailtri format 2: format B2.5 3: format SONNET-DIVIMP 4: format DG-SONNET-B2-EIRENE 5: revised DIVIMP with grid parameters and PSI values The format chosen is : 4 Name of the file containing the carre grid carre.out Select the output format 1: standard mailtri format 2: format B2.5 3: format SONNET-DIVIMP 4: format DG-SONNET-B2-EIRENE 5: revised DIVIMP with grid parameters and PSI values The format chosen is : 2 Help, Prepare, Grid, Save, Convert, sTore, Next, Remove, Input, Output, Quit ? (t)

Storing the grid files `t` (or `<Enter>`)	Help, Prepare, Grid, Save, Convert, sTore, Next, Remove, Input, Output, Quit ? (t) dir OK The grid in DG format is stored as /home/ITER/bonninx/SOLPS-ITER_3/modules/DivGeo/device/cmod/g1070725014.00700.default.pnl.v001.sno dg.dgo copied to /home/ITER/bonninx/SOLPS-ITER_3/modules/DivGeo/device/cmod dg.equ copied to /home/ITER/bonninx/SOLPS-ITER_3/modules/DivGeo/device/cmod dg.str copied to /home/ITER/bonninx/SOLPS-ITER_3/modules/DivGeo/device/cmod dg.trg copied to /home/ITER/bonninx/SOLPS-ITER_3/modules/DivGeo/device/cmod DG model g1070725014.00700.default.pnl.dg copied to /home/ITER/bonninx/SOLPS-ITER_3/modules/DivGeo/device/cmod The grid in B2.5 format is stored as /home/ITER/bonninx/SOLPS-ITER_3/modules/Carre/meshes/cmod/g1070725014.00700.default.pnl.v001.geo You can view the grid in PostScript format as /home/ITER/bonninx/SOLPS-ITER_3/modules/Carre/meshes/cmod/g1070725014.00700.default.pnl.v001.ps carre.dat, structure.dat, rzpsi.dat, and btor.dat copied to /home/ITER/bonninx/SOLPS-ITER_3/modules/Carre/meshes/cmod/g1070725014.00700.default.pnl.v001 Help, Prepare, Grid, Save, Convert, sTore, Next, Remove, Input, Output, Quit ? (q)

Import the mesh into DG

`File > Import > Mesh`

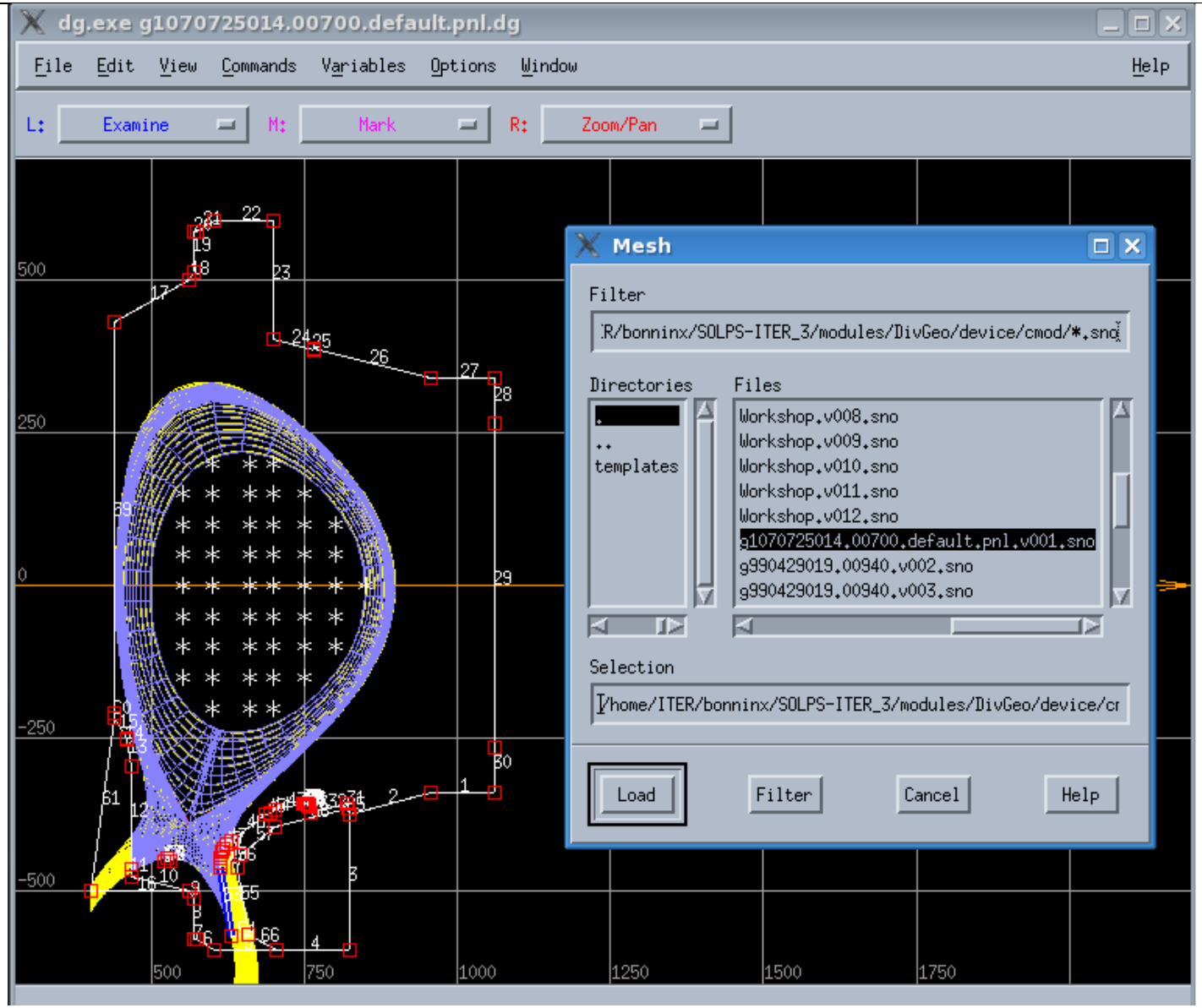
Note that DG can only read files in the Sonnet format (*.sno).

You may see some grid cells that are outlined in magenta. These are concave cells that will yield errors when running Eirene and should be corrected before proceeding. Two methods are available.

The first is to go back to Carre and chose a different set of gridding parameters and try your luck or smarts against an ill-posed mathematical problem. This is where the `Save` option comes in handy!

The second is to modify the grid points by hand (if there are not too many of them). This can be done as follows. Change one of the mouse button functions to `Move mesh point` and select a corner of a magenta grid cell and move it until the cell outline changes colour to lavender. You may need to propagate such changes over a range of cells. Bear in mind however that you are only modifying the *.sno grid file. You will need to save your modifications by exporting the mesh:

`File > Export > Mesh`



Closing Carre or gridding again If you are happy with your grid, or modified it within DG, you can now quit the `carre` script. `q` (or `<Enter>`) If you wish to obtain a new grid (and you have saved the previous set of gridding parameters): `g` (or `<Enter>`) Repeat until satisfied…	Help, Prepare, Grid, Save, Convert, sTore, Next, Remove, Input, Output, Quit ? (q)
Start the triangulation script `triang`	[bonninx@hpc-login4 baserun]$ triang Help, Uinp, B2ag, Eirene, Tria, triaGeom, Plot, View, Store, Convert, List, Remove, reMap, Inquire, Quit ? (u)

Create the input files

On the first call, it is best to use:

```
U
```

Using uppercase `U` instead of the (default) lowercase `u` ensures that the links (from the `lns` command earlier) are correct. `Uinp` is a program that builds several input files for SOLPS-ITER programs according to the data provided in the DG model:

- Eirene input files `input.eir`, `test.eir` and `triang.eir`
- Tria input file header `triang.hed`
- B2.5 pre-processor input files `b2ag.dat`, `b2ai.dat`, `b2ar.dat`, and converter input file `b2yt.dat`.
- B2.5 input file `b2.user.parameters`
- b2plot input file `mesh.extra`
- Stencil files `b2ah.dat.stencil`, `b2.neutral.parameters.stencil` and `b2.boundary.parameters.stencil` which are sample files containing a very rough default physics model for the boundary conditions and recycling parameters, but that have the right format for further modification to adapt to your physics problem at hand.

Uinp will display the list of particles being used, the reactions it will include in the Eirene input file, the boundaries it will define, etc…

```
…
Particle list: natmi,nmoli,nioni,nplsi =    1    1    1    1

 atoms         nmassa  nchara              isrfa   isrta  nmseca   nfola   ngena spcsrna
  D             2          1       1      0      1     -1      0      0      0      1
 molecules     nmassm  ncharm   nprtm  nchrgm   isrfm   isrtm  nmsecm
  D2            4          2       2      0      0      0      0
 test ions     nmassi  nchari   nprti  nchrgi   isrfi   isrti  nmseci   nfoli
  D2+           4          2       2      1      0     -1      0     -1
 plasma ions   nmassp  ncharp   nprtp  nchrgp   isrfp   isrtp  nmsecp spcsrnp
  D+            2          1       1      1      1     -1      0      1
…
Raw reaction list:         17
   species                                  bulk   out1  out2 ig mp mt N1 N2   dlte     enr1     enr2
  1 D       AMJUEL H.4  2.1.5    EI          D+           -1  0  2  1  0 0.00E+00 0.00E+00 0.00E+00
  2 D       AMJUEL H.102.1.5    EI          D+           -1  0  2  1  0 0.00E+00 0.00E+00 0.00E+00
  3 D       HYDHEL H.1  3.1.8    CX   D+     D+     D     -2  2  2  1  1 0.00E+00 0.00E+00 0.00E+00
  4 D       HYDHEL H.3  3.1.8    CX   D+     D+     D     -2  2  2  1  1 0.00E+00 0.00E+00 0.00E+00
  5 D2      AMJUEL H.4  2.2.9    EI          D2+           0  0  4  1  0 1.54E+01 1.00E+00 0.00E+00
  6 D2      AMJUEL H.4  2.2.5g   DS          D             0  0  4  2  0 1.05E+01 3.00E+00 3.00E+00
  7 D2      AMJUEL H.4  2.2.10   DS          D      D+     0  0  4  1  1 2.50E+01 5.00E+00 5.00E+00
  8 D2      AMJUEL H.0  0.3T     EL   D+     D2     D+    -3  2  4  1  1 0.00E+00 0.00E+00 0.00E+00
  9 D2      AMJUEL H.1  0.3T     EL   D+     D2     D+    -3  2  4  1  1 0.00E+00 0.00E+00 0.00E+00
 10 D2      AMJUEL H.3  0.3T     EL   D+     D2     D+    -3  2  4  1  1 0.00E+00 0.00E+00 0.00E+00
 11 D2      AMJUEL H.2  3.2.3    CX   D+     D2+    D      0  2  4  1  1 0.00E+00 0.00E+00 0.00E+00
 12 D2+     AMJUEL H.4  2.2.12   DS          D      D+     0  0  4  1  1 1.04E+01 4.30E+00 4.30E+00
 13 D2+     AMJUEL H.4  2.2.11   EI          D+           0  0  4  2  0 1.55E+01 2.50E-01  2.50E-01
 14 D2+     AMJUEL H.4  2.2.14   DS          D           -4  0  4  2  0 0.00E+00 5.00E-01  5.00E-01
 15 D2+     AMJUEL H.8  2.2.14   DS          D           -4  0  4  2  0 0.00E+00 0.00E+00 0.00E+00
 16 D+      AMJUEL H.4  2.1.8    RC          D           -5  0  2  1  0 0.00E+00  0.00E+00 0.00E+00
 17 D+      AMJUEL H.102.1.8    RC          D           -5  0   2  1  0 1.36E+01 0.00E+00 0.00E+00
…
```

Some common Uinp errors

In this case, Uinp was unhappy…

This sort of error can occur because of a variety of reasons:

- The targets are not specified in the right order (see the table on page 16)
- The surface normal of (some or all) wall elements are not pointing away from the plasma (see page 6)
- The contour indices of the wall elements have the wrong sign (see page 27)
- Your contours contain gaps, or do not begin and end at the edges marked as `PFR edge` or `SOL edge` in the target specifications
- The `PFR edge` or `SOL edge` settings are wrong and these wall are not actually in contact with the grid corners

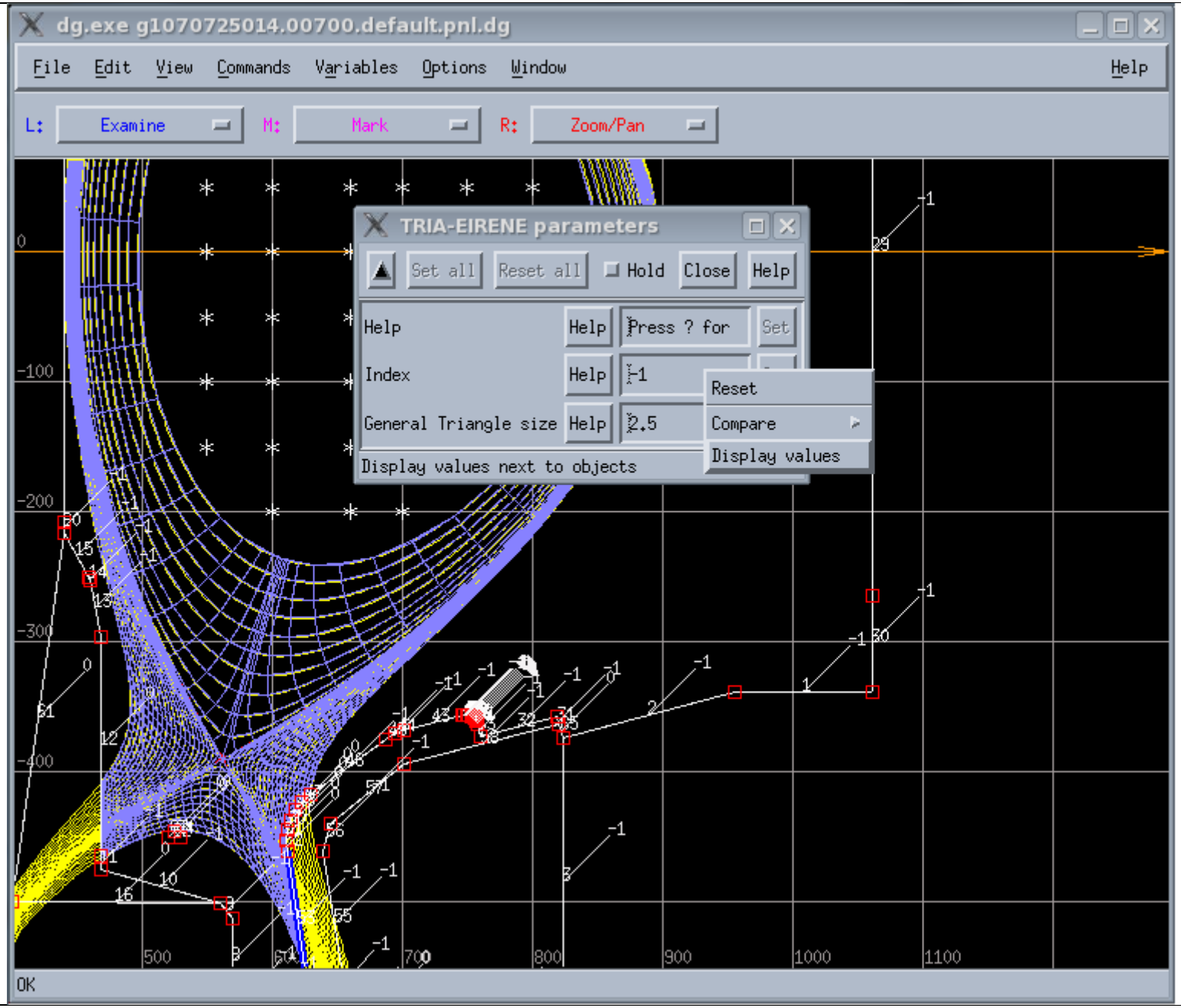
Uinp also provides useful information about what it thinks the contours are. Here, contour -1 (`label`) is initially thought to contain 52 elements (`nni`). The indices (in the DG model) of the wall elements forming the contour are then given. It is thus possible to check these data.

Thus, let's go back to our DG model and check all these things!

```
==== Starting uinptriang
 nattr         2
 attrib        -1         0
 chain number, attrib         1        -1
 nchain        52
chain   1   2   3   4   5   6   7   8   9  10  11  13
  14  15  17  18  19  20  21  22  23  24  25  26  27
  28  29  30  31  32  33  34  35  36  37  38  39  40
  41  42  43  44  45  46  53  54  55  56  57  58  59
  60

Starting uichkchn: label,nni,nnx=         -1        52        50
indelm:
   1   2   3   4   5   6   7   8   9  10  11  13  14  15  17  18  19  20  21  22
  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42
  43  44  45  46  53  54  55  56  57  58  59  60

uichkchn: the chain does not end at the target edges.
label,j,j1,j2=         -1         1         1         0
uichkchn: the chain does not end at the target edges.
label,j,j1,j2=         -1         2         0         0
label,irt=         -1         2
   30
   11  10   9   8   7   6   5   4   3   2   1  30  29  28  27  26  25  24  23  22
   21  20  19  18  17  59  60  15  14  13
   11
   10   9   8   7   6   5   4   3   2   1  30
  ==> Check the "Index" specification in
"TRIA-EIRENE parameters" in DG variables
The targets must be numbered according to the B2 convention:
going clockwise and starting from the bottom left corner.
nn         -4
ERROR in uichkchn, nn         -4
```

Checking the DG contours for compatibility with Uinp

From the information above, we notice that wall #12 is not part of contour -1, which is confirmed by the `Display values` option.



Correcting the contours

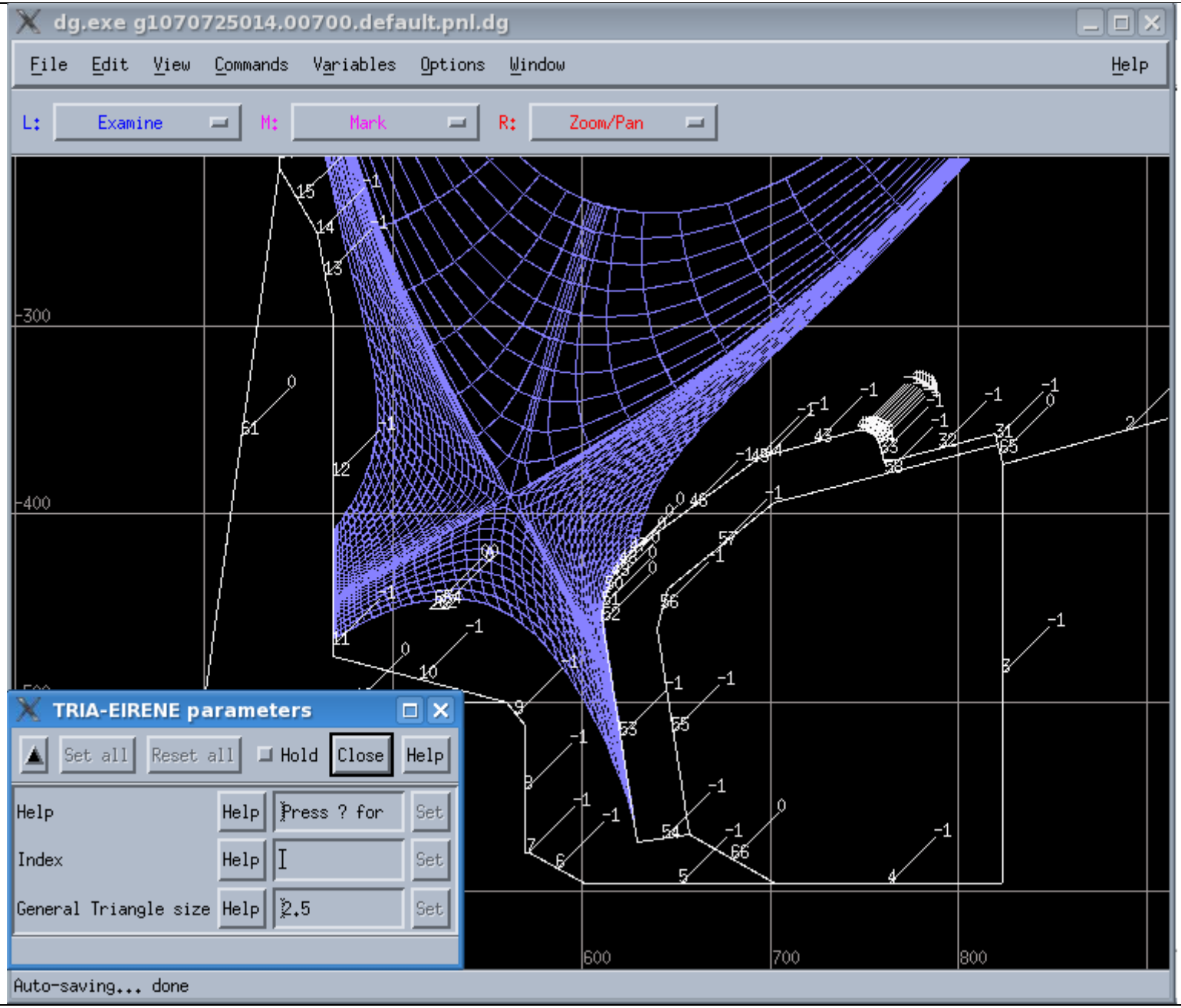
In DG:

`Variables > TRIA-EIRENE parameters`

Set the necessary values…

`File > Save`

`File > Output`



Uinp again… In the `triang` script, re-run Uinp by doing `u` again, overwriting the `b` default. Uinp now succeeds…	Help, Uinp, B2ag, Eirene, Tria, triaGeom, Plot, View, Store, Convert, List, Remove, reMap, Inquire, Quit ? (b)u … nn 3 NSS 5 4 6 chain number, attrib 2 0 ... input.eir created ... test.eir created ... triang.eir created ... triang.hed created ... b2ag.dat file already present, not created ... b2ai.dat file already present, not created ... b2ar.dat file already present, not created ... b2yt.dat file already present, not created No b2.neutrals.parameters found Writing b2.neutrals.parameters.stencil ... b2.neutrals.parameters.stencil created No b2.boundary.parameters found Writing b2.boundary.parameters.stencil ... b2.boundary.parameters.stencil created ... b2.user.parameters.2 created ... mesh.extra created uinp The last grid found is : g1070725014.00700.default.pnl.v001.geo

Modifying the b2ag.dat file If you have modified the grid file by hand in DG, you need to modify the `b2ag.dat` (right) file created by Uinp to point to your modified file instead of the file initially created by Carre. To do this, change the value of the first parameter in the `*param` list from -1.0 to -2.0 and the name of the file given to the `b2agfs_geometry` switch to your modified *.sno file.	*dimens (nx, ny; free format) 80 30 80 30 *param (param(0:99); free format) **-1.0**, 99*0.0 'b2agfs_min_pitch' '0.01' 'b2agfs_geometry' **'g1070725014.00700.default.pnl.v001.geo**'
Creating the fort.30 file Once you are pointing to the correct mesh file, proceed with the `triang` script: `b` (or `<Enter>`) This command runs the `b2ag` program to create a geometry file that provides the mesh geometry used by `Eirene`. This step will be skipped if a `fort.30` file is already present. If you are re-running `triang` to obtain a new geometry in an already populated directory, make sure you have removed the older `b2ag.dat` and `fort.30` files beforehand.	Help, Uinp, B2ag, Eirene, Tria, triaGeom, Plot, View, Store, Convert, List, Remove, reMap, Inquire, Quit ? (b) Command line: gmake -f /home/ITER/bonninx/SOLPS-ITER_3/runs/Makefile b2ag.prt > b2ag.log b2ag real 0m2.009s user 0m0.217s sys 0m0.494s Produced fort.30 file

Eirene triangulation preparation run

`e` (or `<Enter>`)

If this step is successful, the output will look like the one to the right.

If this step fails (i.e. the message "`No valid fort.78 file created`" appears), you should look at the error messages from Eirene that you will find in the `eirtria.log` file. Standard Eirene debugging applies. See the Annex below for more information.

This step uses Eirene to produce the full contours that will be used in the triangulation step next. The contours deduced by Eirene are written in the `fort.78` file. The script then concatenates this file with the `triang.hed` file, which contains the information about the desired triangle sizes and mesh refinement zones from DG to form the `tria.in` file that will be the input for the `tria` program.

If you wish to display these contours, you can run the Eirene version with graphics enabled:

```
ln –s triang.eir fort.1
eirobj < triang.eir >!
eirtria.log
```

and view the `gli.eps` file.

```
Help, Uinp, B2ag, Eirene, Tria, triaGeom, Plot, View, Store, Convert, List, Remove, reMap, Inquire, Quit ? (e)
Creating a link between input.eir and input.dat
Starting Eirene - may take a while...
Command line: /home/ITER/bonninx/SOLPS-ITER_3/modules/Eirene/builds/couple_SOLPS-
ITER.ITER.ifort64.debug/eirobjx < triang.eir > eirtria.log
EIRENE TRIANGULATION PREPARATION RUN COMPLETE
```

Triangulation
 	
`t` (or `<Enter>`)
 	
If this step is successful, the output will look like the one to the right.
 	
The triangulation results will be stored in three files:
`tria.nodes`
`tria.elemente`
`tria.neighbor`
which contain, respectively, the coordinates of the nodes, the vertices assignment for the triangles, and the connectivity information between triangles.
 	
If this step fails, you should look at the error messages from `tria` that you will find in the `tria.log` file. See the Annex to this document for some example errors.
	Help, Uinp, B2ag, Eirene, Tria, triaGeom, Plot, View, Store, Convert, List, Remove, reMap, Inquire, Quit ? (u)t
	Starting a TRIA run.
	Command line: /home/ITER/bonninx/SOLPS-ITER_1/modules/Triang/builds/ITER.ifort64/tria > tria.log
	If it continues more than a couple of minutes, kill with Ctrl-C
	and check the output - probably, the polygon sides are wrong
	In that case, consider re-running DG and asking for larger triangles
	GLI_HOME:/work/imas/opt/gks/4.4.74/lib

Triangulating the Carre grid and merging it with the outer triangles `g` (or `<Enter>`) If this step is successful, the output will look like the one to the right. This steps takes the output from the triangulation step and adds nodes corresponding to the Carre grid cells, each of which is split into at least two triangles.	Help, Uinp, B2ag, Eirene, Tria, triaGeom, Plot, View, Store, Convert, List, Remove, reMap, Inquire, Quit ? (g)g Starting a TRIAGEOM run Command line: /home/ITER/bonninx/SOLPS-ITER_1/modules/Triang/builds/ITER.ifort64/triageom > triageom.log GLI_HOME:/work/imas/opt/gks/4.4.74/lib

The connectivity information is also enriched to indicate in which Carre grid cell, if any, each triangle is embedded. This step also creates the links between the `triageom.[cells,nodes,links]` files and the `fort.3[3-5]` files used later by SOLPS-ITER.	
Storing the triangle files `s` (or `<Enter>`) If this step is successful, the output will look like the one to the right. The triangulation files will be stored in some standard directories. The script may ask you to create these directories if those are not yet present. Links to the triangle files as `fort.33`, `fort.34`, and `fort.35`, as needed by Eirene, are updated to their new location.	Help, Uinp, B2ag, Eirene, Tria, triaGeom, Plot, View, Store, Convert, List, Remove, reMap, Inquire, Quit ? (s)s The serial number selected for the mesh is 006
Converting the triangle files into an "outer" template `c` (or `<Enter>`) If this step is successful, the output will look like the one to the right. The `template.aXXX.tria` file contains the triangles created by the `tria` step, which cover the area outside the Carre grid.	Help, Uinp, B2ag, Eirene, Tria, triaGeom, Plot, View, Store, Convert, List, Remove, reMap, Inquire, Quit ? (c)c Command line: cnvtria template.a006 npoint 1221 reading *.npco_char is finished ntria 1934 reading *.element is finished writing template is finished ... converted into a template template.a006.tria
Converting the triangle files into a "grid" template	Help, Uinp, B2ag, Eirene, Tria, triaGeom, Plot, View, Store, Convert, List, Remove, reMap, Inquire, Quit ? (c)C npoint 3609

`C` (or `<Enter>`) If this step is successful, the output will look like the one to the right. The `template.gXXX.tria` file contains the triangles created by the `triageom` step, which covers the entire domain.	reading *.npco_char is finished ntria 6808 reading *.element is finished writing template is finished ... converted into a template template.g006.tria

Import the templates into DG

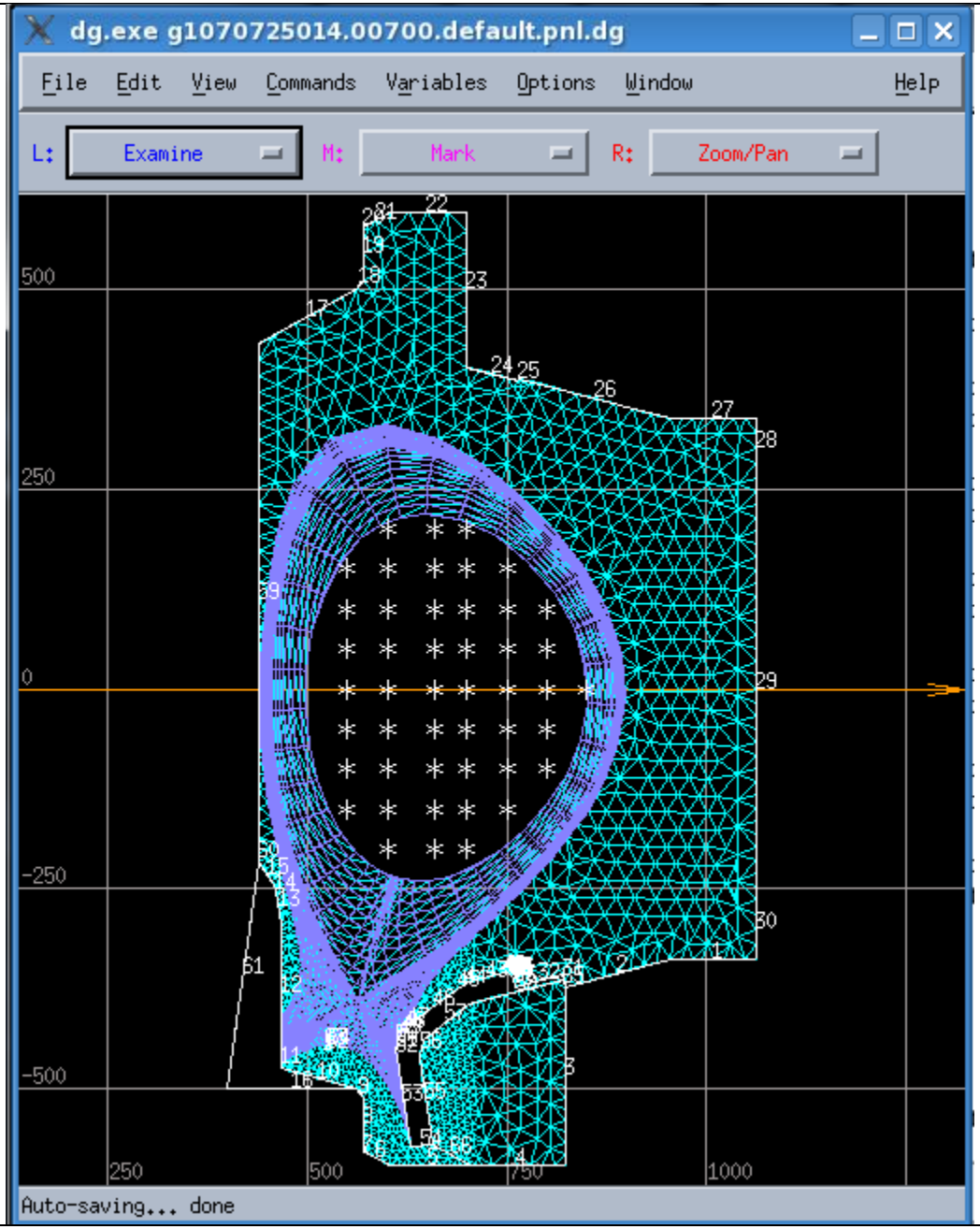
`File > Import > Template`

You can view the triangle grids by importing them as templates, using the template file created during the two conversion steps above. If you are happy with them, you are done! Otherwise, you may wish to modify the triangle size and/or the mesh refinement regions. Also, you can move or create wall vertices, since those vertices act as anchors for the triangulation.

To close DG:

`File > Save`

`File > Exit program`



Quitting Triang `q` (or `<Enter>`) Congratulations, you've got your calculation grid!	Help, Uinp, B2ag, Eirene, Tria, triaGeom, Plot, View, Store, Convert, List, Remove, reMap, Inquire, Quit ? (q)q

Annex: Aargh… `triang` failed. What can I do?

Golden Rule 1: Do **NOT** write to Xavier immediately…

Golden Rule 2: Eirene questions should go to Jülich too, not just ITER, if they refer to code outside the coupling routines.

First, you can use the `(P)lot` command to visualize the current state of the triangulation to get and idea of where and why the problem occurs. The first thing to try is to run with larger triangles (change `TRIA-EIRENE parameters` in the DivGeo model, save and output it, then re-run Uinp and the triangulation). This method is usually effective if the triangulation has strayed outside of the desired domain but the contours are properly defined and closed. If this fails, then see below…

The `Eirene` step may not produce the `fort.78` file. Usually, this will happen because the list of valid Eirene wall segments is not forming closed contours when merged with the grid outline. Check the DG model to verify that the PFR and SOL edges are properly defined (as the wall elements in contact with the grid corners) in the target specifications. Also check the contour index settings from `TRIA-EIRENE parameters` in the DG model to make sure they form complete closed contours with the right contour index sign (negative if we are triangulating inside the contour, positive if we are triangulating outside the contour), not forgetting that the contours will be completed by the Carre grid outline. Also check that the surface normal of these walls are pointing **away** from the region to triangulate. Then, check the `eirtria.log` file for lines like the following:

```
P1 FOR SEGMENT   9 LINKED TO INNER LEFT TARGET
P2 FOR SEGMENT  73 LINKED TO OUTER LEFT TARGET
P2 FOR SEGMENT  21 LINKED TO INNER RIGHT TARGET
P1 FOR SEGMENT  25 LINKED TO OUTER RIGHT TARGET
```

These lines indicate which Eirene wall segments are modified to fit the grid corners. The segment numbering is the Eirene numbering, which does not follow exactly the DG element numbering (because of the elements tagged as “not for Eirene” in DG are, of course, not in the Eirene description!). The correspondence is given in the output file, starting after the line:

```
*** 3B. DATA FOR ADDITIONAL SURFACES
```

In case of doubt, you can look inside the Eirene input files, and, in block 3B which provides the wall segment descriptions, you will find blocks like these:

```
*  73 :  77  - target edge (to be modified by GEOUSR)
2.00000E+00 1.00000E+00 1.00000E+00 1.00000E-05
      1     1     0     0     0     1     0     0     0     0
1.43566E+02 8.73624E+01-1.00000E+20 1.44164E+02 8.38195E+01 1.00000E+20
SURFMOD_2_CARBON_SPT_1160K
```

The first (title) line begins with the Eirene wall segment number, followed by the DG wall element number, and possibly additional text (like here a label to indicate this is a wall that was tagged in DG as a target edge). The fourth line gives you the coordinates (in centimeters!) of the endpoints of the wall elements, as (X1,Y1,Z1,X2,Y2,Z2).

Another set of important informational messages are of the form:

```
geousr_biased: segment            5   is turned off
geousr_biased: segment            4   is turned off
geousr_biased: segment            3   is turned off
geousr_biased: segment            2   is turned off
geousr_biased: segment            1   is turned off
geousr_biased: segment           71   is turned off
geousr_biased: segment           70   is turned off
geousr_biased: segment           69   is turned off
```

…

These messages tell you which of the Eirene wall segments have been turned off and replaced by the grid edges at the targets. I.e, they indicate which wall segments Eirene deduced as being completely wetted by the plasma at the targets.

Check that these wall segments are indeed the correct ones that you expect. You may notice that the wall segments being turned off are all the other wall segments from the contour, as opposed to those actually being wetted by the plasma. This means that Eirene is following the contour in the wrong direction, either because the surface normals are not pointing away from the plasma, or because the targets are not specified in the proper order (see the table on page 16).

`tria` can fail for a variety of reasons. The most common are:

- The contour of the region to triangulate is not closed, or
- The contour to the region to triangulate is self-intersecting, or
- The contour of the region to triangulate is not being travelled in the correct direction, and
- The triangle size has gone done to zero, and the program is stuck in an infinite loop.

Such cases can be detected in a variety of ways. First, it is useful to plot the Eirene contours, as explained in page 46, directly during the Eirene preparatory step. You can also use your own home-made plotting routines to plot the contents of the `tria.in` or `fort.78` files. Another useful trick is to use the DG model to create points at the locations indicated in the `tria.in` file to make sure thate the contour is being travelled in the direction you want, i.e. with the domain to triangulate to the **left**.

Let's look at the `tria.log` output file. If the contour is self-intersecting, you will see a message at the end of the file like this:

```
chkslfxs finished. nsx=           2
Intersections (x,y [mm]):
1.756445E+03   9.858736E+02
1.432138E+03   7.753270E+02
```

The `nsx` number is the number of self-intersections detected by `tria` in the contours provided by the `tria.in` file, followed by their location. Again, it is then useful to create points in the DG model at these locations to locate them

accurately. Two types of self-intersections are common: near the grid corners, or near tangency points at limiting surfaces. In the first case, check carefully that your PFR and SOL edges in the target specifications are correct, in particular that they are indeed the ones corresponding to the Carre mesh file specified in the `b2ag.dat` file (check it is the same as the one currently displayed in your DG model!). In the second case, consider adding an additional strucuture to push the last flux surface a little away from the tangency point, as sometimes discretization errors in the contours will create self-intersections that should not really be there.

If `tria` fails because it is attempting to triangulate infinite space because it has stepped through a hole in the contour or started on the wrong side of the contour in the first place, the error message you will get on the screen is usually:
`error in complete`
or the code will be stuck in an infinite loop and never terminate.
In such cases, you can look at the output file `curtrngl.dat` which gives out the current triangle `tria` was working on where the error occurred, and `triangles.dat` which gives you the location of the vertices of all the triangles created by `tria` (in cm). You can then plot these triangles in your DG model. If you notice that the triangle size is vanishingly small, this is usually due to a very small initial wall segment somewhere in your model (often, it can be one of the modified Eirene wall segments next to the grid edges, if those come very close to a segment endpoint). In this case, modify your DG model so that the guilty wall element is longer. To do this, you may move the endpoint of the wall element considered, or join it with an adjacent element, or add (or lengthen) a limiting "not-for-Eirene" element to push the grid corner away from the element endpoint (in which case you will need to rebuild a grid, starting back at the "`Rebuild Carre objects`" and `carre` scripts steps). If the triangles are of a correct size but outside the region you meant to triangulate, check the usual suspects: the `TRIA-EIRENE parameters` indices form complete closed contours, with the correct signs, the surface normals point away from the plasma, the target definitions are done in the right order, and the PFR and SOL edges are properly defined.

You can also force `tria` to stop after a certain number of triangles by using the `tr(A)p` function in the `triang` script (type `a` and then give the number of triangles to do before you want the program to stop, type `a` and then

<Enter> again to remove the trap). On some systems, you can then use the (P)lot option within triang to see the current status of the triangulation, but this does not work on all distributions (debugging help welcome!).

If the failure happened during the triaGeom step, you can use the (I)nquire function to get a print-out of the triangle data for the triangle containg a given point for which you supply the 2 coordinates (in mm, separated by a space), or directly the desired cell number.

Now, if all the above has failed, then you can write me for help!