Core-Edge Transport Coupling Via Manual Intervention

D. Coster, H.-J. Klingshirn

2011-12-31

Contents

List of Figures

List of Tables

Listings

1 Introduction

The core is usually simulated with 1D transport codes, and the edge with 2d transport codes. In this work, we explore coupling the core, simulated with the ETS, and the edge, simulated with SOLPS5-B2, for the steady state problem — *i.e.* to ensure self-consistency between the 2 simulated domains.

The basic procedure involves:

1. create a consistent grid for the ETS and SOLPS
2. alternate running the 2 codes, using the output from the one code as the input to the other, until converged

2 Creating the grid

We choose the matching surface between the two codes to be at 95% of the normalized polidal flux.

```
echo "import aug_eq; aug_eq.plot(shot=17151, time=2.5, file='17151_2500_psi_cut=0_95.png', cut=0.95)
Axis:   [ 1.74423304  0.12186872] [ 1.05642062]
Xpt:    [ 1.44091878 -0.94590132] [-0.10083864]
cut, psi_cut:  0.95 [-0.04297568]
Carre pntrat (0.95):  0.167573692185
```

This stored the data as run 17151, shot 0 (see `Examples/case_17151_0/README`).

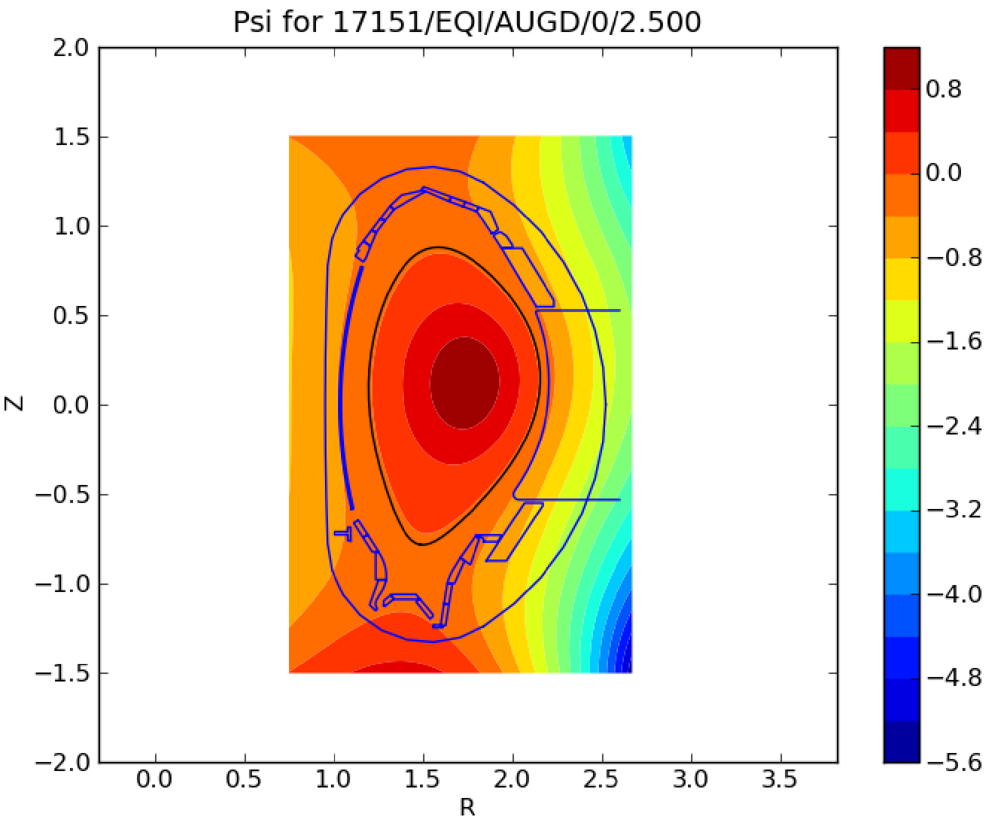


Figure 1: AUG shot 17151 at 2.5s showing the 95% flux surface.

Listing 1: Sample carre.dat input file for creating a 96x36 grid

```
$parameters
repart    =     2
nrelax    =   200
relax     =    0.5
pasmin    =    1.0E-03
rlcept    =    1.0E-06
pntrat    =    0.167573692185
nptseg(1) =     25
deltp1(1) =    2.0e-02
deltpn(1) =    5.0e-03
nptseg(2) =     25
deltp1(2) =    2.0e-02
deltpn(2) =    5.0e-03
nptseg(3) =     49
deltp1(3) =    5.0e-02
deltpn(3) =    5.0e-02
npr(1) =     19
deltr1(1) =   -2.0E-03
deltrn(1) =   -2.0E-03
npr(2) =     19
deltr1(2) =    1.0E-03
deltrn(2) =    1.0E-03
npr(3) =     19
deltr1(3) =    1.5E-03
deltrn(3) =    3.0E-03
tgarde(1) =    1.0000000E-01
tgarde(2) =    5.0000000E-02
$end
```

2.1 SOLPS5-B2 Grid

The B2 grid was created with the grid generator CARRE in the ITM-CARRE version: all input data is read from CPOs, but the code parameters are still read from a text file (`carre.dat`). ITM-CARRE is currently run using a Fortran wrapper routine that does UAL I/O. The CPO parameters (shot, run,...) are currently read from namelist files

Prerequisites for grid generation (information in brackets gives the CPOs that were used for this exercise):

- An equilibrium CPO (17151/0/2.5/equilibrium/coster/aug/4.09a):
  - holding Psi on a structured R,Z grid
  - with X- and O-Point information
  - Namelist file: `ual.namelist.equilibrium`
- A limiter CPO (17151/0/limiter/klingshi/aug/4.09a):
  - Target structures (i.e. intersected by the separatrix) must be closed polygons.
  - All other structures can be open.
  - Namelist file: `ual.namelist.limiter`
- A carre.dat inputfile holding the code parameters, c.f. listing 2.1.

ITM-Carre produces an edge CPO as output, with CPO parameters as specified in the namelist file `ual.namelist.edge` (17151/3/2.5/edge/klingshi/aug/4.09a). The CPO contains only the grid, stored in `edge%grid` in the general grid description. It is a 2d quadrilateral grid, one unstructured space. ITM-Carre also creates normal output files as used in the SOLPS grid generation workflow. Currently these are used to pass the grid to B2.

2.2 ETS Grid

The ETS starting grid is based on running HELENA on the above equilibrium. This is done in `Examples/case_17151_4/` with the command

```
make run_helena_cpo ARGS="17151 0 4 2.5"
```

and stored as shot 17151, run 4.

2.3 Combined Grid

The combined grid can be seen in figure 2.

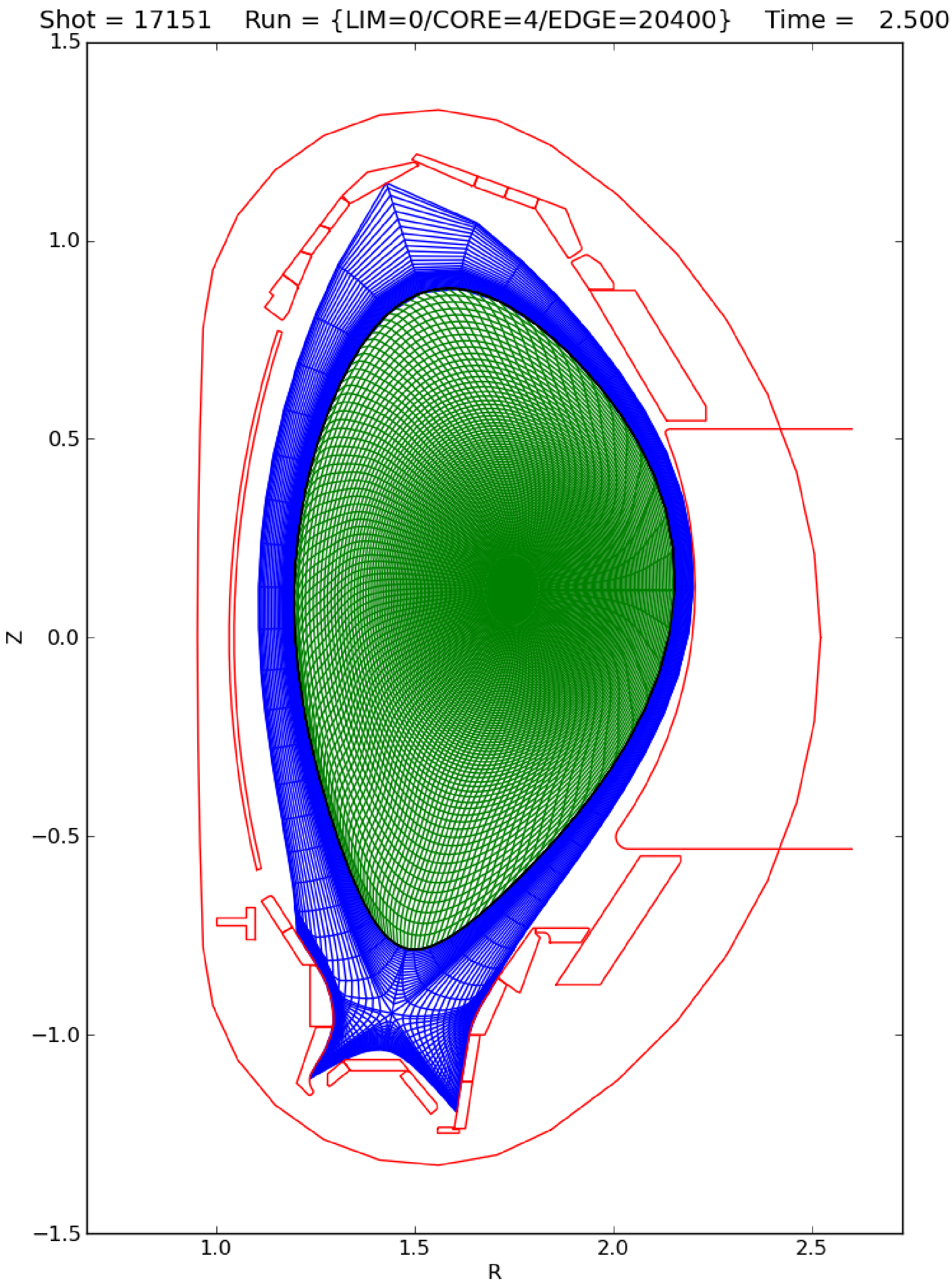


Figure 2: Combined core and grids for AUG shot 17151.

3 Core-Edge Coupling

For the Core-Edge coupling exercise, two cases are consisdered. A low power D only case, and then a higher power case with main ions D and He, together with C as an impurity.

The ETS starting points were based on `PREPARE_INPUT_CPOS_XML/prepare_input_cpos_AUG_17151_6.xml` and `PREPARE_INPUT_CPOS_XML/prepare_input_cpos_AUG_17151_7.xml` and produced with

```
make prepare_input_cpos ARGS=PREPARE_INPUT_CPOS_XML/prepare_input_cpos_AUG_17151_6.xml
make prepare_input_cpos ARGS=PREPARE_INPUT_CPOS_XML/prepare_input_cpos_AUG_17151_7.xml
```

in the ETS directory.

The relevant directories for the coupling are all under `ETS/trunk/Core-Edge/`:

PYTHON contains the python code for the coupling

ETS contains the ETS runs

SOLPS contains the SOLPS runs

A more detailed version of the procedure is as follows

1. set up an initial SOLPS case
2. in the ETS directory, set up an ETS case to run from the starting condition described previously in section 2, and run to stationarity
3. in the PYTHON directory, run the "coupling.core_2_edge" command
4. in the SOLPS directory, modify "b2.boundary.parameters" and "b2_ual_write.dat", and run "b2mn.exe", and then run "b2_ual_write.exe"
5. in the PYTHON directory, run the "coupling.edge_2_core" command
6. in the ETS directory, update "eq_ets.xml" and run the ETS again
7. repeat from 3 above until converged

To speed up the ETS cases, a lower poloidal resolution was used. The actual grids for the coupling cases are shown in figure 3.

3.1 Coupling methodology

The basic idea used in the coupling is that in one direction a value is passed, and then a flux is passed back.

To set this up, the python scripts use a number of arrays

SPECIES a list of length NS, the number of species considered in B2, containing 0, 1, .., NS-1

dest_type a list of length NS containing a flag consisting of

- **0** there is no matching species on the core side
- **1** the species maps to a species in COREPROF, the index given by "dest_index" (see below)
- **2** the species maps to a species in COREIMPUR, the indices given by "dest_index" (see below)
- **3** the species maps to a species in CORENEUTRAL [not yet coded

dest_index a 2D array of length NS by 2 giving the ion species number in coreprof ("dest_type" == 1), or the impurity species and charge state ("dest_type" == 2)

te_flag a flag that is either 0 or 1

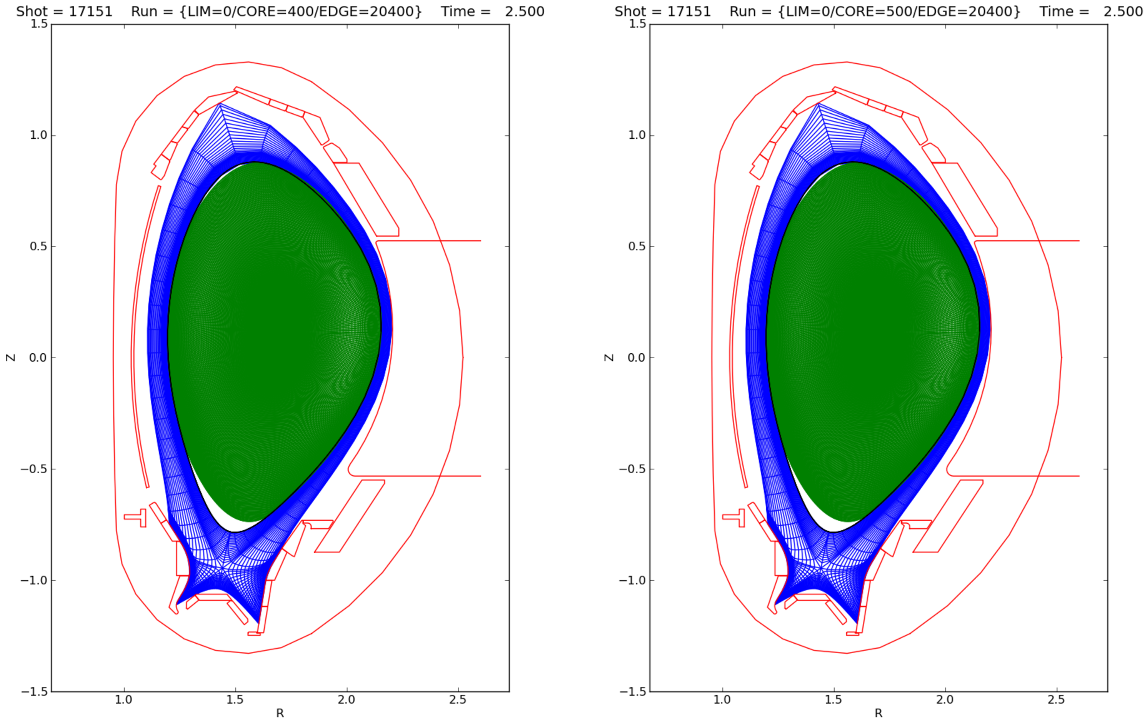


Figure 3: Combined core and grids for AUG shot 17151 as actually used in the core-edge coupling.

- **0** the value of the electron temperature at the interface boundary is passed from SOLPS to the ETS, and the value of the electron energy flux at the interface boundary is passed back from the ETS to SOLPS
- **1** the value of the electron energy flux at the interface boundary is passed from SOLPS to the ETS, and the value of the electron temperature at the interface boundary is passed back from the ETS to SOLPS

ti_flag a flag that is either 0 or 1

- **0** the value of the ion temperature at the interface boundary is passed from SOLPS to the ETS, and the value of the ion energy flux at the interface boundary is passed back from the ETS to SOLPS
- **1** the value of the ion energy flux at the interface boundary is passed from SOLPS to the ETS, and the value of the ion temperature at the interface boundary is passed back from the ETS to SOLPS

na_flag an array of flags of length NS containing either a 0 or 1

- **0** the value of the ion density at the interface boundary is passed from SOLPS to the ETS, and the value of the ion particle flux at the interface boundary is passed back from the ETS to SOLPS
- **1** the value of the ion particle flux at the interface boundary is passed from SOLPS to the ETS, and the value of the ion density at the interface boundary is passed back from the ETS to SOLPS

The data-flow is shown for the case of 17151/403:

1. ETS starts from 17151/10402 and produces 17151/403
2. “coupling.core_2_edge” reads 17151/403 and writes to standard output the set of boundary conditions for SOLPS5-B2 that need to be added (by hand) to “b2.boundary.parameters”
3. 17151/20403 is written by “b2_ual_write.exe” based on the SOLPS run
4. “coupling.edge_2_core” reads 17151/403 and 17151/20403 and produces 17151/10403

3.2 D case

The physics case is a D only plasma, with SOLPS maintaining an upstream separatrix electron density of $3 \times 10^{19} m^{-3}$ through a D gas-puff from the "wall".

Densities/temperatures are passewd from SOLPS to the ETS, and the ETS returns particle and energy fluxes. Neutrals are ignored in the core, and SOLPS uses a zero-flux boundary condition for the neutrals at the core boundary.

The ETS cases form a progression in

ETS/case_17151_400 initial ETS run, started from 17151/6 and producing 17151/400

ETS/case_17151_401 continuation ETS run starting from 17151/10400 and producing 17151/401

ETS/case_17151_402 continuation ETS run starting from 17151/10401 and producing 17151/402

ETS/case_17151_403 continuation ETS run starting from 17151/10402 and producing 17151/403

ETS/case_17151_404 continuation ETS run starting from 17151/10403 and producing 17151/404

ETS/case_17151_405 continuation ETS run starting from 17151/10404 and producing 17151/405

ETS/case_17151_406 continuation ETS run starting from 17151/10405 and producing 17151/406

ETS/case_17151_407 continuation ETS run starting from 17151/10406 and producing 17151/407

The SOLPS cases form a progression in

17151_1.6MW_nesepm=3.0e19_D=0.4_chi=1.6_standalone.000 initial SOLPS run used to provide a starting condition for the 400 case below

17151_1.6MW_nesepm=3.0e19_D=0.4_chi=1.6_standalone.400 SOLPS5-B2 run with an initial plasma state from 000 above, with "b2.boundary.parameters" modified to contain the information from "coupling.core_2_edge" based on the CPO output from "ETS/case_17151_400"; after completion "b2_ual_write.exe" produces 17151/20400

17151_1.6MW_nesepm=3.0e19_D=0.4_chi=1.6_standalone.401 SOLPS5-B2 run with an initial plasma state from 400 above, with "b2.boundary.parameters" modified to contain the information from "coupling.core_2_edge" based on the CPO output from "ETS/case_17151_401"; after completion "b2_ual_write.exe" produces 17151/20401

17151_1.6MW_nesepm=3.0e19_D=0.4_chi=1.6_standalone.402 SOLPS5-B2 run with an initial plasma state from 401 above, with "b2.boundary.parameters" modified to contain the information from "coupling.core_2_edge" based on the CPO output from "ETS/case_17151_402"; after completion "b2_ual_write.exe" produces 17151/20402

17151_1.6MW_nesepm=3.0e19_D=0.4_chi=1.6_standalone.403 SOLPS5-B2 run with an initial plasma state from 402 above, with "b2.boundary.parameters" modified to contain the information from "coupling.core_2_edge" based on the CPO output from "ETS/case_17151_403"; after completion "b2_ual_write.exe" produces 17151/20403

17151_1.6MW_nesepm=3.0e19_D=0.4_chi=1.6_standalone.404 SOLPS5-B2 run with an initial plasma state from 403 above, with "b2.boundary.parameters" modified to contain the information from "coupling.core_2_edge" based on the CPO output from "ETS/case_17151_404"; after completion "b2_ual_write.exe" produces 17151/20404

17151_1.6MW_nesepm=3.0e19_D=0.4_chi=1.6_standalone.405 SOLPS5-B2 run with an initial plasma state from 404 above, with "b2.boundary.parameters" modified to contain the information from "coupling.core_2_edge" based on the CPO output from "ETS/case_17151_405"; after completion "b2_ual_write.exe" produces 17151/20405

17151_1.6MW_nesepm=3.0e19_D=0.4_chi=1.6_standalone.406 SOLPS5-B2 run with an initial plasma state from 405 above, with "b2.boundary.parameters" modified to contain the information from "coupling.core_2_edge" based on the CPO output from "ETS/case_17151_406"; after completion "b2_ual_write.exe" produces 17151/20406

17151_1.6MW_nesepm=3.0e19_D=0.4_chi=1.6_standalone.407 SOLPS5-B2 run with an initial plasma state from 406 above, with "b2.boundary.parameters" modified to contain the information from "coupling.core_2_edge" based on the CPO output from "ETS/case_17151_407"; after completion "b2_ual_write.exe" produces 17151/20407

The PYTHON cases were performed with

```
import coupling
import numpy

# Assume D0, D+
dest_type = numpy.array([0,1])
dest_index = numpy.array([[0, 0],[0, 0]]).transpose()
SPECIES = numpy.arange(2)
te_flag = 0; ti_flag = 0 ; na_flag = numpy.array([0,0])
```

and then a progression of

```
shot = 17151 ; run = 400 ; time=100
coupling.core_2_edge(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
coupling.edge_2_core(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
shot = 17151 ; run = 401 ; time=200
coupling.core_2_edge(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
coupling.edge_2_core(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
shot = 17151 ; run = 402 ; time=300
coupling.core_2_edge(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
coupling.edge_2_core(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
shot = 17151 ; run = 403 ; time=400
coupling.core_2_edge(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
coupling.edge_2_core(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
shot = 17151 ; run = 404 ; time=500
coupling.core_2_edge(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
coupling.edge_2_core(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
shot = 17151 ; run = 405 ; time=600
coupling.core_2_edge(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
coupling.edge_2_core(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
shot = 17151 ; run = 406 ; time=700
coupling.core_2_edge(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
coupling.edge_2_core(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
shot = 17151 ; run = 407 ; time=800
coupling.core_2_edge(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
coupling.edge_2_core(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
```

where the "core_2_edge" run followed the ETS run, and the "edge_2_core" followed the SOLPS run.

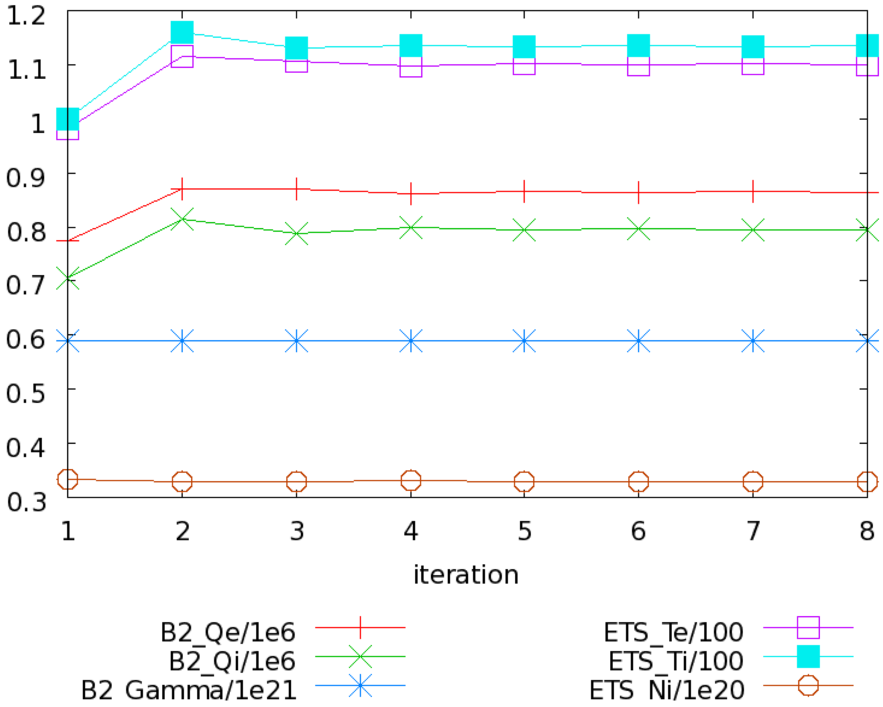


Figure 4: Interface quantities passed to B2 and the ETS, D case.

B2_Qe	**B2_Qi**	**B2_Gamma**	**ETS_Te**	**ETS_Ti**	**ETS_Ni**
774534.944262	705999.944493	5.90317525081e+20	98.2324058893	99.98517951	3.34120635e+19
871330.391735	815549.777211	5.90629479835e+20	111.446502283	115.93981362	3.29450273e+19
870488.226183	788779.534689	5.90503377822e+20	110.557492272	113.2012409	3.29063815e+19
860742.619169	798502.826479	5.90483381736e+20	109.771254433	113.50434584	3.30098752e+19
866667.358018	793566.593866	5.90477049406e+20	110.217911871	113.36355977	3.29783290e+19
863320.985334	796280.435897	5.90474751578e+20	109.990928833	113.46510822	3.29841743e+19
865033.609351	794675.996054	5.90474623423e+20	110.095744098	113.38662477	3.29829469e+19
864076.587323	795588.29576	5.90474142013e+20	110.038343389	113.43349437	3.29837125e+19

Table 1: Interface quantities passed between the codes. D only case, 9 poloidal points in Helena.

Run	**ETS**	**SOLPS**
400	8440.92 (1s)	314.903 (1s)
401	12979.32 (1s)	305.405 (1s)
402	7801.67 (1s)	282.936 (1s)
403	7350.80 (1s)	278.004 (1s)
404	7065.27 (1s)	275.624 (1s)
405	7144.13 (1s)	257.578 (1s)
406	7119.18 (1s)	2783.33 (10s)
407	50236.63 (10s)	2705.32 (10s)

Table 2: Timings for ETS and SOLPS: cpu used (and physics time), 9 poloidal points in Helena.

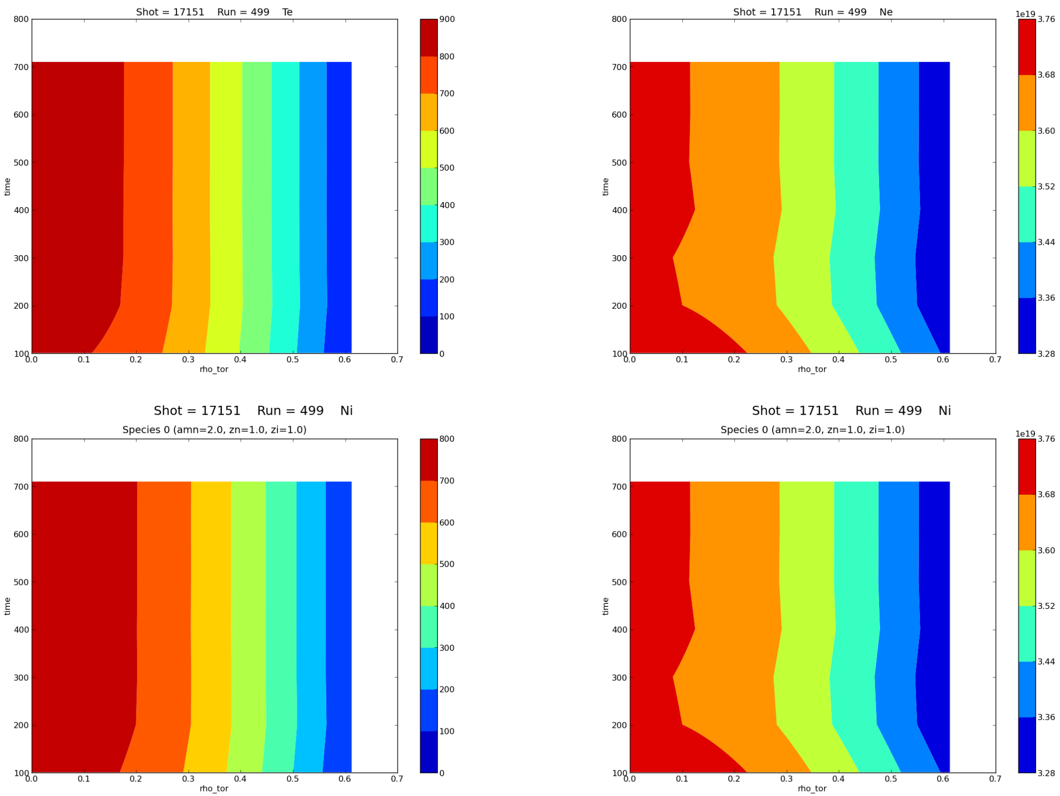


Figure 5: Plots of densities and temperatures in the core at the end of each ETS run for the D case.

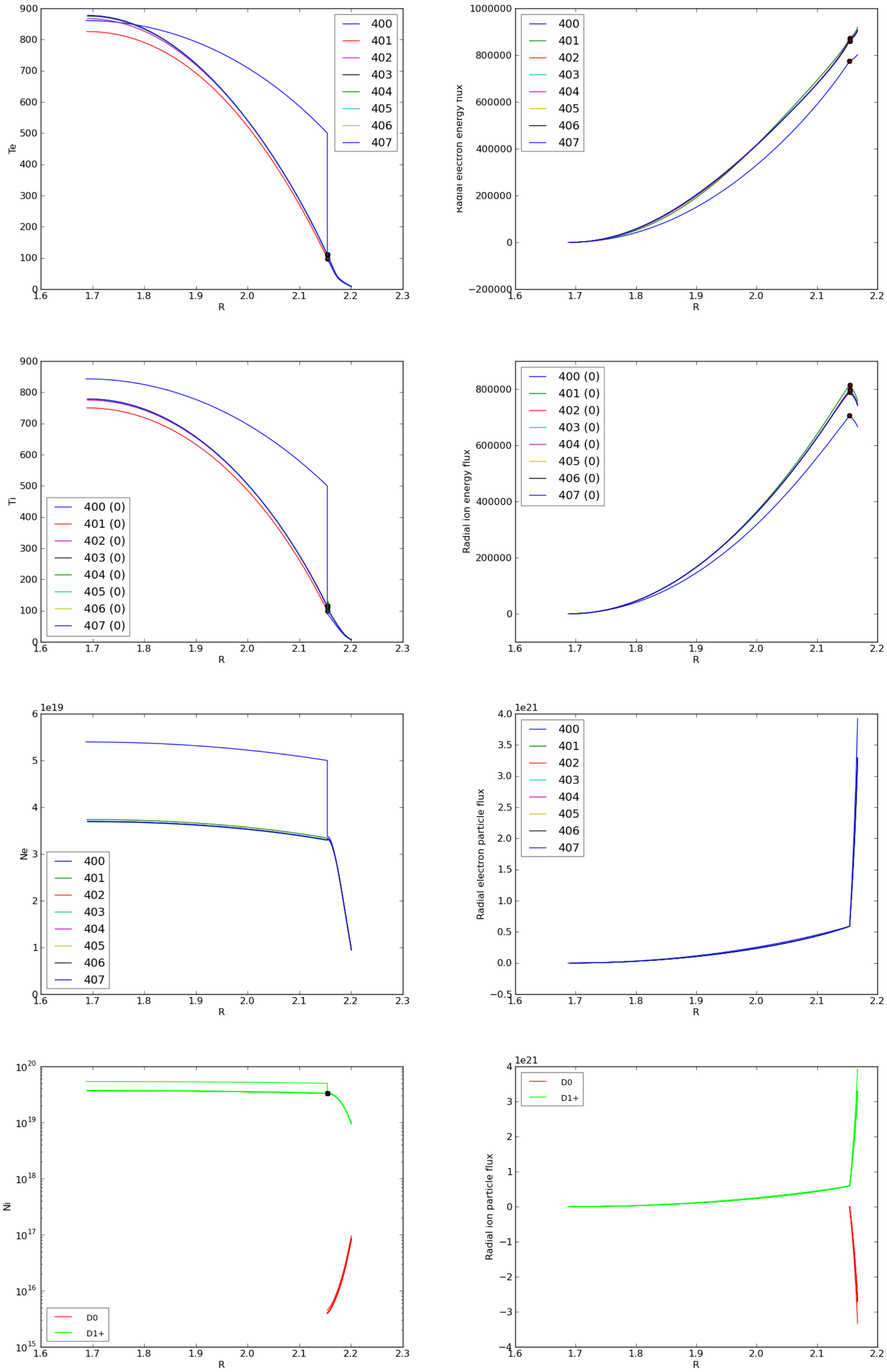


Figure 6: Combined profiles, D case.

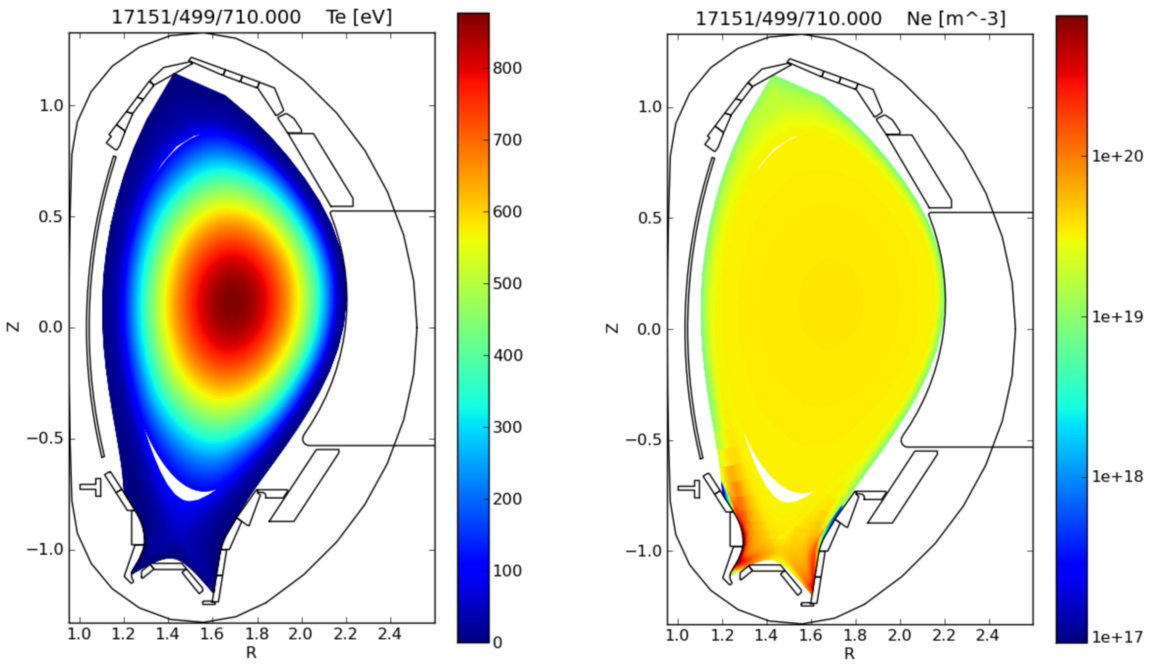


Figure 7: Plots of Te and ne for the final state of the D case.

3.3 D+He + C case

The physics case is a D+C+He plasma, with SOLPS maintaining an upstream separatrix electron density of $4 \times 10^{19} m^{-3}$ through a D gas-puff from the "wall".

Densities/temperatures are passed from SOLPS to the ETS, and the ETS returns particle and energy fluxes. Neutrals are ignored in the core, and SOLPS uses a zero-flux boundary condition for the neutrals at the interface boundary.

The ETS handles D^+ and He^{2+}, with zero flux boundary conditions for D^0, He^0 and He^{1+}. The ETS impurity code is used to treat the carbon charged states, and a zero flux density boundary condition is used for neutral carbon.

For these simulations, the densities of the ionized C charge states were passed from SOLPS to the impurity code, but SOLPS used in the return direction zero flux boundary conditions for all of the C charge states, which summed to the same C flux as that from the ETS (zero).

The ETS cases form a progression in

ETS/case_17151_500 initial ETS run, started from 17151/7 and producing 17151/500

ETS/case_17151_501 continuation ETS run starting from 17151/10500 and producing 17151/501

ETS/case_17151_502 continuation ETS run starting from 17151/10501 and producing 17151/502

ETS/case_17151_503 continuation ETS run starting from 17151/10502 and producing 17151/503

ETS/case_17151_504 continuation ETS run starting from 17151/10503 and producing 17151/504

The SOLPS cases form a progression in

17151_5MW_nesepm=4.0e19_D=0.4_chi=1.6_standalone.000 initial SOLPS run used to provide a starting condition for the 500 case below

17151_5MW_nesepm=4.0e19_D=0.4_chi=1.6_standalone.500 SOLPS5-B2 run with an initial plasma state from 000 above, with "b2.boundary.parameters" modified to contain the information from "coupling.core_2_edge" based on the CPO output from "ETS/case_17151_500"; after completion "b2_ual_write.exe" produces 17151/20500

17151_5MW_nesepm=4.0e19_D=0.4_chi=1.6_standalone.501 SOLPS5-B2 run with an initial plasma state from 500 above, with "b2.boundary.parameters" modified to contain the information from "coupling.core_2_edge" based on the CPO output from "ETS/case_17151_501"; after completion "b2_ual_write.exe" produces 17151/20501

17151_5MW_nesepm=4.0e19_D=0.4_chi=1.6_standalone.502 SOLPS5-B2 run with an initial plasma state from 501 above, with "b2.boundary.parameters" modified to contain the information from "coupling.core_2_edge" based on the CPO output from "ETS/case_17151_502"; after completion "b2_ual_write.exe" produces 17151/20502

17151_5MW_nesepm=4.0e19_D=0.4_chi=1.6_standalone.503 SOLPS5-B2 run with an initial plasma state from 502 above, with "b2.boundary.parameters" modified to contain the information from "coupling.core_2_edge" based on the CPO output from "ETS/case_17151_503"; after completion "b2_ual_write.exe" produces 17151/20503

17151_5MW_nesepm=4.0e19_D=0.4_chi=1.6_standalone.504 SOLPS5-B2 run with an initial plasma state from 503 above, with "b2.boundary.parameters" modified to contain the information from "coupling.core_2_edge" based on the CPO output from "ETS/case_17151_504"; after completion "b2_ual_write.exe" produces 17151/20504

17151_5MW_nesepm=4.0e19_D=0.4_chi=1.6_standalone.505

The PYTHON cases were performed with

```
import coupling
import numpy
```

Run	ETS	SOLPS
500	12930.66 (1s)	9051.37 (1s)
501	9025.31 (1s)	8826.7 (1s)
502	7416.69 (1s)	8770.82 (1s)
503	7277.57 (1s)	8677.77 (1s)
504	7297.45 (1s)	9005.48 (1s)
505	7457.80 (1s)	8809.32 (1s)

Table 3: Timings for ETS and SOLPS: cpu used (and physics time). D+He + C case, 9 poloidal points in Helena.

```
# Assume D0, D+, C0 .. C6+, He0 .. He2+
dest_type =  [0, 1, 0, 2, 2, 2, 2, 2, 2,, 0, 0, 1]
dest_index = numpy.array([[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
                          [0, 0, 0, 0, 1, 2, 3, 4, 5, 0, 0, 0]]).transpose()
species =  [ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11]
te_flag = 0; ti_flag = 0 ; na_flag = numpy.array([0,0,0,0,0,0,0,0,0,0,0,0])

dest_type, dest_index, SPECIES = coupling.get_coupling_destinations(17151,500,17151,20500)
```

and then a progression of

```
shot = 17151 ; run = 500 ; time=100
coupling.core_2_edge(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
coupling.edge_2_core(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
shot = 17151 ; run = 501 ; time=200
coupling.core_2_edge(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
coupling.edge_2_core(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
shot = 17151 ; run = 502 ; time=300
coupling.core_2_edge(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
coupling.edge_2_core(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
shot = 17151 ; run = 503 ; time=400
coupling.core_2_edge(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
coupling.edge_2_core(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
shot = 17151 ; run = 504 ; time=500
coupling.core_2_edge(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
coupling.edge_2_core(shot,run,time,te_flag,ti_flag,na_flag,dest_type,dest_index,SPECIES)
```

where the "core_2_edge" run followed the ETS run, and the "edge_2_core" followed the SOLPS run.

3.3.1 Discussion of interface boundary conditions for C

These simulations passed $C^{1+}..C^{6+}$ densities from SOLPS to the ETS, but passed the integrated flux over all charge states back to SOLPS (zero) which then used this as the boundary condition for all charge states.

If the individual charge states fluxes were used in SOLPS, then with boundary condition type 8, the net C flux across the boundary was 1.23172784978e+20 and with type 13, 1.2225042272e+20 (for the used approach, the net flux was -4.30361121895e+09). The main problem was for those charge states with fluxes out of the SOLPS domain into the core where the code was not able to draw enough flux. It might be possible to swap the coupling direction for the charge states with negative fluxes; some effort should be made to try to ensutre the appropriate net flux across the charge states.

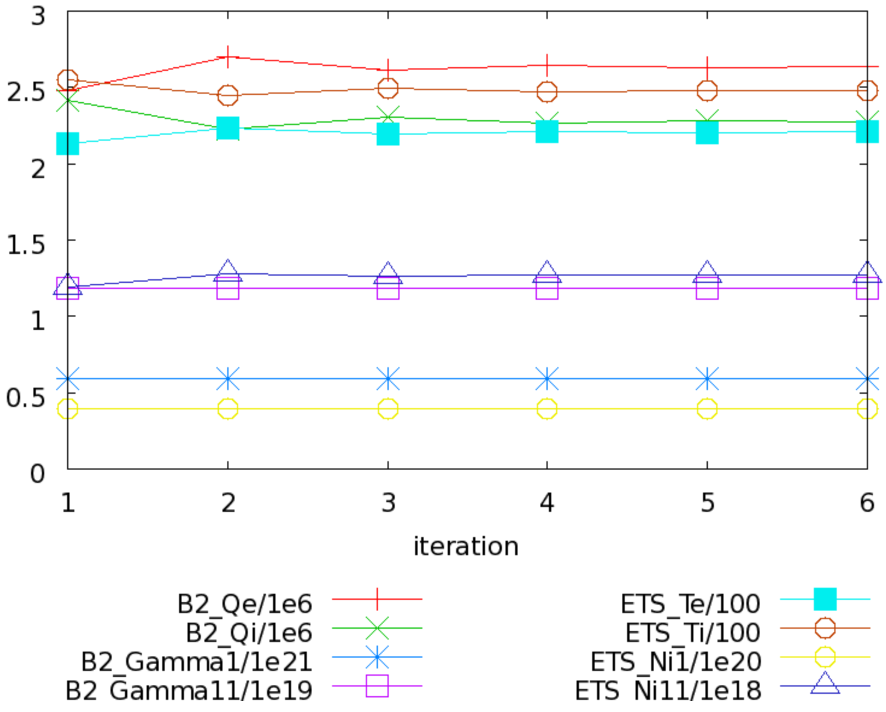


Figure 8: Interface quantities passed to B2 and the ETS, D+He + C case.

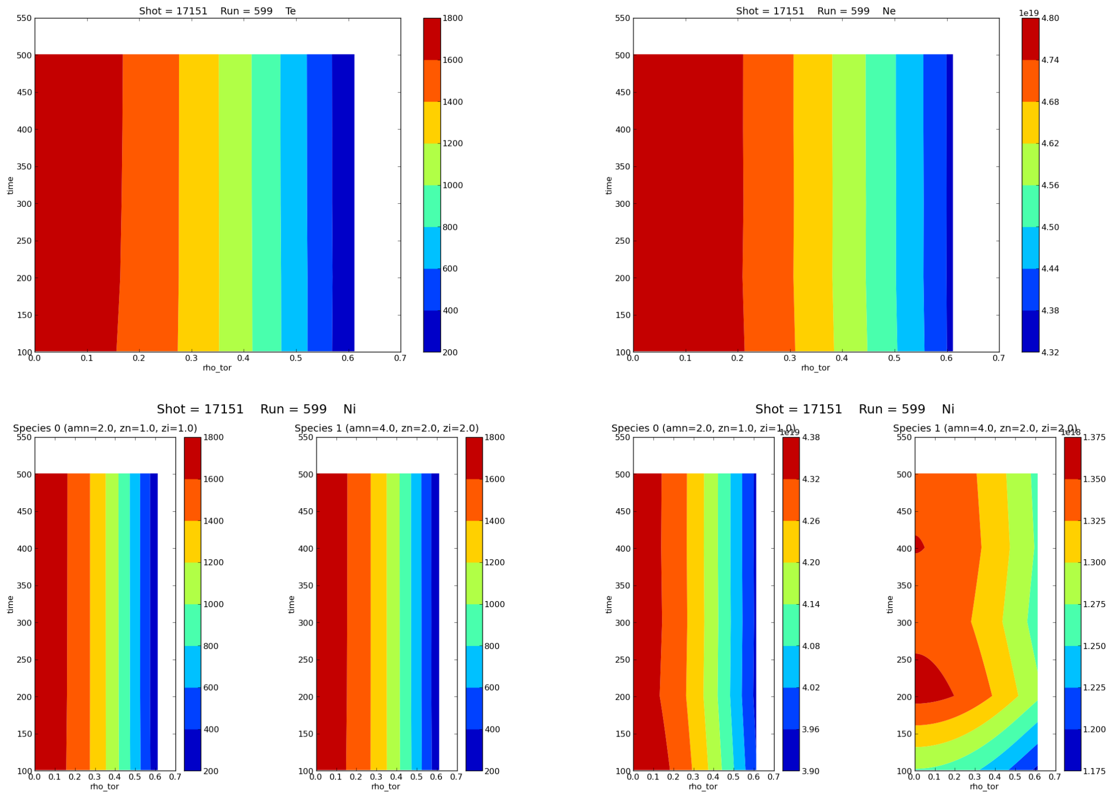


Figure 9: Plots of densities and temperatures in the core at the end of each ETS run for the D+C+He case.

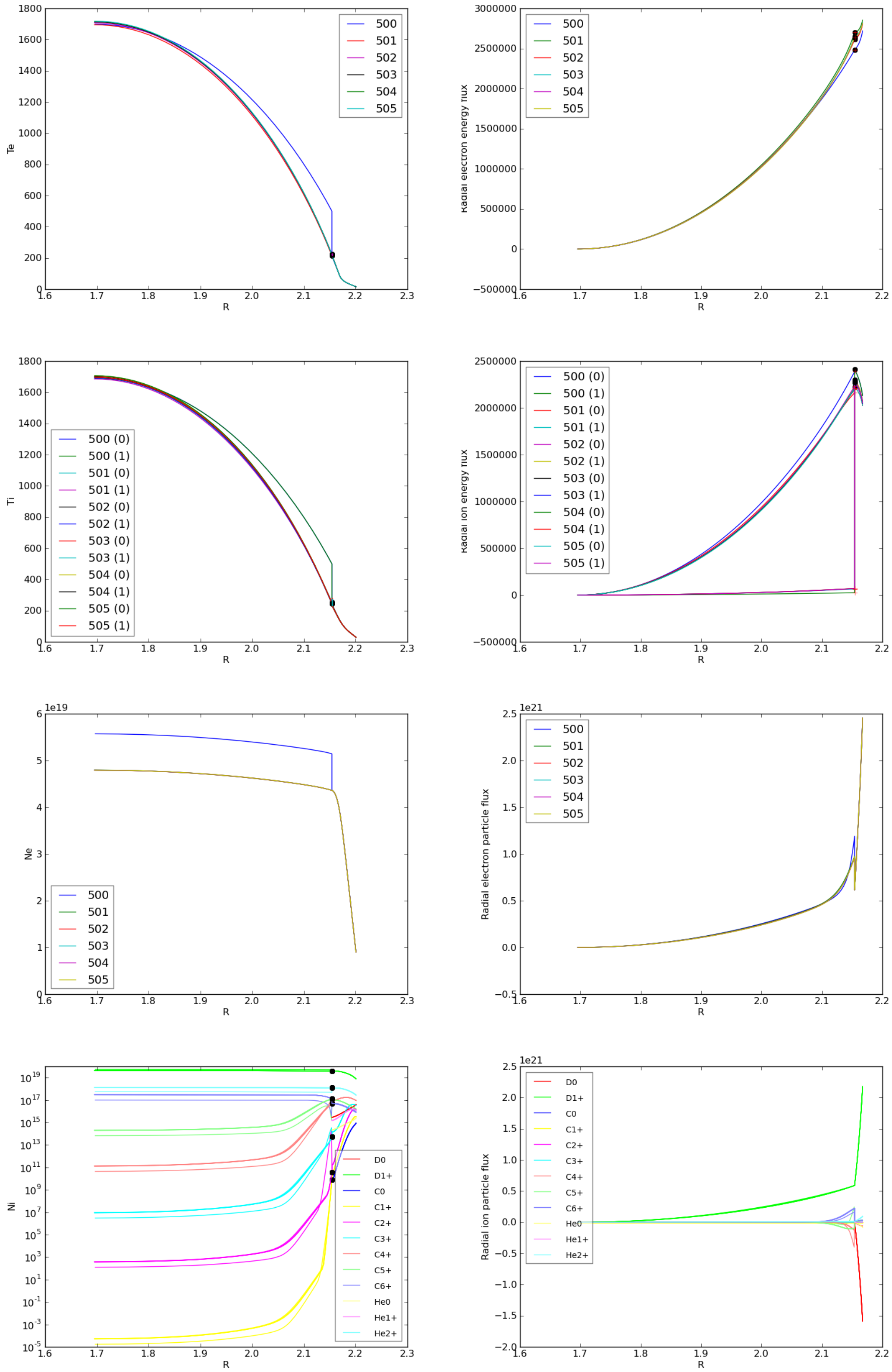


Figure 10: Combined profiles, D+He + C case.

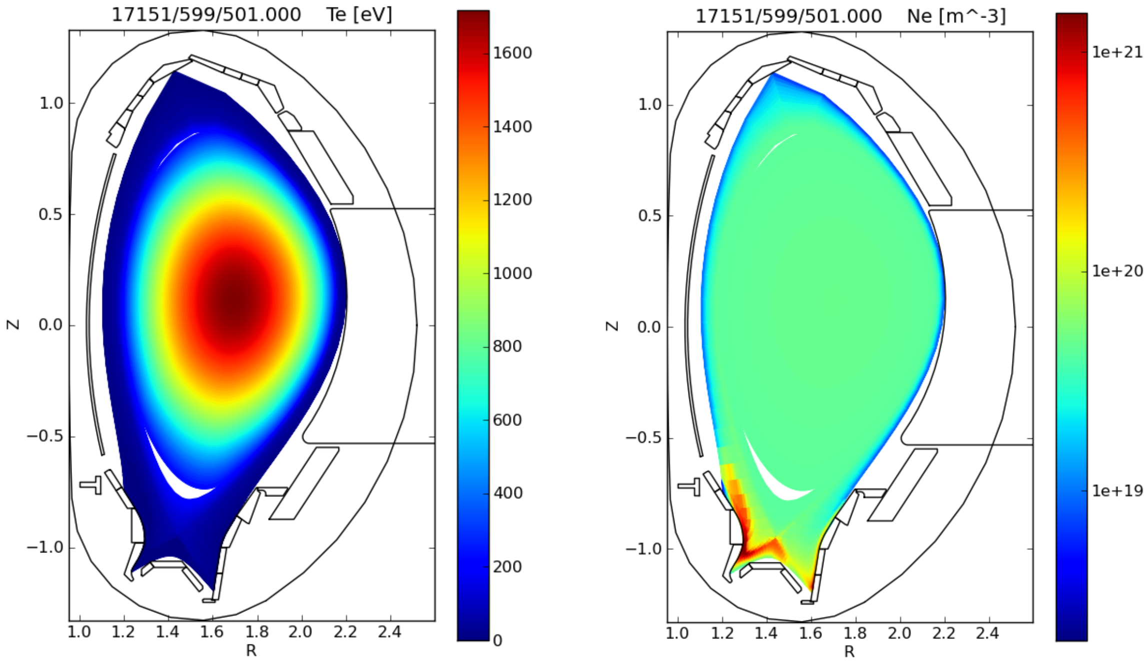


Figure 11: Plots of Te and ne for the final state of the D+He + C case. 9 poloidal points were used in Helena.

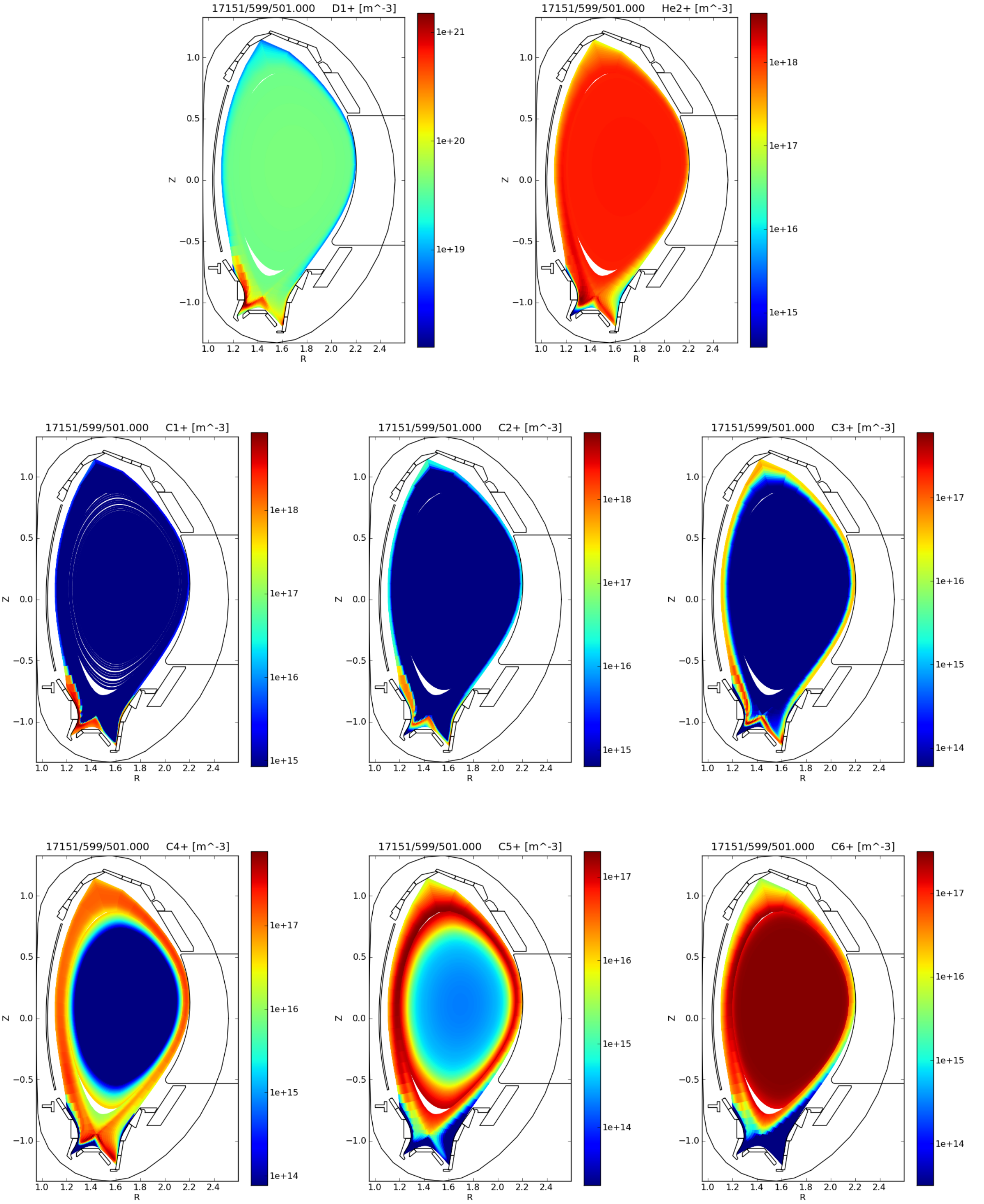


Figure 12: Plots of ion densities for the final state of the D+He + C case. 9 poloidal points were used in Helena.

The boundary condition used here in SOLPS to set the fluxes was option **8**, which is used to set a local flux density on each boundary surface element. This does not mean, though, that the density of the ions (or neutrals) is a constant. For C^{6+}, this is not the case, as can be seen in figure 13.

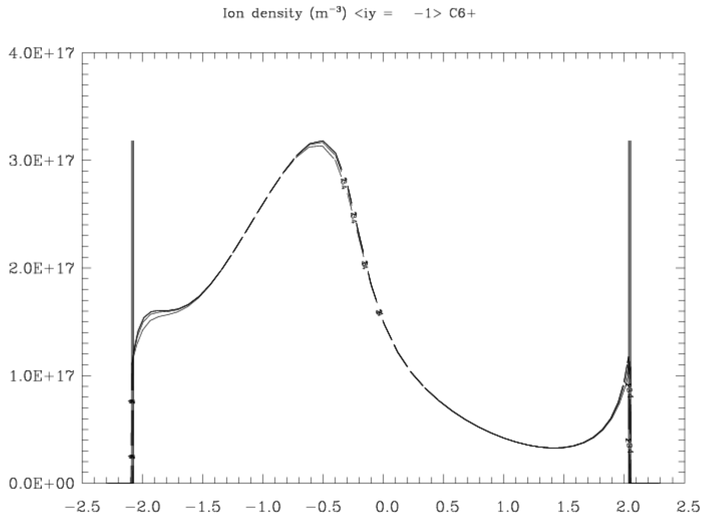


Figure 13: The density of C^{6+} calculated in SOLPS5-B2 plotted on the flux surfaces close to the interface boundary for the case where boundary fluxes were specified using option **8**.

An alternative formulation of the boundary consitions is possible in SOLPS5-B2 where an internal feedback loop is activated which varies the boundary density so that the integral flux across the boundary is the desired one (option **13**). This was used and the resultant C^{6+} densities near the interface boundary are shown in figure 14.

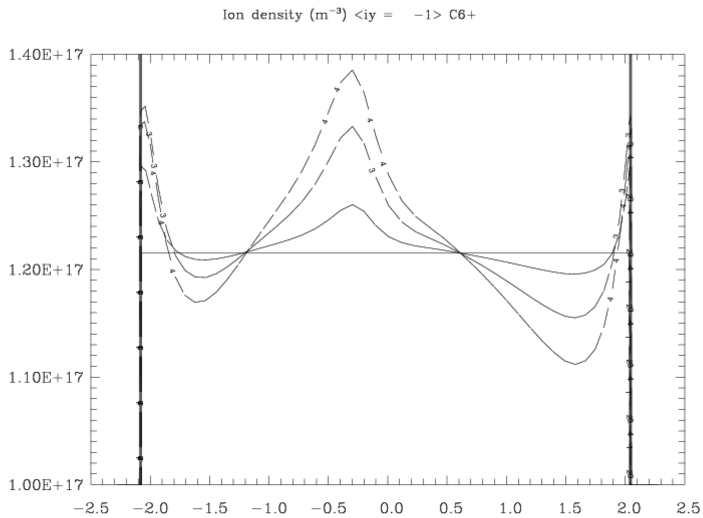


Figure 14: The density of C^{6+} calculated in SOLPS5-B2 plotted on the flux surfaces close to the interface boundary for the case where boundary fluxes were specified using option **13**.

The resultant combined plots are shown in figures 15 and 16, where the number of polodal points in

Run	ETS	SOLPS
700	8705.62 (1s)	8637.08 (1s)
701	8532.55 (1s)	8675.53 (1s)
702	8503.54 (1s)	9263.45 (1s)

Table 4: Timings for ETS and SOLPS: cpu used (and physics time). D+He + C case, 17 poloidal points in Helena.

Run	ETS	SOLPS
800	11467.61 (1s)	8878.92 (1s)
801	10670.55 (1s)	9283.28 (1s)
802	5941.46 (1s)	8844.02 (1s)

Table 5: Timings for ETS and SOLPS: cpu used (and physics time). D+He + C case, 33 poloidal points in Helena.

Helena was also increased.

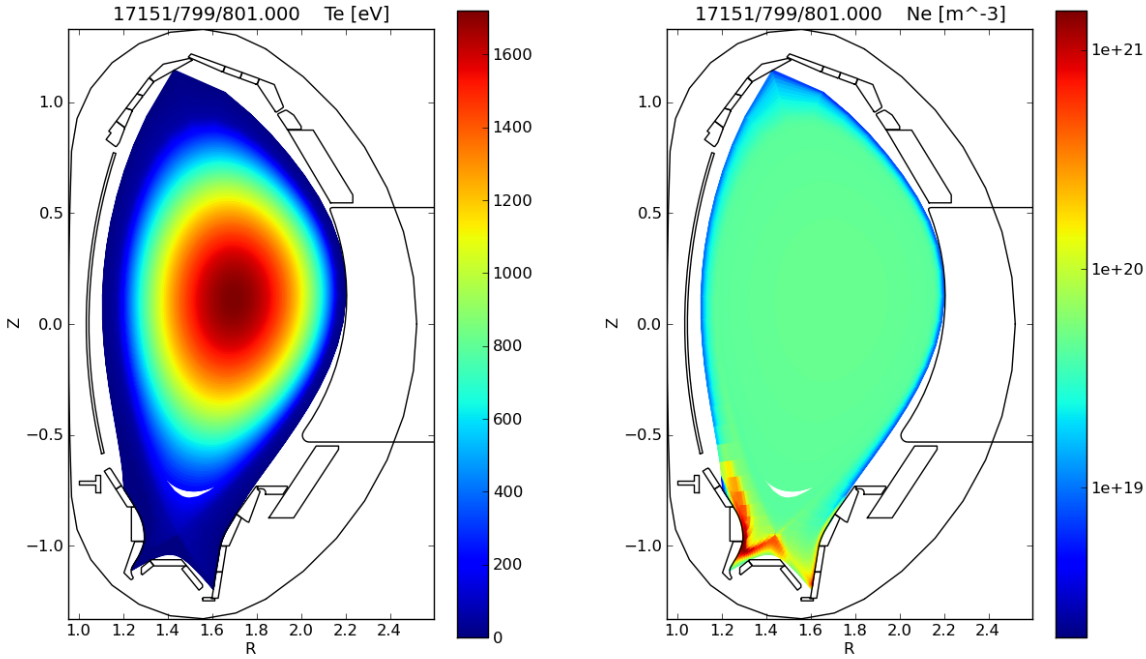


Figure 15: Plots of Te and ne for the final state of the D+He + C case, using SOLPS5-B2 option **13** for the flux boundary consitions. 17 poloidal points were used in Helena.

Figures 17 and 18 show the results with additional poloidal points in Helena.

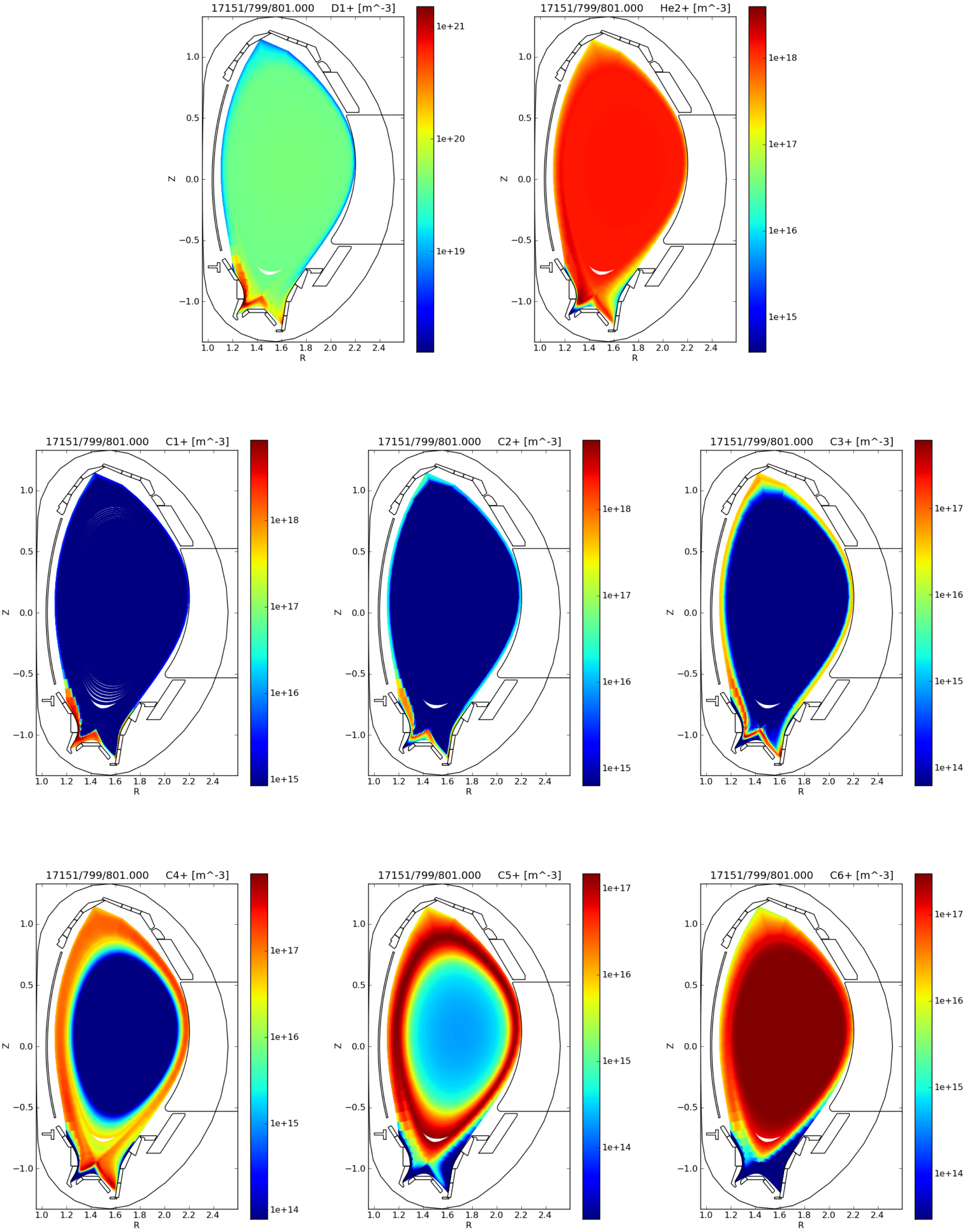


Figure 16: Plots of ion densities for the final state of the D+He + C case, using SOLPS5-B2 option **13** for the flux boundary consitions. 17 poloidal points were used in Helena.

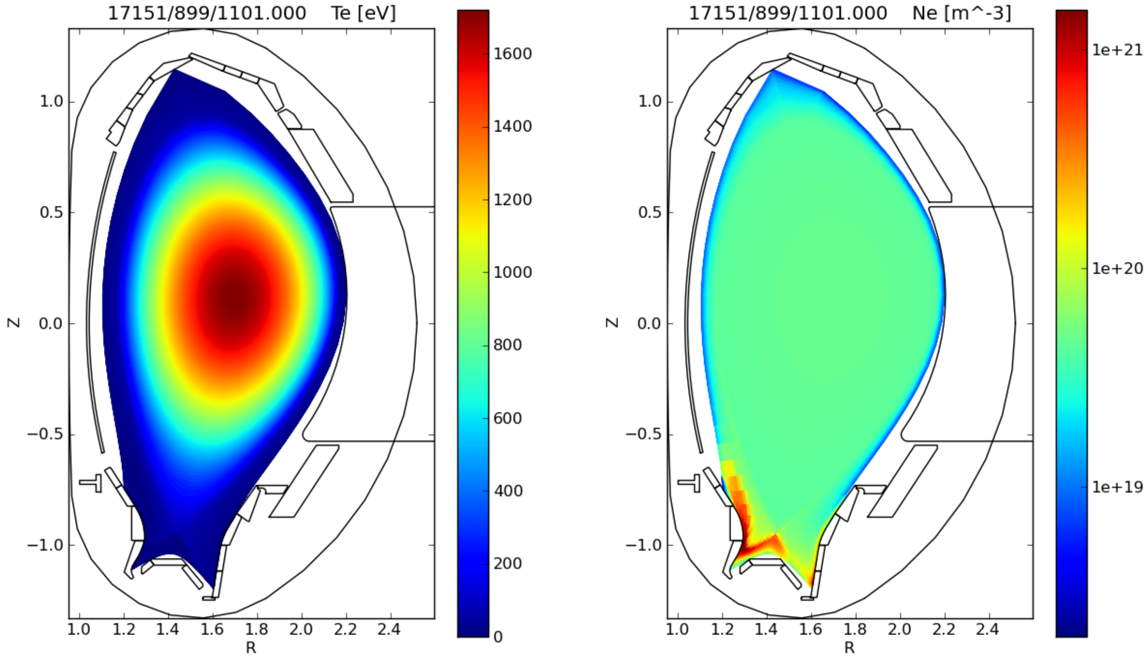


Figure 17: Plots of Te and ne for the final state of the D+He + C case, using SOLPS5-B2 option **13** for the flux boundary consitions. 33 poloidal points were used in Helena.

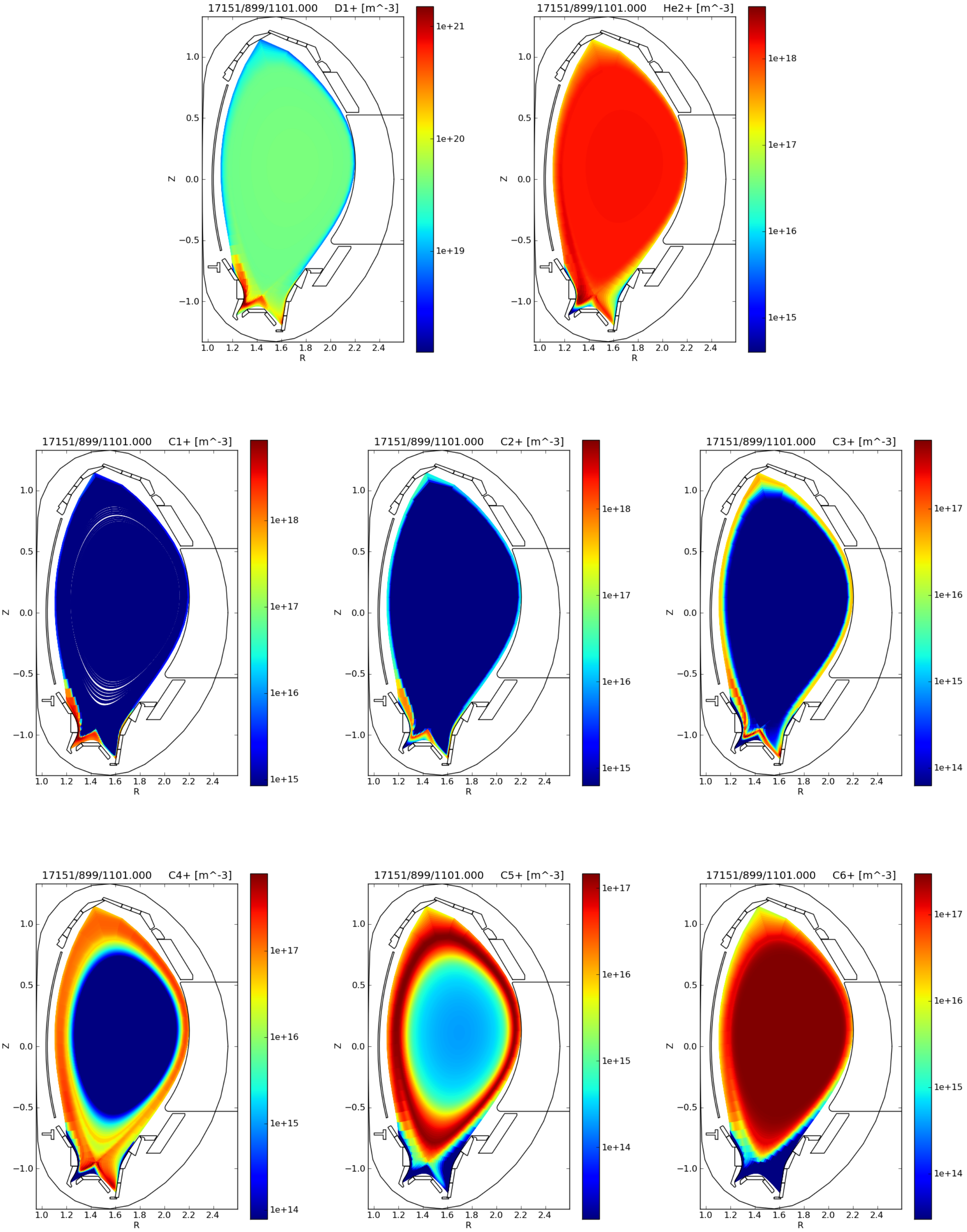


Figure 18: Plots of ion densities for the final state of the D+He + C case, using SOLPS5-B2 option **13** for the flux boundary consitions. 33 poloidal points were used in Helena.

3.4 D+He + C+Ar+Ne case

The physics case is a D+C+Ar+Ne+He plasma, with SOLPS maintaining an upstream separatrix electron density of $4 \times 10^{19} m^{-3}$ through a D gas-puff from the "wall". A fixed gas puff for both Ar and Ne were applied at the outer boundary in SOLPS-B2 (the "wall").

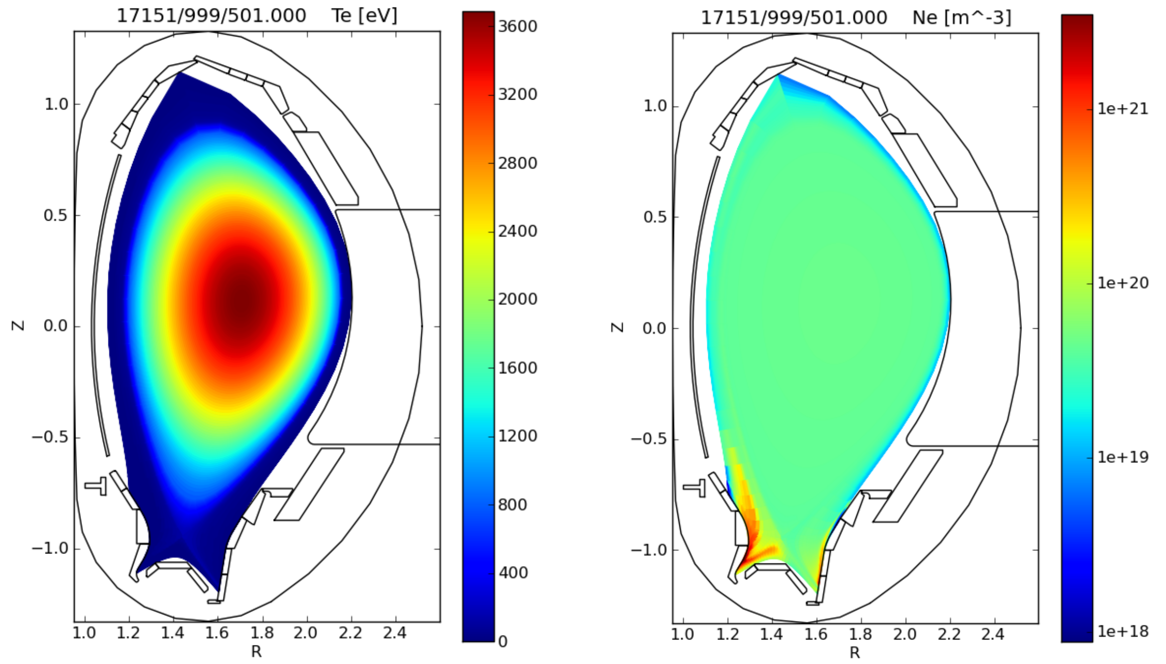


Figure 19: Plots of Te and ne for the final state of the D+He + C+Ar+Ne case, using SOLPS5-B2 option **13** for the flux boundary consitions. 33 poloidal points were used in Helena.

Run	ETS	SOLPS
900	18166.65 (1s)	41721 (1s)
901	13670.60 (1s)	43000.6 (1s)
902	10437.65 (1s)	38613.1 (1s)
903	10365.51 (1s)	37506.6 (1s)
904	10401.43 (1s)	37442.4 (1s)
905	10414.03 (1s)	33931.1 (1s)

Table 6: Timings for ETS and SOLPS: cpu used (and physics time). D+He + C+Ar+Ne case, 33 poloidal points in Helena.

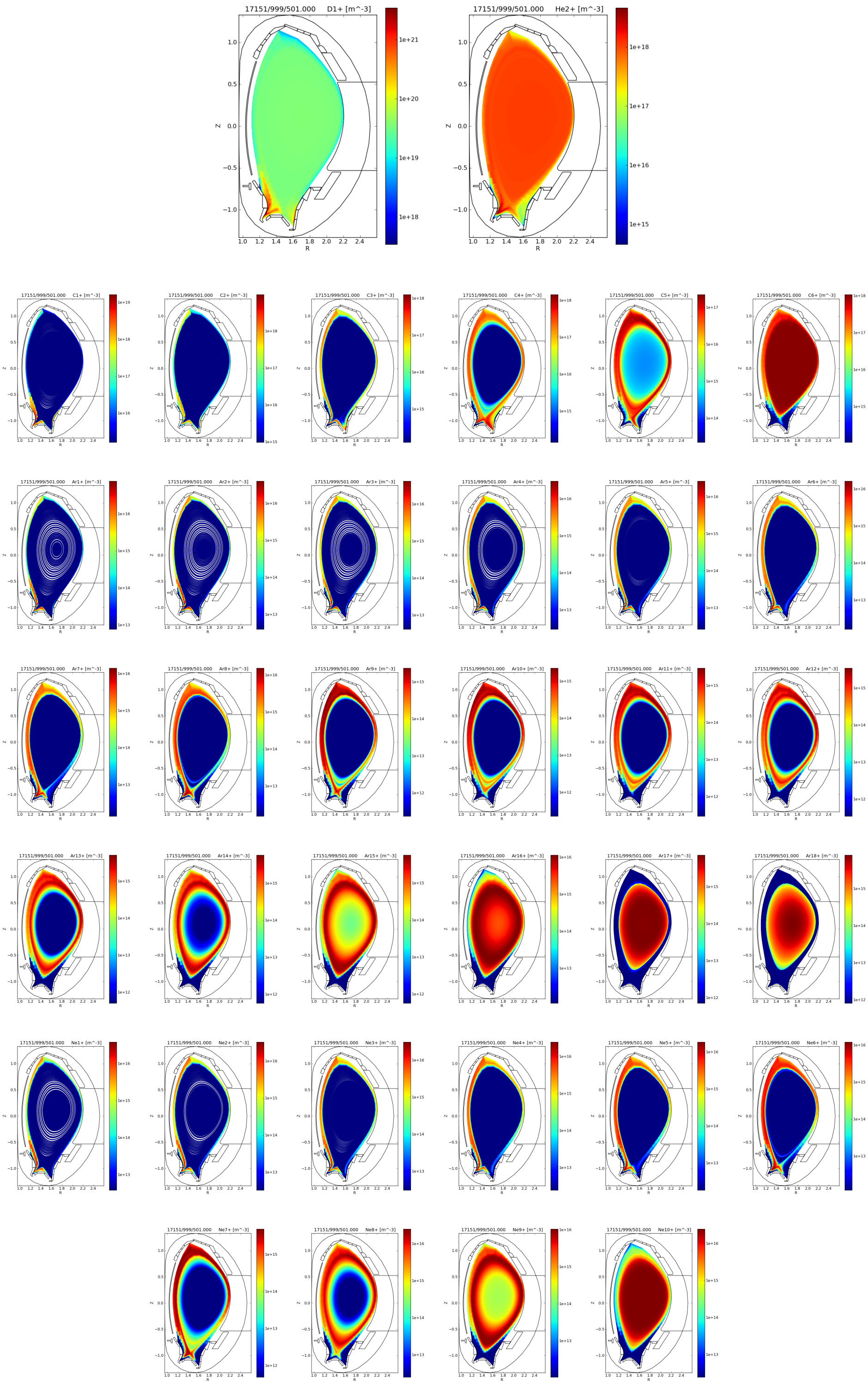


Figure 20: Plots of ion densities for the final state of the D+He + C+Ar+Ne case, using SOLPS5-B2 option **13** for the flux boundary consitions. 33 poloidal points were used in Helena.

3.5 D+He + C+Ar+Ne case, higher density

The physics case is a D+C+Ar+Ne+He plasma, with SOLPS maintaining an upstream separatrix electron density of $5 \times 10^{19} m^{-3}$ through a D gas-puff from the "wall". A fixed gas puff for both Ar and Ne were applied at the outer boundary in SOLPS-B2 (the "wall").

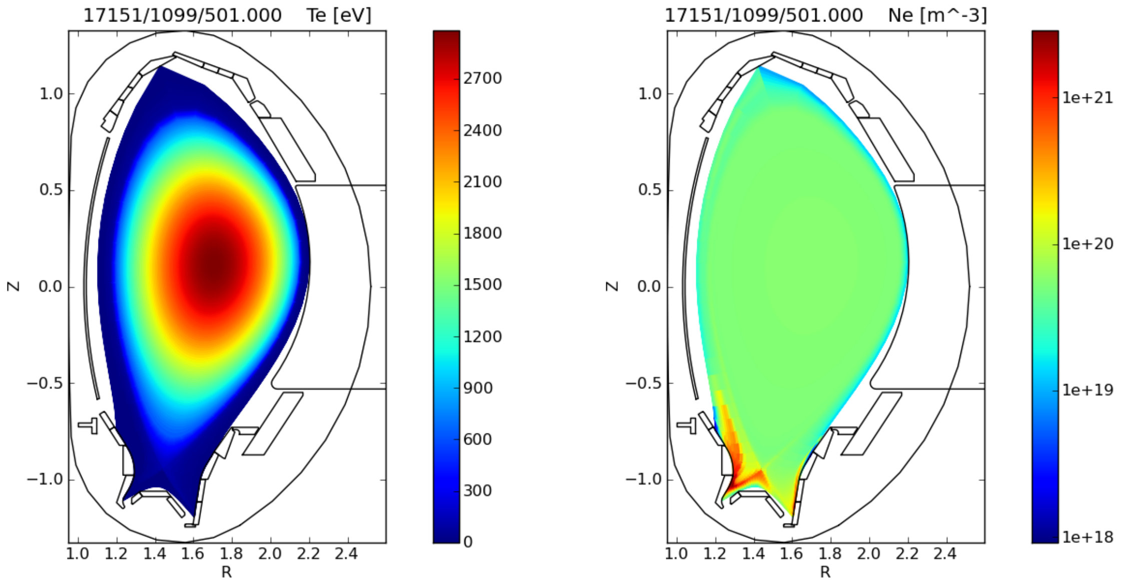


Figure 21: Plots of Te and ne for the final state of the D+He + C+Ar+Ne case, using SOLPS5-B2 option **13** for the flux boundary consitions. 33 poloidal points were used in Helena.

Run	**ETS**	**SOLPS**
1000	18355.44 (1s)	46684.7 (1s)
1001	11948.39 (1s)	44238.4 (1s)
1002	10449.72 (1s)	41591.3 (1s)
1003	10569.40 (1s)	41858.9 (1s)
1004	10755.15 (1s)	40342.4 (1s)
1005	10584.50 (1s)	38641.7 (1s)

Table 7: Timings for ETS and SOLPS: cpu used (and physics time). D+He + C+Ar+Ne case, 33 poloidal points in Helena.

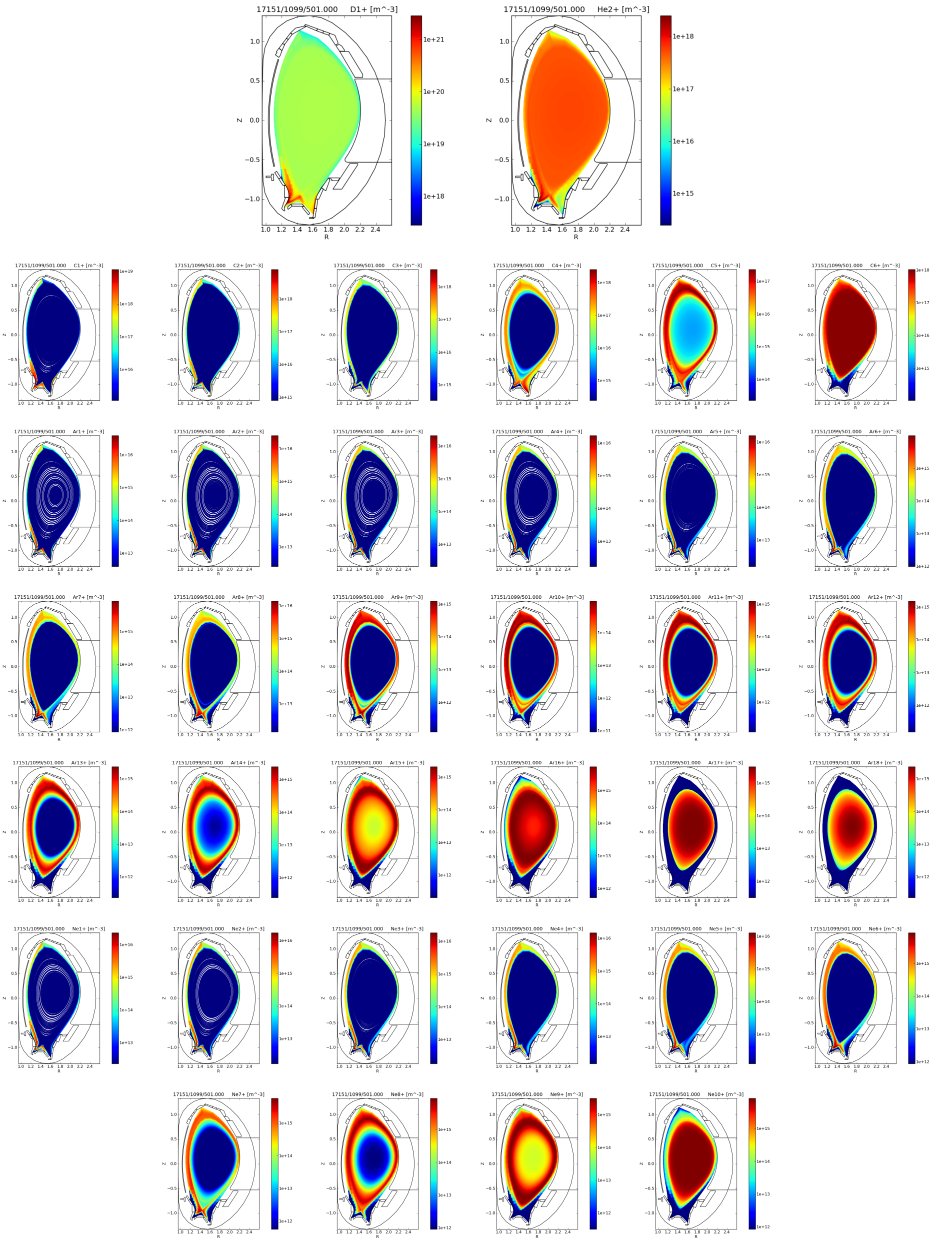


Figure 22: Plots of ion densities for the final state of the D+He + C+Ar+Ne case, using SOLPS5-B2 option **13** for the flux boundary consitions. 33 poloidal points were used in Helena.

4 Combined core-edge plots in VisIt

VisIt can be used for interactive analysis and plotting of data stored in the UAL. This section explains how.

4.1 Preparing CPOs with core data

Currently (August 2011), data on complex grids can only be accessed from VisIt when they are stored in CPOs using the general grid description. Data from any of the core CPOs therefore first has to be transferred into an edge CPO.

A program called `convert_core_data_to_edge_cpo` is provided in the `Core-Edge/FORTRAN` directory. It writes Te, Ti, ne, ni from the coreprof/coreimpur CPOs for a given run/shot to edge CPOs at equivalent time points. The 2d core grid used for this is derived from the equilibrium CPOs at same time points.

To run the tool, first check the input/output shot/run numbers in `convert_core_data_to_edge_cpo.F90`. Then compile and run in the same directory with

```
make
./convert_core_data_to_edge_cpo
```

4.2 Using VisIt to access the UAL

A tool called ualconnector is used to launch VisIt and allow it to access given CPOs via the UAL. Assuming the CPOs of interest are stored in shot #17151, runs #598 and #598, time 2.5s, user coster, tokamak name aug, data version 4.09a, the ualconnector call would be

```
~klingshi/bin/itm-grid/ualconnector -s 17151,598,2.5 -c edge
    -s 17151,599,2.5 -c edge -u coster -t aug -v 4.09a
```

This launches ualconnector and a VisIt instance which is set up to access the CPOs specified on the command line.

Detailed documentation of ualconnector can be found at
https://www.efda-itm.eu/ITM/doxygen/imp3/grid_service_library/python/.

4.3 Creating 2d plots

The CPOs written by SOLPS and `convert_core_data_to_edge_cpo` contain data on 2d grids. Use the Add/Mesh and Add/Pseudocolor menus to create 2d plots of grids and data. An example is shown in figure 24.

Some notes:

- CPOs are presented as hierarchical structures (see figure 23). Hierarchy levels that contain only numbers indicate indices for an array of structures. The names on the last hierarchy level contain the names of the subgrid the data is stored on.
- To get matching colors in different Pseudocolor plots you have to manually adjust the data ranges of the plots to match in the plot attributes.

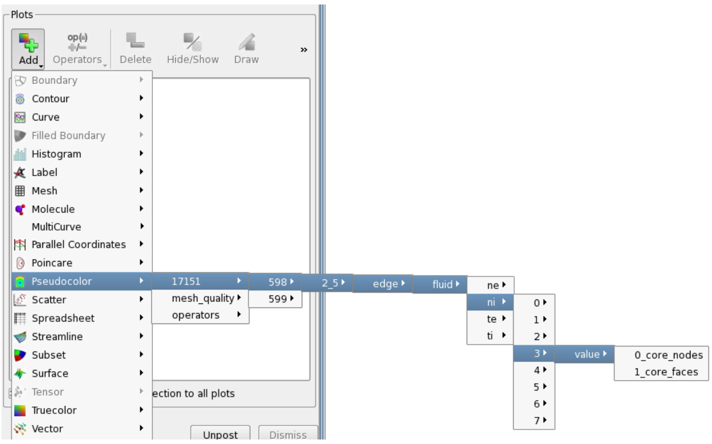


Figure 23: CPO fields as presented by the VisIt "Add..." menu. The numbers in the hierarchy below ni are the ion species indices. The last hierarchy level indicates that the Te values are available both on the faces (2d cells) and the nodes of the core grid.

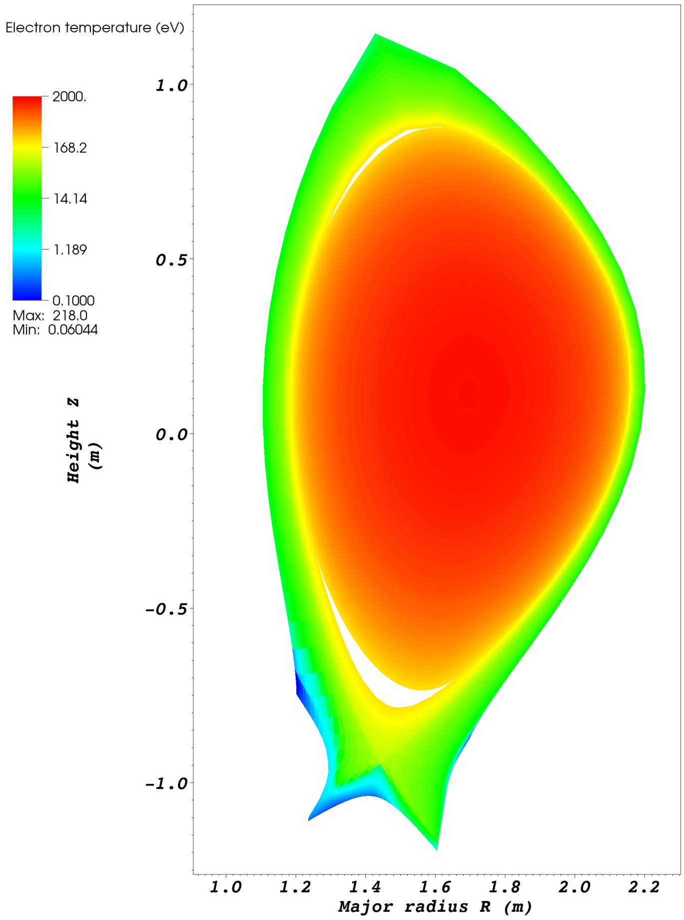


Figure 24: Combined 2d electron temperature produced with VisIt

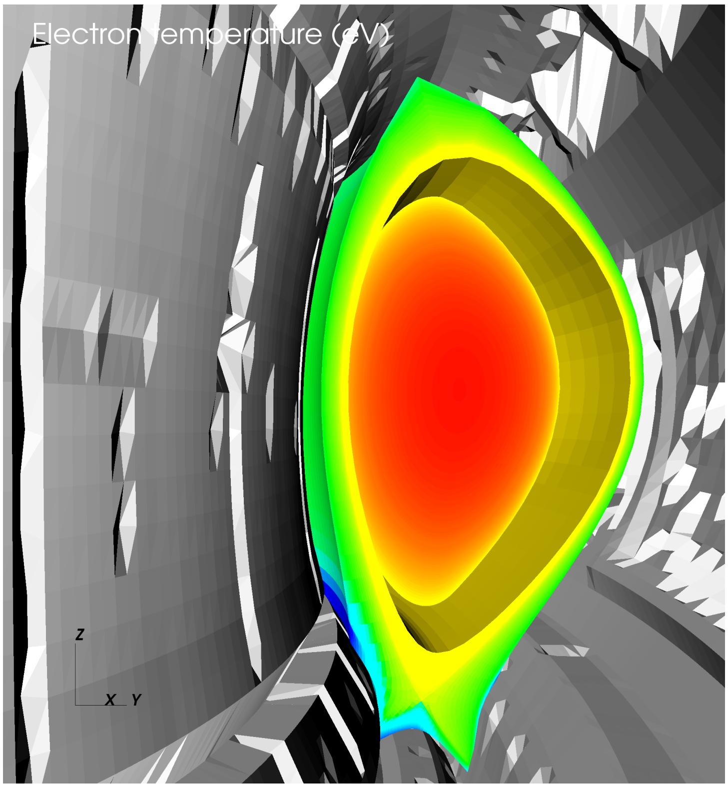


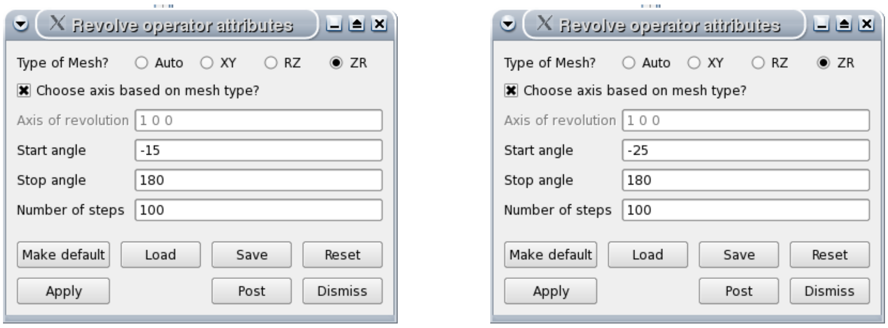
Figure 25: Electron temperature plot extended to 3d and combined with ASDEX Upgrade wall geometry.

4.4 Creating 3d plots

The 2d data in the CPOs can be extended to 3d in order to combine it with other 3d data. This example combines edge and core data with the vessel wall geometry as shown in figure 25.

The basic steps to produce this plot are as follows:

- Launch ualconnector/VisIt as described in section 4.2.
- Data plots:
  - Create 2d data plots as described in section 4.3.
  - Apply the "Revolve" operator to the 2d plots (for settings see figure 26).
- Vessel plot:
  - Open the file `/afs/efda-itm.eu/user/k/klingshi/public/wall/aug/wall.xmf` in VisIt (this adds it as an additional data source).
  - Add a pseudocolor plot for variable "CellCenteredValues".
  - In case the vessel geometry has the wrong orientation, fix it by applying the Geometry/CoordSwap operator (settings show in figure 28).
  - Create a colormap that contains only one color and assign it to the previous plot.
- View:
  - Set the view properties in the Controls/View/3D view menu. Sample settings are shown in figure 27.
- Lights:



(a) Core plot (b) Edge plot

Figure 26: "Rotate" operator settings to extend 2d plots to 3d in toroidal direction

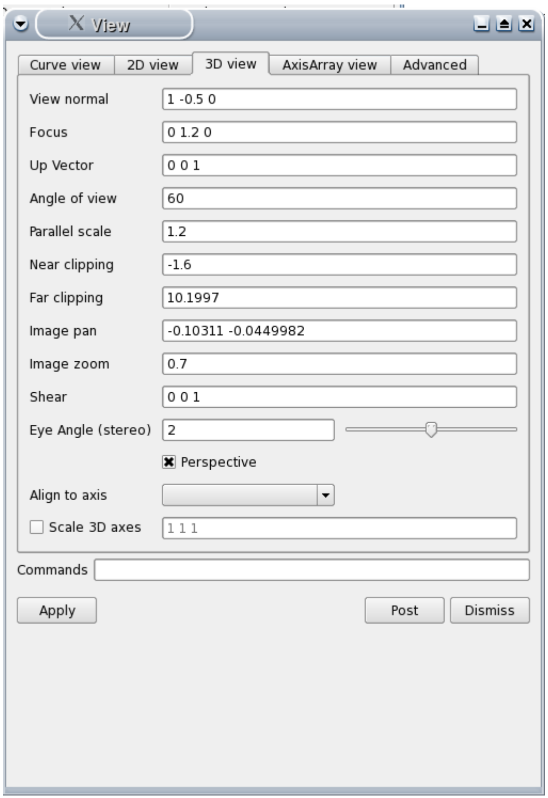


Figure 27: View settings for the 3D view in figure 25.

– Add lights to the 3D scene in the Controls/Lighting menu. The sample plot uses three lights with properties as shown in figure 29.

Notes:

- When applying operators to individual plots, make sure you have deselected "Apply operators ... to all plots" in the VisIt main window to avoid confusion.
- It's currently not possible to retrieve the vessel geometry information via ualconnector in the form required for this plot, thus the approach with the xdmf file (which was actually created from data in a CPO). This will change in the future.

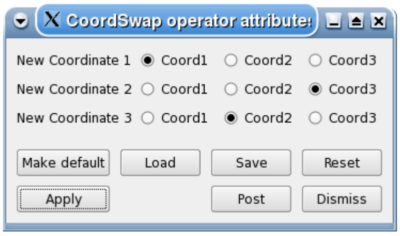
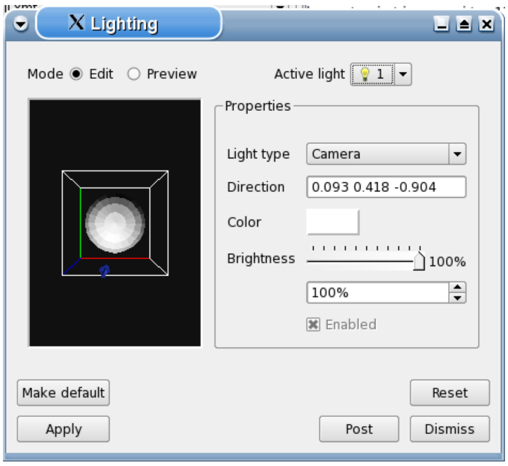
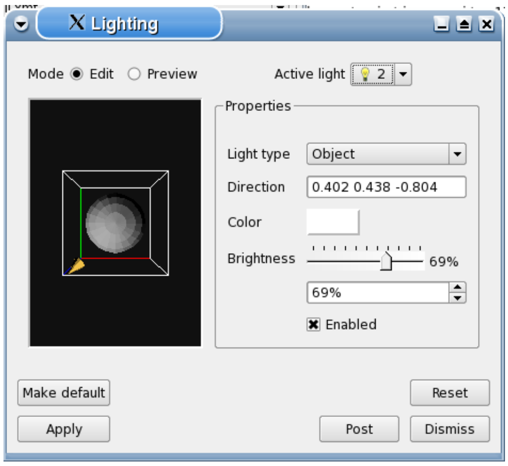


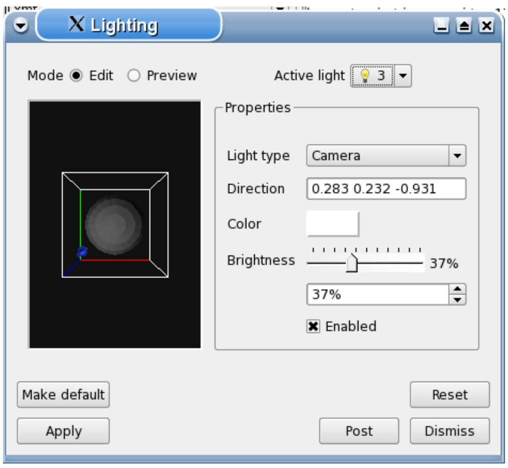
Figure 28: Settings for Geometry/CoordSwap operator to fix vessel geometry.



(a) Light 1



(b) Light 2



(c) Light 3

Figure 29: Light source settings as used in figure 25.

4.4.1 3d plots with transparency

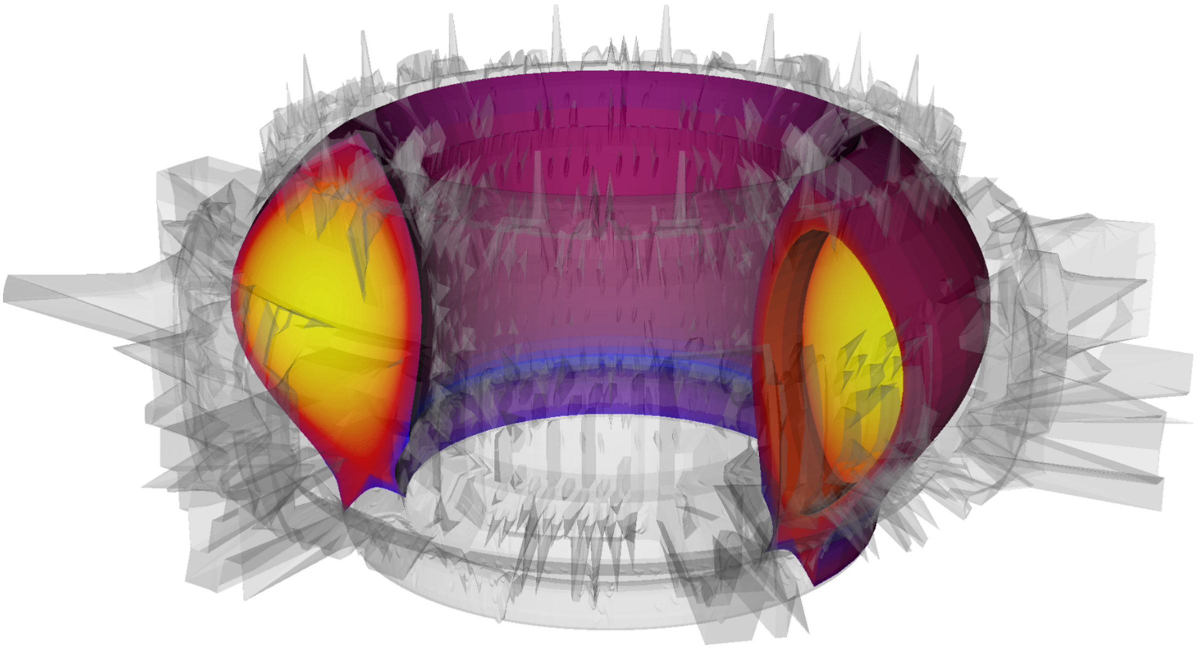


Figure 30: 3d rendering showing the AUG vessel (3d), SOLPS (2d) and ETS (1d) coupled simulations.