**Postprocessing SPARC output**

**hands-and-feet**

First, we need to copy the output files into the “runs” directory. Reminder: green text is what is typed by the user; cyan text is an alias defined in .bashrc.ext (also typed by the user)

It might be a good idea for someone at some point to add code to the batch script (e.g. knl\_702.sh) that copies the ascot output file to the …/ascot5/runs directory so that we don’t have to manually do this tedious copy.

toruns

copy\_to\_runs 36825352

copy\_to\_runs 36837310

copy\_to\_runs 36837314

copy\_to\_runs 36837322

toruns <-- the last command leaves us in directory … ascot/ascot\_work\_36837322

Now run the home-grown ASCOT postprocessor process-ascot.py

python $dir\_mypython/process\_ascot.py 🡨 launches the Python script

enter comma-separated list (w/o spaces) of ascot output .h5 files( default=) **ascot\_36825352.h5,ascot\_36837310.h5,ascot\_36837314.h5,ascot\_36837422.h5**

enter name of geq file( default=v1e.geq) **<cr>** 🡨 a <cr> accepts default value

enter profiles filename( default=v1e\_profiles\_3.txt) **<cr>**

enter kW of lost alphas( default=0.0) **480.**

enter integer that specifies grid resolution for spd calc( default=0) **6**

enter stub for outout filename (none for outout to screen)( default=)

**ascot\_AR\_pshape7\_ashape6** 🡨 note that the “stub” does not include e.g. .txt

… the code will take a minute to several minutes to process, depending on the number of files and the total number of markers that are being processed

Note that you supply a list of one or more ASCOT output files, in single quotes, **without any spaces between the filenames**. The processor will then combine the outputs of the two files before computing e.g. the fraction of lost alpha power. This allows us to break a big calculation of say 100,000 alphas into more manageable runs of 20,000 alphas each.

The second argument (v1e.geq’) is the EFIT equilibrium for the run.

The parameter “kW of lost alphas” is the number of kilowatts of alpha power that hits the wall. One would think that this would be an ouput parameter rather than an input parameter, but for reasons discussed in the section on parent/daughter runs, this is not the case.

The “integer that specified grid resolution for the spd calc” controls the grid size for the poloidal limiters, toroidal belt limiters, RF antennas etc. As shown in the code snippet below the integer (called ishape internally in the construct\_torus function within process\_ascot.py) is used as an index to select say the number of grid points in the phi direction, nphi\_limiter, and the number of grid points in the Z direction, nz\_limiter. There are similar arrays for the toroidal belt limiter, the RF antennas, and all of the other structures that comprise the plasma-facing wall.

# ishape 1 2 3 4 5 6 7 8

nnphi\_limiter = [5, 5, 20, 10, 5, 5, 4, 5] 🡨 these are hard-wired sets of

available grid resolutions.

nnz\_limiter = [6, 6, 20, 10, 6, 10, 10, 15]

nphi\_limiter = nnphi\_limiter[ishape-1] 🡨 this selects one value from the set

nz\_limiter = nnz\_limiter[ishape-1]

Output: e.g. ascot\_pshape7\_ashape6.txt and ascot\_pshape7 need to finish this

Both the text file and the plot file have accumulated a lot of bloat over the past months, mostly to debug particular problems. The plot file in particular is big – about 200 pages of plots.

Sample output is given in file [ascot\_33081964\_multi.pdf](file:///C:\Users\sscott\Documents\ripple\ASCOT_handover\ascot_33081964_multi.pdf) and annotations for these plots are given in [annotations\_for\_figures.docx](file:///C:\Users\sscott\Documents\ripple\ASCOT_handover\annotations_for_figures.docx).