EUROfusion TSVV-5
Python tool(s) for basic tally reading and line integral calculations

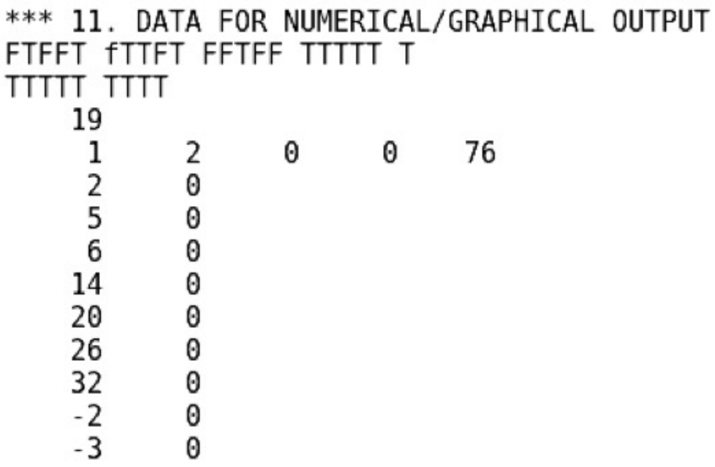
Andreas Holm, Mathias Groth, Vladimir Solokha
– Aalto University

This work has been carried out within the framework of the EUROfusion Consortium and has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement number 633053. The views and opinions expressed herein do not necessarily reflect those of the European Commission.

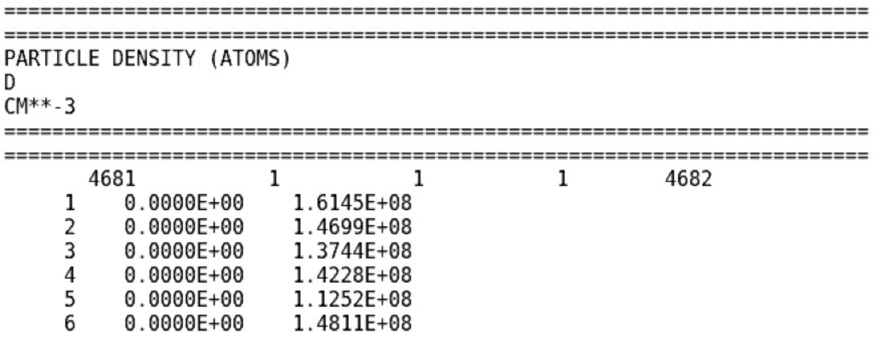
Motivation: standard/unified and (FZJ, TSVV-5) internal vetted Python tool to read out EIRENE tallies

- EIRENE internal graphical and numerical output of tallies controlled under block 11

⇒ Block 11A for numerical tallies, e.g., output of tally 1 (PDENA, see table 5.3 in EIRENE manual) in fort.76



```
*** 11. DATA FOR NUMERICAL/GRAPHICAL OUTPUT
FTFFT fTTFT FFTFF TTTTT T
TTTTT TTTT
    19
     1     2     0     0    76
     2     0
     5     0
     6     0
    14     0
    20     0
    26     0
    32     0
    -2     0
    -3     0
```



```
========================================================================
========================================================================
PARTICLE DENSITY (ATOMS)
D
CM**-3
========================================================================
========================================================================
       4681           1           1           1        4682
     1    0.0000E+00    1.6145E+08
     2    0.0000E+00    1.4699E+08
     3    0.0000E+00    1.3744E+08
     4    0.0000E+00    1.4228E+08
     5    0.0000E+00    1.1252E+08
     6    0.0000E+00    1.4811E+08
```

Motivation: standard/unified and (FZJ, TSVV-5) internal vetted Python tool to read out EIRENE tallies

- EIRENE internal graphical and numerical output of tallies controlled under block 11

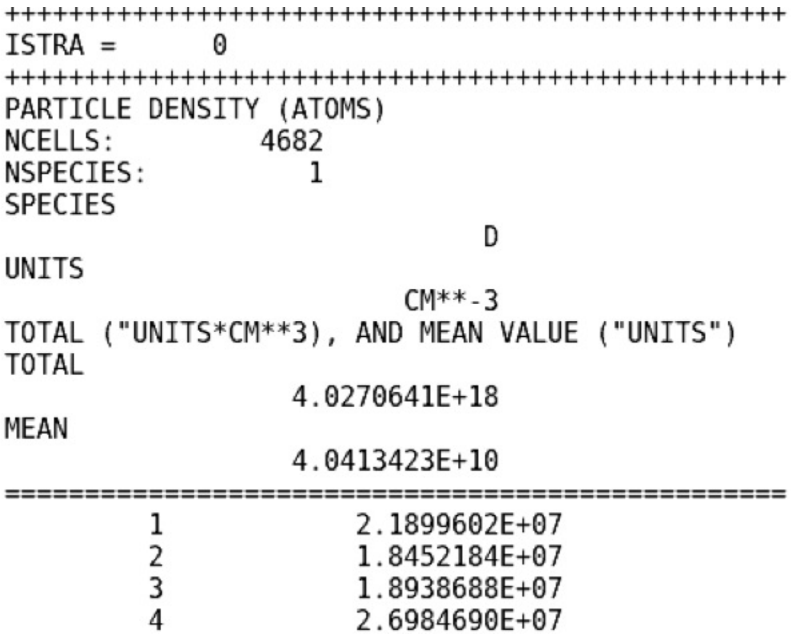
⇒ Block 11A for numerical tallies, e.g., output of tally 1 (PDENA, see table 5.3 in EIRENE manual) in fort.76

- Setting PLIDL = T in block 11B prints output files for further processing in external tools (e.g. for FZJ IDL graphics tool)
  - **Grid** information as well as all **input** (plasma) and **output** tallies (excluding de-activated ones) are printed on files, one file per tally
  - Not all parameters are yet written out, e.g., both Lyman and Balmer emissivities as defined in block 12
  - JET EDGE2D-EIRENE cases: 10-15 MB

Motivation: standard/unified and (FZJ, TSVV-5) internal vetted Python tool to read out EIRENE tallies

- More information in numerical output via PLIDL than when turning on tally output individually

⇒ Working with individually written tallies requires (small) modification of read routine



```
+++++++++++++++++++++++++++++++++++++++++++++++++
ISTRA =      0
+++++++++++++++++++++++++++++++++++++++++++++++++
PARTICLE DENSITY (ATOMS)
NCELLS:         4682
NSPECIES:          1
SPECIES
                              D
UNITS
                         CM**-3
TOTAL ("UNITS*CM**3), AND MEAN VALUE ("UNITS")
TOTAL
                  4.0270641E+18
MEAN
                  4.0413423E+10
=================================================
         1             2.1899602E+07
         2             1.8452184E+07
         3             1.8938688E+07
         4             2.6984690E+07
```

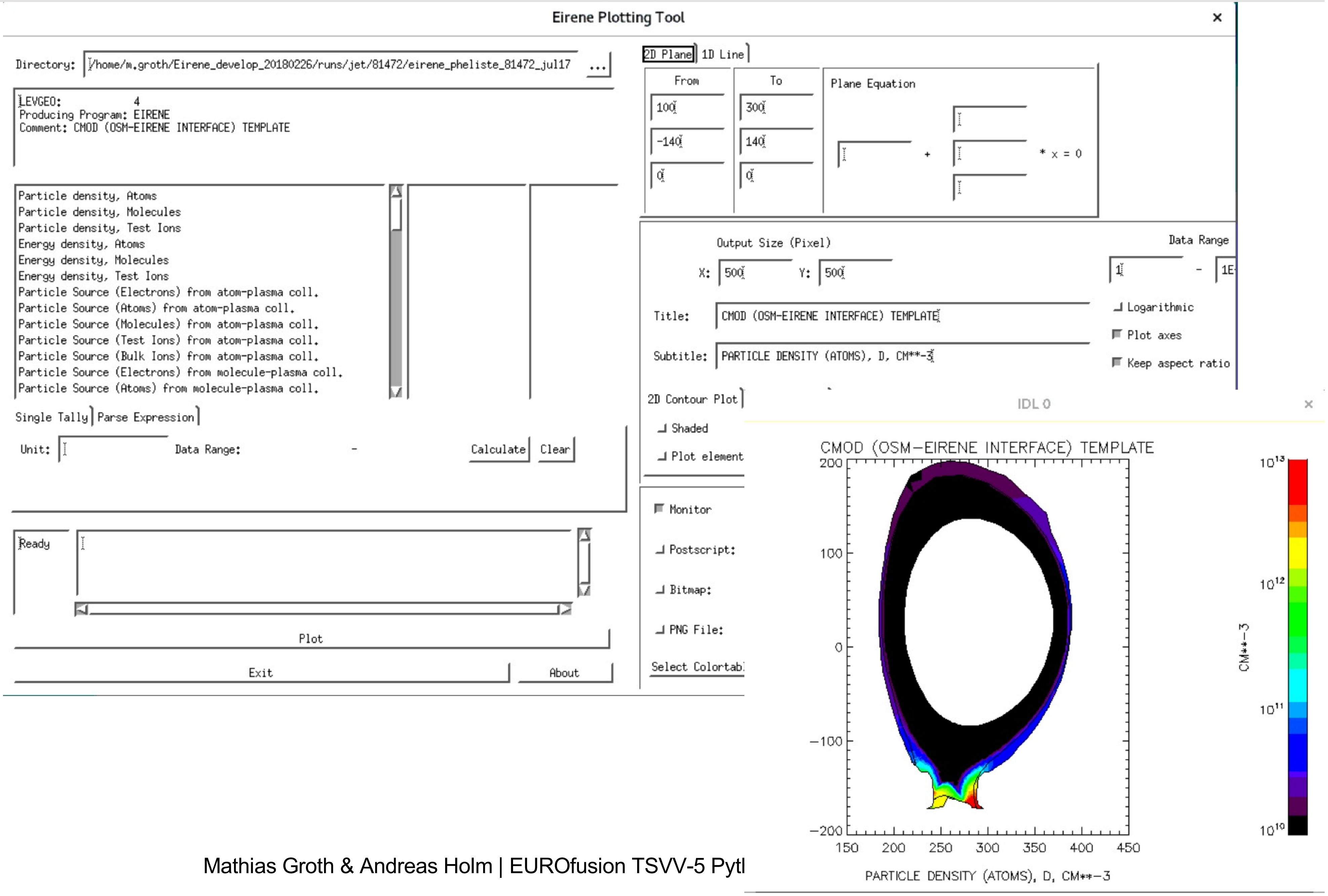
Motivation: standard/unified and (FZJ, TSVV-5) internal vetted Python tool to read out EIRENE tallies

- EIRENE internal graphical and numerical output of tallies controlled under block 11

⇒ Block 11A for numerical tallies, e.g., output of tally 1 (PDENA, see table 5.3 in EIRENE manual) in fort.76

- Setting PLIDL = T in block 11B prints output files for further processing in external tools (e.g. for FZJ IDL graphics tool)
  - Volume tallies, post-processed line-integrals for ADDV

Flexible FZJ internal IDL tool to plot EIRENE tallies, requires IDL license (and FZJ permission)



Eirene Plotting Tool

Directory: /home/m.groth/Eirene_develop_20180226/runs/jet/81472/eirene_pheliste_81472_jul17

```
LEVGEO:              4
Producing Program: EIRENE
Comment: CMOD (OSM-EIRENE INTERFACE) TEMPLATE
```

```
Particle density, Atoms
Particle density, Molecules
Particle density, Test Ions
Energy density, Atoms
Energy density, Molecules
Energy density, Test Ions
Particle Source (Electrons) from atom-plasma coll.
Particle Source (Atoms) from atom-plasma coll.
Particle Source (Molecules) from atom-plasma coll.
Particle Source (Test Ions) from atom-plasma coll.
Particle Source (Bulk Ions) from atom-plasma coll.
Particle Source (Electrons) from molecule-plasma coll.
Particle Source (Atoms) from molecule-plasma coll.
```

Single Tally | Parse Expression

Unit: Data Range: -

Ready

2D Plane | 1D Line

From	To
100	300
-140	140
0	0

Plane Equation: + * x = 0

Output Size (Pixel) X: 500 Y: 500

Data Range: 1 - 1E

Title: CMOD (OSM-EIRENE INTERFACE) TEMPLATE

Subtitle: PARTICLE DENSITY (ATOMS), D, CM**-3

□ Logarithmic

☑ Plot axes

☑ Keep aspect ratio

2D Contour Plot

□ Shaded

□ Plot element

☑ Monitor

□ Postscript:

□ Bitmap:

□ PNG File:

Motivation: standard/unified and (FZJ, TSVV-5) internal vetted Python tool to read out EIRENE tallies

- EIRENE internal graphical and numerical output of tallies controlled under block 11

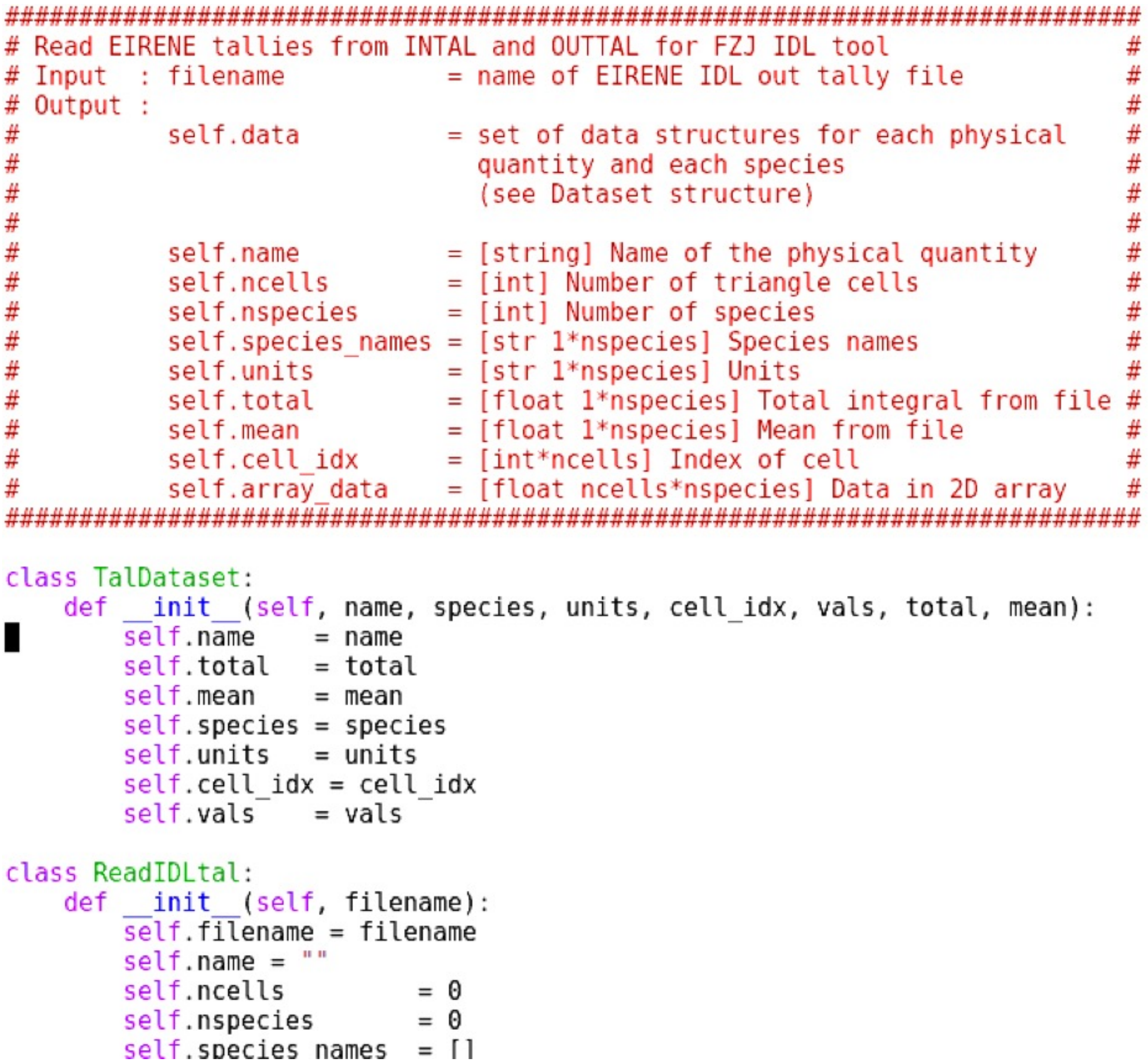
⇒ Block 11A for numerical tallies, e.g., output of tally 1 (PDENA, see table 5.3 in EIRENE manual) in fort.76

- Setting PLIDL = T in block 11B prints output files for further processing in external tools (e.g. for FZJ proprietary IDL graphics tool)

- FZJ-internal IDL tool to plot EIRENE “IDL” tallies, e.g., volume tallies, post-processed line-integrals

⇒ Plotting EIRENE output outside FZJ Linux cluster and further processing of EIRENE tallies requires platform less-dependent tool ⇒ **Python**

Built Python read routines (classes) around reading “IDL” tallies like FZJ IDL

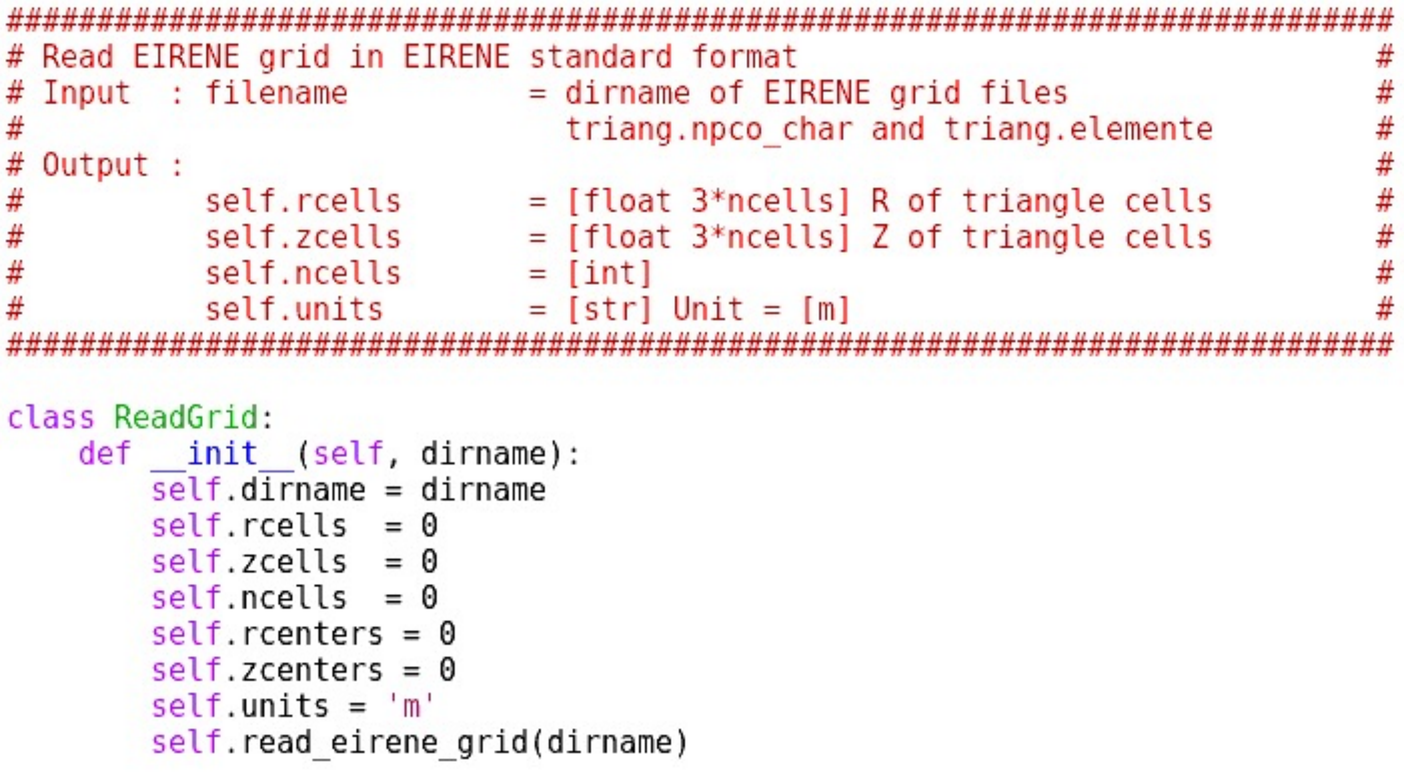


```
##############################################################################
# Read EIRENE tallies from INTAL and OUTTAL for FZJ IDL tool                 #
# Input   : filename          = name of EIRENE IDL out tally file            #
# Output :                                                                   #
#          self.data          = set of data structures for each physical     #
#                               quantity and each species                    #
#                               (see Dataset structure)                      #
#                                                                            #
#          self.name          = [string] Name of the physical quantity       #
#          self.ncells        = [int] Number of triangle cells               #
#          self.nspecies      = [int] Number of species                      #
#          self.species_names = [str 1*nspecies] Species names               #
#          self.units         = [str 1*nspecies] Units                       #
#          self.total         = [float 1*nspecies] Total integral from file  #
#          self.mean          = [float 1*nspecies] Mean from file            #
#          self.cell_idx      = [int*ncells] Index of cell                   #
#          self.array_data    = [float ncells*nspecies] Data in 2D array     #
##############################################################################

class TalDataset:
    def __init__(self, name, species, units, cell_idx, vals, total, mean):
        self.name    = name
        self.total   = total
        self.mean    = mean
        self.species = species
        self.units   = units
        self.cell_idx = cell_idx
        self.vals    = vals

class ReadIDLtal:
    def __init__(self, filename):
        self.filename = filename
        self.name = ""
        self.ncells         = 0
        self.nspecies       = 0
        self.species_names  = []
```

Grid files triang.npco_char and triang.elemente proved to be sufficient to plot grid



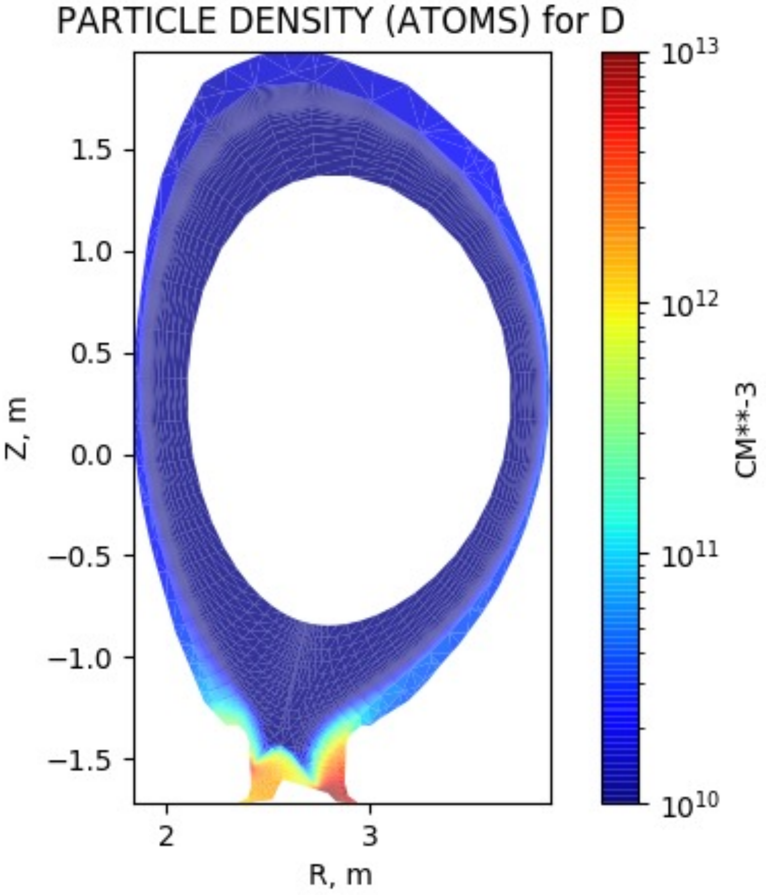
```
#############################################################################
# Read EIRENE grid in EIRENE standard format                                #
# Input   : filename          = dirname of EIRENE grid files                 #
#                               triang.npco_char and triang.elemente         #
# Output :                                                                  #
#          self.rcells        = [float 3*ncells] R of triangle cells         #
#          self.zcells        = [float 3*ncells] Z of triangle cells         #
#          self.ncells        = [int]                                        #
#          self.units         = [str] Unit = [m]                             #
#############################################################################

class ReadGrid:
    def __init__(self, dirname):
        self.dirname = dirname
        self.rcells  = 0
        self.zcells  = 0
        self.ncells  = 0
        self.rcenters = 0
        self.zcenters = 0
        self.units = 'm'
        self.read_eirene_grid(dirname)
```

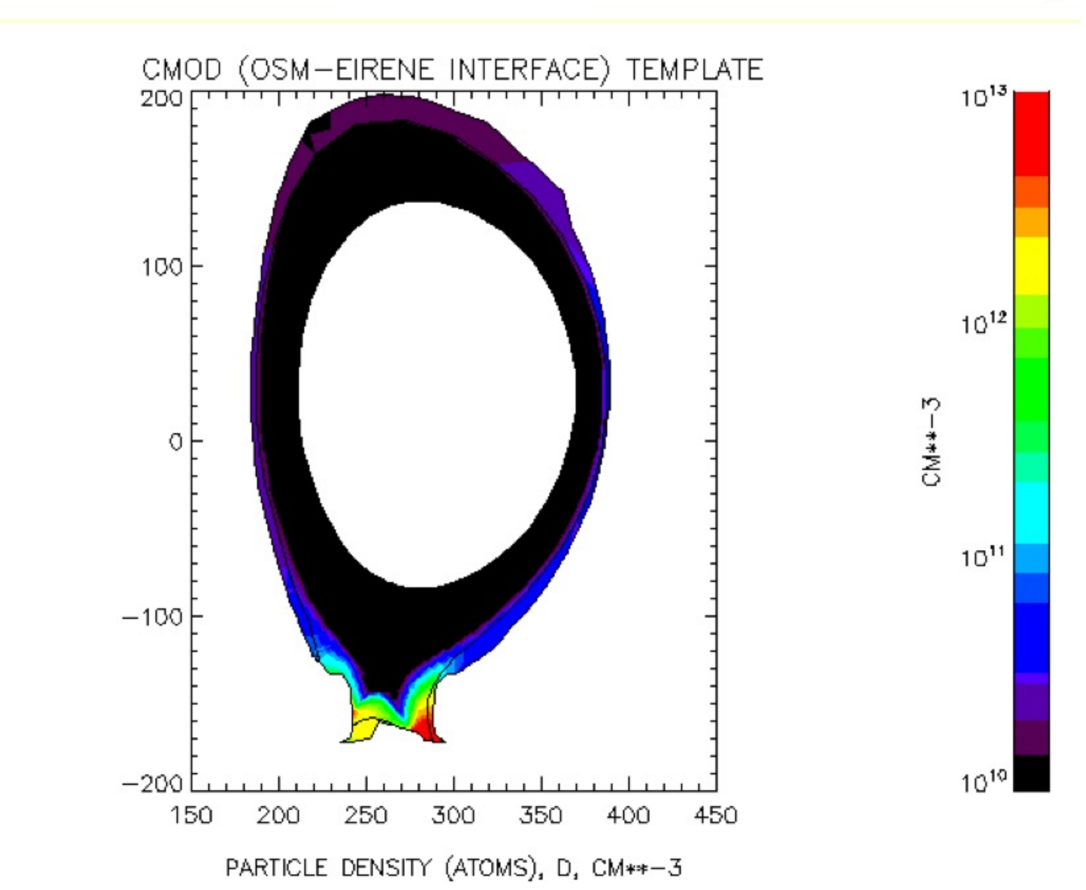
- Third grid output file for FZJ IDL tool, triang.neighbors, not yet used

Classes TalDataSet and ReadGrid + Python Matplotlib produce PDENA

Python



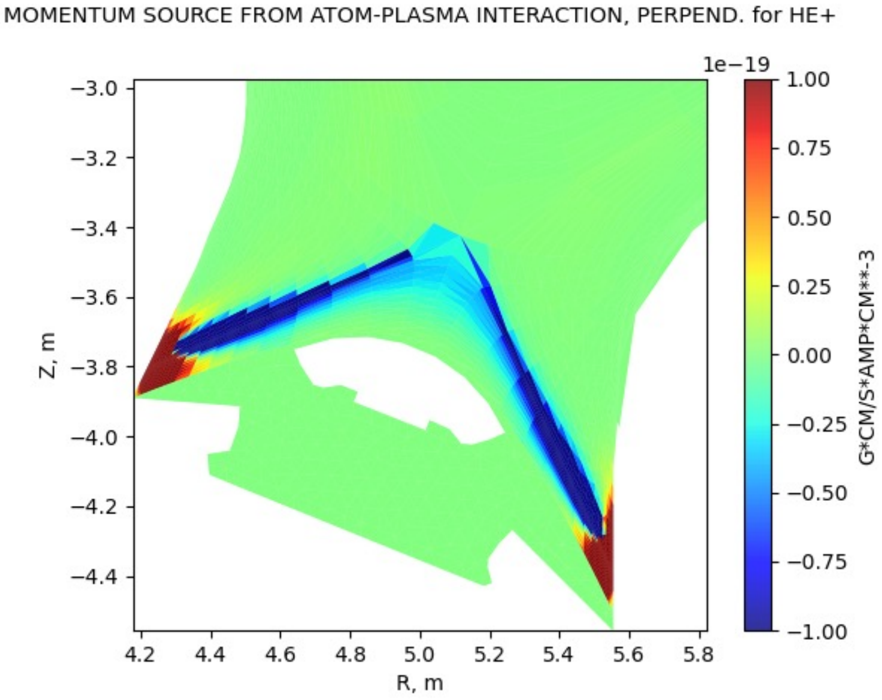
FZJ IDL



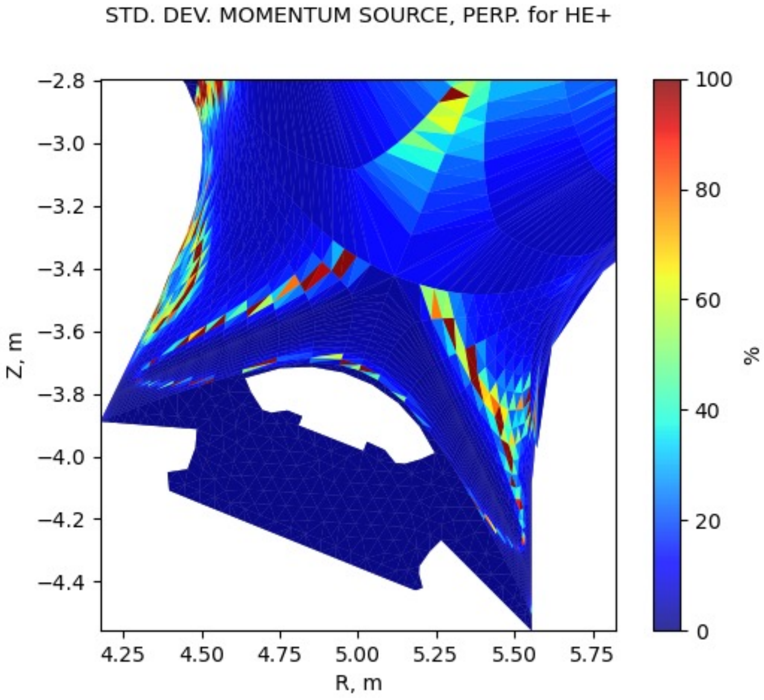
- Volume intal and outtal as in EIRENE manual tables 5.3 and 5.7 ⇒ differences in number of non-zero cells in tallies to be resolved

For NSIGVI ≠ 0 in block 9, variances of tally parameters are also produced

Mean value



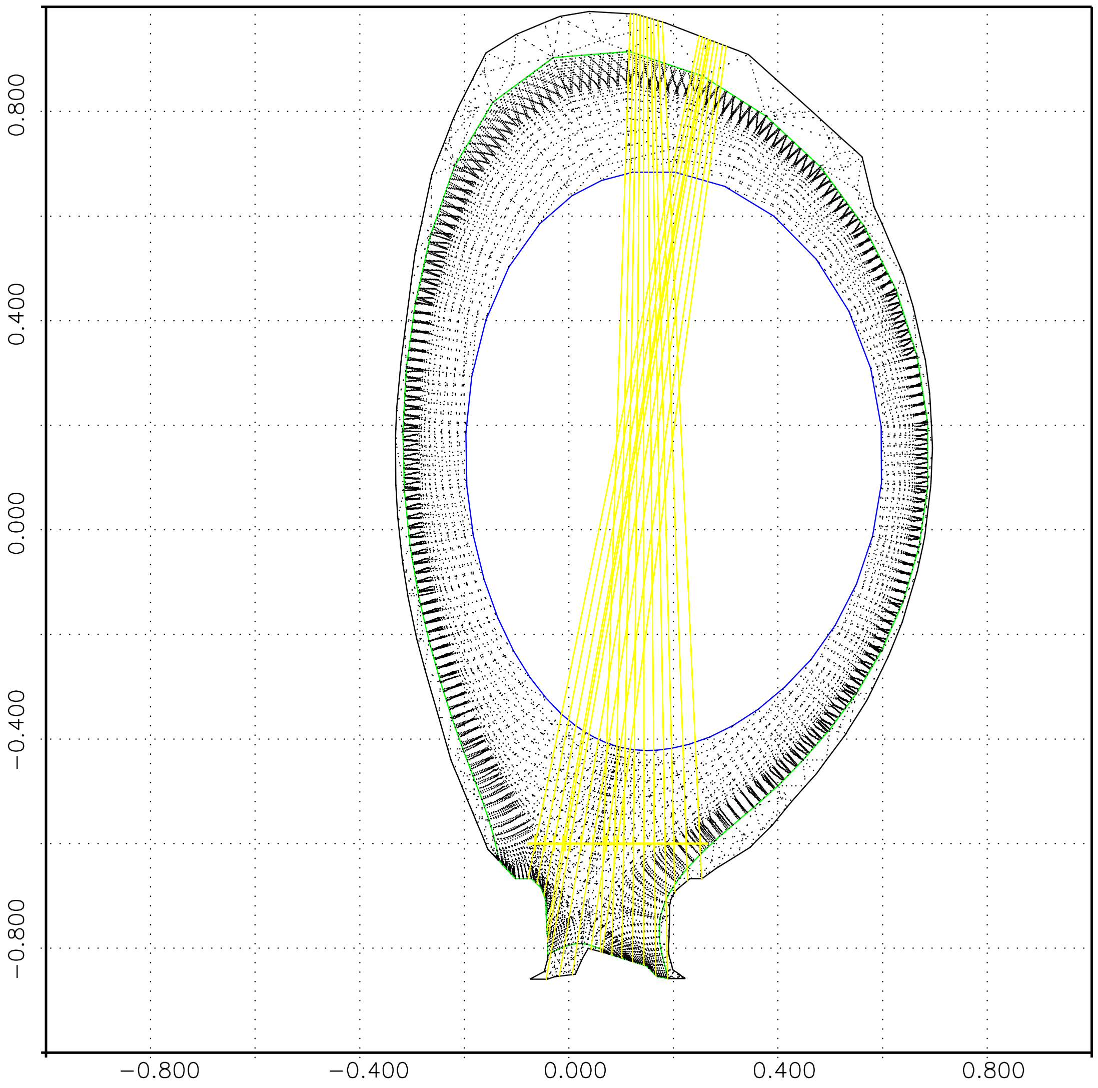
Standard deviation of mean value

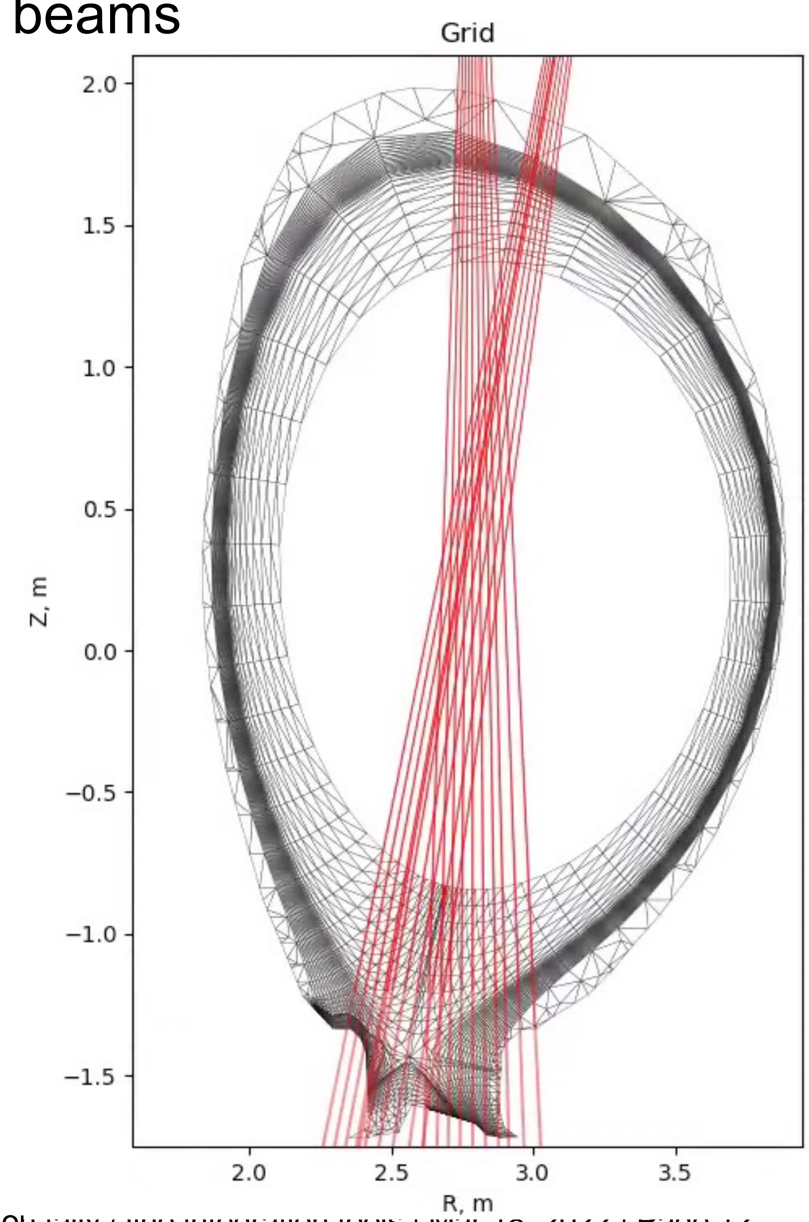


ISFN Feb 2022

Development of line-integration tools for comparison to spectroscopic measurements

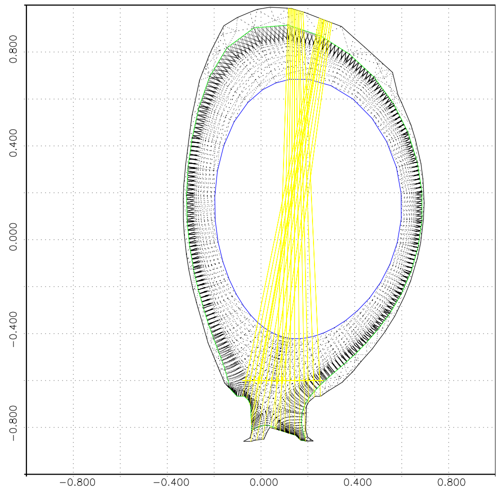
EIRENE internal integration based on pencil beams

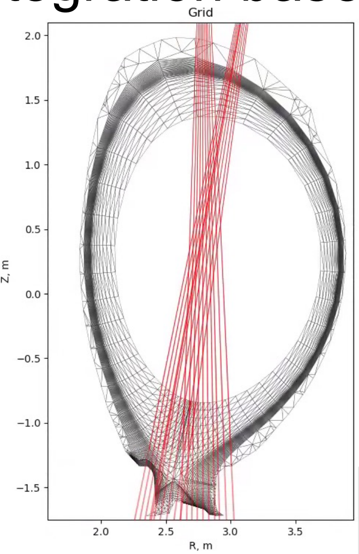


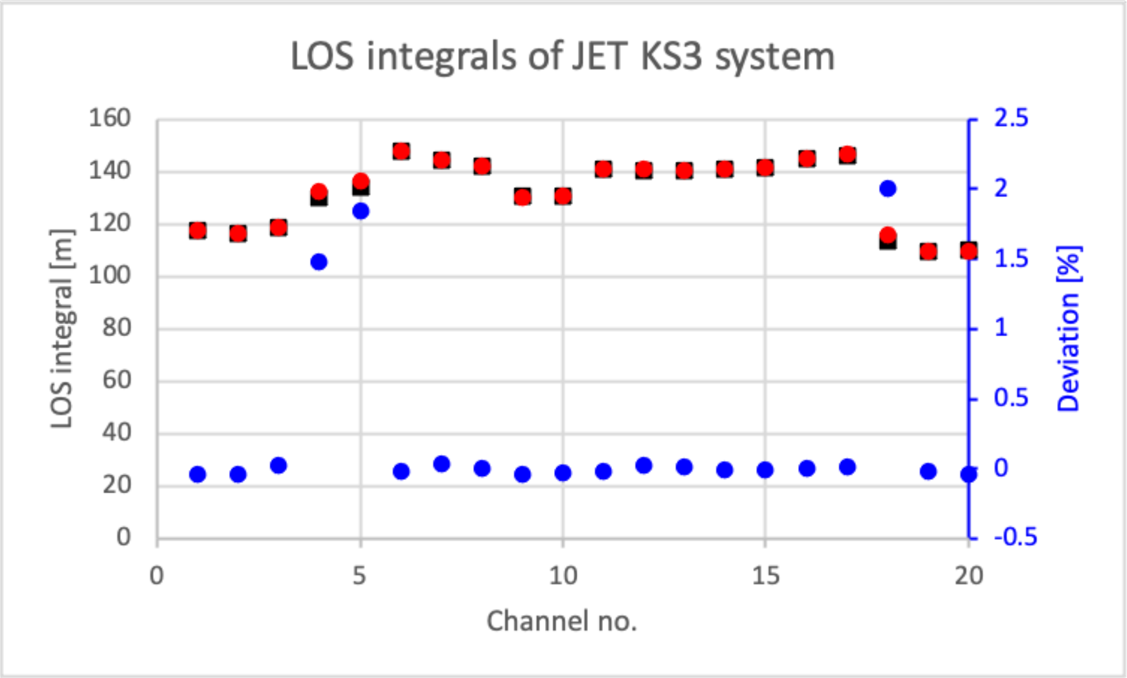


Development of line-integration tools for comparison to spectroscopic measurements

EIRENE internal integration based on pencil beams

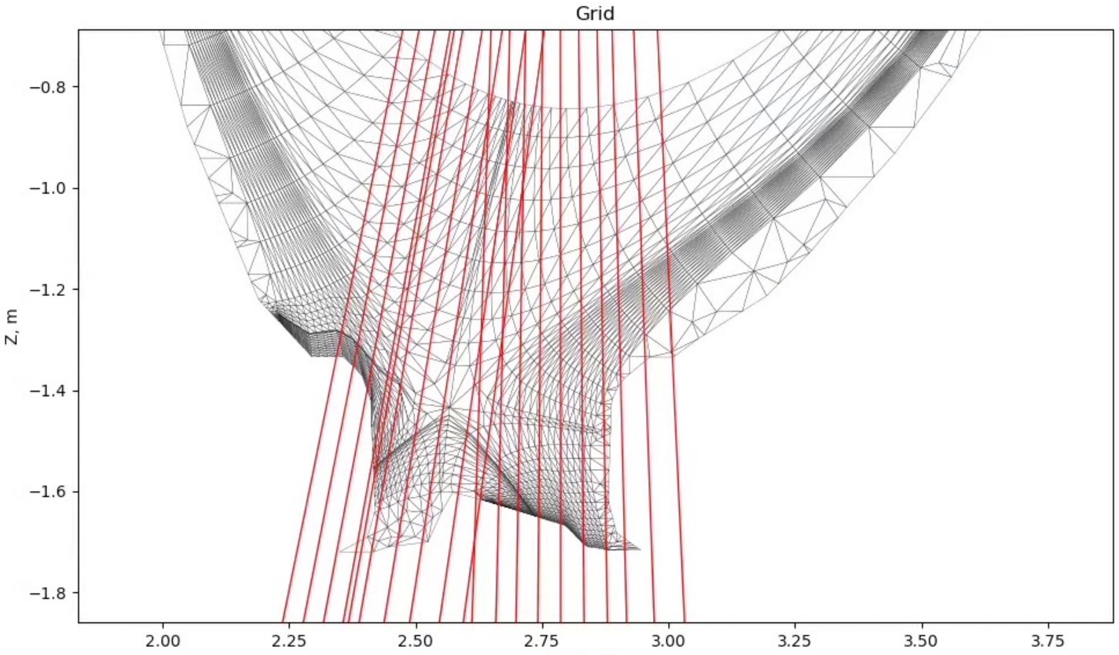


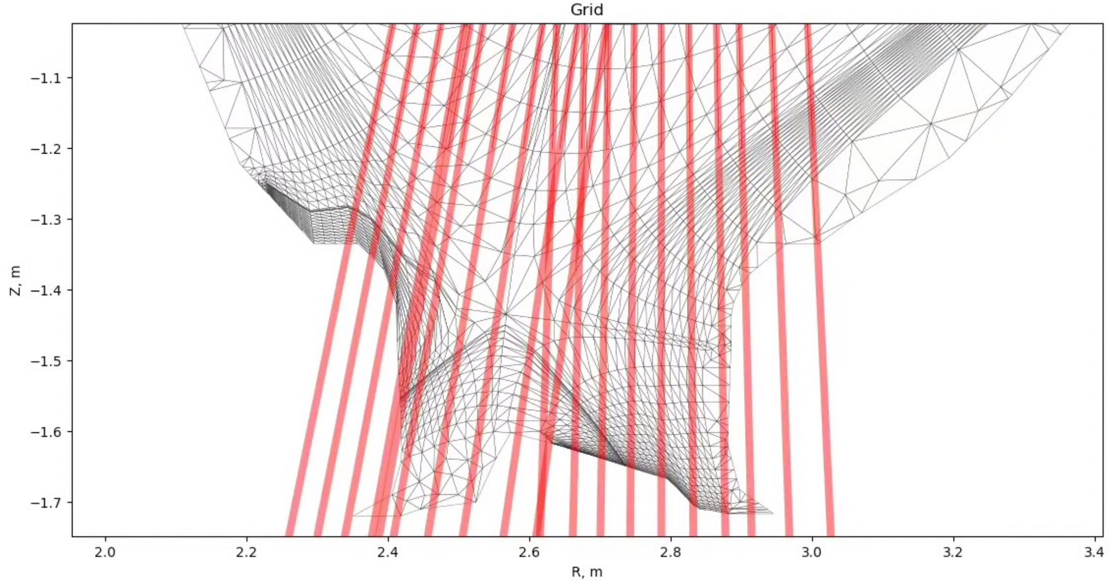




- Identified issue with LOS crossing EIRENE grid second time in divertor corner

Introduction of etendue in lines-of-sight for comparison to spectroscopic measurements





- Connection to SOLPSPY and Pyproc for SOLPS-ITER and EDGE2D-EIRENE

- Cherab (for reflections) ⇒ IMAS objects: LOS geometry definition, spectral lines, etc.

Additional ideas, issues on post-processing tool

- Routines run on FZJ, JET Heimdall and Aalto Triton Linux clusters
- Standalone EIRENE output based on converged EDGE2D-EIRENE cases
- Contour/vector plots for magnetic fields, particle (drift) velocities
- Variances of tally parameters
- Test grid and tally formats from other coupled codes, e.g., SOLPS-ITER, SOLEDEG2D-EIRENE, …
- Common plotting routines can be stored in Git repository in addition to basic routines (avoiding duplicate pain/work)

⇒ Other tally reading and plotting routines?

⇒ Place and maintenance of post-processing routines?

⇒ Fidelity check of post-processing routines, i.e., line integration tool?

Issues outside post-processing tools

- Modification to EIRENE Fortran routines ⇒ push to which Git repository?
    - Sven: angular resolved surface tallies
    - Mathias: line-integration tools in scr/diagno and src/user-routines/default-user
- Fidelity/consistency check and base-case testing of modified routines?