Memorandum

idm@F4E Ref:	F4E_D_2R2H62 v.1.0	Date	2021-03-30

Subject: **Mesh2vtk: a tool for D1S-UNED mesh responses post-processing**

Author: **P. Sauvan, F.Ogando**

Affiliation: Universidad Nacional a Distancia, Madrid

Signatory: **M. Fabbri**

Department: A&C / Engineering Unit / ITER Department

Contact marco.fabbri@f4e.europa.eu

To: **Everybody interested**

In order to keep F4E nuclear analysis capabilities up-to-date with the latest, state-of-the-art method and post-processing tools, a new practical tool called *Mesh2Vtk* to convert the meshes produced by MCNP and D1S-UNED into a VTK format has been developed. The tool is a python 3.6 based script able to read any mesh format in meshtally files produced by D1SUNED, MCNP5 or MCNP6. Mesh format includes usual MCNP column or matrix format and also specific D1SUNED format including cell or isotope contribution binning and source mesh importance format. Both Cartesian and cylindrical meshes can be read. The tool incorporates also simple functions to operate with meshes (add, scale, compare).
The tool is used through a text based interactive menu, and it can be run under Windows or Linus systems.

What Mesh2vtk does

Capabilities of Mesh2Vtk are divided in three categories:

- Mesh information
- Mesh conversion to VTK
- Operation on meshes

Mesh information can display a summary of the meshes present in the meshtally file or the detailed information of a specific mesh.

Mesh conversion writes, in separate files, meshes selected by the user into VTK format.

Operation on meshes allows performing various operations with meshes. Operations are:

- Mesh scaling. Multiplying the mesh value by a factor.
- CuV treatment and different normalization
- Sum of two or various meshes. For this operation, all meshes should have the same structure (same spatial mesh definition and energy bins). Each mesh involved in the sum can be scaled by a user defined factor.
- Mesh difference. Calculate the differences between the mesh elements of two meshes. This operation can be used to compare the variation in the results of two simulations of the same response.
- Sum of the contribution of various energy bin of the same mesh.
- Check if two meshes have the same structure (spatial mesh definition and energy binning)

How to install it

The Mesh2Vtk script is written in python 3.6. It is compound of two modules, the principal module "Mesh2vtk.py" and a modules dependency "meshtal_mod.py". The path of the location of the *meshtal_mod.py* module should be adjusted in the "*dep_path*" variable located in the first line of the mesh2vtk.py script. In addition, the script uses two other package dependencies which are "*numpy*" and "*vtkpython*". If these dependencies are not installed in the system, they could be downloaded from the following links:

- https://www.scipy.org/scipylib/download.html
- https://www.vtk.org/download/:

To install these packages from the command line execute:

```
$ pip install numpy
```

for numpy and:

```
$ pip install vtk
```

for VTKpython.

How to use it

The Mesh2Vtk script operates through an interactive text menu interface. To be launched the user has to type the command:

```
$ python –m mesh2vtk.py
```

The principal menu has four options (plus “Exit” option), see figure 1. The keywords to be used to perform actions are the words between parentheses displayed in the menu.

```
*********************************************
  Process meshtally to VTK
*********************************************

* Append meshtally file    (open)
* Display Mesh information (info)
* Write VTK file           (write)
* Mesh operation           (operate)
* Exit                     (end)

enter action :
```

Figure 1: Principal menu of Mesh2Vtk.

- Append meshtally file: The first option is used to open the meshtally file to be processed. This action is carried out with the keyword “*open*”. This keyword can be used alone or following by the meshtally filename. In case of only “*open*” keyword is entered the script will ask for the meshtally filename in the next step.

Several meshtally files can be opened at the same time. For this, the action “*open*” should be repeated as many times as there are files to be added.

```
enter action :open
enter Meshtally file name:meshmcnp5

Input files :
- meshmcnp5
enter action :open meshmcnp6 █
```

Figure 2: Append file menu

- Display Mesh information: The second option is used to display mesh information (figure 3).

```
* Mesh information          (meshinfo)
* Tally information         (tallyinfo)
* Exit                      (end)
```

Figure 3: Display info sub menu.

Mesh information is divided in two categories, the information of meshes contained in the meshtally file (*meshinfo*) and the information of a specific tally in the meshtally file *(tallyinfo)*.

The “*meshinfo*” option displays for all the meshes present in the meshtally file, the following information: the tally id, the mesh particle type, the mesh type (flux, source, source importance), the first line of the tally comment (comments of the MCNP FC card). If several meshtally files are open, the script will ask to which file the user wants to display the mesh information.

```
enter action :meshinfo
Meshtally file : meshtal
  Tally 24 : photon  flux mesh   'This EQ SDDR tally is the big mesh filtered by cells 5x5x5'
  Tally 34 : photon  flux mesh   'This UP SDDR tally is the big mesh filtered by cells 5x5x5'
  Tally 2124 : photon  flux mesh
  Tally 3124 : photon  flux mesh
enter action :█
```

Figure 4: Example of mesh information in meshtally file.

The “*tallyinfo*” information option displays the detailed information of a specific mesh. An example is shown in figure 5.

```
enter action :tallyinfo
Tallies :
- tally 24
- tally 34
- tally 2124
- tally 3124
Select tally #:24
Meshtally file : meshtal
Tally          : 24
Comments       :
     This EQ SDDR tally is the big mesh filtered by cells 5x5x5
Particle       : photon
Mesh type      : flux
Dose modif     : True
Mesh geometry  : rectangular
Mesh origin    : 0.0 0.0 0.0
X dimensions   :   8.000e+02  159I  1.600e+03
Y dimensions   :  -2.000e+02   79I  2.000e+02
Z dimensions   :  -2.300e+02  107I  3.100e+02
Energy bins    :
        flag : cells
   bin index : bin range
         0   :  Other
         1   : 150150-150181
         2   : 153174-153402
         3   : 180000-180594
         4   : 180595-180644
         5   :  Total
enter action :█
```

Figure 5: Example of detailed information of the mesh.

- Write VTK file: The third option is used to write the mesh into vtk type files. This option is used with the keyword "*write*". When this option is called a list of all meshes available to be written in vtk format is displayed, this list include new meshes produced as the result of mesh operation (see the next script option). To select the mesh to be written in VTK format, the meshtally file name (if several meshtally files are open) and the tally number should be entered. The mesh to be written should be entered one by one (if all meshes present in a meshtally file should be written, the word "all" could be used to refer to all meshes), once the desired meshes have been entered the keyword "end" should be entered for the script starts writing the vtk files. Each mesh is written in a separate vtk file. The files are written in new VTK format using XML structure. The name of the files is "*meshtallyfilename_tallynumber*"+ ".extension". The extension can be ".vtr" or ".vts" depending if the grid used to write the data is rectilinear or structured (cartesian meshes produce rectilinear grids and cylindrical mesh produce structured grids).

If new meshes are produced as the result of operation on meshes, the name of the meshtally file to be entered should be "result" and the name of the mesh is the name set by the user when an operation is carried out.

In figure 6 an example of the write option is shown.

```
*********************************************
  Process meshtally to VTK
*********************************************

* Append meshtally file    (open)
* Display mesh information (info)
* Write VTK file           (write)
* Mesh operation           (operate)
* Exit                     (end)

select mesh to write to VTK :
- meshtal
    - 24
    - 34
    - 2124
    - 3124
- meshnew
    - 114
    - 124
    - 134
    - 154
    - 164
    - 174
    - 184
    - 214
    - 224
    - 234
select mesh "file tally", finish "end":meshtal 24
select mesh "file tally", finish "end":meshnew all
select mesh "file tally", finish "end":end

Write mesh meshtal_24.vtr
Write mesh meshnew_224.vts
Write mesh meshnew_164.vtr
Write mesh meshnew_134.vtr
Write mesh meshnew_234.vts
Write mesh meshnew_174.vtr
Write mesh meshnew_114.vtr
Write mesh meshnew_214.vts
Write mesh meshnew_184.vtr
Write mesh meshnew_154.vtr
Write mesh meshnew_124.vtr
enter action :
```

Figure 6: Example of mesh2vtk write option.

- Mesh operation: The last option is the “*operate*” option. This function is used to perform operations on meshes. In figure 7 an example of the operation menu is shown.

```
* Mesh scale                (scale)
* Mesh sum                  (sum)
* Mesh difference           (diff)
* Energy bin sum            (binsum)
* Check identical mesh      (identical)
* Change correlation        (corr)
  correlation : False
enter action :
```

Figure 7: Example of Mesh2Vtk operation menu.

The operation menu has five functions: mesh scaling, sum of meshes, mesh difference, sum of energy bin, and mesh checking. The first four functions produced a new mesh as the result of the operation. For these functions the operation should be called with the keyword of the operation followed by the name the user wants to assign to this new mesh.

After entering the operation the script will ask for the meshes to be used in the operation. Depending on the desired operation, this number could be one (for "*scale*", "*binsum"*), two (for "*sum"*, "*diff"*, "*identical"*) or higher (for "*sum"*). For the "*sum*" option the list of meshes is finished with the "*end*" keyword.

For the "sum" and "scale" options two values are asked for each mesh. The first one is the tally value and the second is the scaling factor (if several meshes are open three values should be given, in this case the first value is the meshtally filename followed by the tally number and scaling factor).

The result of the sum operation is:

$$newmesh = \sum mesh_i \times factor_i$$

The sum operation is carried out only if all meshes involved in the mesh have the same structure (same mesh type, same spatial dimension and same energy bin). Meshes originated by different particles can be added. The errors of the resulting mesh are also evaluated.

The value of the relative error can be computed considering either correlated or uncorrelated values between the different meshes. The user can change correlated to uncorrelated for the error calculation using the "*corr*" option in the "*operation* " menu.

Correlated errors are calculated as the sum of the standard deviation, uncorrelated errors as the sum of the variance.

Correlated values: $\sigma_t = \sum_i \sigma_i$

Uncorrelated values: $\sigma_t = \sqrt{\sum_i \sigma_i^2}$

The relative error being the standard deviation divided by the mean value: $\varepsilon = \frac{\sigma}{x}$

The "*diff*" and "*identical*" options require only one value for each of the two meshes involved in the operation, which is the tally number (if there are several meshtally files open, the meshtally filename and the tally number).

The result of the "*diff*" option is:

$$newmesh = mesh_1 - mesh_2$$

For this option, the error values of the new mesh have been replaced by the result of the operation:

$$newmesh = \frac{mesh_1 - mesh_2}{\sigma_{12}}$$

Here σ_{12} is the standard deviation obtained adding the errors of mesh1 and mesh2 ($\sigma_{12}^2 = \sigma_{mesh1}^2 + \sigma_{mesh2}^2$). In other words, the values of the "*diff*" operation are the values of the difference between the meshes (preserving the sign) and, in the error part, the relative difference estimated in standard deviation units.

In figures 8 several examples of mesh operation are shown.

```
* Mesh scale                 (scale)
* Mesh sum                   (sum)
* Mesh difference            (diff)
* Energy bin sum             (binsum)
* Check identical mesh       (identical)
* Change correlation         (corr)
  correlation : False
enter action :scale scaled_meshtal
    - 24
    - 34
    - 2124
    - 3124
tally and scaling factor "tally factor":2124 5.
enter action :
```

Figure 8.a: Example of mesh scaling

```
* Mesh scale                 (scale)
* Mesh sum                   (sum)
* Mesh difference            (diff)
* Energy bin sum             (binsum)
* Check identical mesh       (identical)
* Change correlation         (corr)
  correlation : False
enter action :diff tally24-34
    - 24
    - 34
first mesh  "tally":24
second mesh "tally":34
enter action :
```

Figure 8.b: Example of mesh difference

```
* Mesh scale                 (scale)
* Mesh sum                   (sum)
* Mesh difference            (diff)
* Energy bin sum             (binsum)
* Check identical mesh       (identical)
* Change correlation         (corr)
  correlation : False
enter action :sum sum_24_lm
- meshtal
    - 24
    - 34
    - 2124
    - 3124
- meshtam
    - 24
    - 34
    - 2124
    - 3124
meshtally file, tally and scaling factor "file tally factor", finish "end":meshtal 24 1
meshtally file, tally and scaling factor "file tally factor", finish "end":meshtam 24 3
meshtally file, tally and scaling factor "file tally factor", finish "end":end
enter action :
```

Figure 8.c: Example of mesh sum

```
* Mesh scale                  (scale)
* Mesh sum                    (sum)
* Mesh difference             (diff)
* Energy bin sum              (binsum)
* Check identical mesh        (identical)
* Change correlation          (corr)
  correlation : False
enter action :binsum sumcell
    - 24
    - 34
    - 2124
    - 3124
select mesh :24
        flag : cells
   bin index : bin range
         0   :  Other
         1   : 150150-150181
         2   : 153174-153402
         3   : 180000-180594
         4   : 180595-180644
         5   :  Total
Enter bin index list:1 2 3 4
enter action :
```

Figure 8.d: Example of sum of “energy” bins.

All these new meshes created as the result of the operation are stored as if they were stored in the file “result”. To write these meshes in vtk, the file to be selected with the “*write*” option is “result”. The exported vtk files are named as “result_*meshname*.vtr(s)”. An example is shown in figure 9. If only one meshtal file is open, the name assigned to the result file is the filename of the meshtal file (instead of “*result*”) “*meshtalfilename_meshname.vtr(s)*”.

```
* Append meshtally file    (open)
* Display mesh information (info)
* Write VTK file           (write)
* Mesh operation           (operate)
* Exit                     (end)

select mesh to write to VTK :
- mesh
    - 24
    - 34
- meshtal
    - 24
    - 34
    - 2124
    - 3124
- meshtam
    - 24
    - 34
    - 2124
    - 3124
- meshtan
    - 24
    - 34
    - 2124
    - 3124
- Result mesh (result)
    - scale_meshtal
    - sumcell
    - sum_24_lm
    - tally24-34
select mesh "file tally", finish "end":result all
select mesh "file tally", finish "end":end

Write mesh result_scale_meshtal.vtr
Write mesh result_sumcell.vtr
Write mesh result_sum_24_lm.vtr
Write mesh result_tally24-34.vtr
enter action :
```

Figure 9: Example of writing result meshes.

Mesh2Vtk verification

The script has been verified firstly by checking visually that the data written to the vtk file were the same as the data present in the original mesh. For this, a mesh with 2x2x2 spatial bins and 2 energy bins was produced with MCNP5. The mesh elements values were substituted by numbers from 1.0 to 16.0 for the data values and from 0.01 to 0.16 for the corresponding error values. The substitution was performed to assign a different value to each mesh element and to identify them more easily. This modified mesh was translated to vtk with the script and the values written in the file were compared with original one, and checked if they were in the correct location (correctly spatially ordered, correct energy bin location, correct section data/error).

This process was repeated for each MCNP5, MCNP6 and D1SUNED mesh output format (column, matrix, specific D1SUNED format).

The second verification step consist in comparing visually maps produced by Paraview using vtk files (produced by the script) with maps produced by the MCNP mesh plotting function.

In figures 10, 11 and 12, 13, two maps of the ITER Equatorial Port 12 are shown. Figures 10 and 11 are the maps in the plane PY=0, produced by Paraview and MCNP respectively, and figures 12 and 13 are the maps in the plane PZ=139, produced by Paraview and MCNP respectively.

It can be seen in these comparisons that the map produced by Paraview is identical to the MCNP map (for each map the range of value for the color scale were identical but the difference in the color map used in Paraview and MCNP, may slightly changes the perception of the image) .

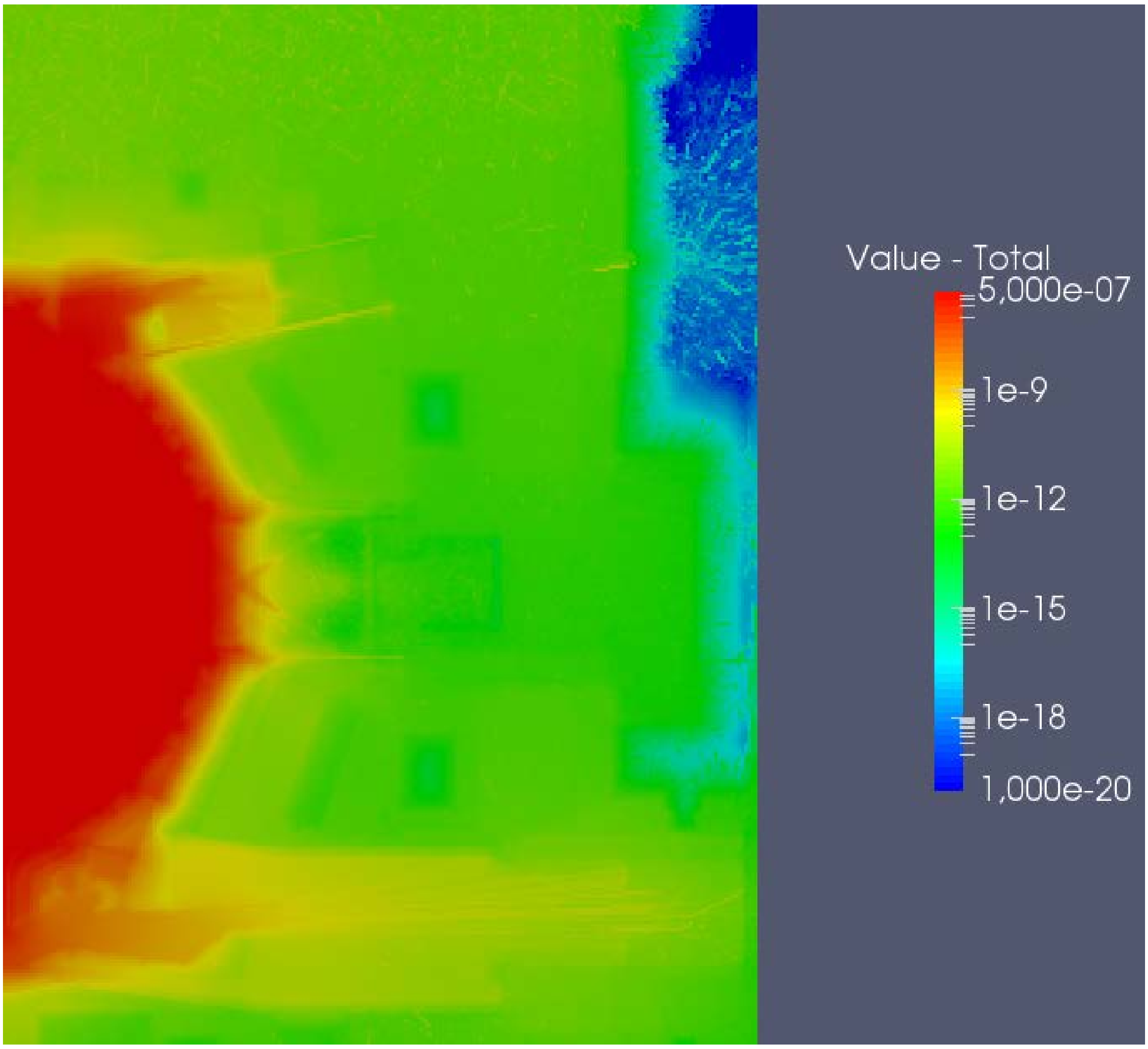

Figure 10: ITER Equatorial port 12. Map obtained with Paraview in PY=0 plane.

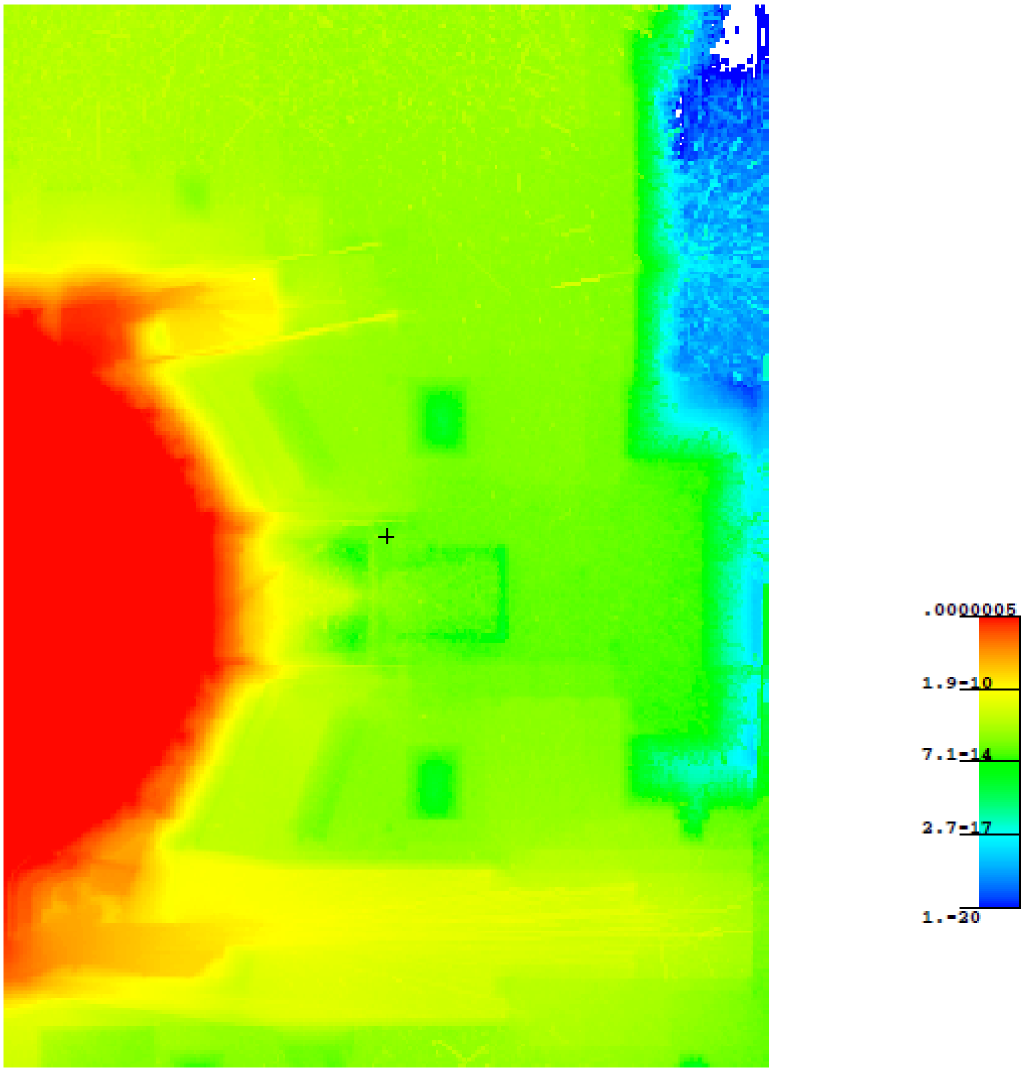

Figure 11: ITER Equatorial port 12. Map obtained with MCNP plotting function in PY=0 plane.

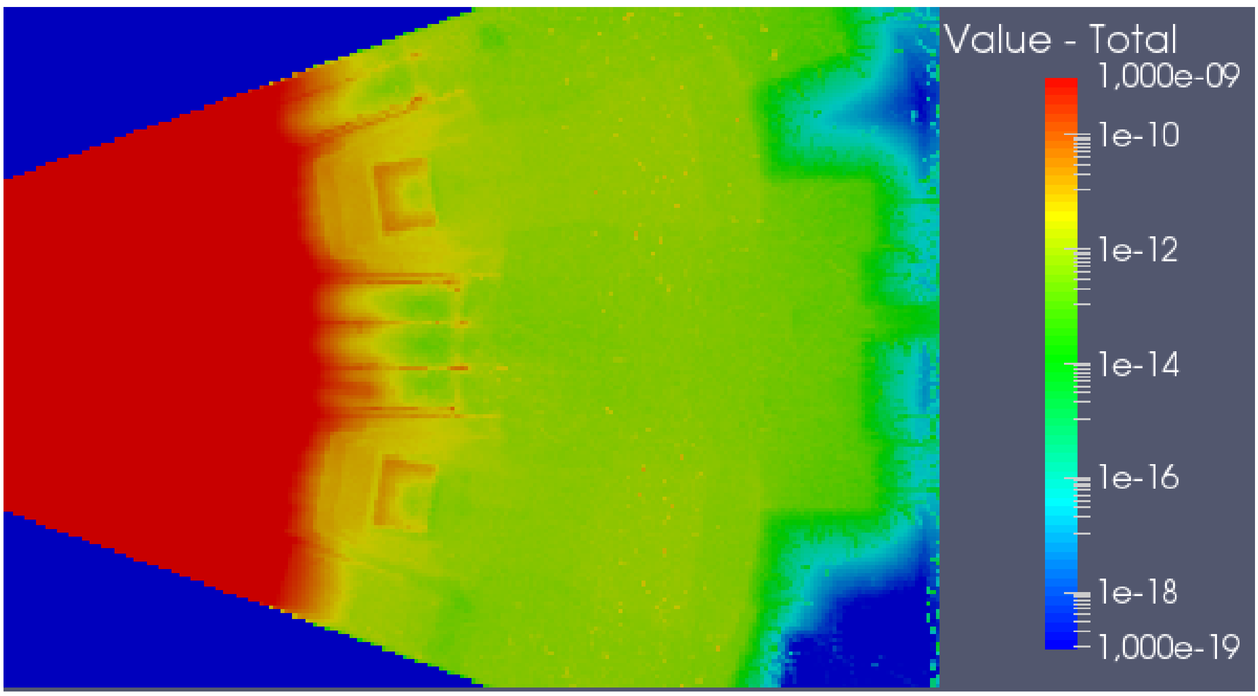

Figure 12: ITER Equatorial port 12. Map obtained with Paraview in PZ=139 plane.

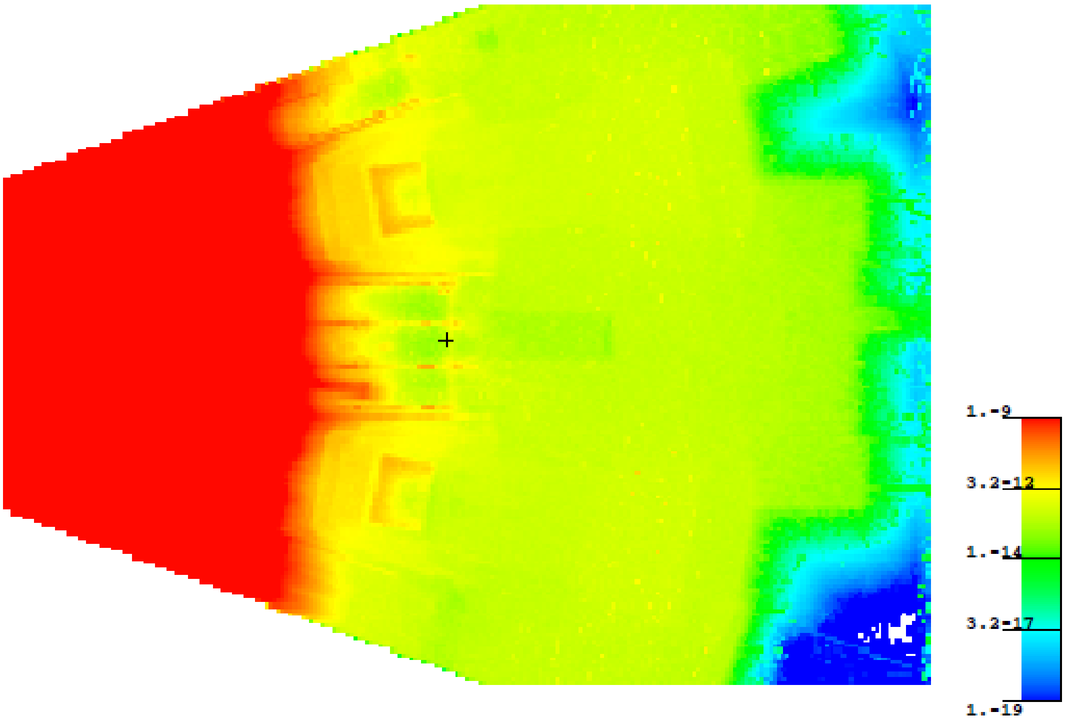

Figure 13: ITER Equatorial port 12. Map obtained with Paraview in PZ=139 plane.

Reference

1. "MCNP User’s Manual - Code Version 6.2", C.J. Werner (editor), LA-UR-17-29981, 2017

Annex I

Voxel normalization

The values stored in the CuV mesh are normalized per cm3, the value of the cell i inside a CuV voxel is $x_i = \frac{X_i}{V_i}$ (V_i: Volume of the cell portion inside the voxel).

The correct averaged value for the vtk voxel is then:

$$x = \sum_i x_i f_i$$

Where:

- x_i is the value of the cell i belonging to the CuV voxel.
- f_i is the volume fraction of the cell i belonging to the CuV voxel ($f_i = \frac{V_i}{V}$, V : volume of the voxel)

The sum is extended to all the cells inside the voxel.

The value to be assigned to the vtk voxel can be either the averaged value without normalization or the averaged value normalized by the volume of the voxel.

Two type of volumes can be considered:

1) The total volume of the voxel (V)
2) The volume of only no-zero value cells (V_o)

In case of volume normalization, the value assigned to the Vtk voxel will be:

$$x_{vtk} = x * V \quad \text{or} \quad x_{vtk} = x * V_o$$

If the cells are filtered, the volume V_o will correspond to the volume of the sum of the no-zero and no-rejected cells belonging to the voxel.

The choice between the three normalizations (no normalization, voxel volume, no-zero volume) is asked to the user when the mesh is loaded. The respective keywords for these options are :

- none : no normalization
- vtot : Voxel volume (i.e., $x_{vtk} = x * V$)
- celf : no-zero value cells volume (i.e., $x_{vtk} = x * V_0$)

Cell Filtering file

The final vtk file can be filtered by including only specific cells from original CuV file.
The list of cells to be included in the final vtk file is provided through a text file. The text file can provide either the cells value to be included in the finale vtk mesh or the cells to be rejected.
The cell value can be inserted individually or by cell range. Several values can be entered in a single or several lines. The cell range are identified by semi colon (":") between two cell values.
An example of cell definition in the filtering file is show in figure 14.

```
20 40  50:60
123 124 125 127
200:350   390:500
```

Figure 14: Example of cell filtering file

If the first line of the file contains number, the cell values of the files are considered as the cells to be included in the vtk mesh.
If the first line has the keywork "reject" the values of the files corresponds to the cell to not be included in the vtk file.

```
reject
30 31 32 90:100
850:1200
```

Figure 15: Example of cell filtering file with cells to be excluded