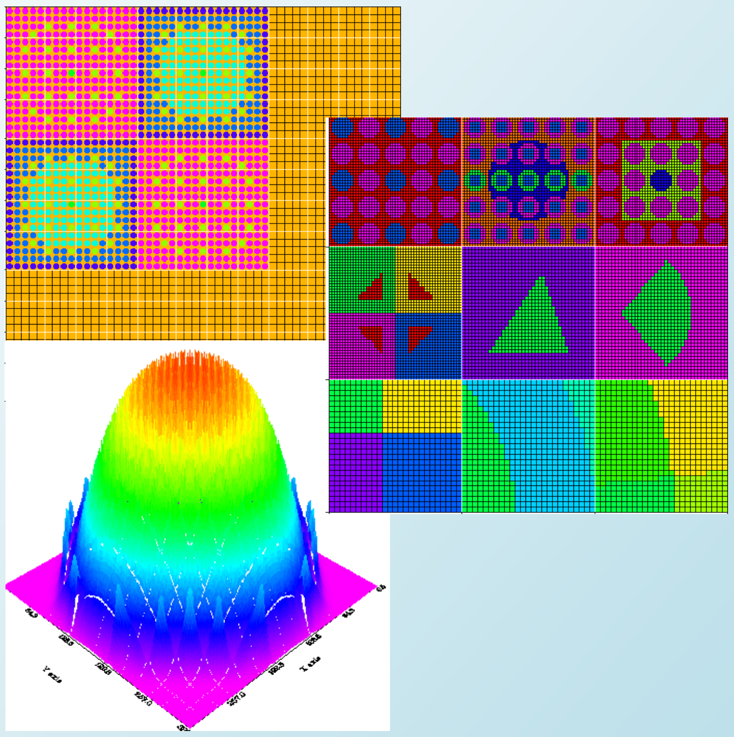
January 2017

PENMSHXP : A Cartesian Geometry Mesh Generator and Post-processing Code for PENTRAN and TITAN

PENMSHXP User Guide

C. Yi and A. Haghighat



Contents

Tables

Figures

Examples

1. Introduction

PENMSHXP, or *PENMSH* Express, is another utility tool, which combines most of the functions of *PENMSH* (Haghighat, 2000) and *PENINP* with some new features. *PENTRAN* (Sjoden and Haghighat, 1996) users can choose either *PENMSH&PENINP*, or *PENMSHXP* to build their problem models. From version 2.0, *PENMSHXP* can also be used to generate *TITAN* (Yi and Haghighat, 2010) input deck. *PENMSHXP* was originally developed for solving some problems where *PENMSH/PENINP* has memory allocation issues. Users familiar with *PENMSH* will find it is straightforward to switch to *PENMSHXP*, since *PENMSHXP* uses almost the same input deck syntax as *PENMSH*, and a *PENMSH* input deck can be used by *PENMSHXP* directly to generate the same geometry model. However, the generated *PENTRAN* input deck is not necessary exactly the same as the one generated by *PENMSH/PENINP*. For example, the FIDO sequences in Block 2 and 5 of the *PENTRAN* input deck could be different, since the same array can be represented by different sets of FIDO sequence. In most cases, the FIDO sequence generated by *PENMSHXP* tends to be more concise. Furthermore, the default values of some *PENTRAN* input deck variables have been updated in *PENMSHXP*. It is important to note that for both utilities, it is still users' responsibility to verify input variables in order to complete a meaningful *PENTRAN/TITAN* calculation.

PENMSHXP is written from scratch in FORTRAN 90 with a number of new features. Some functions of *PENMSH* might be implemented in different ways in *PENMSHXP*. For example, *PENMSHXP* utilizes dynamic memory management, and includes a new set of FIDO sequence generation routines, and a new projection procedure from the source grid to coarse/fine meshing scheme. These new features are used to improve the performance of *PENMSHXP*. Our focus in this manual is mainly on the usage of *PENMSHXP*, and some new functions we have introduced in *PENMSHXP*. Here we assume readers are already familiar with *PENMSH&PENINP*. Details of *PENMSH* and *PENTRAN* input deck syntax can be found in the *PENMSH* (Haghighat, 2000) and *PENTRAN*(Sjoden and Haghighat, 2012) manuals, respectively. We also assume users are familiar with the TECPLOT software. Finally, it is worth to note that DORT/TORT input deck generation is not available in *PENMSHXP*.

2. *PENMSHXP* Input Files

PENMSHXP represents a problem model using a 2-level meshing scheme: coarse meshes and fine meshes in Cartesian geometry. Figure 1 shows a problem model with 3x3x3 coarse meshes, and each coarse mesh is filled with uniform fine meshes. Note that the number of fine meshes could vary among coarse meshes. The main function of

PENMSHXP/PENMSH is to assign a material number for each fine mesh in the problem model.

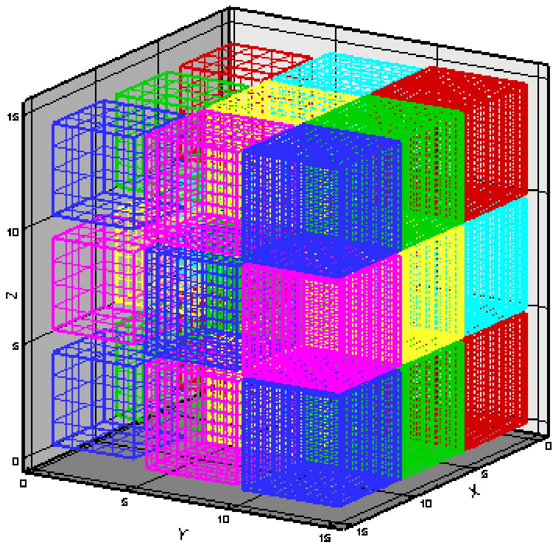


Figure 1 Coarse/fine meshing scheme.

As the original *PENMSH* code, *PENMSHXP* also arranges the coarse meshes by "z-levels" along the *z* axis. In Figure 1, the model has three z-levels, and each z-level has 3x3 coarse meshes. The main input file of *PENMSHXP* is named "*penmsh.inp*". *PENMSHXP* also requires one input file for each z-level, named *prbname#.inp*, where "*prbname*" is a string specified in "*penmsh.inp*". Table 1 compares the input files used by *PENMSH* and *PENMSHXP*.

Table 1: List of input files of PENMSH and *PENMSHXP*

File names	Description	PENMSH	PENMSHXP
penmsh.inp	Meshing parameters	Required	Required
prbname#.inp or prbname.inp	Meshing per z level (Example 9) All-zlev-in-one input (Example 15)	Required	Required
prbname.hdr	Problem header file	Required	Required
mshprm	Memory parameters	Optional	Not used
prbname.src	Source spectrum (Appendix B)	Optional	Optional
prbname.spc	Source spectrum (Example 2)	Optional	Optional
prbname.chi	Fission spectrum	Not available	Optional

prbname.mba	Material balance (Example 3)	Optional	Optional
prbname#.flx* or *prb#.flx* or *.fjn	Group fluxes (*.flx*) or current (*.fjn*), generated by *PENDATA* for PENTRAN calculations, or TITAN with –*flx* command option	Not available	Optional
in4.inp	Additional transport simulation parameters (Appendix	Not available	Optional

Note that only *penmsh.inp* and *prbname#.inp* (# is z-level number) are required by *PENMSHXP*. *penmsh.inp* specifies some general parameters of the model, such as total number of z-levels, S_N order, and so on. The cards in *prbname#.inp*, where # is the *z* level number, will define the geometry for the corresponding *z* level. More details about these cards are discussed in next section with a sample model.

Since the number of fine meshes in a coarse mesh could be quite large, it is not practical to specify a sequence of material numbers for each coarse mesh in the input files. *PENMSHXP/PENMSH* use the "overlay" scheme to assign material numbers to fine mesh. First, fine meshes in a coarse mesh are uniformly assigned a background material number. Then, a sequence of geometry shapes can be overlaid to a coarse mesh. Each geometry shape is associated with a material number. As the geometry shapes pile on inside a coarse mesh, the fine meshes falling into a geometry shape will updates their material numbers to the shape-associated one. The final material number for a fine mesh is determined by the last overlay shape, which occupies the fine mesh. If a fine mesh is not covered by any overlay shapes in the sequence, its material number will be the initial background material number. All the syntax and overlay types in PENMSH are supported in *PENMSHXP*, with one exception: The sphere overlay card is defined by the coordinates of the sphere center and the radius, not by the six boundary positions as in *PENMSH*. See the following example:

Example 1: Sphere overlay card in *PENMSHXP*

```
/overlay cm 1
1                /num of overlay
4                /shape id: 4 for sphere
12               /mat number
5  5  5  3       /overlay  type 4  : center_x, y, z and radius
```

Some new overlay shapes and features will also be discussed in the next section. Two *PENMSH* files, *prbname.hdr* (header file) and *mshprm* (pre-defined variables for memory allocation) are not used in *PENMSHXP*. *prbname.src* (source grid magnitude) and *prbname.spc* (source spectrum) are of the same format as *PENMSH*. A sample of *prbname.src* can be found in Appendix A Example 2 is a sample of *prbname.spc*.

Example 2: *prbname.spc* file in *PENMSHXP* and *PENINP*

```
1.0
0.0
0.0
```

The above *prbname.spc* defines the source spectrum for a three-group model, with source particles only in the first group. By default, all sources have the same spectrum.

prbname.chi is an optional file for *PENMSHXP*, which provides the fission spectrum of Material 1. By default, all the other materials are assigned the same fission spectrum as Material 1. *prbname.chi* has the same format as *prbname.spc*. The total number of entries in both files equals to the number of groups.

prbname.mba has the same format as used by *PENINP*. Example 3 is a sample *prbname.mba* file:

Example 3: *prbname.mba* file in *PENMSHXP*

```
 -PENINP USER INPUT MATERIAL INVENTORY FOR BALANCE-
   (The first 6 lines of this deck are comments)

 Material   Material    Target (cm3)    Density g/cc
   Name     Number      Volume          (see xsec)
---------- -------- -----------------  ------------
 Material1     1           5.475328         6.5060E+00
 Material2     2           342.4973         5.9643E+00
 Material3     3           19.67009         8.032
 Material4     4           179.5754         1.0
 Material5     5           159.6034         1.6
 Material6     6           9.022014         8.032
 Material7     7           82.36524         1.0
 Material8     8           0.0              0.0
 Material9     9           3858.04          1.0
```

With *prbname.mba*, users can provide the volume information of each material in the model. *PENMSHXP* will compare these values with the model volumes for the same materials after meshing. The comparison results will be written in the output file of prbname_out.mba. A sample of *prbname_out.mba* file corresponding to the above *prbname.mba* file can be found in Appendix A.

PENMSHXP can process the calculated fluxes in *prbname#.flx* files, where # is the energy group number. The fluxes, as well as the material and source distributions of the model, will be written into a TECPLOT binary file, called *prbname_mix.plt*. Then, users can use TECPLOT to make various flux/source/material distribution plots. The flux processing function is triggered by the command line parameter of '*-f1*'.

Example 4: Command option to load one set of flux files in specified directory

```
[home/user/]# penmshxp –i venus –f1 venus/s10
```

The '*-i*' option specifies the input file directory. For the above command, *PENMSHXP* will read the input files (*penmsh.inp*, *prbname#.inp* etc.) from the directory of /home/user/venus, and the flux files (*prbname#.flx* or *prb#.flx*) from the directory of /home/user/venus/s10. If no flux file directory is specified, the default directory is the current directory. For example:

Example 5: Command option to load one set of flux files in default directory

```
[home/user/]# penmshxp –i venus –f1
```

PENMSHXP will read the fluxes from the current directory (*/home/user/).* The flux files are generated by *PENDATA* from the results of a valid *PENTRAN* run on the same model, with only the scalar flux column necessary in the files. If there are more than one column in the files, the last column will be used. *PENMSHXP* will first try to read the flux files by the name of *prbname#.flx*. If they don't exist, it will try to read *prb#.flx*. All available flux files are processed at the same time. If a flux file for a certain group is not present in the flux directory, *PENMSHXP* will automatically skip processing that group. *PENMSHXP* can also normalize the group fluxes by the command option '*-n*'.

Example 6: Command option to load one set of flux files with normalization.

```
[home/user/]# penmshxp –i venus –f  -n
```

With the above command, fluxes from all available groups will be normalized to the maxim value. Note that '*-f*' option is the same as '*-f1*' option. The normalized fluxes, instead of values from *prbname#.flx* or *prb#.flx*, will be written into the TECPLOT binary file. Note that *PENMSHXP* can process two sets of fluxes at the same time for comparison by adding another command line parameter *-f2* with *–f1*:

Example 7: Command option to load two set of flux files for comparison.

```
[home/user/]# penmshxp –i venus –f 1  venus/s10 –f 2
```

Assuming users make two *PENTRAN* runs on the same model with two quadrature sets: S10 and S12. And the flux files from the two runs are located in the directories of /home/user/venus/s10 and /home/user/venus/s12, respectively. With the above command, *PENMSHXP* will read both fluxes sets and compare the difference. The relative difference of the two fluxes set will be written into the TECPLOT binary file ($\frac{\phi_{S_{10}}-\phi_{S_{12}}}{\phi_{S_{12}}}$). Note that the second flux set, the S_{12} run in this case, is used as the reference. Another similar command option '*–f3*' or '-fjn' can be used to read the flux current output files (*.fjn*, using njdump=1 in PENTRAN input deck BLOCK VII) generated by PENDATA (Option 9). Five columns (scalar flux moments, current, current projections along x, y and z) are necessary in the *prbname#.fjn* files. These five variables will be written in the TECPLOT file for all available groups. Flux current x, y, z projection data can be used to make a vector plot in TECPLOT.

3. A Sample Input deck

Here we use one z-level model with 3x3 coarse meshes to demonstrate the overlay scheme and other input card syntax. As listed in Table 1, only the *penmsh.inp* and *prbname#.inp* files are required by *PENMSHXP*, other files are optional. The *penmsh.inp* file for this model is given in Example 8:

Example 8: A sample *penmsh.inp* file.

```
/Sample penmsh.inp file
/Problem name
over
/# coarse z-levels, # materials, mathematica flag
1,12,0
/z-level coarse mesh boundaries
0.0, 10.0
/max. number of fine z-mesh per coarse z-level
10
/fine-to-med grid ratio along x, y, z in each coarse z-level
1     /along x for each z-level
1     /along y for each z-level
1     /along z for each z-level
/source format, # x-src mesh, # y-src mesh, # z-src mesh, ngrp, sn, pn
-1,10,10,2,47,8,3
/xsec type, xsec #comment cards, xsec Legendre order, ihm=total # of columns
1, 1, 3, 50
/Bdy conds: ibback(-x),ibfrnt(+x),jbeast(-y),jbwest(+y),kbsout(-z),kbnort(+z)
1,0,1,0,1,0
/source material id (if soure format <0)
1
/source intensity (#/cm^3/second; if <0, total source magnitude #/second)
1.0
```

The format and syntax of the *penmsh.inp* file are the same as *PENMSH*, except for several minor differences. A comment line starts with '/', '!' , or '#' in *PENMSHXP*. The first card specifies the problem name, which is used to construct various output file names. It is recommended that the name is less than 10 characters. The last entry in the second card is not used. Originally, it is the flag for Mathematica data output in *PENMSH*. The third card defines the z-level boundaries. The number of entries should be *# zlevel+1*. The boundaries are entered in an increasing order. The next card specifies the maxim *z* fine meshes for each *z* level (number of entries = number of z level), although the actual *z* fine mesh numbers are given in the z-level input file (*prbname#.inp*). *PENMSHXP* keeps this card here only for compatibility with *PENMSH*. Fine-to-medium grid ratio is a parameter used in *PENTRAN* for multi-grid acceleration. The card has three lines for the three axes *x*, *y* and *z*. Users specify the ratios for every z level (number of entries = number of *z* level) on each line. In this case, the model only has one z level, and the ratio is one.

A negative source format entry (*s_format*) means that a uniform source is distributed in one material. Users can define a number of source materials. For example, by assigning *s_format=-3*, sources will be uniformly located in 3 materials. The material ids and source intensities are given in the last two cards. The number of entries in these last two cards should be equal to the absolute values of *s_format*. Note that the source intensity has a unit of #of particles/sec.cm^3. A negative value can be used to define source magnitude (#of particles/sec) deposited in the material. *s_format=2* is used to specify *k-effective* calculation for PENTRAN. Other positive values of *s_format* are undefined. Besides the sources defined in *penmsh.inp*, *PENMSHXP* always looks for *prbname.src* file in the input directory unless *s_format=2*. If *prbname.src* is present and processed, *PENMSHXP* will add up both sources (uniform material source in *penmsh.inp*, and source grid defined in *prbname.src*). The source grid numbers along *x*, *y* and *z* are defined by the next three entries after *s_format* in the same card. A sample *prbname.src* file is provided in Appendix B.

Example 9: A sample *prbname#.inp* file.

```
/Sample prbname#1.inp file
/ncx, ncy, maxfinz  (maxfinz < 0, add z-fine per cm below y-fine)
3, 3, 1
/ x-fine mesh per CM (# of FM along x axis for each CM)
25 25 25
50 40 40
50 50 50
/ y-fine mesh per CM (# of FM along x axis for each CM)
25 25 25
50 40 40
50 50 50
/ cm boundaries along x-axis
0.0, 5.0, 10.0 15.0
/cm boundaries along y-axis
0.0, 5.0, 10.0 15.0                                    / (to be continued in next page)
```

```
/ Sample prbname#1.inp file (continued)
/ CM type (negative entry indicates overlays in that CM)
1   2    3
-1 -1   -1
-1 -1   -1
/ sub-region: number of material regions per coarse mesh
4  3  4
4  1  1
1  1  1
/ coarse mesh  1 (cm type 1 with 4 rectangular regions)
 0   2   0   2         /left boundaries for the 4 regions
 2   5   2   5         /right boundaries for the 4 regions
 0   0   3   3         /bottom boundaries for the 4 regions
 3   3   5   5         /top boundaries for the 4 regions
 2   4   7   10        /mat numbers for the 4 regions
/ coarse mesh  2 (cm type 2 with 3 circular regions)
 0    7    10            /inside radius for region 1, 2 and 3
 7   10   15            /outside radius for region 1, 2 and 3
 0    0     0            /CM bot not used (legacy penmsh)
 5    5     5            /CM top not used (legacy penmsh)
 7    5     6            /mat numbers for the 3 regions
/ coarse mesh  3   (cm type 3: circle and line)
 0    0    13   13          /inside radius for the 4 regions
 13 13   20   20          /outside radius for the 4 regions
 0    6    0     6            /bottom line in degree, 0=CM bottom)
 6    0    6     0            /top line in degree, 0=CM top)
 7    8    9    10           /mat numbers
/ coarse mesh  4  (cm type 1, 4 rectangular regions with overlay)
 0     2.5    0     2.5        /left boundaries for the 4 regions
 2.5   5     2.5   5           /right boundaries for the 4 regions
 5      5     7.5   7.5        /bottom boundaries for the 4 regions
 7.5  7.5   10   10          /top boundaries for the 4 regions
 1     4       7   10          /mat numbers
 / coarse mesh  5    (only one region)
 2                         /mat #
/ coarse mesh  6
 1
/ coarse mesh  7
 12
/ coarse mesh  8
 11
/ coarse mesh  9
 12
/overlay cm 4 :      4 quarter circles
4                           /num of overlays
21  22  23  24       /quarter circles with different orientation See Figure 2
12  12  12  12       /mat number
3    4    8    9        /overlay #1: left right, bot , top
1    2    8    9        /overlay #2: left right, bot , top
1    2    6    7        /overlay #3: left right, bot , top
3    4    6    7        /overlay  #4:  left right, bot , top    / (to be continued)
```

```
/Sample prbname#1.inp file (continued)
/overlay cm 5: triangle
1                 /1 overlay shape
3                 /shape id=3, triangle
7                 /mat
6 6   9 6   8 9   /A(x,y) B(x,y), C(x,y)
/overlay cm 6: Sector
1            /1 overlay shape
51           /shape id=51, sector overly
7            /mat
11 7.5    12.732 5.5    12.732 9.5   /O(x,y) B(x,y), C(x,y)
/overlay cm 7: lattice repeated circle
2             /1 overlay, negative sign means lattice structure overlay block
-2  -2        /shape id =-2  repeated circle
1  4          /mat
/circle left, right, bot, top (first overlay)
0.1 0.9 10.1 10.9
/rep_i, rep_j, rep_k, pitch_x, pitch_y, pitch_z (lattice number and element dist.)
5   5  1   1  1  0
/circle left, right, bot, top (second overlay)
0.1 0.9 10.1 10.9
/rep_i, rep_j, rep_k, pitch_x, pitch_y, pitch_z (lattice number and pitch size)
3  3  1   2  2  0
/overlay cm 8: lattice rectangular within circle
3               /3 overlays
2 -2 -1         / id: 2 circle (regular) id -2: circle (lattice); id: -1 rect. (lattice);
3 -1  4         /mat numbers. negative -> element mat number card
/circle left, right, bot, top (first overly)
6.0 9.0 11.0 14.0
/circle left, right, bot, top (second overlay)
5.1 5.9 10.1 10.9
/lattice card, triggered by overlay type id=-2 (second overlay)
5  5  1   1  1  0  /num of lattice elements and pitch size, see overlay cm7
/lattice element mat #, triggered by mat#=-1,FIDO support(second overlay)
10R1 5R7 10R1   /mat #'s for 25 elements (5x5x1)
/rectangular left, right, bot, top (third overlay)
5.3   5.7   10.3  10.7
/rep_i, rep_j, rep_k, pitch_x, pitch_y, pitch_z (third overlay)
5  5  1   1  1  0
/overlay cm 9
3              /num of overlays
1  -2  1       /shape id 1, rectangular
9   1  3       /mat number
/left rgt bot top  (first overlay)
11.0 14.0 11.0 14.0
/circle left, right, bot, top (second overlay)
10.1 10.9 10.1 10.9
/rep_i, rep_j, rep_k, pitch_x, pitch_y, pitch_z    (lattice number and pitch size)
5  5  1   1  1  0
/left rgt bot top  (third overlay)
12.1 12.9  12.1  12.9
```

Figure 2 shows the model for the above input files.

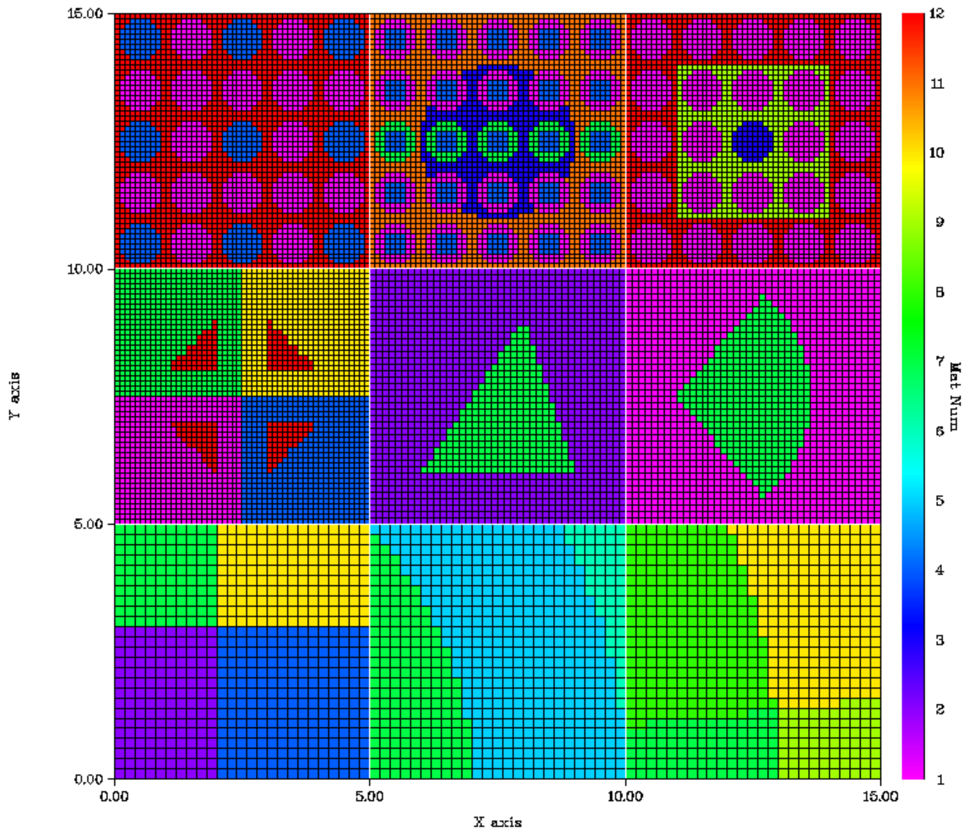


Figure 2 Meshing plot for the sample input files generated by PENMSHXP.

There are nine coarse meshes in this z level. The first three coarse meshes (the bottom row with CM 1 on the left) are CM type 1, 2 and 3 without overlay structures. CM 4 is a coarse mesh with four quarter-circle overlays. They follow the same syntax as *PENMSH*. CM 5 and 6 demonstrate two new overlay shapes in *PENMSHXP*: triangle and sector, with the overlay ID 3 and 51 respectively. Both overlay shapes are specified by the coordinates of three vertexes. Note that for a sector overlay, the first vertex should be the center of circle (See the overlay for CM 5 and 6 cards in the sample file). Table 2 lists all the supported overlay shapes in *PENMSHXP*.

Table 2 Overlay shapes in *PENMSH* and *PENMSHXP*.

Shape	Shape ID	PENMSH Spec.	PENMSH XP Spec.
Rectangular	1	-x, +x, -y, +y	-x, +x, -y, +y , (-z, +z)*
Cylinder (along z)	2	-x, +x, -y, +y	-x, +x, -y, +y, (-z, +z)
Quarter Cylinder	21,22,23,24	-x, +x, -y, +y	-x, +x, -y, +y, (-z, +z)

Cylinder (along y)	25	Not available	-x, +x, -z, +z
Cylinder (along x)	26	Not available	-y, +y, –z, +z
Arbitrary Triangle	3	Not available	Ax, Ay, Bx, By, Cx,Cy, (-z, +z)
Prismatic Right triangle	31,32,33,34	-x, +x, -y, +y	-x, +x, -y, +y, (-z, +z)
Sphere	4	-x, +x, -y, +y ,-z, +z	Ox, Oy, Oz, r, (-z, +z)
Sector	51	Not available	Ox, Oy, Ax, Ay, Bx, By, (-z, +z)
Hexagon	6	Not available	Ox, Oy, Ax, Ay

*( ) means the inside specification numbers are optional. (Brackets are not included)

As discussed in Example 1 in Section 2, the sphere overlay shape is specified differently in *PENMSHXP*. All the other overlay shapes follow the same syntax as *PENMSH*. For any overlay shape card in *PENMSHXP*, two additional z axis boundary values (-z, +z) can be added at the end of the specification sequence. If these two values present, the overlay shape will only appear in between –z and +z.

CMs 7, 8, and 9 are used to demonstrate the lattice structure of overlay shapes. Lattice overlays can be specified by a negative value in the ‘overlay type’ card, as demonstrated in the overlay block for CM7. For example, ‘-2’, as an overlay type, specifies a lattice circle structure. For a lattice overlay, additional specification card(s), after the ‘overlay boundary’ card, are required to describe the lattice structure. As shown in the CM 7 overlay block, the first additional card, titled ‘number of lattice element and pitch size’, is composed of six numbers: the first three are the numbers of lattice elements along x, y, and z, and the next three numbers are the pitch sizes (distance between elements) along x, y, and z. Note only lattice in Cartesian Geometry is supported. The second additional card, called ‘lattice element material’ card, is required only if lattice elements use different materials. This card is required by a negative overlay material number (See CM8 Overlay 2). The card is to specify the material number for each lattice element. This ‘lattice element material’ card is placed after the ‘number of lattice element and pitch size’ card. The total number of entries should be equal to the number of lattice elements. The numbering of lattice elements follows the same ordering as coarse/fine mesh numbering. (i.e. increasing along x first, then y, then z). FIDO input (R and Q) is supported in this card. More details about FIDO can be found in *PENTRAN* manual. NOTE: FIDO (R and Q) input format is supported only in the following cards: overlay type, overlay material, and overlay lattice element material.

Several overlays can be cast onto the same individual fine mesh. A fine mesh takes the last overlay on it as its final material number. By default, all the x, y, and z coordinates used in the CM overlay specification cards are the global coordinates. If a negative number is given in 'number of subregion' card, the entries are taken as local coordinates relative to the left bottom corner of coarse mesh.

4. Practical Notes for Users

4.1. Shared overlay

As demonstrated above, the overly scheme allows users to specify a sequence of shapes to a given coarse mesh. If the sequence is very large and multiple coarse meshes share the same complicated sequence, it is not practical to copy and paste the same sequence repeatedly for different coarse meshes. To overcome this, *PENMSHXP* supports a "shared overlay" feature by simply modifying the "CM type" block. A regular CM type block (5x5 coarse meshes) of z-level input file may look like:

Example 10: A 5x5 coarse mesh type card

```
/ CM type for each CM (<0 indicates overlay)
1   1   1   1   1
1  -1   1   1   1
1   1   1   1   1
1   1   1   1   1
1   1   1   1   1
```

In the above example, all the 5x5 coarse meshes are of type 1, and overlay is present in CM 7 (type "-1"). To copy the CM 7 overlay to other coarse meshes, we can modify the CM type card as shown in Example 11:

Example 11: Shared overlay coarse mesh type card

```
/ CM type for each CM (<0 indicates overlay)
1  1  1   1   1
1 -1  71  71  1
1  1  71  1   1
1  1  1   1   1
1  1  1   1   1
```

Now the coarse mesh 7 overlay structure will be also used for CM 8, 9 and 13, which are indicated by their corresponding "71" CM type number, in which the first digit '7' is the coarse mesh number whose overlay will be copied. '1' is coarse mesh type as usual.

More than one coarse mesh overlay can be shared.

Example 12: Multiple shared overlay coarse mesh type card.

```
/ CM type for each CM (<0 indicates overlay)
1     1    1   1   1
1     1    1   1   1
-1   -1    1   1   1
111   121  1   1   1
1     1    1   1   1
```

In this case, CM16 shares an overlay with CM11, and CM17 shares one with CM12.

Note:

1. Only the cm type card input need to be modified, the rest input cards should remain same. The overlay structure indicated by a negative CM type number can be specified as usual.

2. Shared overlay feature only works when copying an overlay structure from a previously defined coarse mesh. The following card is invalid:

Example 13: Invalid shared overlay coarse mesh type card.

```
/ CM type for each CM (<0 indicates overlay)
1     1    1  1  1
1     1    1  1  1
-1   171   1  1  1
111  -1    1  1  1
1     1    1  1  1
```

In the above example, CM12 tries to use CM17 overlay, which hasn't been read in yet. Therefore, this card is invalid. However, copying from previously copied CM would work:

Example 14: Chained shared overlay coarse mesh type card.

```
/ cm type for each cm (<0 =look for overlay)
1     1    1  1  1
1     1    1  1  1
-1   111   1  1  1
111  121   1  1  1
1     1    1  1  1
```

In this case, CM17 copies from CM12, which itself copies from CM11. Therefore, this card would be the same as:

```
/ cm type for each cm (<0 =look for overlay)
1     1   1  1  1
1     1   1  1  1
-1   111  1  1  1
111  111  1  1  1
1     1   1  1  1
```

4.2. All-zlev-in-one input file

Since version 2.65b, PENMSHXP also reads an ‘all-zlev-in-one’ input file (*prbname.inp*), instead of a number of z-level input file (*prbname#.inp*). The ‘all-in-one’ input file contains information for all z-levels. Note that *penmsh.inp* file remain the same.

The following example shows a sample all-zlev-in-one input file:

Example 15: A sample all-zlev-in-one input file.

```
/All-zlev-in-one input file: prbname.inp
/ncx, ncy
3, 2
/ cm bounds along x-axis
0.0  15.0  20.0  25.0
/cm bounds along y-axis
0.0000E+00  15.0000E+00  2.0000E+01
/CM cards
cm=1 1 1
2           /cm(1,1,1) mat num
5 5 5       /cm(1,1,1) fine mesh number along x y z
cm=2 1 1
2
15 5 12
cm=3 1 1
2
5 5 12
cm=1   2 1
2
15 5 12
cm= 2  2 1
2
5 5 12
cm=3 2 1
2
5 15 12
cm=1  1 2
-1      /negative mat number -> overlay exist
40 40 40
/overlay
2
-2  -4
5   -1
/first overlay (circle)
0.0 1.0 0.0 1.0
3 3 1    2.5 2.5 0
/second overlay (sphere)
10.0 10.0 1.0   1.0
3 3 3    2.5 2.5 2.5
5 7 9 2Q3  2Q9
cm=2 1 2
2
5 15 12
cm=3  1 2
1
15 15 12
cm=1   2  2
2
5 15 12
cm=2 2  2
2
5 5 12
cm=3 2 2
2
15 5 12
```

The first card in the ‘all-zlev-in-one’ input file specifies the number of coarse meshes along x and y. Note number of coarse meshes along z axis is specified in the *penmsh.inp* file. The next two cards specify the coarse mesh boundaries along x and y axis, respectively. The rest of the cards are coarse mesh cards, used to define each coarse mesh individually. A coarse mesh card starts with a marker ‘*cm=*’, followed the coarse mesh’s x, y, and z index. For example, ‘*cm=1 1 1*’ marks the definition of the coarse mesh (1 1 1). Each coarse mesh requires a marker, ‘*cm=i j k*’, but they can appear in any order. Following the coarse mesh marker, two cards are required: the first card defines the coarse mesh material; the second card specifies the number of fine meshes along x, y, and z axis in the coarse mesh. A negative coarse mesh material number indicates that there are overlay structures in the coarse mesh. Overlay cards follows the same syntax as in the ‘*prbname#.inp*’.

4.3. Automatic mesh size generation

PENMSHXP supports automatic mesh size generation by using the command option “-maxmesh *max_fm_size*”, where *max_fm_size* is a real number, representing the global maxim fine mesh size (in centimeters) along any axis for all coarse meshes.

Example 16: Command option to set maxim size for all fine meshes.

```
[home/user/]# penmshxp –maxmesh 1.0
```

The above command ensures that the fine mesh size along any axis will not exceed 1.0 cm in all coarse meshes. For a given coarse mesh, we use S_i and d_i to denote the coarse mesh and fine mesh size, respectively, where $i = x, y, or\ z$, is one of the three Cartesian axes. Generally, the number of fine meshes along an axis, n_i, is the same as the entry, m_i, given in the input file (file: *prbname#.inp*; card: # of fine mesh per CM for x, y, or z), and d_i can be calculated by $d_i = \frac{S_i}{n_i}$. When *max_fm_size* is given by the “-maxmesh *max_fm_size*” command option, *PENMSHXP* takes m_i specified in the input file, as a multiplier for the given coarse mesh, and the number of fine meshes along an axis, n_i, is calculated by:

$$n_i = ceiling(\frac{S_i}{\max\_fm\_size}) \times m_i$$

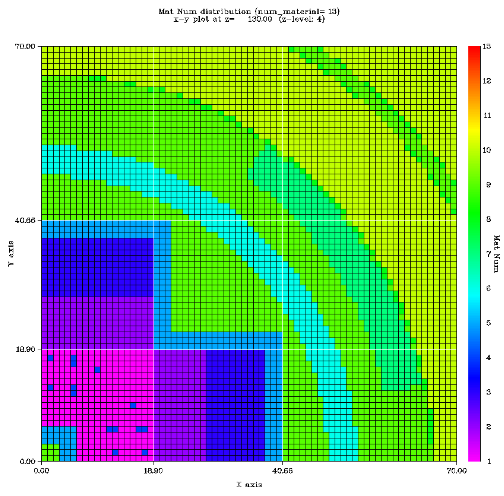
Note that $ceiling$ is a FORTRAN intrinsic function, which returns the smallest integer that is larger than the function’s argument.

The following examples illustrate the usage of the “-maxmesh” command option.

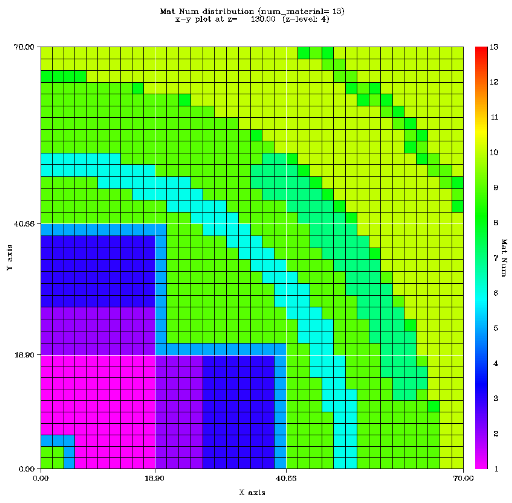
Example 17: Input cards for –maxmesh option with unity multipliers.

```
/ncx, ncy, maxfinz  (maxfinz < 0, add z-fine per cm below y-fine)
 3,3,1
/x-fine mesh per fuel pin CM (coarse mesh)
 1 1 1
 1 1 1
 1 1 1
/y-fine mesh per CM
 1 1 1
 1 1 1
 1 1 1
```

With the above cards, the effect of –maxmesh is shown in Figure 3.



(A) `penmshxp` *`-maxmesh 1.0`*



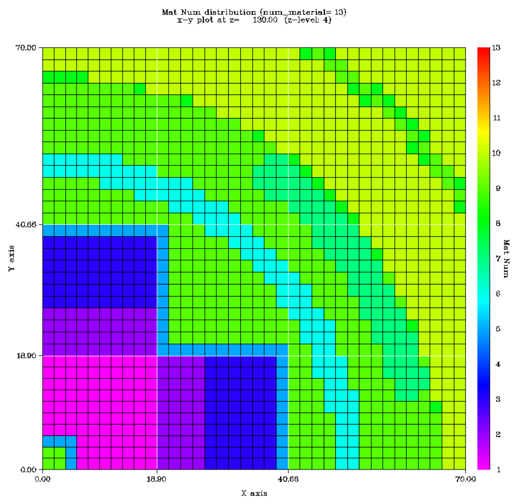
(B) `penmshxp` *`-maxmesh 2.0`*

Figure 3 Automatic fine mesh size with the –maxmesh command option

The multipliers in the input file can be used to refine specific coarse meshes as shown in the following example.

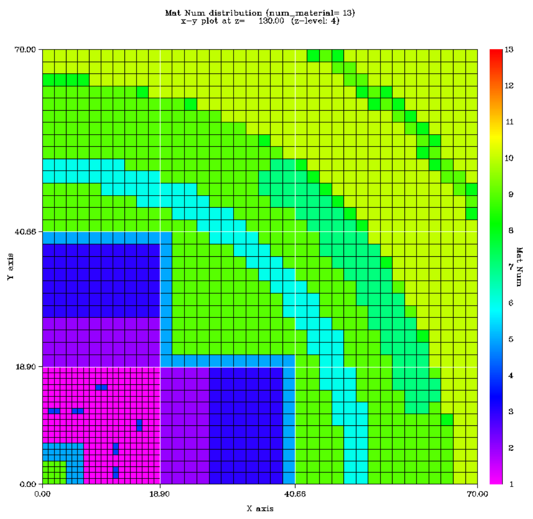
Example 18: Input cards for –maxmesh option with non-uniform multipliers.

```
/ncx, ncy, maxfinz  (maxfinz < 0, add z-fine per cm below y-fine)
 3,3,1
/x-fine mesh for each coarse mesh (CM)
 2 1 1
 1 1 1
 1 1 1
/y-fine mesh for each CM
 2 1 1
 1 1 1
 1 1 1
```



(A) `penmshxp -maxmesh 2.0`

with Example 17



(B) `penmshxp -maxmesh 2.0`

with Example 18

Figure 4 Automatic fine mesh size with non-uniform multipliers.

As shown in Figure 4B, the fine mesh density for CM1 is increased by a factor of 2 due to the non-uniform multiplier setting used in Example 18. When using the –maxmesh option, care should be taken to avoid very large models.

Example 19: Command option to set maxim sizes along x, y, and z axes.

```
[home/user/]# penmshxp -maxmesh 1.0 2.0 3.0
```

The above example will set the maxim mesh sizes along x, y, and z axes as 1.0, 2.0, and 3.0cm, respectively.

5. ***PENMSHXP* Output Files**

Table 3 lists the output files by *PENMSHXP*.

Table 3 *PENMSHXP* output flies.

File names	Description
prbname_out.pen	PETRAN input deck
prbname_out.mba	Material balance tables
read.log	Processing log file
prbname_mix.plt.	TECPLOT binary file, contains all the meshing data
prbname.mcr	TECPLOT macro file
prbname#.png	z-level plots, where # is the z level number

prbname_out.pen is the generated *PENTRAN* input file. For very large models, it could take a while to generate the FIDO characters in the PENTRAN input deck. Users can use "*-offf90*" option to turn off the .f90 output file, if *PENMSHXP* is used only for testing the model geometry. Another command option in version 1.6 or later is *–nofido*, which turns off the FIDO sequence generation in the source section of *prbname_out.pen*.

prbname_out.mba contains meshing and material balance information of the model. A sample of *prbname_out.mba* can be found in Appendix A, which contains two sections: The first section of the file lists the material information for each coarse mesh in the model. The second section provides the volume information for each material in the model. If the prbname.mba input file exists in the input file directory, the second section also provides the comparison results between the target volume (provided in prbname.mba) and the model volume (calculated by *PENMSHXP*) for each material. More details can be found in the PENTRAN manual.

read.log is the processing log file. All the data processed in the input files will be recorded in the log file. This file can be useful in case of *PENMSHXP* running into unexpected errors. Error messages and warnings will be displayed both on screen and in the log file. The default log file name is *read.log*. Users can use '*-l*' option to specify the log file name. For example:

Example 20: Command option to specify log file name

```
[home/user/]# penmshxp –l mylog.txt
```

In the above command, *PENMSHXP* reads the input files from current directory, and all the processing log will be written into a file named mylog.txt.

prbname_mix.plt is a TECPLOT binary file with a point-wise structure. A fine mesh corresponds to one point. Each fine mesh has at least five variables: x, y, z, mat_num, and source intensity. If the flux files are properly processed, the group flux variables will be added in the list: grp1, grp2, etc. A coarse mesh forms a zone in TECPLOT, and all the coarse meshes in one z level form a zone group.

prbname.mcr is a TECPLOT macro file. It can be loaded within TECPLOT: menu -> file -> macro -> play, then select the file. The macro will make a scatter plot in TECPLOT, with the *mat_num* as the contour variable. Therefore, after the macro file is loaded, a material distribution plot of the model should appear in the plot window. The TECPLOT output files can be disabled by using a command option *"-offplt"*.

prbname#.png is an image file to show the x-y plot in the middle of each z-level, which is similar to the z-level postscript files generated by PENMSH. In *PENMSHXP*, the default format of the plots is PNG, which is compatible with most image viewer software. Details of plotting options are discussed in the next section.

6. *PENMSHXP* Plotting Options

PENMSHXP generates z-level plots using the DISLIN graphic library, which is written by Helmut Michels, and can be found at http://www.dislin.de. The library is free for non-commercial use. A separate version of *PENMSHXP* is available without the support of this DISLIN plotting function. Users can still use TECPLOT to generate various plotting. Table 4 lists the command options for DISLIN plotting options.

Table 4 *PENMSHXP* plotting options with DISLIN library

Options	Arguments	Description
-plot ***n***	*n* is an integer to specify x, y, and/or z plot	Mid-level plots *n*=1, 2, or 4: plotting on each z, y or x level mid-plane. Users can add the numbers up to plot along multi axes. e.g. n=5 will plot along z and x axes; n=7 will plot along all there axes.
-plotx ,-ploty, -plotz ***pos1, pos2…***	*pos1*, *pos2* … are position values	Specified position plots To plot on a number of positions along an axis. e.g. *–plotx 1.2 3.0* will plot on the planes of x=1.2 and x=3.0
-msf ***m***	*m* is an integer to specify to plot *mat*, *src*,	To specify material, fixed source, or flux plots *m*=1, 2, or 4: plotting material, fixed source, or

	and/or *flux*	flux. Users can add the numbers up to plot multi variable distributions. e.g. n=5 will plot mat and flux; n=7 will plot *mat*, *src*, and *flux*. If a negative number is used, an ASCII data file (.cat) is also generated.
-3d	N/A	Turn on 3-d contour plotting. Without using this option, 2-d contour plots are generated.
-3dpos x y z	View point coordinates	Default (3 3 3)
-size *f*	*f* is a real number to specify a size multiplication factor	To control the *.png* file size e.g. –size 2.0 will increase the plot size by a factor of 2. *f* ranges from 0.1 to 9. By default *f=1*. The plot size is 900 pix wide. The height depends on the model x-y size ratio.
-nofm		Turn off fine mesh line drawing for all CM. Note FM lines automatically turned off if FM size is less than 8 pixel in a CM
-offpng	N/A	Turn off plotting
-z	z lever number	Plot a single z level.
-color	<0-8>	Define a color map, default is 3, a rainbow color map.
-cm		Color map help info

By default, *PENMSHXP* will generate a material distribution plot per *z* level at the z-level mid-plane. The default size of the image is 900 pixels along x axis, and y-size is automatically calculated based on the model geometry (x-y ratio).

Figure 5 shows a sample z-level plot from a reactor model.

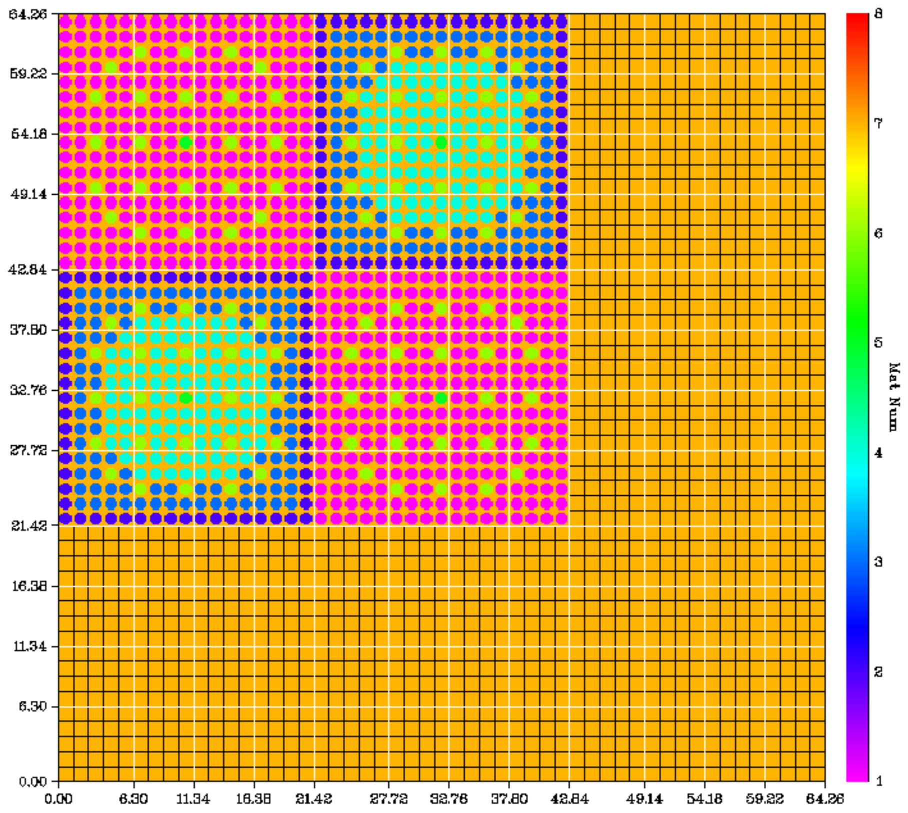
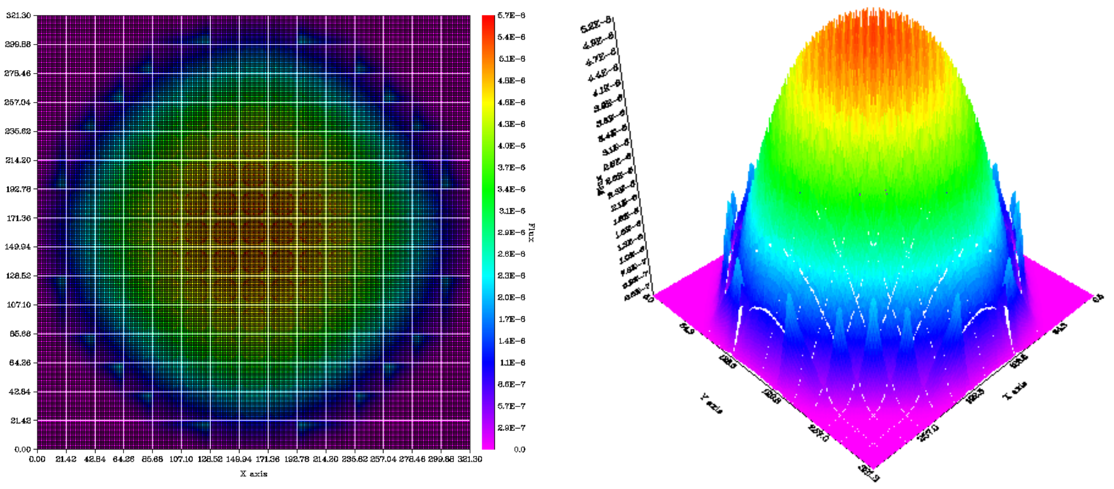


Figure 5 2-D material distribution plot for a reactor model

In Figure 5, coarse meshes boundaries are drawn with white lines. Fine meshes are represented with black lines. Note if the size of a fine mesh is less than 8 pixels, fine mesh lines will be automatically disabled in the coarse mesh for a clearer view of the material distribution. As shown in Figure 5, the fine mesh boundary lines (black lines) are not visible in the fuel region because of the small size of the fine mesh, but can be seen in the reflector region. The coordinates of the coarse mesh boundaries are labeled along the axis. Each fine mesh is filled with color based on the material number. The legend is shown on the right side of the figure.

Users can use '-plot' option to specify the axis along which the mid-plane will be plot on (See Table 4). *PENMSHXP* can also plot on user-specified positions along an axis with the *-plotx*, *-ploty* and *–plotz* options. The *'-msf'* option is used to specify different plotting variables: materials, fixed source and/or flux. If *-msf 4* or higher is used, *PENMSHXP* will try to load the flux files. Users can use *'-f'* option to specify the flux file directory. (See Section 2). If '*-f2'* option is used, the difference between two flux sets will be plot. If *–msf -4* is used, beside the plot graphic files, *PENMSHXP* will also output an ASCII data file (*.cat*) for each plot. The '-3d' option will turn on the 3-D contour plot. Figure 3 shows 2-D and 3-D flux distribution plots on the same plane.



(a) 2-D plot (b) 3-D contour plot

Figure 6 2-D and 3-D contour plots with PENMSHXP

Note that for version 2.5 and above, PENMSHXP will overwrite the existing PNG files with the same name, instead of generating new files. The naming convention is defined as:

<prbname's first 3 letters>_<xyz> <plot number><ic><msf>.png

For example, if the problem name is 'test', '*tes_z1cm.png*' will be the first level material distribution plot (at mid-level). Here '*c*' means a mid-zlevel plot, and '*m*' indicates a material distribution. While '*i*' indicate is a specific location plot (when –plotx, -ploty or –plotz is used), instead of a mid-level plot, '*s*' and '*f*' mean source distribution plot and flux distribution plot, respectively.

For a 2-group model, a command option:

'-ploty 1.3 4.5 –msf 7 –f '

will generate 8 plots assuming flux files (*.flx*) are successfully loaded from current director. They are material, source, and flux distribution at y=1.3cm and 4.5cm. The file names are:

tes_y1im.png : material plot at y=1.3
tes_y1is.png : source plot at y=1.3
tes_y1if1.png : group 1 flux plot at y=1.3
tes_y1if2.png : group 2 flux plot at y=1.3
tes_y2im.png : material plot at y=4.5
tes_y2is.png : source plot at y=4.5
tes_y2if1.png : group 1 flux plot at y=4.5
tes_y2if2.png : group 2 flux plot at y=4.5

A large set of graphic files could be generated, especially if the number of group is large. User can refer the screen output, the log file, or the title line within a graphic file for information on each plot. Users can turn off plotting entirely by using a command option *–offpng*. '-z' option is used to plot a single z level material distribution. This is designed to help user to adjust meshing parameters while building the problem model level by level. When *'-z n'* option is used (where *n* is the z-level number), only *penmsh.inp* and *prbname#n.inp* are required to build z-level *n*.

7. Phantom Binary File Handling

PENMSHXP can build a model based on a phantom binary file by using the command line option *'-hrt'*, followed by the name of the phantom input file (user-defined).

Example 21: Command option to load binary medical phantom file.

```
[home/user/]# penmshxp –hrt  phantom.inp
```

With the above command, *PENMSHXP* attempts to open *phantom.inp* in the current directory. *penmsh.inp* and *prbname#.inp* files in Table 1 are not required for a phantom model. Instead, *PENMSHXP* builds the model material and source distributions based on the phantom binary files: a voxel-wise material attenuation file and an optional voxel-wise source activity file. Some parameters of the phantom are defined in *phantom.inp*, such as the binary file name(s), number of voxels in the phantom, etc. *PENMSHXP* renders each voxel as a fine mesh. Users can define the coarse mesh boundaries in the unit of # voxel in the *phantom.inp*. All the other input files in Table 1 remain the same syntax. If the input file name is not specified, the default name is *heart.inp*.

Example 8: A sample of phantom input file

```
/phantom input file: hrt2.inp
/prbname
hrt2
/# of voxels along x, y,and z
65 ,61 ,58
/voxel size along x, y, and z
0.3125, 0.3125, 0.3125
/mat binary file name, src binary file name (optional)
hrt_atn.bin      hrt_act.bin
/data type for each binary file
/binary datatype= 0  :  4 byte integer (default)
/binary datatype= 1  :  4 byte real
/binary datatype= -1 : 4 byte integer as mat #
1  1
/number of coarse meshes along x, y and z
3  3  3
/number of voxels per coarse mesh along x, y, and z
10 45 10
10 41 10
10 38 10
/source format,# source fine meshes (x,y,z),#group,sn,pn
0,30,30,30,5,18,3
/cross-section file: format, # comment lines,Leg. order, table length
1,1,3,8
/BCs
0,0,0,0,0,0
```

In the above example, the first three cards specify problem name, number of voxels and voxel size along x, y and z, respectively. Two binary files are specified by the following binary file name card. The first file is the attenuation/material number binary file, named *hrt_atn.bin* in this model. The file contains the material attenuation coefficients or the material numbers for each voxel/fine mesh depending on the following data type card, in which users can specify the data format in each binary file. *PENMSHXP* reads the binary data as 4-byte integer (Type 0) or as 4 byte real number (Type 1). If Type 0 or 1 is specified for the attenuation/material number binary file (hrt_atn.bin), *PENMSHXP* reads data as attenuation coefficients (with the unit of cm-1), and assigns a unique material number for every different value of attenuation coefficient automatically (first come, first serve). The total number of materials is determined by the total number of different attenuation values in the binary file. *PENMSHXP* can also read the data as material number (4 byte integer) if *datatype=-1* is specified for the attenuation/material number binary file. And the total number of material is the maxim integer in the binary file. In either case, the material numbers and their associated attenuation values (if *datatype=0* or *1*) will be written in an output file called *prbname_data.out*. The second file, *hrt_act.bin*, is an optional file. *hrt_act.bin* contains the radiation activity value for each voxel/fine mesh, which can be used to specify the fixed source distribution in the model. Generally, datatype=1 (4 byte real number) should be used for the activity file. Both the attenuation/material number binary file and the activity binary file should have the same size of the total number of voxels multiplying by 4 bytes. Besides the radiation activity binary file, users can also define sources uniformly deposited in one or more materials, and/or a source distribution file (*prbname.src*), as the same way in the *penmsh.inp* file. If multi-sources are defined in different ways, *PENMSHXP* combines all the sources together, and projects them on the fine meshes.

8. References

Haghighat, A., 2000. Manual of PENMSH version 3.4 A Cartesian-based 3-D mesh generator.

ROYSTON, K., HAGHIGHAT, A., YI, C., 2011. Sensitivity Analysis of the TITAN Hybrid Deterministic Transport Code for SPECT Simulation, in: M&C 2011, International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering, Rio de Janeiro, RJ, Brazil.

Sjoden, G., Haghighat, A., 1996. PENTRAN, A 3-D Scalable Transport Code with Complete Phase Space Decomposition. Trans. Am. Nucl. Soc. 74, 181–183.

Sjoden, G.E., Haghighat, A., 2012. PENTRAN Code System: Parallel Environment Neutral-Particle TRANsport - Users Guide to Version 9.4X.6 Series.

Yi, C., Haghighat, A., 2010. A three-dimensional block-oriented hybrid discrete ordinates and characteristics method. Nucl. Sci. Eng. 164, 221–247.

Appendix A Sample *prbname_out.mba* file (material balance output file of PENMSHXP)

```
Material Balance Summary for Model: core

Section 1 : Coarse Mesh information
 CM #  x_size y_size   z_size      fm num       tot_fm    fm_vol num_mat              # of fm per mat
                                   x    y    z                          mat 1    2       3      4     5     6     7    8     9
   1  11.530   11.284    5.000   60    60    2   7200   9.035E-02    5        14   1000    48   540     0     0     0    0  5598
   2  11.530   11.284    5.000   60    60    2   7200   9.035E-02    5        14   1000    48   540     0     0     0    0  5598
   3  11.530   11.284    5.000   60    60    2   7200   9.035E-02    5        14   1000    48   540     0     0     0    0  5598
   4  11.530   11.284    4.050   60    60    2   7200   7.319E-02    5        14   1000    48   540     0     0     0    0  5598
   5  11.530   11.284    4.000   60    60    2   7200   7.228E-02    4         0      0     0     0  1014    48   540    0  5598
   6  11.530   11.284    4.738   60    60    2   7200   8.561E-02    4         0      0     0     0  1014    48   540    0  5598
   7  11.530   11.284    4.000   60    60    2   7200   7.228E-02    1         0      0     0     0     0     0     0    0  7200
   8  11.530   11.284    4.000   60    60    2   7200   7.228E-02    1         0      0     0     0     0     0     0    0  7200

Section 2: Model mat. Information

Contents of file: ./ntest/test4/core.mba

  -PENINP USER INPUT MATERIAL INVENTORY FOR BALANCE-
    (The first 6 lines of this deck are comments)

  Material  Material    Target (cm3)    Density g/cc
    Name     Number       Volume         (see xsec)
 ---------- -------- -----------------  ------------
  Material1    1        5.475328        6.5060E+00
  Material2    2        342.4973        5.9643E+00
  Material3    3        19.67009        8.032
  Material4    4        179.5754        1.0
  Material5    5      159.6034        1.6
  Material6    6      9.022014        8.032
  Material7    7      82.36524        1.0
  Material8    8      0.0             0.0
  Material9    9      3858.04         1.0

Model Material Inventory/Volume Table

Targeted Mean Density:   1.435529E+00 g/cm3 from .mba file
   Targeted Volume:   4.656249E+03 cm3    Mass:    6.684180E+03 g
      Model Volume:   4.656249E+03 cm3    Mass:    6.657266E+03 g

 Material   Mat    Fine       Model (cm3)      % of      Mdl/Trg     Model Mass      Model
  name      No.   Meshes        volume         Total     V-Ratio      Excess (g)      Mass (g)
---------- ------  --------   -------------   -------   ---------   ------------    ----------
Material1    1       56     4.819413E+00     0.104    8.802E-01   -4.267E+00    3.136E+01
Material2    2     4000     3.442438E+02     7.393    1.005E+00    1.042E+01    2.053E+03
Material3    3      192     1.652370E+01     0.355    8.400E-01   -2.527E+01    1.327E+02
Material4    4     2160     1.858916E+02     3.992    1.035E+00    6.316E+00   1.859E+02
Material5    5     2028     1.601037E+02     3.438    1.003E+00    8.004E-01   2.562E+02
Material6    6       96     7.578871E+00     0.163    8.400E-01   -1.159E+01    6.087E+01
Material7    7     1080     8.526230E+01     1.831    1.035E+00    2.897E+00    8.526E+01
Material8    8        0     0.000000E+00     0.000     NaN          0.000E+00    0.000E+00
Material9    9    47988     3.851826E+03    82.724     9.984E-01   -6.214E+00    3.852E+03

   TOTAL          57600    4.656249E+03

Material Vol in Cm^3
  Total Vol    :       4.656249E+03
Total Mat Vol:    4.656249E+03
```

Appendix B A Sample *prbname.src* file (fixed source file)

Prbname.src has two sections. The first section specifies the source grid boundaries along x, y and z (in unit of cm). The second section defines the source intensities for each grid node (in unit of #of particles/sec.cm3). The number of source meshes is given in the source specification card in *penmsh.inp*. For the input file of example 3 in Section 3, the numbers of source meshes are 10, 10, and 2 for x, y and z axis. Note that source grid are independent of the model meshing. Therefore, the boundaries of source meshes are not required to agree with the model meshing.

Sample prbname.src file corresponding penmsh.inp in example 3 as follows:

```
0.0 1.26  2.52  3.78  5.04  6.3  7.56  8.82  10.08  11.34  12.6          /x src. mesh boundary. # meshes=10
0.0 1.26  2.52  3.78  5.04  6.3  7.56  8.82  10.08  11.34  12.6          /y src mesh  boundary  # meshes=10
0.0  5.0  10.0                                                           /z src mesh boundary  # meshes=2

Z level 1 (in this example, 10x10 values to specify the source intensities for each src mesh
-----------------------------------------------------------              / two comment lines. (do not start with / ! or #)
0.034 0.040 0.043 0.041 0.000 0.043 0.045 0.041 0.000 0.038
0.033 0.041 0.043 0.044 0.043 0.045 0.045 0.044 0.040 0.040
0.035 0.041 0.045 0.045 0.042 0.044 0.045 0.044 0.040 0.040
0.037 0.042 0.044 0.043 0.000 0.043 0.044 0.042 0.000 0.038
0.045 0.047 0.048 0.049 0.048 0.047 0.045 0.042 0.041 0.041
0.046 0.048 0.049 0.049 0.049 0.048 0.045 0.000 0.041 0.041
0.047 0.049 0.049 0.049 0.049 0.048 0.046 0.043 0.042 0.041
0.044 0.048 0.049 0.049 0.049 0.049 0.047 0.045 0.043 0.041
0.045 0.047 0.048 0.048 0.049 0.048 0.047 0.046 0.044 0.040
0.045 0.045 0.045 0.046 0.047 0.047 0.047 0.045 0.044 0.039
```

```
Z level 2 (in this example, 10x10 values to specify the source intensities for each src mesh
-----------------------------------------------------------                    / two comment lines. (do not start with / ! or #)
0.043 0.042 0.000 0.043 0.045 0.046 0.045 0.044 0.041 0.037
0.043 0.042 0.000 0.043 0.045 0.046 0.045 0.044 0.041 0.037
0.040 0.041 0.041 0.041 0.041 0.040 0.039 0.037 0.034 0.030
0.049 0.049 0.049 0.048 0.048 0.047 0.045 0.043 0.039 0.039
0.046 0.046 0.046 0.045 0.044 0.043 0.041 0.039 0.034 0.041
0.044 0.043 0.043 0.041 0.040 0.039 0.037 0.035 0.031 0.036
0.042 0.041 0.041 0.040 0.039 0.038 0.036 0.034 0.032 0.028
0.040 0.039 0.038 0.037 0.036 0.035 0.033 0.031 0.029 0.026
0.036 0.034 0.035 0.034 0.033 0.032 0.031 0.028 0.026 0.023
0.033 0.032 0.031 0.030 0.029 0.028 0.027 0.025 0.023 0.021
```

Appendix C PENMSHXP Command Line Options

SYNOPSIS

penmshxp [flag1] [flag1 argument] [flag2] [flag2 argument] ...

DESCRIPTION

Flag Flag_argument

-h, --help
Display help screen

-v, --version
Display version information

-i, --input *input_dir*
Set *input_dir* as the input file directory. If *input_dir* is not provided or *'-i'* option is not used, the current directory will be the input file directory

-f1, -f, --flux *flux_dir*
Set *flux_dir* as the flux directory, where flux files (*proname#.flx* or *prb#.flx*) are located. Default directory is the current directory. See the examples in section 2.

-f2 *ref_flux_dir*
Set *ref_flux_dir* as the reference flux directory, where reference flux files (*proname#.flx* or *prb#.flx*) are located. Default directory is the current directory. See the examples in Section 2.

-f3 or -fjn *flux_current_dir*
Set *flux_current_*dir as the flux current directory, where flux current files (*proname#.fjn* or *prb#.fjn*) are located. If this option is used, -f1 and –f2 options are ignored.

-l, --log *logfile_name*
Set *logfile_name* as the log file name.

-dot *DotBit*
DotBit is an integer, which controls the *spacpf* value format in the *PENTRAN* input file (Block V). The absolute value of *DotBit* specifies the number of digits after the point for a *spacpf* FIDO entry. A negative number means the *spacpf* value is in the scientific format.

Example 22: -*dot* command option usage with floating point format

```
[home/user/]#  penmshxp –dot 3
```

The above command will generate a FIDO sequence for *spacpf* like:
spacpf=2R1.345 3R241.123 20R0.012   …

Example 23: -*dot* command option usage with scientific format

```
[home/user/]#  penmshxp -dot -5
```

The above command will generate a FIDO sequence for *spacpf* like:
spacpf=2R1.34512E+00 3R2.41123R+02 20R1.21157E-02   …

-nd, or –ndmesh   *ndmesh*

Specify the differencing scheme id*: ndmeth* in *PENTRAN* input file (BLOCK 4)

Example 24: -*nd* command option usage

```
[home/user/]# penmshxp –nd -2
```

The above command will set *ndmeth=-2* in the generated PENTRAN input deck. If this option is not used, the default one is 2.

-w   *max_warning*

Set the number of the same warning messages displayed. The default value of *max_warning* is 5.

-offpen

Skip generating the *PENTRAN* input file.

-offtan

Skip generating TITAN input file.

-offplt

Skip generating TECPLOT binary file.

-offpng

Turn off the *.png* file plotting. More plotting options are listed in Table 4.

-nofido

Turn off FIDO in source block (Block 5, Variable *spacpf*) in *PENTRAN/TITAN* input file.

-hrt *filename*

Turn on the phantom binary file mode. See Section 7.

-fgm

Output *flux.fgm*, an ASCII file containing average flux per material zone for each group. This option should be used with -f option.

-agm

Output *flux.agm*, an ASCII file containing average adjoint flux per material zone for each group. This option requires the all group flux are read in using -f option. Note that in *PENTRAN/TITAN* adjoint simulations, group numbers are in the reverse order. This option will automatically restore the ordering.

-ff

Output *prbname.flx.out*, an ASCII file containing forward flux distribution. This option requires the all group flux are read using -f option.

-fa

Output *prbname.adj.out*: an ASCII file containing adjoint flux distribution. This option requires the all group flux are read in using -f option. Note that in *PENTRAN/TITAN* adjoint simulations, group numbers are in the reverse order. This option will automatically restore the ordering.

-maxmesh *max_fm_size*

Set maxim fine mesh size, See Section 4.3.

Appendix D A Sample in4.inp Input File

As shown in Table 1, an optional input file, named *in4.inp*, can be used to specify some additional parameters related to transport simulations, such as iteration limits and tolerances. *PENMSHXP* will read and process this file if it is present in the input folder. The file is organized into four sections, marked as #0, #1, #2, and #3, respectively. Section #0 accepts one card with three entries: number of quadrature, global differencing scheme id, and SPECT simulation switch (ROYSTON et al., 2011). Section #1 specifies parameters (such as quadrature type, order, associated ordinate splitting type and split directions) for each quadrature set. For each coarse mesh, Section #2 assigns a solver id, a quadrature id if number of quadrature is more than 1, and a differencing scheme id if the global differencing scheme id is set to 0. Coarse mesh individual solvers and quadrature sets are supported by the TITAN code. More information related to differencing schemes and ordinate splitting techniques can be found in TITAN and PENTRAN manuals(Sjoden and Haghighat, 2012). Note that the entire Section #2 is optional. Section #3 contains parameters for the iteration limits and tolerance, and initial k-effective. If the SPECT simulation switch is set to 1 in Section #0, another section (#10) is required for SPECT simulation parameters.

File: *in4.inp*

```
/ Additional input variables in optional input file: in4.inp
/**********************************************************
#0
/
/ number of quads, global ndmeth (differencing scheme), SPECT flag
2 1 0
/
/*********************************************************
#1 Quadrature Settings
/quad 1
/Quadrature id, order, number of split directions
/id=0, level symmetric
0 14 2
/split directions
46 47  /direction index
10 10  /split order
1  1   /split id: 1= pn-tn splitting
2  2   /topnum
/quad 2
/ Quadrature id , order, number of split directions
/id=1, Pn-Tn
1 18 1
/split directions
37  /direction index
8   /split order
1   /split id: 1- pn-tn splitting
2   /topnum
/
/*********************************************************
```

```
/#2 Solver and Quad id for each coarse mesh
/Solver_id
 1 1 1
 1 1 1
 1 1 1

 1 1 1
 1 0 1
 1 1 1
/quad_id
 2 1 1
 1 1 1
 1 1 1

 1 1 1
 1 1 1
 1 1 1
/*********************************************************
#3 control parameters and iteration limits
/
/tolerances: outer, inner
1.0E-6 1.0E-4
/iteration limit: outer inner
200  10
/initial keff, for k-effective problems
1.00
```

Appendix E A Hexagon Lattice Model

The input files (*penmsh.inp* and *he3demo1.inp*) for a hexagon lattice model (as shown in Figure 7) are included.

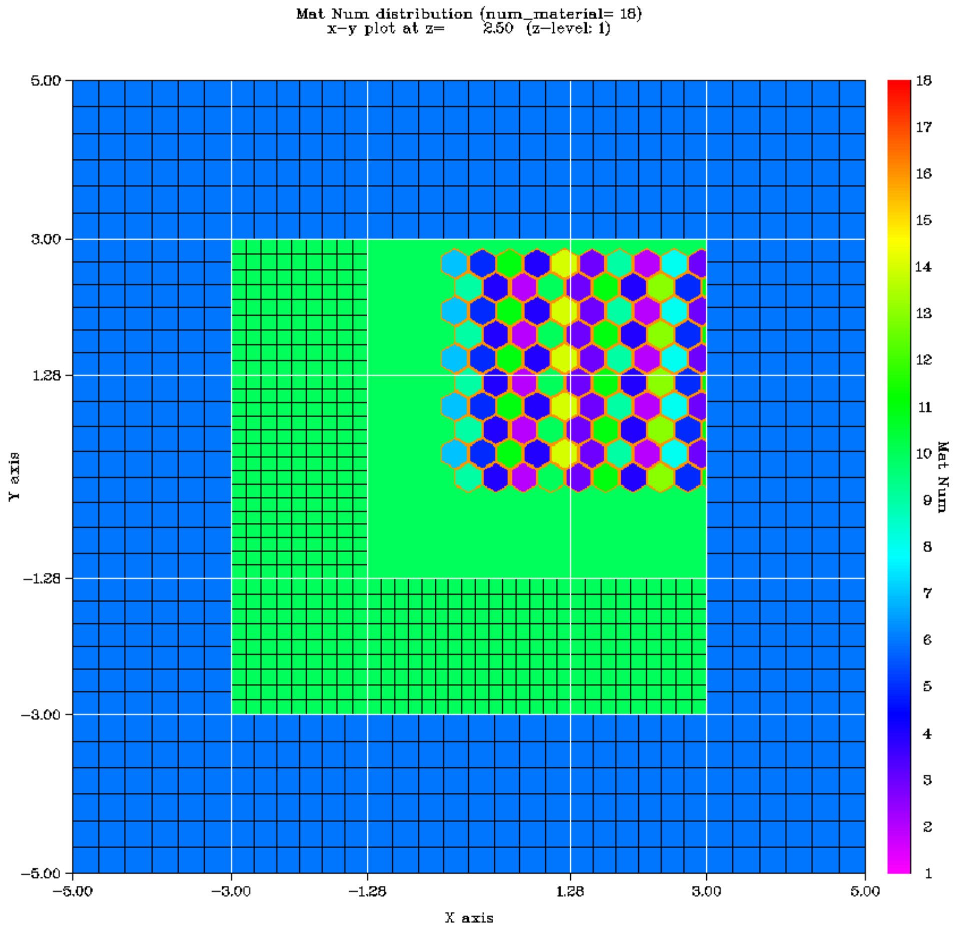


Figure 7 A Hexagon lattice model

File : *penmsh.inp*

```
/Problem name
he3demo
/no. coarse z-levels, no. of materials, imath
1,18,0
/z-level coarse mesh boundary
0.00, 5.00
/max. number of fine z-mesh per coarse z-level
20 20
/fine-to-med grid ratio along x.., y.., z.. in each coarse z-level
2
2
2
/source format, #x-src mesh, #y-src mesh, #z-src mesh, #grps, Sn, Pn
-1,0,0,0,47,8,2
/xsec type, xsec #comment cards, xsec Leg order, ihm
1,2,2,50
/Bdy conds: ibback(-x),ibfrnt(+x),jbeast(-y),jbwest(+y),kbsout(-z),kbnort(+z)
0,0,0,0,0,0
/source material id (if id <0, need that number of sources)
16
/source magnitude(s)
1.0
```

File he3demo1.inp

```
/ ncx, ncy, maxfinz  (maxfinz < 0, add z-fine per cm below y-fine)
5,5,-30
/ x-fine mesh per fuel pin cm (# seq along cm rows of x)
 6 6  8   6    6
 6 9 15   9    6
 6 9 160 120   6
 6 9 160 120   6
 6 6  8 6      6
/ y-fine mesh per fuel pin cm (# seq ... y)
 6  6  6  6   6
 6  9  9  9   6
 8 15 160 160  8
 6  9 120 120  6
 6  6  6  6  6
/ z-fine mesh per fuel pin cm (# seq ... y)
1 1 1 1 1
1 1 1 1 1
1 1 1 1 1
1 1 1 1 1
1 1 1 1 1
/ cm bounds along x-axis (in seq ...x)
-5.0  -3.0  -1.28  1.28  3.0  5.0
/ cm bounds along y-axis (in seq ...x)
-5.0  -3.0  -1.28  1.28  3.0  5.0
/ cm type, each cm (<0 =look for overlay)
1 1 1 1  1
1 1 1 1  1
1 1 -1  131 1
1 1 131 131 1
1 1 1 1 1
/ number of material regions per coarse mesh
1 1 1 1 1
1 1 1 1 1
1 1 1 1 1
1 1 1 1 1
1 1 1 1 1
/ coarse meshes background mat numbers (total 25 entries)
6R6 3R10 2R6 3R10 1Q5 6R6
/ shared overlay for CMs 13 14 18 and 19
2
-6 -6
16 -16
0 0  0 0.2
10 10 1 0.34641 0 0
0 0  0 0.18
10 10 1 0.34641 0 0
9 4 2 10 3 11 4 13 5 11  7 5 11 4 14 3 9 2 8 3 4Q20
```