PIRS: Python Interfaces for Reactor Simulations

Package to facilitate neutronics and T-H calculations

A.Travleev

Institute for Neutron Physics and Reactor Technology

24.04.14

Outline

What PIRS is

PIRS: Python Interfaces for Reactor Simulations

A set of packages for *Python* programming language, to facilitate *interaction* with reactor calculation codes.

Python

- www.python.org
- Free
- Interpreted
- Big community
- Lot of packages

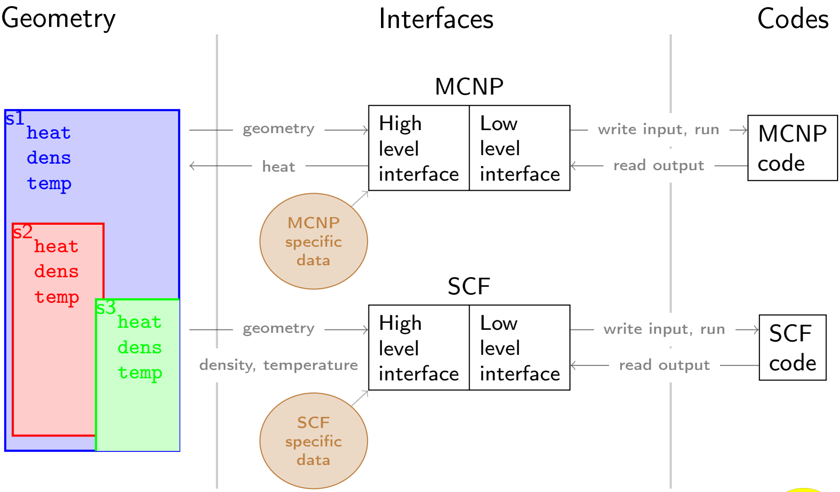
Interaction with code

- Model description
- Generation of Input file(s)
- Job submission
- Reading of calculation results

What PIRS is for

- Routine preparation of input files, reading output files
- Framework for coupled calculations

PIRS concept



Python Interpreter

```
$ python --version
Python 2.7.3
$ python
```

```
>>> from math import pi
>>> print pi
3.14159265359

>>> from pirs.solids import Box
>>> b = Box()
>>> b.X = 5

>>> print b
>>> print repr(b)

>>> help(b)
>>> dir()
>>> dir(b)
```

Python script

```
$ python script.py
```

```
l = [1, 2, 3, 4, 5]
s = 0
for e in l:
    s += e
    print e, s

for c in 'abs':
    print c
```

Python version

Python 2 or 3?

"Python 2.x is legacy, Python 3.x is the present and future of the language" from www.python.org/doc

PIRS uses Python 2.x

PIRS is developed for Python 2.6.x or above. Not for Python 3.x!

Usually Python 2.6.x or 2.7.x is already installed. If not, can be installed using OS installer (e.g. Ubuntu Software Center), or compiled from sources.

Required packages

uncertainties

Needed to preserve information about errors in MC calculations.

```
$ wget https://pypi.python.org/.../uncertainties-2.4.6.tar.gz
$ tar -xzf uncertainties-2.4.6.tar.gz
$ cd uncertainties-2.4.6
$ python setup.py install --user
```

matplotlib

Optional. Needed to plot geometry and results of calculations. Simplest way to install – using OS installer. Compilation from sources is difficult due to lot of dependencies.

Environmental variables

Several enviromental variables should be defined.

$DATAPATH

Path to xsdir file and files with cross-sections.

```
$ echo $DATAPATH
/home/data/mcnp
```

$MCNP

Path to MCNP executable

```
$ echo $MCNP
/home/bin/mcnp5_linux_i386_omp
```

$SCF

Path to SCF executable

```
$ echo $SCF
/home/bin/scf25_intel
```

How to install PIRS

Standard method

```
$ tar -xzf pirs-0.2a.0.tar.gz
$ cd pirs-0.2a.0
$ python setup.py install --user
```

Dependencies should be installed separately.

2-nd method

It uses pip – python package installer that is not available by default.

```
$ pip install pirs-0.2a.0.tar.gz --user
```

Dependencies will be installed by pip, if necessary.

Test

```
from pirs.solids import Box
from pirs import McnpInterface
b = Box()
m = McnpInterface(b)
```

PIRS classes and functions

Classes for geometry description

```python
from pirs.solids import zmesh
from pirs.solids import Cylinder, Box, Sphere
```

High-level interfaces

```python
from pirs import McnpInterface
from pirs import ScfInterface
```

Low-level interface to MCNP

```python
from pirs.mcnp import Material, MaterialCollection
from pirs.mcnp import MeshTally, TallyCollection
from pirs.mcnp import Xsdir
from pirs.mcnp import Surface, Volume, SurfaceCollection
from pirs.mcnp import Cell, Model
from pirs.mcnp import xs_interpolation
```

PIRS classes and functions

Low-level interface to SCF

```
        from pirs.scf2 import Input
        from pirs.scf2 import read_output, OutputTable
```

Tools

```
        from pirs.tools import LoadMap
        from pirs.tools import load, dump

        from pirs.tools.plots import MeshPlotter, colormap
```

Base classes

Not needed to end-user of PIRS.

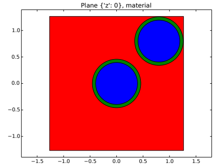
```
        from pirs.core.tramat import Nuclide, Mixture, zai
        from pirs.core.trageom import Vector3, pi, pi2
        from pirs.core.scheduler import Job, Scheduler, enva, WorkPlace, InputF
```

Geometry I

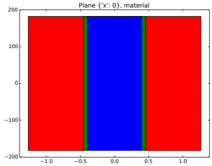
```
from pirs.solids import Box, Cylinder

cnt = Box(X=2.53, Y=2.53, Z=365, material = 'water')
r = Cylinder(R=0.4583, Z=365, material = 'steel')
f = Cylinder(R=0.3951, Z=365, material = 'fuel')

r.insert(f)
cnt.insert(r)

r2 = cnt.insert(r.copy_tree())
r2.pos.x = 0.8
r2.pos.y = 0.8

if __name__ == '__main__':
    from pirs.tools.plots import colormap
    pz = colormap(cnt, plane={'z':0})
    px = colormap(cnt, plane={'x':0}, aspect='auto')
    py = colormap(cnt, plane={'y':0}, aspect='auto')
    pz.get_figure().savefig('geom1_pz.pdf')
    px.get_figure().savefig('geom1_px.pdf')
```

Geometry I: plots

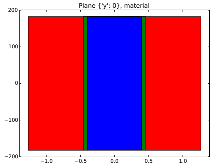
z plane



x plane



y plane

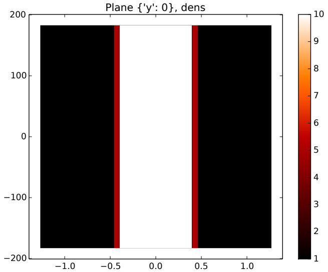


System variables

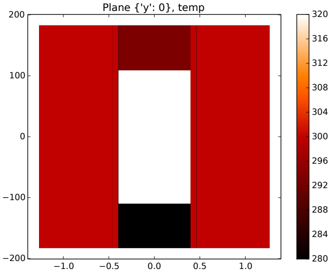
```
from geom1 import cnt

for v in cnt.values(True):
    v.temp.set_values(300.)

cnt.dens.set_values(1.0)
cnt.children[0].dens.set_values(5.0)
cnt.children[1].dens.set_values(5.0)

f1 = cnt.get_child((0,0))
f2 = cnt.get_child((1,0))

f1.temp.set_grid([1, 3, 1])
f1.temp.set_values([280, 320, 293])

f2.temp.set_grid([1, 2, 1])
f2.temp.set_values([275, 314, 293])

f1.dens.set_values(10.)
f2.dens.set_values(10.)

if __name__ == '__main__':
    from pirs.tools.plots import colormap
```

System variables: plot

density at plane y=0



temperature at plate y=0

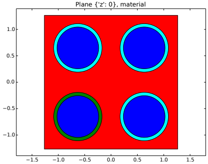


Geometry II

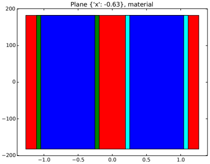
```
from geom2 import cnt

r1 = cnt.children[0]
r2 = cnt.children[1]

r1.material = 'zirc'
r1.ijk = (0, 0, 0)
r2.ijk = (1, 0, 0)
r2.pos *= 0.

cnt.grid.x = 1.26
cnt.grid.y = 1.30
cnt.grid.z = cnt.Z
r3 = cnt.grid.insert((1, 1, 0), r2.copy_tree())
r4 = cnt.grid.insert((0, 1, 0), r2.copy_tree())
# r4.R = 0.56

# cnt.grid.set_origin((0, -1, 0), (0.,-0.5,0.))
cnt.grid.center()
```

Geometry II: plots

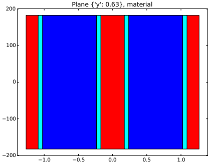
z plane



x plane at -1



y plane at 1



High-level interface to MCNP

```
from pirs import McnpInterface
from geom3 import cnt

mi = McnpInterface(cnt)

if __name__ == '__main__':
    mi.wp.prefix = 'm1_'
    mi.run('P')
```

High-level interface to MCNP: input file

```
MESSAGE:  datapath=/home/likewise-open/KIT/rx8040/data/mcnp/all_jeff

c title
1 0  -1 fill=1 imp:n=1
2 1 -1.0 -2 3 -4 5  fill=-1:2 0:1 0:0
c  k=0
     1 2 5 1
     1 5 5 1 imp:n=1 lat=1 u=1
3 0  -6 fill=3 imp:n=1 u=2
4 0  -7 fill=4 imp:n=1 u=3
5 2 -10.0 -8 imp:n=1 tmp=2.412856e-08 u=4
6 3 -10.0 8 -9 imp:n=1 tmp=2.757550e-08 u=4
7 4 -10.0 9 imp:n=1 tmp=2.524881e-08 u=4
8 1 -5.0 7 imp:n=1 tmp=2.585203e-08 u=3
9 1 -1.0 6 imp:n=1 tmp=2.585203e-08 u=2
10 0  -6 fill=6 imp:n=1 u=5
11 0  -7 fill=7 imp:n=1 u=6
12 5 -10.0 -10 imp:n=1 tmp=2.369769e-08 u=7
```

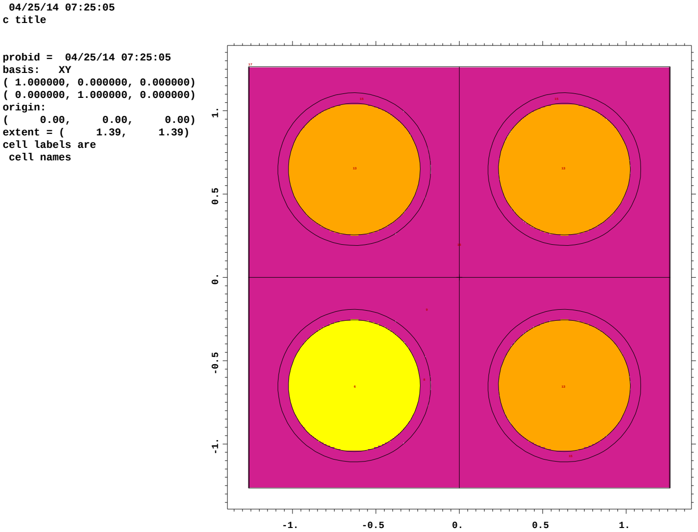
High-level interface to MCNP: input file continued

```
12 5 -10.0 -10 imp:n=1 tmp=2.369769e-08 u=7
13 6 -10.0 10 -11 imp:n=1 tmp=2.705846e-08 u=7
14 4 -10.0 11 imp:n=1 tmp=2.524881e-08 u=7
15 1 -5.0 7 imp:n=1 tmp=2.585203e-08 u=6
16 1 -1.0 6 imp:n=1 tmp=2.585203e-08 u=5
17 0  1 imp:n=0 tmp=2.526174e-08

c surfaces
1 rpp  -1.265  1.265  -1.265  1.265  -182.5  182.5
2 px  0.0
3 px  -1.26
4 py  0.0
5 py  -1.3
6 c/z  -0.63  -0.65  0.4583
7 c/z  -0.63  -0.65  0.3951
8 pz  -109.5
9 pz  109.5
10 pz  -91.25
11 pz  91.25
```

High-level interface to MCNP: input file continued

```
11 pz  91.25

c data cards
c materials
m1
c density 1.00000000000000e-05 g/cc, 5.97538539870074e-06 1/cm-barn
     1001.31c 1.0000000e+00
m2
c density 1.00000000000000e-05 g/cc, 5.97538539870074e-06 1/cm-barn
     1001.31c 1.0000000e+00
m3
c density 1.00000000000000e-05 g/cc, 5.97538539870074e-06 1/cm-barn
     1001.31c 7.8801773e-01     1001.32c 2.1198227e-01
m4
c density 1.00000000000000e-05 g/cc, 5.97538539870074e-06 1/cm-barn
     1001.31c 1.0000000e+00
m5
c density 1.00000000000000e-05 g/cc, 5.97538539870074e-06 1/cm-barn
     1001.31c 1.0000000e+00
m6
```

High-level interface to MCNP: plot 1



MCNP-specific data

Data in general model

- geometry
- axial behaviour of system variables (temperature, density – to define MCNP geometry, heat – to define tallies)

What else MCNP needs

- Material specifications
- Boundary conditions
- Description of (criticality) source

Material specifications: water

```
from pirs.mcnp import Material

m1 = Material(('H', 2, 1),
              ('O', 1, 1))
m1.sdict[8018] = 8016
m1.thermal = 'lwtr'

m1.T = 300
print m1.card()

m1.T = 450
print m1.card()
```

Material specifications: water continued

```
mcnp.Xsdir.read(/home/likewise-open/KIT/rx8040/data/mcnp/all_jeff/xsdir) callec
m{0:<}
     1001.31c 1.9997700e+00
     1002.31c 2.3000000e-04
     8016.31c 9.9757000e-01
     8017.31c 3.8000000e-04
     8016.31c 2.0500000e-03
mt{0:<} lwtr01.31t
m{0:<}
     1001.32c 9.7215298e-01     1001.33c 1.0276170e+00
     1002.32c 1.1181045e-04     1002.33c 1.1818955e-04
     8016.32c 4.8495109e-01     8016.33c 5.1261891e-01
     8017.32c 1.8473031e-04     8017.33c 1.9526969e-04
     8016.32c 9.9657141e-04     8016.33c 1.0534286e-03
mt{0:<} lwtr05.31t
```

Material specifications: steel

```
from pirs.mcnp import Material
# zircaloy-2, Table 5, p.7
zr = Material(('Zr', 98.23),
              ('Sn', 1.50),
              ('Fe', 0.12),
              ('Cr', 0.10),
              ('N',  0.05))

print zr.card()
```

Material specifications: steel continued

```
mcnp.Xsdir.read(/home/likewise-open/KIT/rx8040/data/mcnp/all_jeff/xsdir) calle
m{0:<}
     40090.31c 5.0539335e+01
     40091.31c 1.1021406e+01
     40092.31c 1.6846445e+01
     40094.31c 1.7072374e+01
     40096.31c 2.7504400e+00
     50118.31c 3.6330000e-01
     50122.31c 6.9450000e-02
     50120.31c 4.8870000e-01
     50112.31c 1.4550000e-02
     50114.31c 9.9000000e-03
     50115.31c 5.1000000e-03
     50116.31c 2.1810000e-01
     50117.31c 1.1520000e-01
     50119.31c 1.2885000e-01
     50124.31c 8.6850000e-02
     26054.31c 7.0140000e-03
     26056.31c 1.1010480e-01
     26057.31c 2.5428000e-03
     26058.31c 3.3840000e-04
     24050.31c 4.3450000e-03
     24052.31c 8.3789000e-02
```

Material specifications: mox

```
from pirs.mcnp import Material
# depleted u, mass fractions
ud = Material((92235,  0.2, 2),
              (92238, 99.8, 2))
# simplified o for oxide
o = Material(8016)
# pu
pu = Material((94239, 93.6, 2),
              (94240,  5.9, 2),
              (94241,  0.4, 2),
              (94242,  0.1, 2))
# oxides and mox
ux = Material((ud, 1), (o, 2))
px = Material((pu, 1), (o, 2))
mox = Material((ux, 0.5), (px, 0.5))
print mox.report()
# objective function
def obj(mix):
    a1 = mix.how_much(2, ZAID=[92235, 94239, 94241, 94243])
    a2 = mix.how_much(2, Z=[92, 94])
    return a1/a2 - 0.025
mox.tune(obj, [ux, px])
print mox.report()
```

Material specifications: mox continued

before tune() method

```
Mixture O-U-Pu
        <O-U       89.2383>: 0.5 mol
        <O-Pu      89.5943>: 0.5 mol
        total: 1.0 mol or 90.1910541098 g
Nuclide composition:
                    Nuclide         At.frac       Wgt.frac
        <     8016  15.8575>    6.66667e-01    1.18230e-01
        <    92235 233.0248>    3.37589e-04    8.79779e-04
        <    92238 236.0058>    1.66329e-01    4.39010e-01
        <    94239 236.9986>    1.56046e-01    4.13600e-01
        <    94240 237.9916>    9.79516e-03    2.60709e-02
        <    94241 238.9861>    6.61316e-04    1.76752e-03
        <    94242 239.9793>    1.64645e-04    4.41880e-04
```

after tune() method

```
Mixture O-U-Pu
        <O-U       89.2383>: 0.9755859375 mol
        <O-Pu      89.5943>: 0.0244140625 mol
        total: 1.0 mol or 90.0202779671 g
Nuclide composition:
                    Nuclide         At.frac       Wgt.frac
        <     8016  15.8575>    6.66667e-01    1.18454e-01
        <    92235 233.0248>    6.58694e-04    1.71986e-03
        <    92238 236.0058>    3.24537e-01    8.58209e-01
        <    94239 236.9986>    7.61941e-03    2.02336e-02
        <    94240 237.9916>    4.78279e-04    1.27541e-03
        <    94241 238.9861>    3.22908e-05    8.64685e-05
        <    94242 239.9793>    8.03929e-06    2.16171e-05
```

Set materials to high-level MCNP interface

```
from hmcnp1 import mi
from mcnp_water import m1 as w
from mcnp_zirc import zr
from mcnp_mox import mox

mi.materials['water'] = w
mi.materials['fuel'] = mox
mi.materials['steel'] = zr
mi.materials['zirc'] = zr

if __name__ == '__main__':
    mi.wp.prefix = 'm2_'
    mi.run('P')
```

MCNP input file

```
MESSAGE:  datapath=/home/likewise-open/KIT/rx8040/data/mcnp/all_jeff

c title
1 0  -1 fill=1 imp:n=1                                          $  container for None
2 1 -1.0 -2 3 -4 5  fill=-1:2 0:1 0:0
c  k=0
     1 2 5 1
     1 5 5 1 imp:n=1 lat=1 u=1                                  $  Lattice cell for
3 0  -6 fill=3 imp:n=1 u=2                                      $  container for None
4 0  -7 fill=4 imp:n=1 u=3                                      $  container for None
5 2 -10.0 -8 imp:n=1 tmp=2.412856e-08 u=4                       $  layer of
6 3 -10.0 8 -9 imp:n=1 tmp=2.757550e-08 u=4                     $  layer of
7 4 -10.0 9 imp:n=1 tmp=2.524881e-08 u=4                        $  layer of
8 5 -5.0 7 imp:n=1 tmp=2.585203e-08 u=3                         $  layer of
9 1 -1.0 6 imp:n=1 tmp=2.585203e-08 u=2                         $  layer of
10 0  -6 fill=6 imp:n=1 u=5                                     $  container for None
11 0  -7 fill=7 imp:n=1 u=6                                     $  container for None
12 6 -10.0 -10 imp:n=1 tmp=2.369769e-08 u=7                     $  layer of
13 7 -10.0 10 -11 imp:n=1 tmp=2.705846e-08 u=7                  $  layer of
14 4 -10.0 11 imp:n=1 tmp=2.524881e-08 u=7                       $  layer of
15 5 -5.0 7 imp:n=1 tmp=2.585203e-08 u=6                         $  layer of
16 1 -1.0 6 imp:n=1 tmp=2.585203e-08 u=5                         $  layer of
17 0  1 imp:n=0 tmp=2.526174e-08                                 $  layer of

c surfaces
1 rpp  -1.265  1.265  -1.265  1.265  -182.5  182.5
2 px  0.0
3 px  -1.26
4 py  0.0
5 py  -1.3
```

MCNP input file continued 1

```
5 py  -1.3
6 c/z  -0.63  -0.65  0.4583
7 c/z  -0.63  -0.65  0.3951
8 pz  -109.5
9 pz  109.5
10 pz  -91.25
11 pz  91.25

c data cards
c materials
m1                                                                            $  H-O at 300.0 K
     1001.31c 1.9997700e+00
     1002.31c 2.3000000e-04
     8016.31c 9.9757000e-01
     8017.31c 3.8000000e-04
     8016.31c 2.0500000e-03
mt1 lwtr01.31t                                                                $  thermal data at 293
m2                                                                            $  O-U-Pu at 280 K
     92235.31c 6.5869406e-04
     92238.31c 3.2453662e-01
     8016.31c 6.5039062e-01
     94239.31c 7.6194114e-03
     94240.31c 4.7827937e-04
     94241.31c 3.2290797e-05
     94242.31c 8.0392860e-06
     8016.31c 1.6276042e-02
m3                                                                            $  O-U-Pu at 320 K
     92235.31c 5.1906260e-04    92235.32c 1.3963146e-04                       $  0.78802 300.000K,
     92238.31c 2.5574061e-01    92238.32c 6.8796009e-02                       $  0.78802 300.000K,
     8016.31c 5.1251934e-01     8016.32c 1.3787128e-01                        $  0.78802 300.000K,
     94239.31c 6.0042313e-03    94239.32c 1.6151801e-03                       $  0.78802 300.000K,
```

MCNP input file continued 2

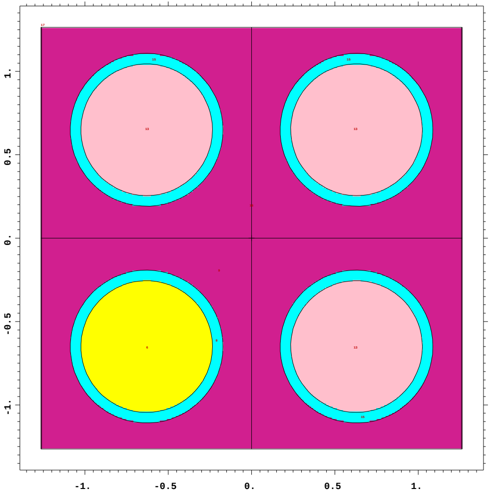
```
     94239.31c 6.0042313e-03   94239.32c 1.6151801e-03              $  0.78802 300.000K,  (
     94240.31c 3.7689263e-04   94240.32c 1.0138675e-04              $  0.78802 300.000K,  (
     94241.31c 2.5445720e-05   94241.32c 6.8450764e-06              $  0.78802 300.000K,  (
     94242.31c 6.3350999e-06   94242.32c 1.7041861e-06              $  0.78802 300.000K,  (
     8016.31c 1.2825809e-02    8016.32c 3.4502322e-03               $  0.78802 300.000K,  (
m4                                                                  $  O-U-Pu at 293 K
     92235.31c 6.5869406e-04
     92238.31c 3.2453662e-01
     8016.31c 6.5039062e-01
     94239.31c 7.6194114e-03
     94240.31c 4.7827937e-04
     94241.31c 3.2290797e-05
     94242.31c 8.0392860e-06
     8016.31c 1.6276042e-02
m5                                                                  $  Zr-Sn-Fe- at 300.0 K
     40090.31c 5.0539335e+01
     40091.31c 1.1021406e+01
     40092.31c 1.6846445e+01
     40094.31c 1.7072374e+01
     40096.31c 2.7504400e+00
     50118.31c 3.6330000e-01
     50122.31c 6.9450000e-02
     50120.31c 4.8870000e-01
     50112.31c 1.4550000e-02
     50114.31c 9.9000000e-03
     50115.31c 5.1000000e-03
     50116.31c 2.1810000e-01
     50117.31c 1.1520000e-01
     50119.31c 1.2885000e-01
     50124.31c 8.6850000e-02
     26054.31c 7.0140000e-03
```

MCNP input file continued 3

```
     94241.31c 3.2290797e-05
     94242.31c 8.0392860e-06
     8016.31c 1.6276042e-02
m7                                                                          $  O-U-Pu at 314 K
     92235.31c 5.6048132e-04    92235.32c 9.8212741e-05                    $   0.8509 300.000K,
     92238.31c 2.7614749e-01    92238.32c 4.8389128e-02                    $   0.8509 300.000K,
     8016.31c 5.5341594e-01     8016.32c 9.6974681e-02                     $   0.8509 300.000K,
     94239.31c 6.4833403e-03    94239.32c 1.1360711e-03                    $   0.8509 300.000K,
     94240.31c 4.0696686e-04    94240.32c 7.1312513e-05                    $   0.8509 300.000K,
     94241.31c 2.7476168e-05    94241.32c 4.8146292e-06                    $   0.8509 300.000K,
     94242.31c 6.8406106e-06    94242.32c 1.1986753e-06                    $   0.8509 300.000K,
     8016.31c 1.3849248e-02     8016.32c 2.4267938e-03                     $   0.8509 300.000K,
c tallies
c kcode 500  1.0  20  100  j  j  100000  j
prdmp j j 1                                                                 $  write mctal file
```

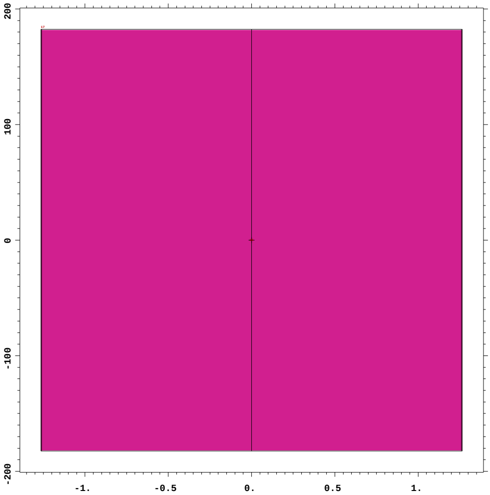
MCNP plot 1

```
 04/25/14 07:25:07
c title

probid =  04/25/14 07:25:06
basis:   XY
( 1.000000, 0.000000, 0.000000)
( 0.000000, 1.000000, 0.000000)
origin:
(      0.00,      0.00,      0.00)
extent = (      1.39,      1.39)
cell labels are
 cell names
```



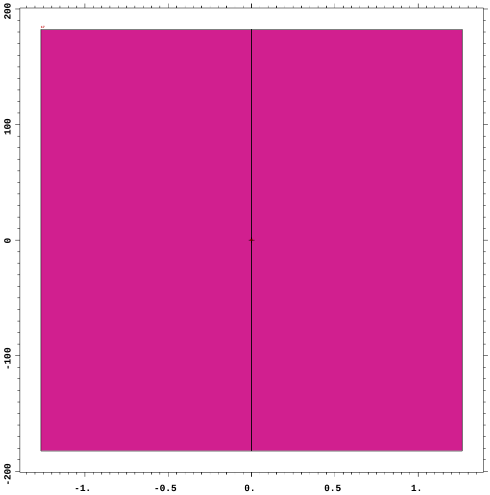
MCNP plot 2

```
 04/25/14 07:25:07
c title

probid =  04/25/14 07:25:06
basis:   XZ
( 1.000000, 0.000000, 0.000000)
( 0.000000, 0.000000, 1.000000)
origin:
(     0.00,      0.00,      0.00)
extent = (      1.39,    200.75)
cell labels are
 cell names
```



MCNP plot 3

```
 04/25/14 07:25:07
c title

probid =  04/25/14 07:25:06
basis:   YZ
( 0.000000, 1.000000, 0.000000)
( 0.000000, 0.000000, 1.000000)
origin:
(     0.00,     0.00,     0.00)
extent = (     1.39,    200.75)
cell labels are
 cell names
```



Set boundary conditions to high-level MCNP interface

```
from hmcnp2 import mi

mi.bc['radial'] = '*'
mi.bc['axial'] = ''

if __name__ == '__main__':
    mi.wp.prefix = 'm3_'
    mi.run('P')
```

MCNP input file

without boundary conditions

```
3 px  -1.26
4 py  0.0
5 py  -1.3
6 c/z  -0.63  -0.65  0.4583
7 c/z  -0.63  -0.65  0.3951
8 pz  -109.5
9 pz  109.5
10 pz  -91.25
11 pz  91.25

c data cards
c materials
m1
     1001.31c 1.9997700e+00
     1002.31c 2.3000000e-04
     8016.31c 9.9757000e-01
```

with boundary conditions

```
*3 px  1.265
*4 px  -1.265
*5 py  1.265
*6 py  -1.265
7 px  0.0
8 px  -1.26
9 py  0.0
10 py  -1.3
11 c/z  -0.63  -0.65  0.4583
12 c/z  -0.63  -0.65  0.3951
13 pz  -109.5
14 pz  109.5
15 pz  -91.25              $  H-O at 300.0 K
16 pz  91.25

c data cards
c materials
m1
     1001.31c 1.9997700e+00
     1002.31c 2.3000000e-04
     8016.31c 9.9757000e-01
```

Specify additional cards for MCNP input

```
from hmcnp3 import mi
# additional cell cards
mi.acc.append('c commented cell card')

# additional surface cards
mi.asc.append('c commented surface card')

# additional data cards
ksrc = 'ksrc '
for v in mi.gm.values(True):
    if v.material == 'fuel':
        x, y, z = v.abspos().car
        ksrc += '  {0} {1} {2}'.format(x, y, z-v.Z*0.49)
        ksrc += '  {0} {1} {2}'.format(x, y, z)
        ksrc += '  {0} {1} {2}'.format(x, y, z+v.Z*0.49)
mi.adc.append(ksrc)

# kcode card
mi.kcode.active = True # otherwise commented
mi.kcode.Nh = 1000 # histories per cycle
mi.kcode.Ncs = 20 # cycles to skip
mi.kcode.Nct = 100 # total num of cycles

if __name__ == '__main__':
    mi.wp.prefix = 'm4_'
    mi.run('P')
```

MCNP input file

Additional cell cards

```
*3 px  1.265
*4 px  -1.265
*5 py  1.265
*6 py  -1.265
```

Additional surface cards

```
     1002.31c 2.3000000e-04
     8016.31c 9.9757000e-01
     8017.31c 3.8000000e-04
     8016.31c 2.0500000e-03
mt1 lwtr01.31t                                                                           $  thermal data at 293
m2                                                                                       $  O-U-Pu at 280 K
     92235.31c 6.5869406e-04
```

Additional data cards

```
     -178.85  0.63 -0.65 0.0  0.63 -0.65 178.85  0.63 0.65 -178.85  0.63 0.65
     0.0  0.63 0.65 178.85  -0.63 0.65 -178.85  -0.63 0.65 0.0  -0.63 0.65
     178.85
```

Tallies for MCNP input

```
from hmcnp4 import mi

# set heat meshes
for v in mi.gm.values():
    if v.material == 'fuel':
        v.heat.set_grid([1]*20)

if __name__ == '__main__':
    mi.wp.prefix = 'm5_'
    mi.run('R', tasks=3)

    from pirs.tools import dump
    dump('m5_.dump', gm=mi.gm)


    from pirs.tools.plots import colormap
    hx1 = colormap(mi.gm, plane={'x':-0.63}, var='heat', aspect='auto')
    hx2 = colormap(mi.gm, plane={'x': 0.63}, var='heat', aspect='auto')
    hy1 = colormap(mi.gm, plane={'y':-0.63}, var='heat', aspect='auto')
    hy2 = colormap(mi.gm, plane={'y': 0.63}, var='heat', aspect='auto')
    hx1.get_figure().savefig('hmcnp5_hx1.pdf')
    hx2.get_figure().savefig('hmcnp5_hx2.pdf')
    hy1.get_figure().savefig('hmcnp5_hy1.pdf')
    hy2.get_figure().savefig('hmcnp5_hy2.pdf')
else:
    from pirs.tools import load
    mi.gm = load('m5_.dump')['gm']
```

MCNP input file with tallies

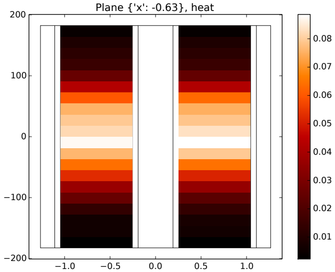
```
     jmesh= 0.0 1.265
     kmesh= -164.25 -146.0 -127.75 -109.5 -91.25 -73.0 -54.75 -36.5 -18.25 -0.0
     18.25 36.5 54.75 73.0 91.25 109.5 127.75 146.0 164.25 182.5
fm14 -1 0 -6 -8
kcode 1000  1.0  20  100  j  j  100000  j
prdmp j j 1                                                                 $  write mctal file
ksrc   -0.63 -0.65 -178.85  -0.63 -0.65 0.0  -0.63 -0.65 178.85  0.63 -0.65
     -178.85  0.63 -0.65 0.0  0.63 -0.65 178.85  0.63 0.65 -178.85  0.63 0.65
     0.0  0.63 0.65 178.85  -0.63 0.65 -178.85  -0.63 0.65 0.0  -0.63 0.65
     178.85
```

MCNP meshtal file

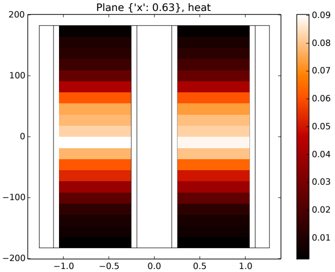
```
      -1.262    -0.632   100.375 0.00000E+00 0.00000E+00
      -1.262    -0.632   118.625 0.00000E+00 0.00000E+00
      -1.262    -0.632   136.875 0.00000E+00 0.00000E+00
      -1.262    -0.632   155.125 0.00000E+00 0.00000E+00
      -1.262    -0.632   173.375 0.00000E+00 0.00000E+00
      -1.262     0.632  -173.375 0.00000E+00 0.00000E+00
      -1.262     0.632  -155.125 0.00000E+00 0.00000E+00
      -1.262     0.632  -136.875 0.00000E+00 0.00000E+00
      -1.262     0.632  -118.625 0.00000E+00 0.00000E+00
      -1.262     0.632  -100.375 0.00000E+00 0.00000E+00
      -1.262     0.632   -82.125 0.00000E+00 0.00000E+00
      -1.262     0.632   -63.875 0.00000E+00 0.00000E+00
      -1.262     0.632   -45.625 0.00000E+00 0.00000E+00
      -1.262     0.632   -27.375 0.00000E+00 0.00000E+00
      -1.262     0.632    -9.125 0.00000E+00 0.00000E+00
      -1.262     0.632     9.125 0.00000E+00 0.00000E+00
      -1.262     0.632    27.375 0.00000E+00 0.00000E+00
      -1.262     0.632    45.625 0.00000E+00 0.00000E+00
      -1.262     0.632    63.875 0.00000E+00 0.00000E+00
      -1.262     0.632    82.125 0.00000E+00 0.00000E+00
      -1.262     0.632   100.375 0.00000E+00 0.00000E+00
      -1.262     0.632   118.625 0.00000E+00 0.00000E+00
      -1.262     0.632   136.875 0.00000E+00 0.00000E+00
      -1.262     0.632   155.125 0.00000E+00 0.00000E+00
      -1.262     0.632   173.375 0.00000E+00 0.00000E+00
      -0.630    -0.632  -173.375 2.76141E-03 1.31865E-01
      -0.630    -0.632  -155.125 5.98696E-03 7.95926E-02
      -0.630    -0.632  -136.875 6.30306E-03 7.48691E-02
      -0.630    -0.632  -118.625 1.34508E-02 5.35425E-02
      -0.630    -0.632  -100.375 2.58262E-02 4.10274E-02
      -0.630    -0.632   -82.125 3.69181E-02 3.27159E-02
```

MCNP results

x plane at -1



x plane at 1



High-level interface to SCF

```
from pirs import ScfInterface
from hmcnp5 import mi

si = ScfInterface(mi.gm)

if __name__ == '__main__':
    si.wp.prefix = 's1_'
    si.run('r')
```

Limitations

- Rectangular container (i.e. an instance of the pirs.solids.Box() class)
- Rods inserted into grid element centers, no empty grid elements.
- Container and rods (and all internal structure) have the same height.

SCF input file: channels

```
&channel_layout
file = this_file
channel_number          channel_area   wetted_perimeter   heated_perimeter   x_position    y_position
             1    2.25560841552e-05    0.0071989595657    0.0071989595657          1.1           1.2
             2    4.44971683104e-05    0.0143979191314    0.0143979191314          1.1           1.2
             3    2.25560841552e-05    0.0071989595657    0.0071989595657          1.1           1.2
             4    4.95571683104e-05    0.0143979191314    0.0143979191314          1.1           1.2
             5    9.78143366208e-05    0.0287958382628    0.0287958382628          1.1           1.2
             6    4.95571683104e-05    0.0143979191314    0.0143979191314          1.1           1.2
             7    2.25560841552e-05    0.0071989595657    0.0071989595657          1.1           1.2
             8    4.44971683104e-05    0.0143979191314    0.0143979191314          1.1           1.2
             9    2.25560841552e-05    0.0071989595657    0.0071989595657          1.1           1.2
!
file = this_file
channel    max_40_x_(neighbour+gap+distance)
      1                                    2   0.001567   0.009475   4   0.001767   0.009575   /
      2                                    3   0.001567     0.0126   5   0.003434      0.013   /
      3                                    6   0.001767   0.009575   /
      4                                    5   0.003834   0.009475   7   0.001767      0.013   /
      5                                    6   0.003834     0.0126   8   0.003434      0.013   /
      6                                    9   0.001767      0.013   /
      7                                    8   0.001567   0.009475   /
      8                                    9   0.001567     0.0126   /
      9                                    /
!
!
```

SCF input file: rods

```
file = this_file
rod_number   material_type   outer_diameter   power_fraction   x_position   y_position
         1               1         0.009166              0.1        -0.63        -0.65
         2               1         0.009166              0.1         0.63        -0.65
         3               1         0.009166              0.1         0.63         0.65
         4               1         0.009166              0.1        -0.63         0.65
!
file = this_file
rod   max_6_x_(channel+fraction)
  1                          1   0.25   2   0.25   4   0.25   5   0.25   /
  2                          2   0.25   3   0.25   5   0.25   6   0.25   /
  3                          5   0.25   6   0.25   8   0.25   9   0.25   /
  4                          4   0.25   5   0.25   7   0.25   8   0.25   /
!
file = this_file
```

SCF input file: power axial profile

```
file = this_file
time   inlet_flow
!
file = this_file
channel_number   time   inlet_flow
!
file = this_file
time   heat_flux_factor
!
file = this_file
power_map_time
!
file = this_file
axial_cell_number   rod_number    power_map
                1            1   0.00276141
                2            1   0.00598696
                3            1   0.00630306
                4            1    0.0134508
                5            1    0.0258262
                6            1    0.0369181
```

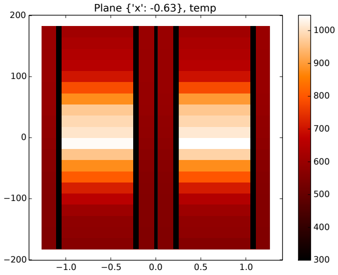
SCF-specific parameters

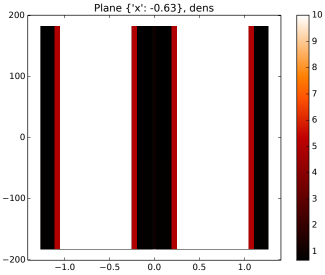
```
from hscf1 import si

# set boundary conditions
si.find('total_power')[0].value = 7e4*4 # W
si.find('average_heat_flux')[0].value = 0.0
si.find('inlet_temperature')[0].value = 280 # C
si.find('inlet_flow_rate')[0].value = 0.28*4 # g/sec
si.find('inlet_mass_flux')[0].value = 0.
si.find('set_driving_pressure_condition')[0].state = 'set_pure_flow_condition'
si.find('exit_pressure')[0].value = 15.45e6 # Pa
si.find('v', 'pressure_drop')[0].value = 0.
si.find('inlet_boron_concentration')[0].value = 0.
si.find('heat_fraction_moderator')[0].value = 0.

si.find('number_of_fuel_nodes')[0].value = 10

# variables:
si.find('blasius_laminar_prefactor')[0].value = 64.0
si.find('blasius_laminar_reynolds_exponent')[0].value = -1.0
si.find('blasius_laminar_constant')[0].value = 0

si.find('blasius_turbulent_prefactor')[0].value = 0.316
si.find('blasius_turbulent_reynolds_exponent')[0].value = -0.25
si.find('blasius_turbulent_constant')[0].value = 0

si.find('dittus_boelter_prefactor')[0].value = 0.023
si.find('dittus_boelter_reynolds_exponent')[0].value = 0.8
si.find('dittus_boelter_prandtl_exponent')[0].value = 0.4
si.find('dittus_boelter_constant')[0].value = 0.

si.find('set_buoyancy')[0].state = 0
```

SCF-specific parameters: rod material specifications

```
from pirs.scf2 import RodMaterial
from hscf2 import si

cld = RodMaterial()
cld.fp = 'benpwr'
cld.fd = -1
cld.ct = -1
cld.cp = 'zircaloy'

si.materials['steel'] = cld
si.materials['zirc'] = cld

if __name__ == '__main__':
    si.wp.prefix = 's3_'
    si.run('R')

    from pirs.tools import dump
    dump('s3_.dump', gm=si.gm)

    from pirs.tools.plots import colormap
    fltr = lambda e: e.material not in ['zirc', 'steel']
    tz  = colormap(si.gm, plane={'z':1},      var='temp', aspect='auto')#   , filter_=fltr
    tx1 = colormap(si.gm, plane={'x':-0.63}, var='temp', aspect='auto')#   , filter_=fltr
    tx2 = colormap(si.gm, plane={'x': 0.63}, var='temp', aspect='auto')#   , filter_=fltr
    ty1 = colormap(si.gm, plane={'y':-0.63}, var='temp', aspect='auto')#   , filter_=fltr
    ty2 = colormap(si.gm, plane={'y': 0.63}, var='temp', aspect='auto')#   , filter_=fltr
    tz.get_figure().savefig('hscf3_tz.pdf')                            #
    tx1.get_figure().savefig('hscf3_tx1.pdf')                          #
    tx2.get_figure().savefig('hscf3_tx2.pdf')                          #
    ty1.get_figure().savefig('hscf3_ty1.pdf')                          #
    ty2.get_figure().savefig('hscf3_ty2.pdf')                          #
```

SCF-specific parameters: SCF results

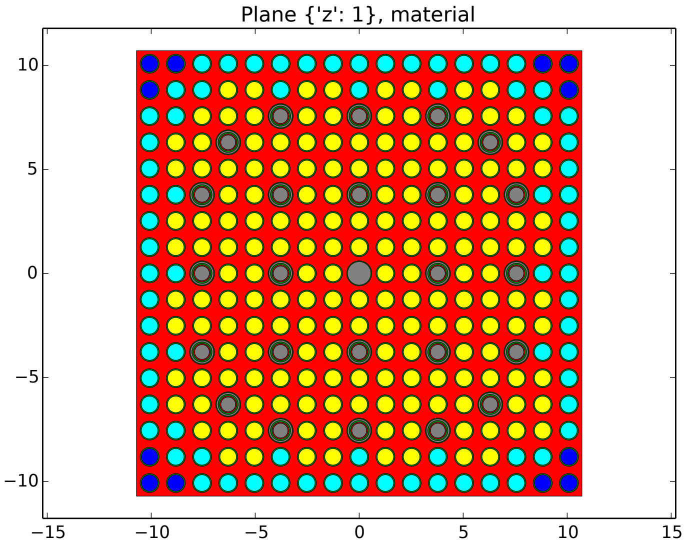




Data exchange between interfaces

```
from pirs.tools import dump
from hscf3 import si
from hmcnp5 import mi

mi.wp.prefix = 'cm_'
si.wp.prefix = 'cs_'

I = 0
while I < 5:
    # mcnp run
    mi.gm = si.gm.copy_tree()
    mr = mi.run('R', tasks=3)
    # relaxed power
    for (em, es) in zip(mr.heats(), si.gm.heats()):
        h = 0.5 *em.heat + 0.5 * es.heat
        es.heat.update(h)
    # scf run
    si.run('R')
    # store results
    dump('{}_coupling.dump'.format(I),
         Keff = mi.keff(),
         mr = mr,
         sr = si.gm,
         I = I)

    I += 1
```

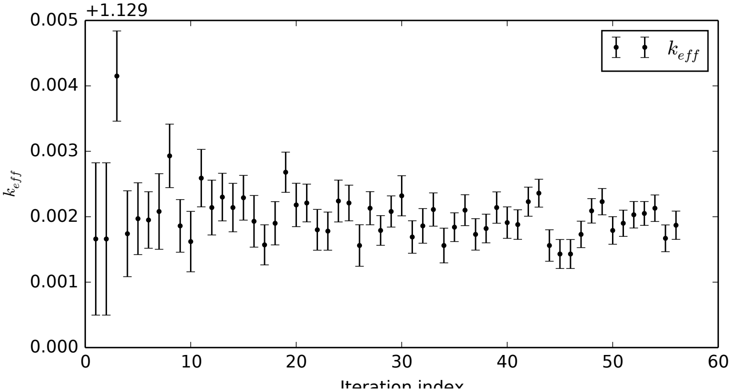
PWR assembly model



PWR assembly model

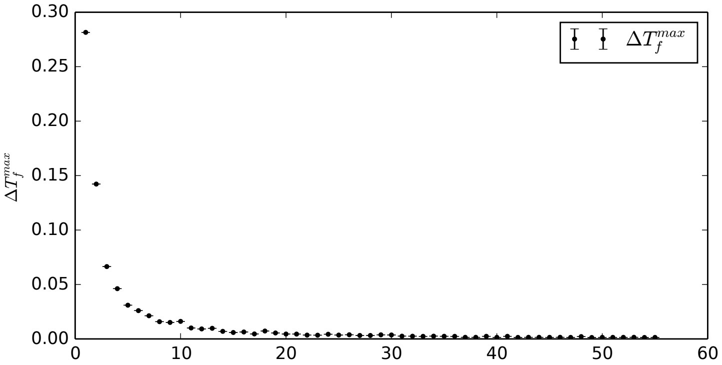
- Computed on IC2 cluster
- Initial statistics 5000 10 100
- Conv. criteria: Keff, Tfuel.
- Statistics at 56-th iteration: 149227 10 100
- MCNP input: 12K cells, 18K lines.

Keff vs. iterations

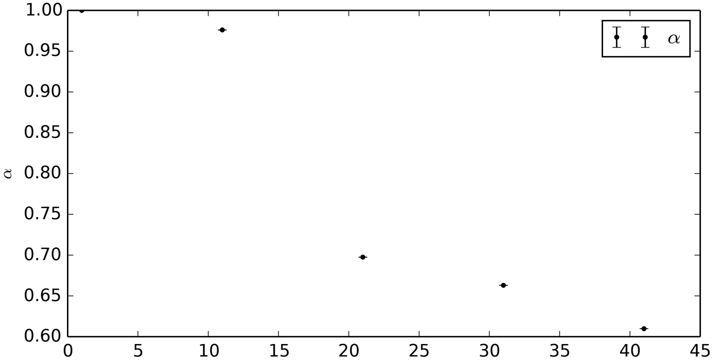


dTf vs. iterations

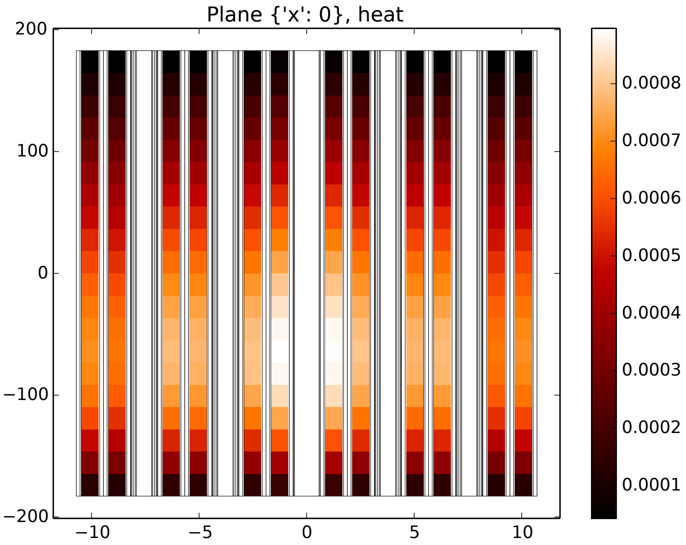
Maximal change in fuel temperature from I-1-th to I-th iteration



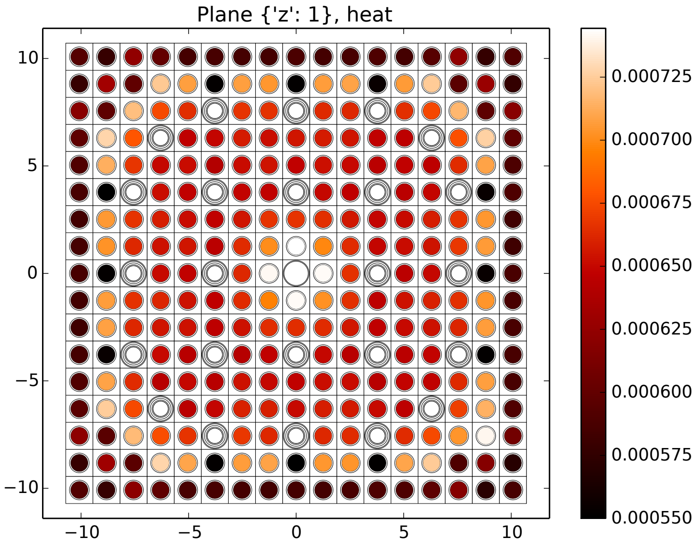
Relaxation factor vs. iterations



Power profile



Power profile



Fuel temperature profile