Putting it all Together, Part 3.

Parts 1 and 2 instructed you how to use multiple features of Serpent to build a simple LWR core model. The homogeneous fuel loading in the previous example, however, is unrealistic. Real reactor cores are highly heterogeneous in order to optimize fuel burnup and power distribution.

GOAL: To increase the realism of the previous model of light water reactor (LWR). To plot key quantities in the analysis of reactors.

BACKGROUND: The neutron multiplication factor keff is just one of the important quantities in reactor analysis. Another quantity is the neutron energy spectrum, which Monte Carlo codes like Serpent spends much of its runtime computing. However, the neutron energy spectrum is dependent on many parameters, two of which are burnup and location.

The previous exercise's input deck improved on the homogeneous deck of Part 1. However, real reactors are fueled with assemblies of fuel pins, and a realistic simulation must take that into account. We'll use the previous exercise's input deck as a starting point to create a reactor core composed of 180 individual fuel assemblies, in order to develop familiarity with the lattice approach one would take in order to do this. While the model will consist of 180 assemblies, we'll assume we can divide this reactor into 3 homogeneous regions of 60 assemblies each. Each region will have a single fuel material associated with it.

We will then use Serpent's detectors feature (referred to as tallies in MCNP) to compute the energy dependent neutron flux in the fuel regions and the coolant at beginning- and end-of-cycle (BOC and EOC).

EXERCISES:

1) Open the template provided. You may notice some new organizational tools at use. Instead of cramming all of your geometries and materials into the input deck it can be easier, especially when you start using lots of materials/geometries to separate things out into individual text files and then combine them in the deck using the command “include “filename.txt””. Try to use the spaces outlined with this file structure.

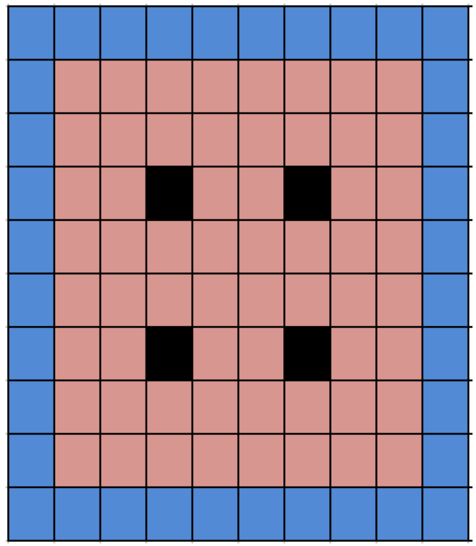
2) We are now going to construct a lattice of lattices! Lattiception!!!

Start by moving your fuel pins from the last exercise into the folder pins.txt. You will notice there is already one pin (pin 41, feel free to rename) in this folder. This is called a guide pin. It is where the control rod would be inserted into the assembly.

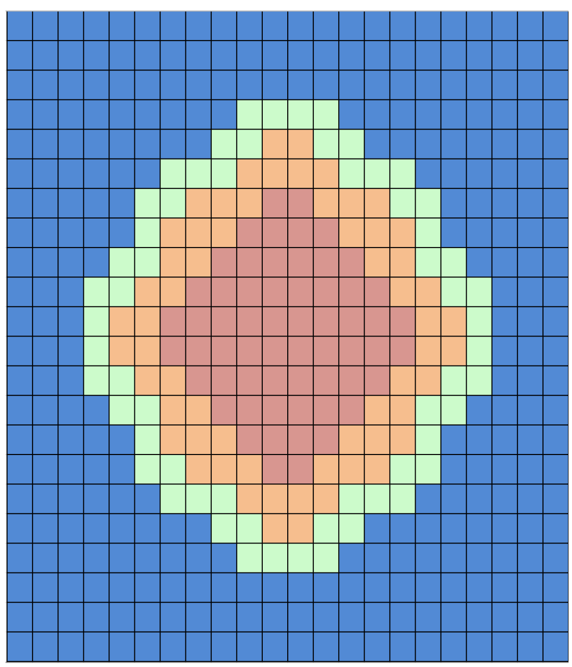
Next move your materials from the previous exercise into the folders fuel1.txt, fuel2.txt, and fuel3.txt. These once again represent the three burn up campaigns.

I left some surfaces in your deck. Read the comments and understand what these are for.

Open the file assemblies.txt and start making your assemblies. These will be 10 by 10 lattices with pin pitch 1.3 cm. There will be water filled boxes on the parameter (universe 42, also feel free to rename) as well as 4 guide pins. See the geometry bellow. You will need one lattice for each campaign. Use lattice type 1.



Open the file lattice.txt. This is where lattiception starts. Make the reactor shown bellow. Your fresh fuel will go on the outside, with older fuel towards the center. There will be 60 of each fuel assembly, for a total of 180 assemblies Be sure to add water boxes (universe 150). This will be a 22 by 22 lattice of type 1. Set the lattice pitch to 11.2 cm and label this lattice universe 300.



3) Recompute the volume of fuel in your core. Make sure you change the fuel volumes in your material cards.

4) Set up detectors for your fuel and coolant regions. For example, if your fuel materials are called fuel1, fuel2 and fuel3, you would first set up the energy grid:

ene eqlethgrid 3 100 1e-4 20

meaning you are requesting a 100 bin energy grid from 10-4 to 20 MeV. Then the detectors themselves:

det 1 dm fuel1 de eqlethgrid
det 2 dm fuel2 de eqlethgrid
det 3 dm fuel3 de eqlethgrid
det 4 dm coolant de eqlethgrid

5) Burn your input deck at 3GWth, for 18 months, using time steps of 30 days.

6) Serpent produces .m files for most of its output. Run the output files Serpent provides in Matlab. This will populate your workspace with variables. Find the variables associated with your detectors, and plot the neutron flux vs energy for each fuel region and the coolant, at BOC and EOC. Compare them and comment.