

Advanced reactor modeling with TRITON-NEWT

Best Practices for SCALE Modeling & Simulation

SCALE Users' Group Workshop
July 27–29, 2020

F. Bostelmann, B. Ade, B. Betzler, M. Jessee

ORNL is managed by UT-Battelle, LLC for the US Department of Energy



U.S. DEPARTMENT OF
ENERGY

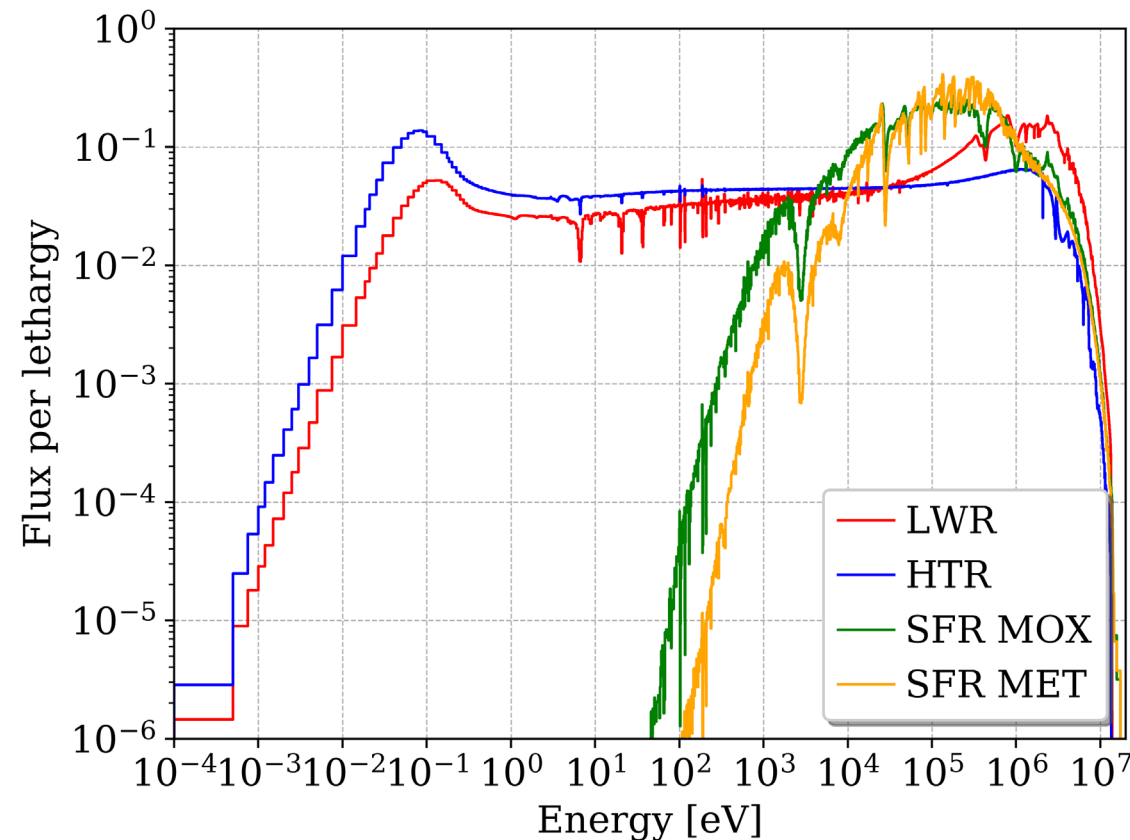
Objectives

- Provide modeling recommendation for TRITON-NEWT
 - Cross section library
 - Geometry checks
 - Transport settings
 - Extra tips on hexagonal geometries

Cross Section Libraries

Which multigroup cross section library should I use?

- Depends on:
 - The spectral conditions of the model (thermal vs. fast)
 - The desired solution accuracy
 - The available simulation time
- For thermal systems and with limited applicability for other spectral conditions:
 - If time allows, use the 252-group ENDFB-VII.1 (**v7-252**) library
 - If a quick answer is needed, use the 56-group ENDFB-VII.1 (**v7-56**) library
- Additional multigroup libraries to be provided with SCALE 6.3:
 - Very fine group library with **1597 groups**
 - **302-group** library optimized for sodium-cooled fast reactor systems



- Especially useful for modeling advanced reactor concepts with non-thermal spectra, such as:
 - Sodium-cooled fast reactors
 - Heat pipe reactor
 - Some molten salt reactors
- Tests with the 1597-group library can be used to increase confidence for current choice of multigroup library

Cross Section Processing

Which cross section processing options should I use?

- The most accurate solution is obtained using CENTRM and a fine-group cross section library
- **Latticecell**: sufficient for most unit cells
 - Assumes that the fuel pin lies in an infinite lattice of other fuel pins – a good assumption in many cases
- **Multiregion**: to be used if unit cell needs to be divided in multiple rings or more complex scenarios
 - Example: Gadolinium-bearing fuel pins; Gadolinium is a strong absorber of neutrons so the flux shape across the fuel pin changes significantly
 - Example: Ag-In-Cd control rods
 - Example: Complex geometry not easily modeled using Latticecell
- **Doublehet**: to be used for double heterogeneous systems with fuel particles dispersed in a fuel component
 - Allowed fuel components are annular and solid rod, pebble and slab
- All options allow for different lattice specifications: square, triangular (hexagonal), slab cell
- If approximation of infinite lattice of fuel pins is insufficient, modify the **Dancoff factor** for that fuel pin
 - Example: Edge and corner fuel pins in high-void BWR lattices
- If a unit cell geometry is widely different from repeated fuel lattices, try to create a multiregion with representative radii and perform thorough comparisons to continuous-energy calculations (see last slide)
 - Example: Many molten salt reactor systems have irregular repeating geometries

Cross Section Processing

What are common input errors?

- Check that all necessary lattice parameters are included: all radii, pitches and heights
- Check that boundary conditions are correct (e.g., **right_bdy=vacuum** – most likely incorrect)
- Do not confuse pitches (pitch) and half pitches (**h**pitch) or radii (fuel**r**) and diameter (fuel**d**)
- For doublehet cells:
 - Specify either volume fraction of fuel particles **OR** number of particles
 - Be aware that coatings can be specified via radius, diameter or thickness: coat**r**, coat**d**, coat
 - You need to define a doublehet mixture ID → don't confuse this ID with the fuel mixture ID from a possible CE model
- Triple check material IDs in cell specifications, especially after a copy/paste from an old input

NEWT Geometry Construction

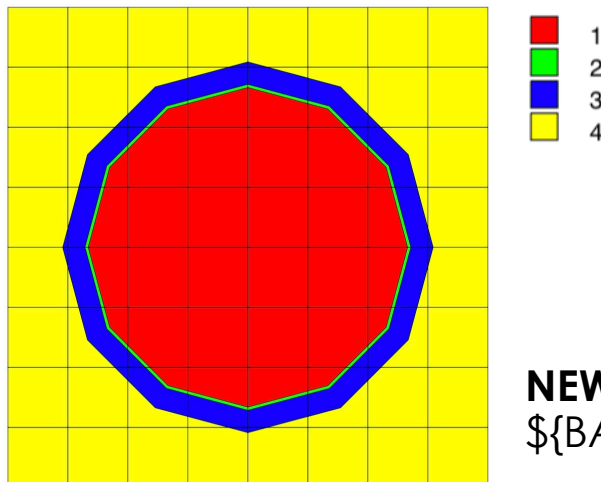
Are there recommendations on how to set up my geometry?

- We recommend giving the unit containing the **fuel unit cell** the same ID as the **fuel material** in that cell
 - Helps eliminate confusion
 - Makes lattice array more visibly intuitive in the input file
- **Use units**
 - You *could* specify everything you need in one global unit, but this can be very confusing
 - For each logical building block in the physical model, build a unit
 - Use arrays and holes to place all the different units into the global unit
 - Example: BWR control blades – if they are constructed as different units, you can simply comment them in or out of the global unit to insert or remove them

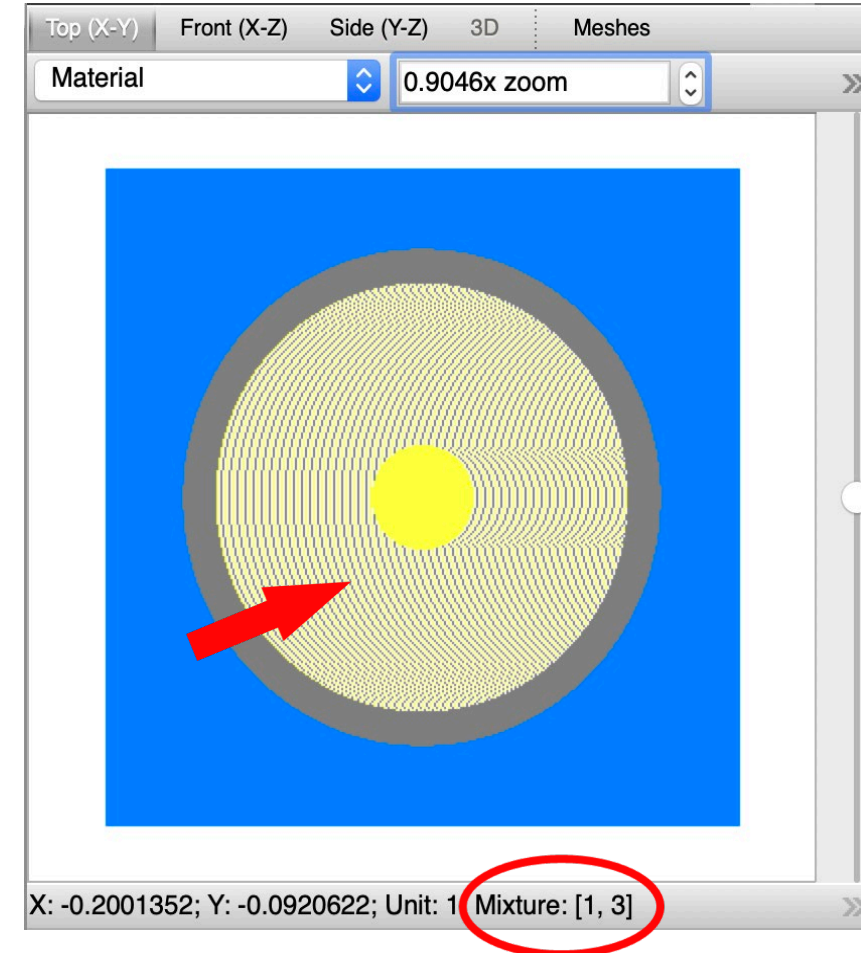
Geometry Check

How do I check that my geometry is correct?

1. Use Fulcrum to check for input validation errors
2. Use Fulcrum to view model
3. Run TRITON-NEWT with *check* parameter:
 - **=t-newt parm=(check)**
 - This checks for so far undetected input errors
 - Can be used to plot the geometry in conjunction with **drawit=yes** in the NEWT parameter block
4. Also helpful: Use Fulcrum to calculate volumes via “Run” --> “Calculate volumes”



NEWT geometry plot:
\${BASENAME}.newtmatl.ps



Fulcrum view:

Typical geometry error due to incorrect media placement, undetectable with the validation check

NEWT transport settings

What options should I use in my NEWT transport solution?

- Currently we use and recommend the basic defaults for NEWT, which are a compromise between accuracy and runtime
- At least S_6 should be used for NEWT calculations
- P_0 in the gap, P_1 in fuel, clad and structural materials, and P_2 in moderators
- Spatial mesh of at least 4x4 in fuel cells
- The global grid is sized such that each grid cell is approximately the size of one unit cell
- CMFD is performed on the global grid
- Epsilon = 1e-5

NEWT hexagonal geometries

- Hexagonal geometries are used in many advanced reactor concepts, such as HTGR, FHR, SFR (and also VVER)
- Unit cell level is straightforward:

- Define appropriate unit cell:

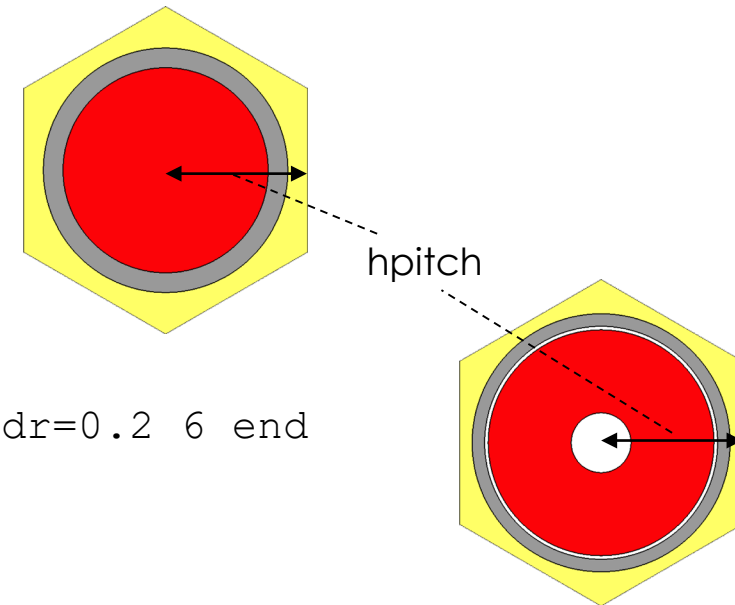
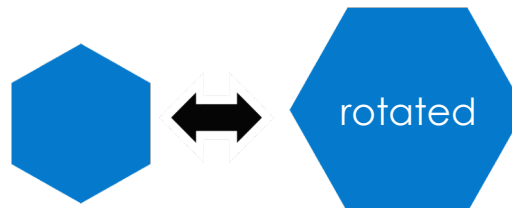
```
latticecell triangpitch hpitch=1.5 1  
    fuelr=1.0 2 gapr=1.2 3 cladr=1.4 4 end
```

```
latticecell attriangpitch hpitch=1.5 1  
    fuelr=1.0 2 gapr=1.2 3 cladr=1.4 4 igapr=0.1 5 icladr=0.2 6 end
```

```
doublehet: rod/pebble + triangpitch/atriangpitch
```

- Define appropriate geometry with consistent materials:

```
hexprism 1 1.0  
rhexprism 2 2.0
```

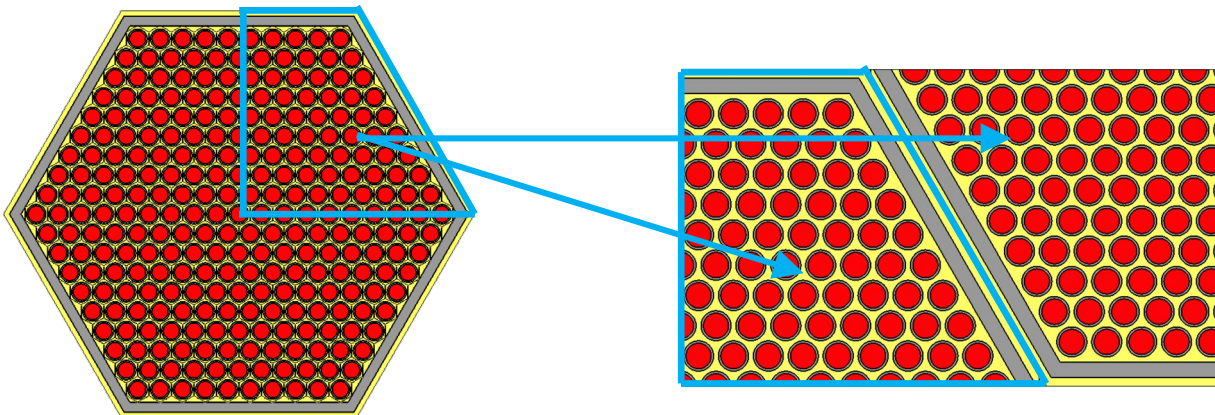


NEWT hexagonal geometries

- Fuel assembly can be trickier, especially if you have an outer wrapper
- Definition of hexagonal array often leads to geometry errors
→ Issues visible in NEWT geometry plot
- Recommendation: explicitly place fuel pin units as **holes** in assembly
- Simplify geometry if possible: reflected assembly can be changed to a rectangular geometry

```
hole 5 origin x=0      y=0
hole 3 origin x=0.89   y=0
hole 1 origin x=0.44   y=0.77
hole 1 origin x=1.34   y=0.77
hole 4 origin x=0      y=1.55
hole 1 origin x=0.89   y=1.55
```

- Further recommendations for this type of geometry:
 - Only global grid
 - cmfd=rect
 - grid_tol=1.0e-5
 - cell_tol=1.0e-8
 - inrcvrg=yes
 - **Check NEWT plot of geometry**
 - Play with grid, npolar, nazim, etc. parameters to check if your solution is converged



Comparisons with Monte Carlo KENO-CE and KENO-MG

I'm doing lattice physics, why do I need Monte Carlo? → To validate your assumptions!

- CE-KENO compared to MG-KENO:
 - Reveals the bias due to cross section processing and use choice of the multigroup cross section library
 - Modify the cross section library, or cross section processing options (Dancoff factors, CENTRM options)
- MG-KENO compared to NEWT:
 - Reveals the bias due to geometry approximations
 - Modify the number of grid cells, angular quadrature, number of sides on cylinders, etc.
- Note: It is relatively easy to generate a KENO model from an existing TRITON model
 - For hexagonal assemblies, you can easily model a hexagonal array in KENO, i.e., without holes
- SCALE 6.3 will include Monte Carlo code Shift that allows the generation of nodal data in t16 files
 - This provides an additional great opportunity for verification with TRITON

A KENO calculation can offer tremendous help in troubleshooting NEWT models and in verifying your results
→ We strongly recommend to perform comparisons with KENO!

Questions?

