

# Kraken workshop

Short overview of KrakenTools

Ville Valtavirta

24/05/2022 VTT – beyond the obvious

# krakentools Python package

Collection of Python scripts, functionalities and definitions useful for the Kraken framework.

Examples:

- Producing water and fuel material definitions for Serpent.
- Producing material volumes for Serpent.
- Regression testing.
- Radial reflector homogenization.
- Condensation of Serpent group constant data.

# Producing water and fuel material definitions for Serpent

# Producing water compositions for Serpent

krakentools.utils.write\_water\_composition()

krakentools.utils.write\_p\_t\_water\_composition()

krakentools.utils.write\_saturated\_water\_composition()

Tests 01, 02, 14, 15.

Feed in basic data: (p, T, boron, ...)

pressure = 15e6 # Pa

temperature = 550 # K

boron = 1000 # ppm (weight)

Get Serpent material definition:

```
mat cool_1000B_76D -0.76978 tmp 550
moder lw550K 1001
rgb 200 200 255
```

```
O-16.03c 3.323385e-01
O-17.03c 1.265963e-04
80180    6.829536e-04
H-1.03c  6.662196e-01
H-2.03c  7.662406e-05
B-10.03c 1.105860e-04
B-11.03c 4.451225e-04
```

# Producing fuel compositions for Serpent

```
krakentools.utils.write_fuel_composition()  
Krakentools/tests/17_fuel_composition/input.py
```

Feed in basic data: (density, enrichment, gadolinium, ...)

```
density = 10.2 # g/cm3  
wt_frac_U234 = 0.0002 # of U  
wt_frac_U235 = 0.04 # of U  
wt_frac_U236 = 0.0002 # of U  
wt_frac_Gd2O3 = 0.02 # of fuel density
```

Get Serpent material definition:

```
mat fuel_20U4_40U5_20U6_20GO -10.2 tmp 800  
92234.06c -1.727707e-04  
92235.06c -3.455413e-02  
92236.06c -1.727707e-04  
92238.06c -8.289537e-01  
8016.06c -1.187942e-01  
64152.06c -3.352815e-05  
64154.06c -3.702705e-04  
64155.06c -2.530124e-03  
64156.06c -3.522011e-03  
64157.06c -2.709997e-03  
64158.06c -4.328777e-03  
64160.06c -3.857778e-03
```

# Producing material volumes for Serpent

# krakentools.utils.process\_material\_volumes()

- Serpent needs information on the volumes of each depletion zone for burnup calculation.
- Depletion zone numbering may be hard to follow.
- Clever use of symmetries increases statistics, but means that some depletion zones can have 2, 4 or 8 times the base volume.
- Serpent can estimate volumes based on Monte Carlo sampling, but that leaves statistical uncertainties.

```
% --- Material volumes:
```

```
% Produced Wed Jul 17 10:40:00 2019 by MC volume calculation routine by  
% sampling 10000000 random points in the geometry.
```

```
set mvol
```

```
fuelNoGad    35 1.98263E+00 % (0.005)  
fuelNoGad    34 1.96811E+00 % (0.004)  
fuelNoGad    33 3.98636E+00 % (0.003)  
fuelNoGad    32 1.99161E+00 % (0.005)  
fuelNoGad    31 3.96966E+00 % (0.004)  
fuelNoGad    30 3.98779E+00 % (0.003)  
fuelNoGad    29 3.96966E+00 % (0.003)
```

```
...
```

# krakentools.utils.process\_material\_volumes()

- Some clean up can be made to the Monte Carlo calculated volumes based on basic information on the depletion zone setup.
- KrakenTools/tests/06\_material\_volume\_processing/input.py
- KrakenTools/tests/62\_process\_arbitrary\_fuel\_volumes/input.py

```
% --- Material volumes:
```

```
set mvol
```

```
fuelNoGad      35 1.98557E+00    %% original volume 1.98263E+00 was a multiple of 4 (3.994 as float)
fuelNoGad      34 1.98557E+00    %% original volume 1.96811E+00 was a multiple of 4 (3.965 as float)
fuelNoGad      33 3.97113E+00    %% original volume 3.98636E+00 was a multiple of 8 (8.031 as float)
fuelNoGad      32 1.98557E+00    %% original volume 1.99161E+00 was a multiple of 4 (4.012 as float)
fuelNoGad      31 3.97113E+00    %% original volume 3.96966E+00 was a multiple of 8 (7.997 as float)
fuelNoGad      30 3.97113E+00    %% original volume 3.98779E+00 was a multiple of 8 (8.034 as float)
fuelNoGad      29 3.97113E+00    %% original volume 3.96966E+00 was a multiple of 8 (7.997 as float)
```

```
...
```



# Simple regression testing module

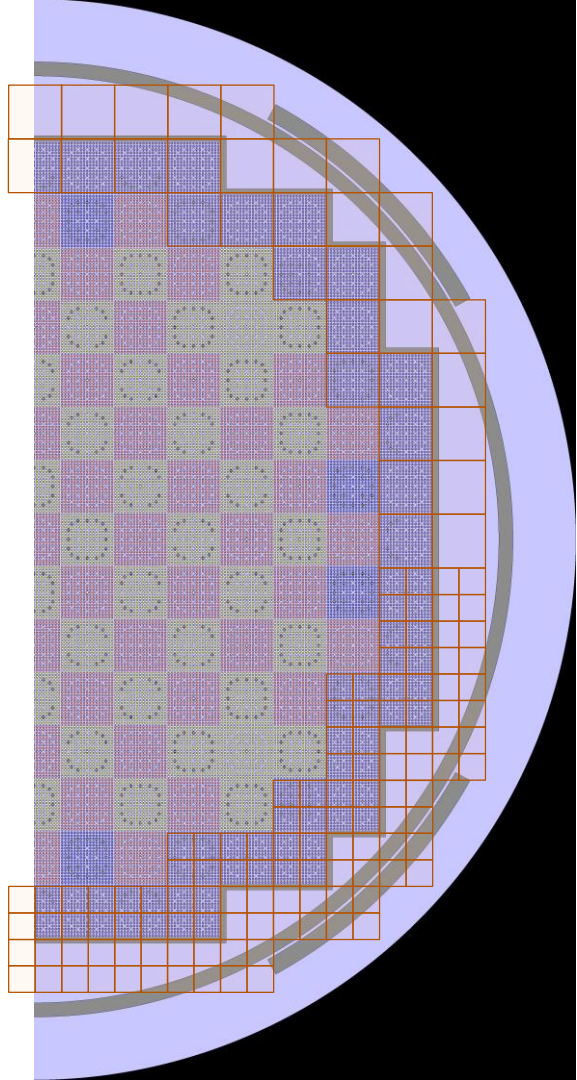
# krakentools.testing

- Provides functionalities for regression testing during Kraken development.
- Runs all tests or some subset.
- Compares results to an earlier reference in three potential ways:
  1. File fully matches the reference.
  2. The reference file is a part of the new file.
    - New file is allowed to contain some additional parts.
  3. A user/developer written script checks the output to see if it is acceptable.
    - Flexible and extendable.
- Utilized in regression testing of the Kraken modules:
  - Serpent, Ants, SuperFINIX, Kharon, KrakenTools, Cerberus, ...

# Radial reflector homogenization

# Radial reflector homogenization

- We homogenize the radial reflector for Ants from a Serpent 2D full core simulation (square or hexagonal lattice).
- The tedious parts are automated with [krakentools.CoreLattice](#) and [krakentools.reflectorhg](#):
  - Setting up the superimposed universes to the Serpent model.
    - 1x1 or 2x2, one layer of reflector or multiple.
  - Collecting Serpent output data and averaging it over symmetry sectors.
  - Running single node Ants calculations to evaluate homogeneous surface fluxes for discontinuity factors.
  - (potentially condensing the data further)
  - Group constant fitting and library production.
- See tests 08, 11, 13, 16, 48, 49
- Also see: [Evaluating the X2 Initial Core Zero Power Physics Tests with Serpent-Ants](#)  
Ville Valtavirta *et al.*  
Tue 1:30 pm, Brighton III/IV, *Data, Methods, Code Validation: V*



# Condensation of Serpent group constant data

# Condensing Serpent group constant data further

- Generate group constants with Serpent into some energy group structure.
  - Read the data into NodeBranch objects with
    - `krakentools.serpent.read_coe_and_res()`
- Reading utilizes functionalities from `serpentTools`: Andrew Johnson, Dan Kotlyar, Stefano Terlizzi, and Gavin Ridley, “serpentTools: A Python Package for Expediting Analysis with Serpent.”, Nuclear Science and Engineering, 194 (2020)
- Each `krakentools.containers.NodeBranch` object contains data for a single universe/`node` at specific conditions (`branch`).
  - Use `NodeBranch.give_condensed_version()` to obtain an energy condensed variant (both infinite spectrum and leakage corrected data if available).
  - See tests 50 and 51.

May be extended for conducting leakage correction outside of Serpent in the future.

# Summary

- KrakenTools collects many functionalities for Kraken related tasks.
  - Built primarily to serve VTT internal purposes as most of the Kraken framework.
- For general Serpent purposes you may have your own scripts already:
  - Producing material compositions
  - ...
- For Kraken purposes you may find the capabilities of KrakenTools invaluable:
  - Full core 2D radial reflector homogenization.
  - Group constant parametrization.
  - ...

Serpent users should also consider testing out serpentTools:  
<https://github.com/CORE-GATECH-GROUP/serpent-tools>  
<https://serpent-tools.readthedocs.io/en/master/overview.html>

# bey<sup>0</sup>nd

## the obvious

Ville Valtavirta  
Ville.Valtavirta@vtt.fi

@VTTFinland

[www.vtt.fi](http://www.vtt.fi)