Monte Carlo Methods and Simulations in Nuclear Technology

Introduction to SERPENT and XS Libraries

Jan Dufek

2022

KTH Royal Institute of Technology

Contents

Nuclear data libraries

Nuclear data libraries

What other data than cross sections are included in neutron cross section libraries?

Nuclear data libraries contain:

- Cross sections for reactions:
  - elastic scattering, (n,2n), (n,3n),...
  - inelastic scattering (n,n'), (n,γ), (n,p) and (n,α)
  - total, absorption, proton prod. and α production
- Angular distributions for different reactions including (n,n'), (n,2n) and (n,3n)
- Energy distribution for some reactions

Nuclear data libraries

What format of nuclear data libraries is used by SERPENT?

SERPENT uses nuclear data in ACE format.

How can ACE libraries be generated?

- ACE libraries can be generated for any temperature from raw ENDF format libraries by the NJOY code.
- SERPENT comes equipped with ACE XS libraries generated at several temperatures from actual JEFF and ENDF/B libraries.

Nuclear data libraries

What is the SERPENT directory file for?

SERPENT directory file contains data necessary for locating the XS libraries. Each line in the directory file has the following format:

`<alias> <zaid> <type> <ZA> <I> <AW> <T> <bin> <path>`

where

- `<alias>` is the name identifying the nuclide in the input file
- `<zaid>` is the actual nuclide name in the data library
- `<type>` is the type of the data
- `<ZA>` is the nuclide identifier (1000*Z + A)
- `<I>` is the isomeric state number (0 = ground state)
- `<AW>` is the atomic weight
- `<T>` is the nuclide temperature (in K)
- `<bin>` is the binary format flag (0 = ASCII, 1 = binary)
- `<path>` is the data path for the library

Example of SERPENT directory file

```
   1001.06c  1001.06c 1  1001 0   1.00783  600.0 0 /xs/1001_06.ace
    H-1.06c  1001.06c 1  1001 0   1.00783  600.0 0 /xs/1001_06.ace
   8016.06c  8016.06c 1  8016 0  15.99492  600.0 0 /xs/8016_06.ace
   O-16.06c  8016.06c 1  8016 0  15.99492  600.0 0 /xs/8016_06.ace
  40000.06c 40000.06c 1 40000 0  91.21963  600.0 0 /xs/40000_06.ace
 Zr-nat.06c 40000.06c 1 40000 0  91.21963  600.0 0 /xs/40000_06.ace
  92235.09c 92235.09c 1 92235 0 235.04415  900.0 0 /xs/92235_09.ace
  U-235.09c 92235.09c 1 92235 0 235.04415  900.0 0 /xs/92235_09.ace
  92238.09c 92238.09c 1 92238 0 238.05078  900.0 0 /xs/92238_09.ace
  U-238.09c 92238.09c 1 92238 0 238.05078  900.0 0 /xs/92238_09.ace
  95342.09c 95342.09c 1 95242 1 242.05942  900.0 0 /xs/95342_09.ace
Am-242m.09c 95342.09c 1 95242 1 242.05942  900.0 0 /xs/95342_09.ace
   lwtr.03t  lwtr.03t 3     0 0   0.00000    0.0 0 /xs/tmccs1
 Np-237.30y 93237.30y 2 93237 0 239.10201    0.0 0 /xs/llldos1
```

Figure 1: Example of SERPENT directory file

Note that the path to the directory file has to be present in the input file.

SERPENT input file

SERPENT input file

Can you give examples of data that need to be specified in order to carry out a neutron transport simulation?

- Type of the problem (criticality or fixed-source, steady-state or transient, burnup, etc.)
- Required results (detector response, eigenvalue, reaction rates)
- Geometry of the system (shapes of volume regions)
- Material properties of different volume regions
  - nuclide composition
  - XS library
  - temperature (cross section are dependent on temperature)
- Additional information for criticality calculations:
  - Selection of the initial fission source, batch size, number of inactive and active cycles
  - Selection of variance reduction techniques (depending on required results)

SERPENT input file

What denotes the term "card" in the SERPENT terminology?

The SERPENT input file consists of so-called "cards". A card is a data block that consists of a key word following by parameters specific to the key word.

Basic cards:

- `surf` - surface definition
- `cell` - defines a cell (volume region)
- `lat` - lattice definition
- `pin` - pin definition
- `mat` - material definition
- `set` - definition of various parameters

SERPENT input file

Can you include your comments in the SERPENT input file?

Yes. Anything that follows after the percent-sign (%) or hash (#) is considered to be a user comment.

Are basic units specified in the input file?

No. Basic units in SERPENT input and output files are set by default:

Distance	cm
Volume	cm^3
Energy	MeV
Mass	g
Mass density	g/cm^3
Atomic density	$10^{24}/cm^3$
Power	W
Neutron flux	$1/cm^2s$ (per neutron history)
Burnup time	days

SERPENT input file

Geometry description consists of cells. Each cell is shaped by a surface that is described by a special card. The surface card syntax is:

```
surf <id> <type> <param 1> <param 2> ...
```

What are the inputs after the keyword?

- `<id>` is the surface identifier
- `<type>` is the surface type (next slide)
- `<param 1> <param 2> ...` are the surface parameters

Surface types

Type	Description	Parameters
`inf`	all space	-
`px`	plane perpendicular to x-axis	x_0
`py`	plane perpendicular to y-axis	y_0
`pz`	plane perpendicular to z-axis	z_0
`sph`	sphere	x_0, y_0, z_0, r
`cyl`	circular cylinder parallel to z-axis	x_0, y_0, r
`sqc`	square cylinder parallel to z-axis	x_0, y_0, r, r_0
`cube`	cube	x_0, y_0, z_0, r
`hexxc`	x-type hexagonal cylinder parallel to z-axis	x_0, y_0, r, r_0
`hexyc`	y-type hexagonal cylinder parallel to z-axis	x_0, y_0, r, r_0
`cross`	cruciform cylinder parallel to z-axis	x_0, y_0, r, d, r_0
`pad`	(see description below)	$x_0, y_0, r_1, r_2, \theta_1, \theta_2$
`cone`	cone oriented in the z-axis	x_0, y_0, z_0, r, h
`dode`	dodecagonal cylinder parallel to z-axis	x_0, y_0, r_1, r_2

Figure 2: Surface types

SERPENT input file

Can you interpret the following example of a surface card?

```
surf 11 sph 0. 0. 0. 10.
```

This card denotes a surface of a sphere with radius of 10 cm, positioned at the origin (0.,0.,0.); the surface is identified by number 11.

SERPENT input file

Material cards are described as:

The syntax of the material card is:

```
mat <name> <dens> [<options>]
<iso 1> <frac 1>
<iso 2> <frac 2>
```

where

- `<name>` is material name (arbitrary)
- `<dens>` is the density (mass if negative, atomic if positive)
- `<iso 1> <iso 2> ...` are names of nuclides
- `<frac 1> <frac 2> ...` are the corresponding fractions

Are the fractions `<frac 1> <frac 2> ...` giving the mass or atomic fractions?

`<frac 1> <frac 2> ...` give mass fractions when they are negative, and atomic fractions when they are positive.

SERPENT input file

Can you interpret the following example of a material card?

```
mat UO2 -10.
92235.09c -0.02699
92238.09c -0.85447
8016.09c -0.11853
```

This card defines a new material identified as UO2 with density 10 g/cm^3, consisting of three nuclides: ^{235}U of rel. density fraction .02699, ^{238}U of rel. density fraction .85447, and ^{16}O of rel. density fraction .11853.

Does the sum of mass or atomic fractions need to be normalised to one (in its absolute value)?

No, but make it your habit to normalise the fractions to one unless you have a good reason not to. (It will be much easier for you to debug your input file then.)

SERPENT input file

What is the thermal scattering library and what is its syntax?

The syntax of the thermal scattering card is:

```
therm <thname> <lib>
```

where

- `<thname>` is an arbitrary name for this data
- `<lib>` is the library identifier as defined in the directory file

Note:

The thermal scattering library is applied to a specific nuclide in a specific material via an additional parameter "`moder`", see the example on the next slide.

SERPENT input file

Example of a definition of a moderator material

The two cards

```
mat water -0.7 moder lwtr 1001
1001.06c 2
8016.06c 1

therm lwtr lwj3.11t
```

define a new material identified as "`water`" with a density 0.7g/cm^3, where hydrogen uses thermal scattering library "`lwj3.11t`".

SERPENT input file

What does the positive and negative region mean and how are they used to shape cells?

- Geometry in SERPENT consists of volume regions - cells.
- The cells are defined as intersections of space regions shaped by various surfaces.
- Each surface splits the space into two regions; these are marked as positive and negative.
- Negative regions are inside the closed surfaces or facing the negative coordinate axis. For instance, the space region -11 denotes the inside space shaped by the spherical surface 11.

SERPENT input file

What does the "universe" mean in SERPENT terminology? Why do we define various universes?

- In SERPENT, a cell can be filled with a "universe" - a structure of other cells. For instance, a box can be filled with a lattice of fuel rods, creating a fuel assembly.
- Various cell structures are needed, e.g. fuel pins with different fuels, various fuel assemblies (FA), etc. Therefore many universes can be defined, each with a unique id number.
- Every cell must belong to a certain universe.
- A universe must be defined in the whole space.
- Only universe 0 is reflected in the model directly; all other universes are virtual and their parts are used to fill cells of other universes.
- Thus, universes are linked together in a branch-like structure, all ending in universe 0.

SERPENT input file

The syntax of the cell card is:

```
cell <name> <u> <mat> <surf 1> <surf 2> ...
```

where:

- `<name>` is the cell unique id number (arbitrary)
- `<u>` is the universe number of the cell
- `<mat>` is the cell material
- `<surf 1> <surf 2> ...` are the space regions

What are the options for `<mat>`?

- name of the material filling the cell
- "void" - in case the cell is empty
- "outside" - outside of the system
- "fill" `<u>` - the cell is filled with another universe with id `<u>`

SERPENT input file

Interpret the example of the following cell cards.

```
cell 1 0 fuel -11
cell 2 0 clad 11 -12
cell 3 0 water 12
```

This card defines directly universe 0. Assuming surface 11 is the sphere defined previously then the above cells define a sphere that is filled with a material with id “fuel”, covered by a “clad” layer, and surrounded by a material with id “water”.

What are lattice cards used for?

Lattices are special universes, filled with regular structure of other universes.
The syntax for square and hexagonal lattices is:

`lat <u0> <type> <x0> <y0> <nx> <ny> <p> <u_1> <u_2> ...`

where

`<u0>` is the universe number of the lattice

`<type>` is the lattice type (1 for square lattice)

`<x0>` is the x coordinate of the lattice origin

`<y0>` is the y coordinate of the lattice origin

`<nx>` is the number of lattice elements in the x-direction

`<ny>` is the number of lattice elements in the y-direction

`<p>` is the lattice pitch

`<u_1>, <u_2>,...` are universe numbers that define the layout.

Example of a lattice card

```
lat 10  1 0.0 0.0 17 17 1.26

1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1
1 2 3 3 3 2 3 3 2 3 3 2 3 3 3 2 1
2 3 3 3 3 4 3 3 4 3 3 4 3 3 3 3 2
2 3 3 4 3 3 3 3 3 3 3 3 3 4 3 3 2
2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2
2 2 4 3 3 4 3 3 4 3 3 4 3 3 4 2 2
2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2
2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2
2 2 4 3 3 4 3 3 4 3 3 4 3 3 4 2 2
2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2
2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2
2 2 4 3 3 4 3 3 4 3 3 4 3 3 4 2 2
2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2
2 3 3 4 3 3 3 3 3 3 3 3 3 4 3 3 2
2 3 3 3 3 4 3 3 4 3 3 4 3 3 3 3 2
1 2 3 3 3 2 3 3 2 3 3 2 3 3 3 2 1
1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1
```

Figure 3: Example of a lattice card

SERPENT input file

Which card specifies free parameters for criticality calculations?

Neutron batch size and criticality cycles in criticality calculation are specified by the card:

```
set pop <npop> <cycles> <skip> [<keff0>]
```

where

`npop` - is the batch size

`cycles` - is the number of active cycles

`skip` - is the number of inactive cycles

`keff0` - is the initial guess for the k-eigenvalue

SERPENT input file

Which card specifies the directory file path?

The directory file path is specified as:

```
set acelib "<file>"
```

where
`<file>` is the file path for the directory file.

SERPENT input file

How are the boundary conditions specified?

Boundary conditions Boundary conditions determine the fate of neutrons escaping outside the defined geometry. The boundary conditions are set using:

```
set bc <mode>
```

where
<mode> is the boundary condition mode (1 - black [neutron is killed], 2 - reflective).

Running the SERPENT simulation

Running the SERPENT simulation

How to connect to our SERPENT server

- install an SSH client on your computer,
- connection setting (for now): IP: 130.237.70.233, port: 2222,
- user name: (- *given in person* -),
- password: **** (- *given in person* -),
- SERPENT, XS libraries and examples in /codes/SERPENT
- Linux terminal command to connect:
  `ssh -p 2222 username\@130.237.70.233`

Running SERPENT

Command to run SERPENT To run SERPENT type in a Linux terminal window the command

```
sss <input file> [<options>]
```

basic options:

- `-checkvolumes <N>` (estimates the material volumes by sampling `<N>` random points)
- `-testgeom <N>` (tests the geometry using `<N>` randomly sampled neutron histories)
- `-plot` (terminates run after geometry plot)

Example:

```
sss bwr_input
```

Running SERPENT

Command to run SERPENT as a background process

When you need to run a long simulation, run it as a background process. You can close the terminal window then. Do it by giving the command:

```
nohup sss <input file> [<options>] > /dev/null &
```

where

`> /dev/null`

trashes the (not-terribly-useful) terminal output text that you'd normally see running on the screen.

Example:

```
nohup sss bwr_input > /dev/null &
```

Terminating a calculation

If you realise you made a mistake in the input file and want to terminate a calculation running in the background, do it by giving the command

```
kill <process_id>
```

where `<process_id>` is an id number of the process of your calculation. You can find out this number using the command "top" (to quit the "top" program just press "q").