Delayed neutron emission model for time dependent simulations with the Serpent 2 Monte Carlo code – First results

PHYSOR 2016, Sun Valley, Idaho
May 3, 2016

V. Valtavirta[†], M. Hessan[‡] and J. Leppänen[†]
[†] VTT Technical Research Center of Finland
[‡] RWTH Aachen

Outline

- History/background/objectives.
- Overview of the transient simulation approach.
- Implementation.
  - Source generation
  - Transient simulation.
- Test calculations
  - Infinite reactor with various reactivity insertions.
  - Critical 3D assembly case.
- Current features and limitations.
- Summary.

Background

As a part of the multi-physics development of Serpent 2, we are interested in coupled transient analysis for reactor applications. As an independent solver and for benchmarking deterministic methods.

Serpent has has a dynamic simulation mode for several years[1] but two technical limitations had to be overcome in order to apply the mode to reactor modeling:

While the live neutron source in the beginning of a transient could be generated, no method existed for generating the initial delayed neutron source.

Delayed neutron emission should be handled in a slightly different manner to ensure good performance in all cases.

---

[1] J. Leppänen. "Development of a Dynamic Simulation Mode in the Serpent 2 Monte Carlo Code". In: M&C 2013. Sun Valley, ID, May 5-9, 2013.

Overview

Transient simulations with Serpent are executed in two parts:

1. A criticality source simulation to generate the steady state neutron and delayed neutron precursor sources for the time dependent simulation.
2. A time dependent simulation to model the time evolution of the system starting from the steady state source distributions.

Instead of sampling delayed neutron emission in fission, we will create delayed neutron precursors. The precursor populations will be tracked instead and delayed neutrons will be emitted from them[2].

There are several advantages in tracking precursors: No need to use expensive memory to store particles that will exist only at some point in the future. Any number of delayed neutrons can be emitted from the known precursor distribution. The precursors can be produced, not only in fissions, but in every interaction using an implicit estimator.

---

[2] This is in many ways similar to the work done by Bart Sjenitzer as a part of his Ph.D. thesis "The Dynamic Monte Carlo Method for Transient Analysis of Nuclear Reactors", Delft University of Technology (2013)

Implementation

Source generation

In order to start the transient simulation, we'll need to know the initial source distributions in the system. We can think of the source generation as a snapshot of the system at a random time and looking at the neutrons we captured in our picture:

1. "Live" neutrons travelling at a certain position (x, y, z) to a certain direction (u, v, w) at a certain energy E.
2. Delayed neutrons "waiting" in precursor atoms at a certain position (x, y, z) and with a certain decay constant λ.

We will create both the live neutron source distribution and the precursor source distribution during a single criticality source (k-eigenvalue) simulation.

Implementation

Source generation (live neutrons)

To create the live neutron source distribution we store neutrons at random points of their life. These random points should be distributed uniformly in time.

We can store neutrons during the simulation at tentative interaction sites. However, the interactions of a neutron are not distributed uniformly in time. Fast moving (high energy) neutrons take less time between interactions as do neutrons travelling in materials with a high material cross section.

The mean interaction frequency of a neutron with energy E travelling in a material with a total macroscopic cross section of $\Sigma_{\mathrm{tot}}(E)$ is

$$f_{\mathrm{mean}}(E) = \frac{1}{t_{\mathrm{mean}}(E)} = \Sigma_{\mathrm{tot}}(E)v(E). \qquad (1)$$

To store neutrons at uniformly distributed random points in time, the neutron is stored at a tentative interaction site with a probability

$$P \propto \frac{1}{v\Sigma_{\mathrm{path}}}. \qquad (2)$$

For each live neutron, the important data to store is the location (x, y, z) the direction (u, v, w) and the energy E of the neutron.

Implementation

Source generation (precursors)

By generating the precursor source, we essentially want to estimate the precursor distribution in the system at the beginning of the transient.

This can be done in a straightforward manner if the initial system is critical and in steady-state. In this case it is enough to calculate the precursor production rates, which can be converted to stable populations by dividing the production rate by the decay constant of the precursor.

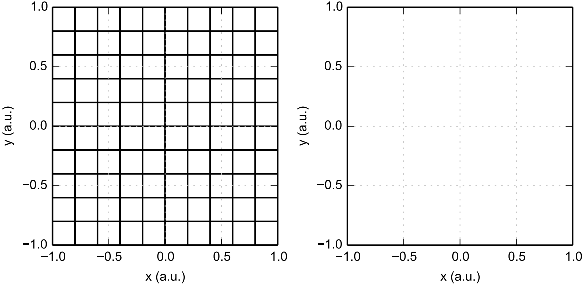
Two ways to track the precursor populations were implemented

1. Mesh-based approach.
2. Point-wise approach.

Serpent currently tracks each precursor group separately.

Implementation

Source generation (precursors)



The spatial fidelity of the mesh-based approach will be limited by the choice of the mesh size. Running more neutrons will give better statistic in each of the mesh bins.

The spatial fidelity of the point-wise approach will be limited by the number of stored precursors. Running more neutrons will increase the number of stored precursors and thus the spatial fidelity[3]. If the precursor points have to be held in RAM[4], population control may be required to limit the memory consumption.

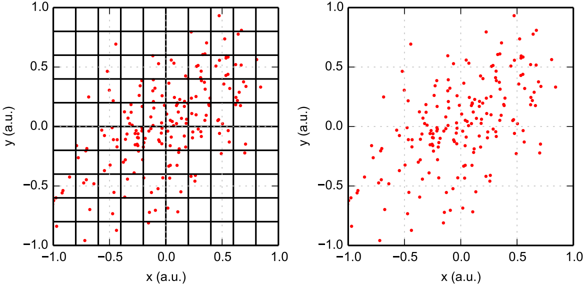
---

[3] The stored precursors can be thought of as point-wise tally bins.

[4] As is the case in the transient simulation

Implementation

Source generation (precursors)



The spatial fidelity of the mesh-based approach will be limited by the choice of the mesh size. Running more neutrons will give better statistic in each of the mesh bins.

The spatial fidelity of the point-wise approach will be limited by the number of stored precursors. Running more neutrons will increase the number of stored precursors and thus the spatial fidelity[3]. If the precursor points have to be held in RAM[4], population control may be required to limit the memory consumption.

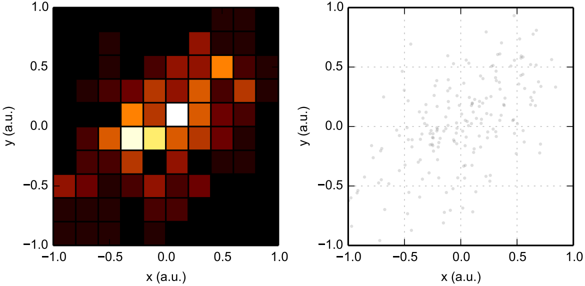
---

[3] The stored precursors can be thought of as point-wise tally bins.

[4] As is the case in the transient simulation

Implementation

Source generation (precursors)



The spatial fidelity of the mesh-based approach will be limited by the choice of the mesh size. Running more neutrons will give better statistic in each of the mesh bins.

The spatial fidelity of the point-wise approach will be limited by the number of stored precursors. Running more neutrons will increase the number of stored precursors and thus the spatial fidelity[3]. If the precursor points have to be held in RAM[4], population control may be required to limit the memory consumption.

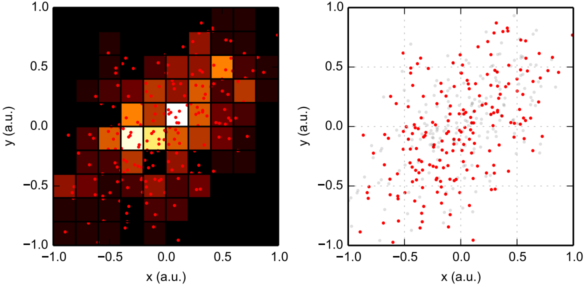
---

[3] The stored precursors can be thought of as point-wise tally bins.

[4] As is the case in the transient simulation

Implementation

Source generation (precursors)



The spatial fidelity of the mesh-based approach will be limited by the choice of the mesh size. Running more neutrons will give better statistic in each of the mesh bins.

The spatial fidelity of the point-wise approach will be limited by the number of stored precursors. Running more neutrons will increase the number of stored precursors and thus the spatial fidelity[3]. If the precursor points have to be held in RAM[4], population control may be required to limit the memory consumption.

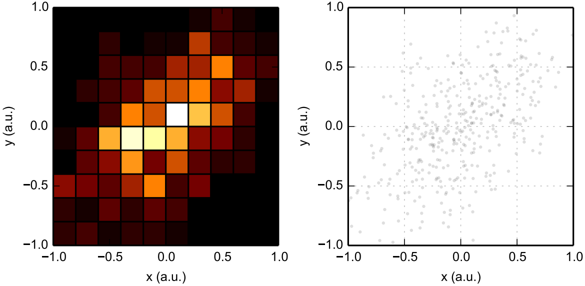
---

[3] The stored precursors can be thought of as point-wise tally bins.

[4] As is the case in the transient simulation

Implementation

Source generation (precursors)



The spatial fidelity of the mesh-based approach will be limited by the choice of the mesh size. Running more neutrons will give better statistic in each of the mesh bins.

The spatial fidelity of the point-wise approach will be limited by the number of stored precursors. Running more neutrons will increase the number of stored precursors and thus the spatial fidelity[3]. If the precursor points have to be held in RAM[4], population control may be required to limit the memory consumption.

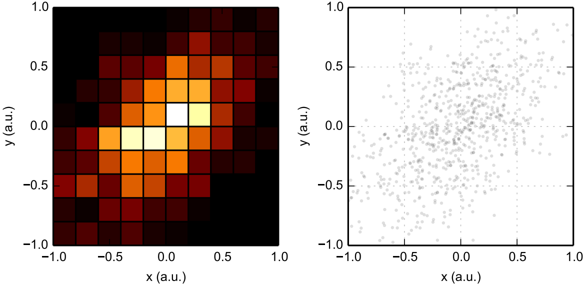
---

[3] The stored precursors can be thought of as point-wise tally bins.

[4] As is the case in the transient simulation

Implementation

Source generation (precursors)



The spatial fidelity of the mesh-based approach will be limited by the choice of the mesh size. Running more neutrons will give better statistic in each of the mesh bins.

The spatial fidelity of the point-wise approach will be limited by the number of stored precursors. Running more neutrons will increase the number of stored precursors and thus the spatial fidelity[3]. If the precursor points have to be held in RAM[4], population control may be required to limit the memory consumption.

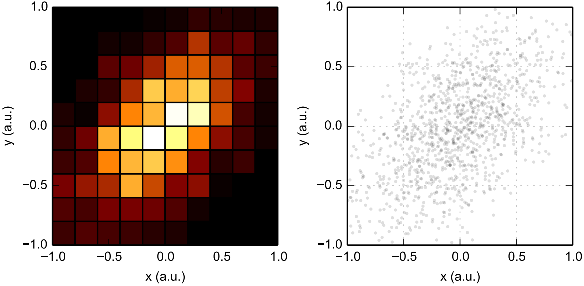
---

[3] The stored precursors can be thought of as point-wise tally bins.

[4] As is the case in the transient simulation

Implementation

Source generation (precursors)



The spatial fidelity of the mesh-based approach will be limited by the choice of the mesh size. Running more neutrons will give better statistic in each of the mesh bins.

The spatial fidelity of the point-wise approach will be limited by the number of stored precursors. Running more neutrons will increase the number of stored precursors and thus the spatial fidelity[3]. If the precursor points have to be held in RAM[4], population control may be required to limit the memory consumption.

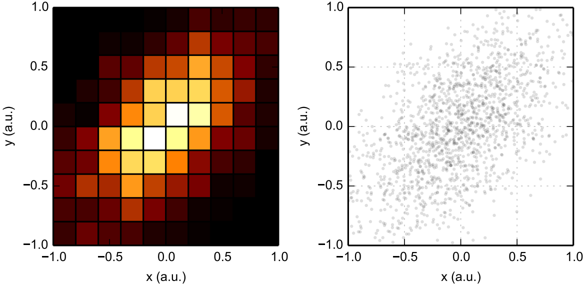
---

[3] The stored precursors can be thought of as point-wise tally bins.

[4] As is the case in the transient simulation

Implementation

Source generation (precursors)



The spatial fidelity of the mesh-based approach will be limited by the choice of the mesh size. Running more neutrons will give better statistic in each of the mesh bins.

The spatial fidelity of the point-wise approach will be limited by the number of stored precursors. Running more neutrons will increase the number of stored precursors and thus the spatial fidelity[3]. If the precursor points have to be held in RAM[4], population control may be required to limit the memory consumption.

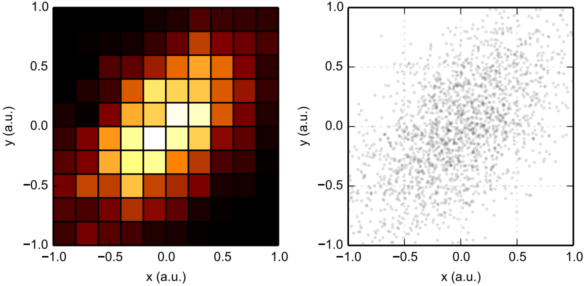
---

[3] The stored precursors can be thought of as point-wise tally bins.

[4] As is the case in the transient simulation

Implementation

Source generation (precursors)



The spatial fidelity of the mesh-based approach will be limited by the choice of the mesh size. Running more neutrons will give better statistic in each of the mesh bins.

The spatial fidelity of the point-wise approach will be limited by the number of stored precursors. Running more neutrons will increase the number of stored precursors and thus the spatial fidelity[3]. If the precursor points have to be held in RAM[4], population control may be required to limit the memory consumption.

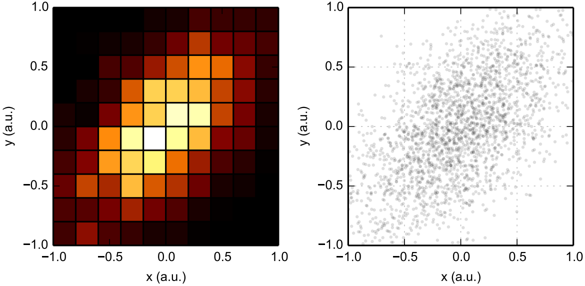
---

[3] The stored precursors can be thought of as point-wise tally bins.

[4] As is the case in the transient simulation

Implementation

Source generation (precursors)



The spatial fidelity of the mesh-based approach will be limited by the choice of the mesh size. Running more neutrons will give better statistic in each of the mesh bins.

The spatial fidelity of the point-wise approach will be limited by the number of stored precursors. Running more neutrons will increase the number of stored precursors and thus the spatial fidelity[3]. If the precursor points have to be held in RAM[4], population control may be required to limit the memory consumption.

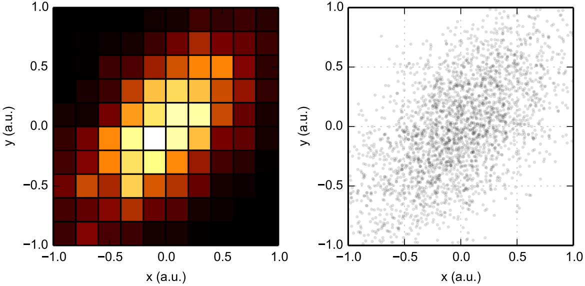
---

[3] The stored precursors can be thought of as point-wise tally bins.

[4] As is the case in the transient simulation

Implementation

Source generation (precursors)



The spatial fidelity of the mesh-based approach will be limited by the choice of the mesh size. Running more neutrons will give better statistic in each of the mesh bins.

The spatial fidelity of the point-wise approach will be limited by the number of stored precursors. Running more neutrons will increase the number of stored precursors and thus the spatial fidelity[3]. If the precursor points have to be held in RAM[4], population control may be required to limit the memory consumption.

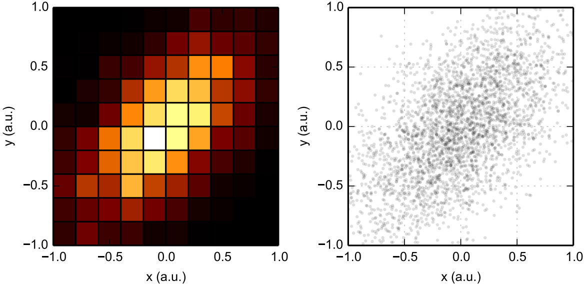
---

[3] The stored precursors can be thought of as point-wise tally bins.

[4] As is the case in the transient simulation

Implementation

Source generation (precursors)



The spatial fidelity of the mesh-based approach will be limited by the choice of the mesh size. Running more neutrons will give better statistic in each of the mesh bins.

The spatial fidelity of the point-wise approach will be limited by the number of stored precursors. Running more neutrons will increase the number of stored precursors and thus the spatial fidelity[3]. If the precursor points have to be held in RAM[4], population control may be required to limit the memory consumption.

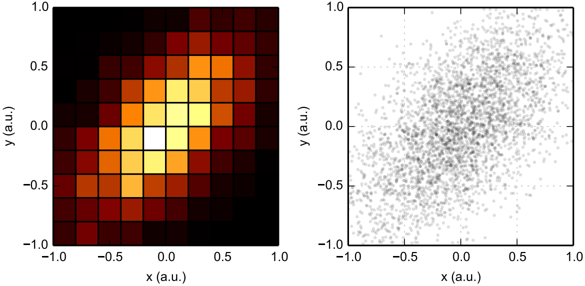
---

[3] The stored precursors can be thought of as point-wise tally bins.

[4] As is the case in the transient simulation

Implementation

Source generation (precursors)



The spatial fidelity of the mesh-based approach will be limited by the choice of the mesh size. Running more neutrons will give better statistic in each of the mesh bins.

The spatial fidelity of the point-wise approach will be limited by the number of stored precursors. Running more neutrons will increase the number of stored precursors and thus the spatial fidelity[3]. If the precursor points have to be held in RAM[4], population control may be required to limit the memory consumption.

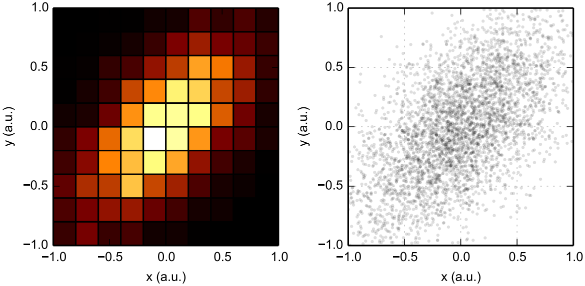
---

[3] The stored precursors can be thought of as point-wise tally bins.

[4] As is the case in the transient simulation

Implementation

Source generation (precursors)



The spatial fidelity of the mesh-based approach will be limited by the choice of the mesh size. Running more neutrons will give better statistic in each of the mesh bins.

The spatial fidelity of the point-wise approach will be limited by the number of stored precursors. Running more neutrons will increase the number of stored precursors and thus the spatial fidelity[3]. If the precursor points have to be held in RAM[4], population control may be required to limit the memory consumption. ☐

---

[3] The stored precursors can be thought of as point-wise tally bins.

[4] As is the case in the transient simulation

Implementation

Source generation (result)

After the source generation we have a number of stored neutrons

```
x0 y0 z0 u0 v0 w0 E0
x1 y1 z1 u1 v1 w1 E1
...
```

mesh-tally values for precursor population in each group

```
group0 meshidx00 val00 err00
group0 meshidx01 val01 err01
...
group1 meshidx11 val11 err11
...
```

and a number of point-wise precursors belonging to different groups

```
x0 y0 z0 group0 pop0
x1 y1 z1 group1 pop1
...
```

Each of the point-wise precursors represents a certain population of precursors.

We also store the total physical neutron population of the system to be used in the normalization of the transient simulation.

Transient simulation

Overview

The simulation time can be divided into sub-intervals for population control. In the simplest case there is a single time-interval. For each time-interval the process is following:

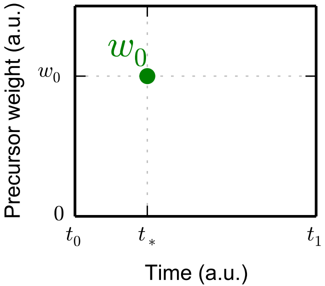
Create primary source based on initial live neutron source and the emission of delayed neutrons from the precursor tallies over the time-interval. The initial source points are divided between live neutrons and delayed neutrons based on proportion of physical population.

Simulate primaries and secondaries that are emitted during the time-interval. Tally the precursor production during the neutron tracking.

Store neutrons and precursors that reach the end of the time interval and start again for the next time interval.

Transient simulations

Precursor production

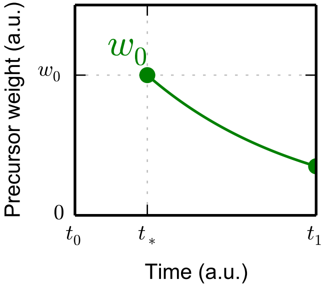


Delayed neutrons are not produced in fissions. Precursors can be produced in each sampled interaction in an implicit manner. If a neutron with weight w_{in} interacts with matter, the average produced precursor weight is

$$w_0 = w_{\mathrm{in}} \frac{\Sigma_f}{\Sigma_{\mathrm{tot}}} \nu \beta_g$$

Transient simulations

Precursor production

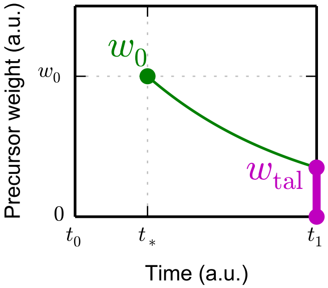


Part of the produced precursor weight will decay before the end of the interval.

$$w_0 = w_{\rm in}\frac{\Sigma_f}{\Sigma_{\rm tot}}\nu\beta_g$$

Transient simulations

Precursor production



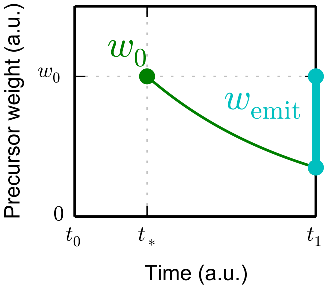
The surviving part of the weight is added to the end-of-interval tallies.

$$w_{\mathrm{tal}} = w_0\, e^{-\lambda_g(t_1 - t_*)}$$

In case of point-wise precursor tracking a new precursor has to be stored to memory. Since w_{tal} can be very small, Russian roulette is played to either store the precursor with a higher weight or to not store the precursor.

Transient simulations

Precursor production



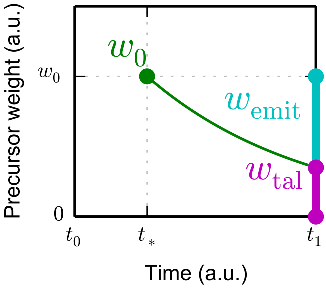
The non-surviving part of the weight is emitted as a delayed neutron.

$$w_{\mathrm{emit}} = w_0 - w_{\mathrm{tal}} = w_0(1 - e^{-\lambda_g(t_1 - t_*)})$$

Since w_{emit} can be very small, Russian roulette is played to either increase the weight to a reasonable level or not emit at all.

Transient simulations

Precursor production



Contribution to the precursor and delayed neutron populations can be tallied in each interaction.

$$w_0 = w_{\mathrm{in}} \frac{\Sigma_f}{\Sigma_{\mathrm{tot}}} \nu \beta_g$$

$$w_{\mathrm{tal}} = w_0 \, e^{-\lambda_g (t_1 - t_*)}$$

$$w_{\mathrm{emit}} = w_0 - w_{\mathrm{tal}} = w_0 (1 - e^{-\lambda_g (t_1 - t_*)})$$

Transient simulations

Delayed neutron emission

When the delayed neutrons are emitted, several variables have to be sampled for them:

Emission time t is sampled based on the decay law between t_* and t_1 from

$$t_{\mathrm{emit}} = t_* - \frac{1}{\lambda} \log \left[1 - \xi (1 - e^{-\lambda (t_1 - t_*)}) \right] , \tag{3}$$

where ξ is a sample from the uniform random distribution over the interval [0, 1).

For delayed neutrons emitted from interactions (during the time interval), the emission position (x, y, z) is the position of the interaction. For delayed neutrons emitted from tallies (beginning of time interval), the emission position is based on the tally.

The emission direction (u, v, w) is sampled isotropically.

The emission energy E is sampled from the emission spectrum of the delayed neutron group.

Test cases

I: Infinite Homogeneous Reactor

The first test case was an infinite homogeneous mixture of water and 3 wt-% enriched UO_2 made critical by addition of Boron 10. Since the system is infinite and homogeneous, there is a known theoretical solution given by the point-kinetics equations.

The initial source distributions were generated for a critical system ($\rho = 0 \pm 0.5$pcm).

For the transient simulation, the boron content of the mixture was adjusted to yield various reactivity insertions: $-0.21\$$, $0\$$, $0.24\$$, $0.85\$$ and $1.5\$$.

The total simulation time was 3 ms divided into 100 sub-intervals.

The neutron population and the power level of the system were tallied into 100 bins of equal time-width.

The theoretical prediction was calculated by numerical integration of the point-kinetics equations starting from known initial neutron and precursor populations:

$$\frac{\partial n(t)}{\partial t} = \frac{\rho(t) - \beta}{\Lambda} n(t) + \sum_{g=1}^{n_g} \lambda_g C_g(t)$$

$$\frac{\partial C_g(t)}{\partial t} = \frac{\beta_g}{\Lambda} n(t) - \lambda_g C_g(t)$$

Test cases

I: Infinite Homogeneous Reactor

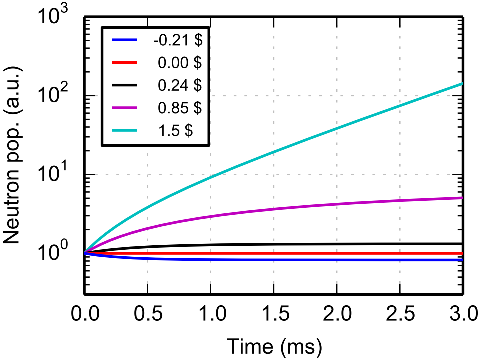


Figure 1 : Point-kinetics prediction for the neutron population in the different transient scenarios.

Test cases

I: Infinite Homogeneous Reactor

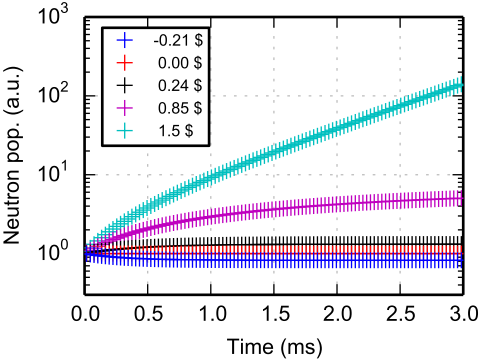


Figure 2 : Point-kinetics prediction for the neutron population in the different transient scenarios. Integrated to 100 bins of equal time width.

Test cases

I: Infinite Homogeneous Reactor

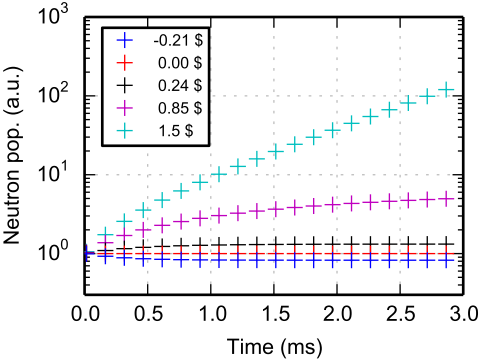


Figure 3 : Point-kinetics prediction for the neutron population in the different transient scenarios. Integrated to 100 bins of equal time width. Every 5th bin shown.

Test cases

I: Infinite Homogeneous Reactor

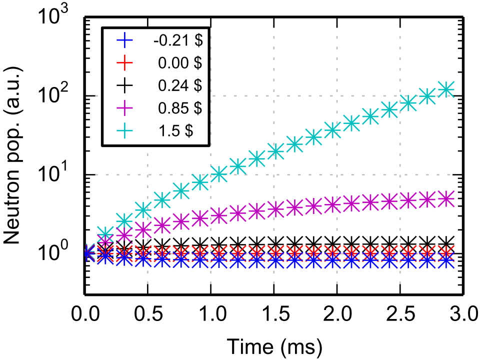


Figure 4 : Point-kinetics prediction (x) and the calculated (+) neutron population in the different transient scenarios. Integrated to 100 bins of equal time width. Every 5th bin shown.

Test cases

II: 3D LWR assembly

To test the methodology in a more realistic geometry, a 3D LWR assembly geometry (Peach Bottom 2 BWR) was chosen.

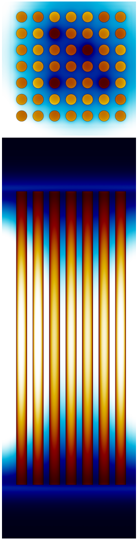
The system was radially infinite with reflective boundary conditions and axially finite with a total axial length of 500 cm. The active length of the system was 365.76 cm.

The initial source distributions were again generated in a system made critical by soluble $^{10}_{5}\mathrm{B}$ in the coolant/moderator. The power level of the system was set to 4 MW.

Only the critical system was modeled. The expected result is that the neutron population (and system power) stays constant in time.

Point-wise precursor tracking was used.

Simulation time was limited to 48 wall-clock hours on a 20 core Intel Xeon E5-2690 v2 @ 3.00GHz.



Test cases

II: 3D LWR assembly

100 microsecond simulation time:

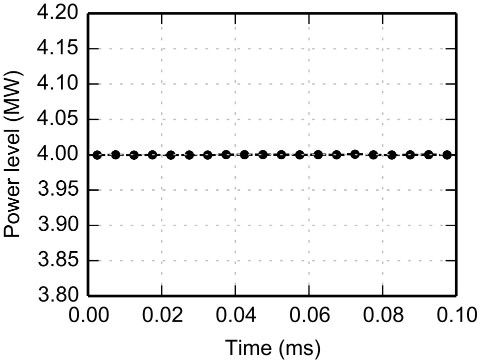


Figure 5 : Tallied average power level (tallied energy)/(bin time width). 2 sigma errorbars

Test cases

II: 3D LWR assembly

1 millisecond simulation time:

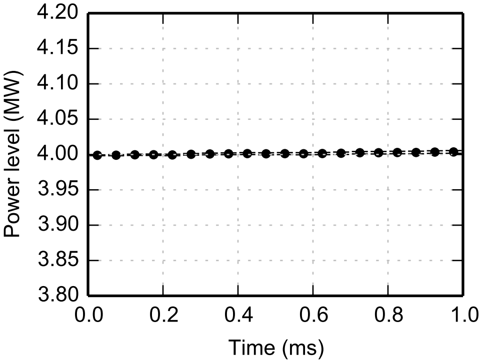


Figure 6 : Tallied average power level (tallied energy)/(bin time width). 2 sigma errorbars

Test cases

II: 3D LWR assembly

10 millisecond simulation time:

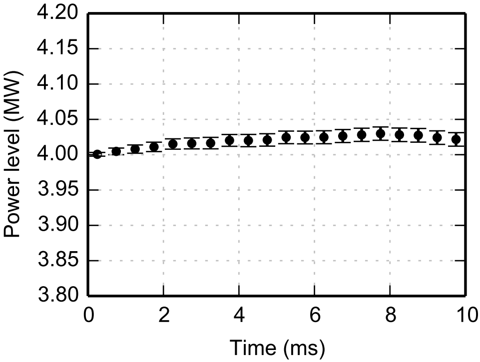


Figure 7 : Tallied average power level (tallied energy)/(bin time width). 2 sigma errorbars

Test cases

II: 3D LWR assembly

100 millisecond simulation time:

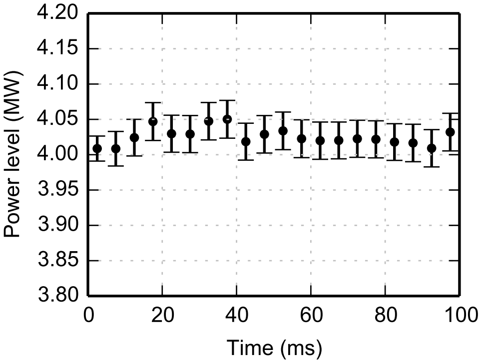


Figure 8 : Tallied average power level (tallied energy)/(bin time width). 2 sigma errorbars

Test cases

II: 3D LWR assembly

1 second simulation time:

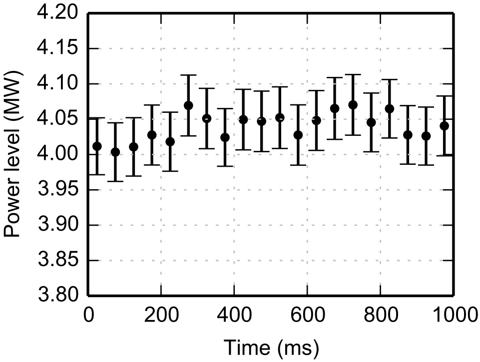


Figure 9 : Tallied average power level (tallied energy)/(bin time width). 2 sigma errorbars

Validation of the methodology

The two test cases provide initial verification for the implementation of the transient simulation routines.

Further validation is needed to ensure the performance of the implemented models. The validation will be conducted in two stages:

- No thermal-hydraulic feedback.
- With thermal-hydraulic feedback (coupled multi-physics calculations).

For the first part, rod drop (insertion) experiments with small test reactors should provide a good benchmark. For the second part we can look at experiments such as TRIGA pulse experiments, the SPERT III E REA experiments (LWR) or TRACY and SILENE (uranyl nitrate). Any suggestions for transient benchmarks/experiments, where delayed neutron emission is an important factor are welcomed.

Current capabilities and limitations

Any reactivity insertions can be modeled (sub, delayed super and prompt super critical). There is no inherent limit to the simulation time.

The initial source distributions are currently generated in a critical system in steady state.

End-of-simulation source can be saved to a file for further simulations. Allows "time-dependent" perturbations.

The methodology can be used with any delayed neutron group structure (read from ACE-libraries). However, the number of delayed neutron groups must be equal for all nuclides.

Solving time-dependent neutronics with Monte Carlo is slow. Even more so for coupled calculations. Such is the price one has to pay for accuracy.

Effect of variance reduction should be studied (implicit reaction modes can reduce branching of neutron histories). Branchless collision method by Sjenitzer should be tested.

Summary

Serpent 2 can model transient scenarios starting from a critical system.

This has been achieved by the implementation of a new delayed neutron emission model based on tracking the delayed neutron precursors. New source generation routines have also been implemented.

The methodology gives good results in the simple test cases tested thus far. Further validation is required.

The work on the transient capabilities in Serpent 2 is still very much in progress and comprehensive testing is still required.