Monte Carlo methods and challenges for the neutron transport equation

Alexander M. G. Cox

Bath, September 2021

Based on joint work with E. Horton, D. Villemonais, S. Harris, A. Kyprianou, T. Davis, M. Wang.

Funded by EPSRC EP/P009220/1

University of Bath

Outline

Aim: To give an overview of common methods for solving the NTE numerically using Monte Carlo methods.

Two key application areas:

- Criticality: what is ‘growth rate’ (= eigenvalue) of a system (typically, a reactor). What does the system look like in criticality. Important for reactor design, operation.
    - Compute e.g. rates of burn-up of fissile material.
    - What are typical temperatures, etc in the reactor?
    - What by-products will appear in the reactor, e.g. Xenon poisoning
- Shielding Problem: Given a source of radiation in a system, how much radioactivity will reach a given ‘detector’. E.g. in nuclear waste management, for canister of fuel waste: will someone standing near the canister receive a dangerous dose. Also relevant in medical, space, etc applications.
    - Typically interested in (very) rare-events.
    - What are ‘typical’ paths for events reaching the detector?
    - Source of particles typically important (unlike criticality).

Monte Carlo Basics

Fundamentally, (see Emma's talks!) we want to compute averages over particles following Neutron Branching Process, i.e. which undergo:

- Transport: particles move in a straight line between collisions.
- Scattering: particles collide with background objects (fuel rod, shielding, control rods, moderator, etc.) and change velocity (direction and energy/speed).
- Absorption: particles may be absorbed by environment and leave system.
- Fission: particles collide with fissile material: multiple new particles are emitted with new velocities.

In addition, particles can also exit through the boundary of the system, after which we assume they will not reenter (hard killing).

A Toy reactor

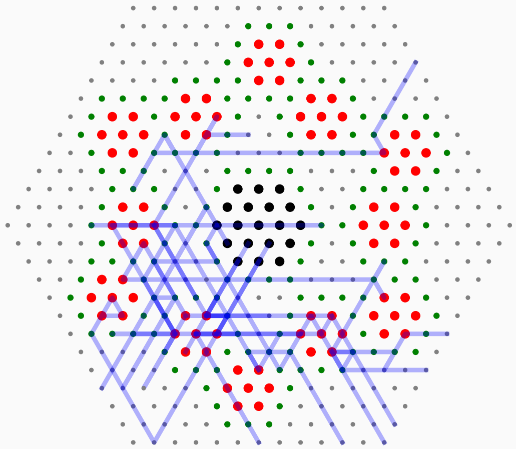

Figure 1: A simple (Branching Markov Chain) model of a reactor.

Monte Carlo Basics

Monte Carlo is seen as the gold standard vs deterministic methods because:

- Underlying geometry can be very complex.
- Many physical quantities are highly sensitive to energy (cross sections, angular direction).
- Particle tracks can be of very different lengths depending on position, direction $\implies$ Diffusive approximation is not appropriate.

As a result, MC estimates have benefit of being unbiased vs deterministic methods, which necessarily must make assumptions about discretising the system.

A Cross Section

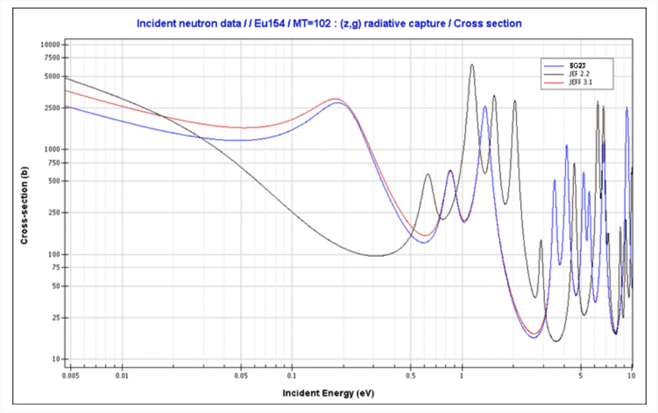

Figure 2: Cross Section of an element as a function of incident neutron energy. Source: JEFF Report 22.

Numerical Implementations

A range of open-source and commercial software packages exist. Most based on the same core algorithms (I will explain some of these!), but with a range of different strengths/weaknesses.

- For practical purposes, need to be able to work with ‘real’ physics. Large (and complex) libraries of real physical data, e.g. reaction rates, cross-sections, decay rates, etc. based on experimental data.
- Underlying geometry needs to be specified! Reactors can be complex.
- May also need to interact with other physical processes (thermal dynamics, fluids, etc.)

Some important implementations: TRIPOLI-4, MCNP, MONK, SCONE, . . .

Reactor Geometry

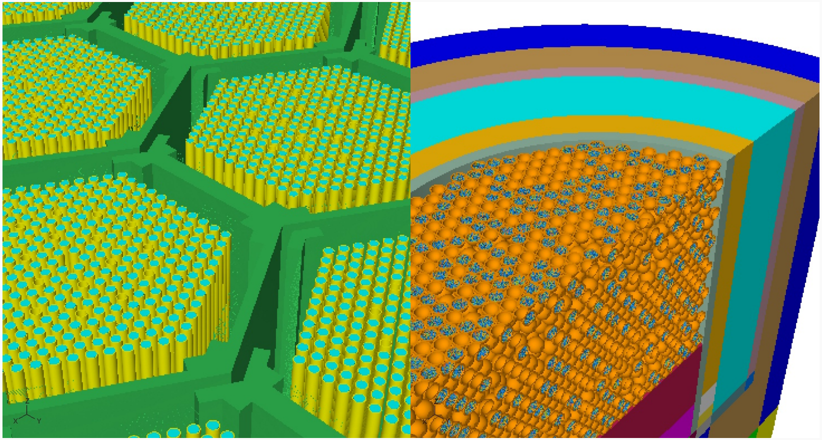

Figure 3: Example Reactor Geometry. Source: Jacobs/ANSWERS

Neutron Flux & Criticality

Recall the neutron flux is the length travelled by neutrons per given volume, per second. In criticality problems, the fundamental problem is to compute the stationary flux Ψ and k_{eff}, which are the solutions to the equation:

$$0 = \left(\overrightarrow{\mathtt{T}} + \overrightarrow{\mathtt{S}}\right)\Psi + \frac{1}{k_{\text{eff}}}\overrightarrow{\mathtt{F}}\,\Psi.$$

Here:

$\overrightarrow{\mathtt{T}}\,\Psi$ is the change in flux due to transport

$\overrightarrow{\mathtt{S}}\,\Psi$ is the change in flux due to scatter

$\overrightarrow{\mathtt{F}}\,\Psi$ is the change in flux due to fission

Idea: How much do I need to reduce/increase fission offspring to get critical system? If e.g. $k_{\text{eff}} = 2$, need to ‘kill’ half of fission offspring to make the system critical.

Neutron Generational Process

Recall Emma's talk. We define the Neutron Generational Process (NGP) as the postion of (weighted) particles after n fission events. I.e. if $\mathcal{X}_n = \sum_{i=1}^{\mathcal{N}_n} w_i^{(n)} \delta_{(r_i^{(n)}, v_i^{(n)})}$ is the position of $\mathcal{N}_n$ particles after n fission events, and $m_0 = \delta_{(r,v)}$ is the initial particle configuration then we can define:

$$\psi_n[g](r, v) = \mathbb{E}_{m_0}[\langle g, \mathcal{X}_n \rangle]$$

and ψ_n satisfies:

$$\psi_n[g](r, v) = \int_0^\infty \mathtt{Q}_s \left[\overleftarrow{\mathtt{F}} \psi_{n-1}[g] \right] (r, v) \, \mathrm{d}s$$

where $\mathtt{Q}$ is the expectation semigroup corresponding to just transport/scatter ($\overleftarrow{\mathtt{T}} + \overleftarrow{\mathtt{S}}$).

Estimating k_{eff} using 'naïve' methods

Theorem (C.-Horton-Kyprianou-Villemonais)

Under relatively mild technical conditions, there exists a positive right eigenfunction φ, a positive left eigenmeasure η and $k_{\text{eff}} \in \mathbb{R}_+$ such that for all g,

$$\langle \eta, \psi_n[g] \rangle = k_{\text{eff}} \langle \eta, g \rangle, \quad \textit{resp. } \psi_n[\varphi] = k_{\text{eff}}^n \varphi.$$

In addition, for large n, we get

$$\psi_n[g](r, v) \sim k_{\text{eff}}^n \langle \eta, g \rangle \varphi(r, v).$$

'Naïve' MC algorithm:

- Start with a collection of particles at sites (r, v), run until M fission generations. Average population size grows like k_{eff}, i.e. $k_{\text{eff}} \approx M^{-1} \log(\#\text{Particles alive at time } t)$

But... Computationally challenging. For large M, $k_{\text{eff}} \neq 1$, population size is either exponentially large, or exponentially small!

Understanding the semi-group

Can look to understand what is happening at the level of the semi-group.

- Start with particles with law η_0. Define a sequence of probability measures of the form:

$$\eta_m[g] := \frac{\langle \eta_{m-1}, \psi_1[g] \rangle}{\langle \eta_{m-1}, \psi_1[\mathbf{1}] \rangle}$$

  I.e. start particles with law η_{m-1} and run the process until after the next fission event. Look at the normalised density.
- By the theorem, η_m converges to the left-eigenmeasure (η — suitably normalised), and $k_{\text{eff}} \approx \langle \eta_{m-1}, \psi_1[\mathbf{1}] \rangle$.

The operation which maps probability measures to probability measures,

$$\mu \mapsto \Phi(\mu), \quad \text{where } \Phi(\mu)[g] := \frac{\langle \mu, \psi_1[g] \rangle}{\langle \mu, \psi_1[\mathbf{1}] \rangle}$$

is known as the Boltzmann-Gibbs transformation. Formally, we can solve our eigenvalue problem by $k_{\text{eff}} = \langle \eta, \psi_1[\mathbf{1}] \rangle$ where $\eta = \Phi(\eta)$, or $\eta = \lim_{n\to\infty} \Phi^n(\eta)$.

(Simplified) Power Iteration Algorithm

Aim: Compute k_{eff}.

Fix N, number of particles in each ‘batch’, M the number of generations.

1. Start with a collection of N particles approximating some distribution: $\eta_0^N = \sum_{i=1}^N \delta_{(r_i^{(0)}, v_i^{(0)})}$. Fix k_0, an estimate of k_{eff}. Set $m = 0$.
2. Simulate each particle from η_m^N independently until it leaves the system, or a fission event happens.
3. Where a fission event should happen with (on average) ν offspring, instead generate ν / k_m offspring (on average).
4. Let N_m be the number of particles after fission.
5. Normalise the particle population to N particles by copying/deleting randomly selected particles.
6. Set $k_{m+1} := k_m \times N/N_m$. $m \mapsto m + 1$.
7. If $m = M$, STOP, else GOTO 2.

Return: Estimate $k_{\text{eff}} \approx \frac{1}{\alpha M} \sum_{m=(1-\alpha)M}^{M} k_m$, α corresponds to burn-in period.

k_{eff} Computations

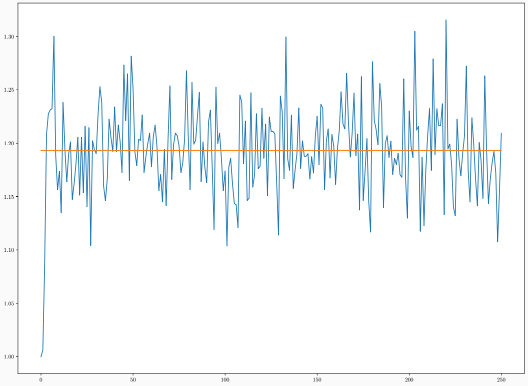

Figure 4: Estimates of k_{eff} in the Toy Reactor. Line denotes true value.

Comments on Power Iteration

1. Algorithm is a type of Interacting Particle System (IPS). The normalisation step ($\sim$ Boltzmann-Gibbs) induces interaction between particles.
2. In most examples, convergence for k_{eff} is fast, although easy to construct pathological examples.
3. Some analysis of convergence properties in the Nuclear Engineering literature (e.g. Brissenden and Garlick (1986), Ueki (2002)), and lots of experimental validation.
4. Particles often 'stored' when not selected. Use a 'particle bank' to supply extra particles if we end up with too few.

Flux Calculations

In addition to the value of k_{eff}, often want to also compute stationary flux Ψ.

Can do this using tallies: count up e.g. number of collisions happening in a given space-direction-energy region. Tallies give an indication of e.g. burn-up rates in fuel rods. Need to model fuel burn-up over time to understand how reactor behaviour may change.

Difficulty: most particles spend time in 'hot' region of the reactor. Can be hard to get accurate estimates of flux in the 'cold' regions.

One solution: variance reduction.

Variance Reduction for Criticality

Main idea: introduce an importance function h (cf Minmin's talk). Bias particles towards areas of greater weight.

Mathematically: introduce a change of measure to bias the paths towards the desired area, but introduce a weight to ensure the 'expectation' remains unbiased.

'Optimal' choice of weighting function (Zero-variance scheme, e.g. Christoforou & Hoogenboon, 2011) corresponds to the adjoint function, given as φ above.

But: φ is hard to compute. Two approaches:

- Deterministic solver of a (discrete) approximation.
- Iterated Fission Probability (Nauchi and Kameyama, 2011)

Iterated Fission Probability

Recall we wanted to estimate φ, where φ is given by

$$\psi_n[\varphi] = k_{\text{eff}}^n \varphi$$

Nauchi & Kameyama (2011) suggest defining the Iterated Fission Probability:

$$I_{FP}(r, v) = k^0_{(r,v)} k^1_{(r,v)} \cdots k^L_{(r,v)}$$

where $k^i_{(r,v)}$ is the estimated multiplication factor in the power iteration routing for particles started at (r, v). For L large, all estimates will be k_{eff}.

Key point: $I_{FP}(r, v) \propto \varphi(r, v)$ (approximately), but I_{FP} can be estimated by Monte Carlo!

Population Control, Russian Roulette and Splitting

General picture: simulate a particle process with weights. Have several ways of managing process to control the particle numbers:

- Population Control: At end of batches (fission events), ‘normalise’ the population to keep a fixed number of particles at start of each generation.
- Russian Roulette: If the weight of a particle gets too small, kill it with given probability. Otherwise increase the weight.
- Splitting: If the weight of a particle gets too high, split the particle into independent copies.
- Implicit Capture: Instead of allowing a particle to be absorbed, reduce weight according to capture probability.
- Weight Windows: Fix levels to determine when to do roulette/splitting.

Population Control, Russian Roulette and Splitting

All modify the branching/killing behaviour of the process while maintaining a ‘fair-game’ or unbiasedness of the particles.

Fundamental trade-off: too much branching ($=$ splitting) increases correlation of particles. Too little branching increases variance (higher weights). Too much killing ($=$ roulette) increases correlation of particles, too little gives too many small particles.

Big Picture: Variance reduction

- Lots of ways of constructing stochastic representations of the underlying physical process ψ_n:
  - Theoretical: *h*-transform methodology/many-to-one, etc.
  - Practical: Iterated fission probability, population control, Russian Roulette, etc.
- Essentially, all are different probabilistic representations of the same Interacting Particle System. (c.f. Pierre's talk)
  - Lots of practical experience, but limited theoretical understanding of how to choose the 'best' combination.
  - In practice, lots of mystery constants, fixed based on experience.
- Large literature on Interacting Particle Systems, e.g also Sequential Monte Carlo (SMC). Can we use this theory to better understand these algorithms in practice?
  - Better ways of parallelising interaction?
  - Spine biasing? (C.f. Athreya (2000), Whitely & Lee (2014)).
  - Better understanding of genealogical structure? Can this profide e.g. clustering heuristics?

Advanced Monte Carlo Methods

Other advanced acceleration techniques are implemented in practice. E.g.:

- Wielandt acceleration: introduce an overestimate of k_{eff}, k^*, say. Solve a modified transport equation with new branching rate:

$$(\overrightarrow{\mathtt{T}} + \overrightarrow{\mathtt{S}} - \frac{1}{k^*}\overrightarrow{\mathtt{F}})\Psi_{n+1} = \left(\frac{1}{k_n} - \frac{1}{k^*}\right)\overrightarrow{\mathtt{F}}\Psi_n$$

  Idea: Effectively add in generations between ‘normal’ fission events. Adjust estimate of k_{eff} appropriately. (Yamamoto & Miyoshi, (2004)).
    - Gives faster convergence to equilibrium!
- Superhistory Powering (Brissenden and Garlick (1986)). Idea: Run each ‘stage’ for L generations of neutron transport with a given choice of k_n estimating the number of branching offspring.
    - Reduces correlation between generational estimates of k_{eff} and flux. May improve accuracy of standard deviation estimates.
    - Not clear how to choose L.

Other Challenges in Criticality

- Clustering:
  - See Eric's talk! Particles in simulation tend to cluster due to branching effects (mostly aphysical). Are there good ways of adapting the simulation to account for this?
- Flux Convergence:
  - In general, k_{eff} converges faster than the flux. If we want to measure flux, need to know if $\psi_n \approx \psi_\infty$.
  - Current practice: Divide the space into regions, compute number of particles in each bin. Compute Shannon Entropy,

    $$H_{src} = -\sum p_i \log(p_i),$$

    where p_i is the proportion of particles in box i. When H seems to stabilise, source is in equilibrium!
- Simulation Parameters
  - How to determine key parameters of simulation: e.g. burn-in time vs total no of stages vs no particles per stage.

Source Convergence vs k_{eff} convergence

Figure 3. Convergence plots of k_{eff} and H_{src} vs. batch for OECD Benchmark 1

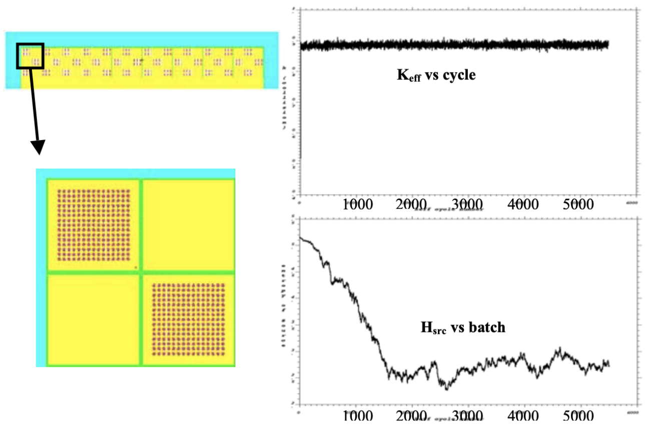

Figure 5: k_{eff} computation vs Shannon Entropy. From: Brown, F.B., PHYSOR, (2006)

Shielding Calculations

Second main class of problem is fixed source calculations, or shielding calculations. Generally relevant in subcritical cases, where there is a source of neutrons (e.g. nuclear waste), and we want to work out how much radiation/what type will reach an external point.

Application examples:

- Waste management
- Space radiation: how much shielding is needed on a shuttle/space station?
- Health applications: how well shielded is an operator of a proton beam.
- Fusion reactors: how much shielding is necessary?
- etc.

Shielding basics

- Simple problem: compute flux at a detector. Equivalent to probability of a path starting at the source reaching the detector. Mathematically, want to solve flux equation of the form:

$$0 = (\overrightarrow{\mathrm{T}} + \overrightarrow{\mathrm{S}})\Psi + Q$$

  where Q is a source term representing particles coming into the system. If e.g. $R = \int_D \Psi(r, v) g(r, v) \, \mathrm{d}r \, \mathrm{d}v$ is the response at the detector, we want to estimate R using Monte Carlo.
- Challenge 1: Sometimes desired paths are very rare. May need $\sim 10^{\text{lots}}$ of simulations. Numerically too intensive!
- Challenge 2: In some applications, paths may not be that rare, but e.g. may want to know 'how' paths reach the detector, i.e. flux estimates of particles reaching detector that pass through a given location, or sensitivity to e.g. material composition.

One solution: Use biasing, or *h*-transform to compute a new law of the process. Use this to direct particles to the detector.

Biasing the NRW

Let (R_t, V_t) be the NRW corresponding to the generator $(\overrightarrow{\mathtt{T}} + \overrightarrow{\mathtt{S}})$ with source Q. Fix a positive function h (Doob's h function/importance function) which is identically 1 on the detector. Then the process

$$Z_t := \frac{h(R_t, V_t)}{h(r, v)} \exp\left\{ - \int_0^t \frac{(\overleftarrow{\mathtt{T}} + \overleftarrow{\mathtt{S}})h}{h}(R_s, V_s)\,\mathrm{d}s \right\}$$

is a positive martingale with $Z_0 = 1$.

If we suppose a point source and detector, let τ_D be the first hitting time of the detector, and τ_∂ the absorption time:

$$\begin{aligned}
&\mathbb{P}\left(\tau_D < \tau_\partial | (R_0, V_0) = (r, v)\right) \\
&\quad = h(r, v)\mathbb{E}\left[Z_{\tau_D} \exp\left\{ \int_0^{\tau_D} \frac{(\overleftarrow{\mathtt{T}} + \overleftarrow{\mathtt{S}})h}{h}(R_s, V_s)\,\mathrm{d}s \right\} \mathbf{1}_{\tau_D < \tau_\partial} | (R_0, V_0) = (r, v) \right] \\
&\quad = h(r, v)\mathbb{E}^h\left[\exp\left\{ \int_0^{\tau_D} \frac{(\overleftarrow{\mathtt{T}} + \overleftarrow{\mathtt{S}})h}{h}(R_s, V_s)\,\mathrm{d}s \right\} \mathbf{1}_{\tau_D < \tau_\partial} | (R_0, V_0) = (r, v) \right]
\end{aligned}$$

Now $\mathbb{P}^h$ is the new probability given by the change of measure Z.

Adjoint flux

Now we need to compute

$$\mathbb{E}^h\left[\exp\left\{\int_0^{\tau_D} \frac{(\overleftarrow{\mathtt{T}} + \overleftarrow{\mathtt{S}})h}{h}(R_s, V_s)\,\mathrm{d}s\right\}\mathbf{1}_{\tau_D<\tau_\partial}|(R_0, V_0) = (r, v)\right]$$

using the new probability measure. Still a hard computation unless we can find h such that:

$$(\overleftarrow{\mathtt{T}} + \overleftarrow{\mathtt{S}})h = 0$$

In which case h is called the adjoint flux.

(It's exactly the function φ we constructed above, in the case where $\overleftarrow{\mathtt{F}} \equiv 0$).

NB: Can compute dynamics of process under $\mathbb{P}^h$: essentially, introduce altered flight length, scattering distribution depending on the function h.

Adjoint flux approximation I

Can't generally compute adjoint flux explicitly, but several alternatives exist: CADIS methodology (Haghighat & Wagner, 2003) is deterministic solver. E.g.

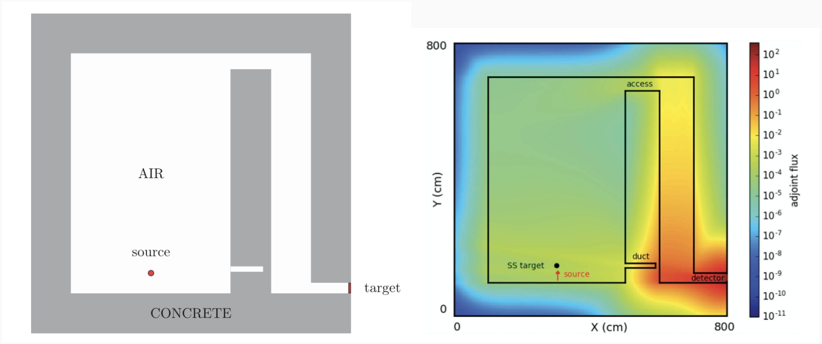

Figure 6: Importance map for a Bunker created by IDT. From Nowak et al. (2019).

Adjoint flux approximation II

An alternative approach is a Response Matrix approach (Leppännen, 2019). Discretises space and uses Monte Carlo to approximate the 'Markov Chain' on the discretised space. Computes adjoint weights for this chain.

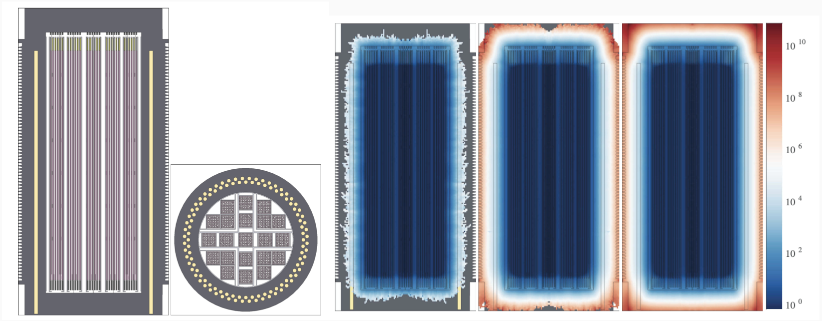

Figure 7: Response Matrix computation of adjoint flux for a storage cask. From Leppännen (2019).

Shielding: Adaptive Multilevel Splitting

See Tony's talk!

Beyond Criticality and Shielding!

Another significant challenge for MC comes in Dynamic Monte Carlo. E.g. Sjenitzer & Hoogenboom (2011), Faucher et al. (2018). Here we want to simulate the behaviour of a reactor over the timescale of a real event, e.g. inserting/removing a control rod, or reactor start up.

- Average generation time in a reactor $\sim 10^{-5}$s, while neutrons can also create a precursor during a fission event. Precursors decay over timescales $\sim 10^{1}$s.
- In normal criticality calculations, delayed neutrons are not a problem — by stationarity, can assume that the decay happens immediately.
- Storing all precursor neutrons during MC is not feasible: for each active neutron, $\sim 10^{4}$ precursors.

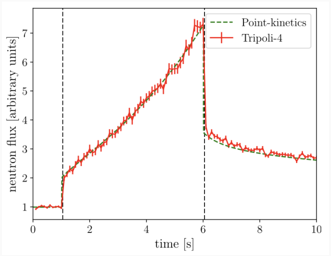

Figure 8: Flux in a reactor during removal/insertion of a control rod. From Faucher et al. (2018).

Summary

- Monte Carlo methods are pervasive in neutron transport modelling.
- Lots of challenging problems and interesting methods. Number of existing software implementations.
- Most successful algorithms rely on Interacting Particle Models, along with a suite of variance reduction and other tricks…
- …but there seems to be relatively little understood about the theoretical properties and advances in the Interacting particles literature.
- Lots of interesting challenges for mathematicians from a range of different communities!

Useful References

Some useful introductions to Monte Carlo for the NTE:

- Forrest Brown: Monte Carlo Techniques for Nuclear Systems - Theory Lectures. *Lecture slides for a course at UNM.* (2016)
- Lux and Koblinger: Monte Carlo Particle Transport Methods: Neutron and Photon Calculations. *Comprehensive book — standard reference*. (1991)