Monte Carlo Techniques for Nuclear Systems –

Theory Lectures

Forrest Brown

National Laboratory Professor, NE Dept., UNM
Senior Research Scientist, XCP-3, LANL
# Lecture Topics

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>0. Course Information</td>
<td>3</td>
</tr>
<tr>
<td>1. Nuclear Eng. Review &amp; MC</td>
<td>10</td>
</tr>
<tr>
<td>2. Random Numbers &amp; Sampling</td>
<td>49</td>
</tr>
<tr>
<td>3. Computational Geometry</td>
<td>105</td>
</tr>
<tr>
<td>4. Collision Physics</td>
<td>154</td>
</tr>
<tr>
<td>5. Tallies &amp; Statistics</td>
<td>175</td>
</tr>
<tr>
<td>6. Eigenvalue Calculations – I</td>
<td>201</td>
</tr>
<tr>
<td>7. Eigenvalue Calculations – II</td>
<td>238</td>
</tr>
<tr>
<td>8. Eigenvalue Calculations – III</td>
<td>278</td>
</tr>
<tr>
<td>9. Variance Reduction</td>
<td>313</td>
</tr>
<tr>
<td>10. Parallel Monte Carlo</td>
<td>336</td>
</tr>
<tr>
<td>11. Parameter Studies</td>
<td>392</td>
</tr>
<tr>
<td>12. Fission Matrix &amp; Higher Eigenmodes</td>
<td>432</td>
</tr>
<tr>
<td>13. Doppler Broadening</td>
<td>473</td>
</tr>
<tr>
<td>14. Monte Carlo Depletion</td>
<td>512</td>
</tr>
<tr>
<td>15. HTGR Modeling</td>
<td>528</td>
</tr>
<tr>
<td>16. Coupled MC &amp; T/H Calculations</td>
<td>560</td>
</tr>
<tr>
<td>17. Fission Energy Deposition</td>
<td>587</td>
</tr>
</tbody>
</table>

Principal lectures

Supplemental
Monte Carlo Techniques for Nuclear Systems

Forrest Brown

National Laboratory Professor, Nuclear Engineering, UNM
Senior Research Scientist, Monte Carlo Codes, LANL
fbrown@unm.edu, fbrown@lanl.gov

@ Copyright 2016, F.B. Brown, UNM, LANL. For educational use only
Course Information

NE 462 - Monte Carlo Techniques for Nuclear Systems - Description

Theory: Solving particle transport problems with the Monte Carlo method is simple - just simulate the particle behavior. The devil is in the details, however. These lectures provide a balanced approach to the theory and practice of Monte Carlo simulation codes. The first lectures provide an overview of Monte Carlo simulation methods, covering the transport equation, random sampling, computational geometry, collision physics, and statistics. The next lectures focus on the state-of-the-art in Monte Carlo criticality simulations, covering the theory of eigenvalue calculations, convergence analysis, dominance ratio calculations, bias in Keff and tallies, bias in uncertainties, a case study of a realistic calculation, and Wielandt acceleration techniques. The remaining lectures cover advanced topics, including HTGR modeling and stochastic geometry, temperature dependence, fission energy deposition, depletion calculations, parallel calculations, and parameter studies.

Practice: This portion of the class focuses on using MCNP to perform criticality calculations for reactor physics and criticality safety applications. It is an intermediate level class, intended for those with at least some familiarity with MCNP. Class examples provide hands-on experience at running the code, plotting both geometry and results, and understanding the code output. The class includes lectures & hands-on computer use for a variety of Monte Carlo calculations. Beginning MCNP users are encouraged to review LA-UR-09-00380, "Criticality Calculations with MCNP: A Primer (3rd Edition)" (available at http://mcnp.lanl.gov under "Reference Collection") prior to the class.

Computing: No Monte Carlo class can be complete without having students write their own simple Monte Carlo routines for basic random sampling, use of the random number generator, and simplified particle transport simulation.
Course Information (2)

NE 462 - Monte Carlo Techniques for Nuclear Systems - Description

• Three Credit Hours; One 170-minute lecture per week.

• Textbook: None
  Supplemental materials: a DVD provided to all students, containing:
  Lecture notes for 17 lectures on the theory of Monte Carlo methods
  Lecture notes for 19 lectures on the practical use of Monte Carlo methods
  Lecture notes for 16 lectures on computing methods related to Monte Carlo
  186 example problems for practical use of MCNP
  737 technical reports on Monte Carlo methods, including tutorials, a complete introductory book, technical workshops, and references

  Supplemental materials: MCNP6 Monte Carlo code package. Students are required to obtain this (at no cost) from the US DOE computer code center, RSICC. The code is used for the practical part of the course in learning to apply Monte Carlo methods to nuclear engineering problems.

• Specific Course Information
  Catalog description: Monte Carlo methods for nuclear criticality and reactor analysis and radiation shielding calculation using production Monte Carlo codes, understand basics of probability and statistics and of particle transport in the context of Monte Carlo methods.
  Corequisite: NE 410.
  Restriction: admitted to School of Engineering.
  Required course in the program
Course Information (3)

NE 462 - Monte Carlo Techniques for Nuclear Systems - Specific Goals

• Students should develop a basic understanding of Monte Carlo simulation techniques and be able to explain and discuss the following topics:
  – Random number generation and random sampling methods used in the Monte Carlo simulation of radiation transport
  – Basic statistical concepts including mean, variance, probability density function, cumulative distribution function
  – 3-dimensional computational geometry used to represent physical systems
  – Simulation of the basics physics interactions for particle collisions with the nuclides in problem materials
  – Obtaining engineering results from tallies performed during the Monte Carlo simulation

• Students should develop a basic understanding of how Monte Carlo methods are applied to realistic nuclear systems and be able to explain and discuss the following topics:
  – How Monte Carlo methods can be used to determine basic physical quantities including scalar flux, nuclear cross-sections, material reaction rates with radiation
  – How Monte Carlo methods can simulate basic aspects of nuclear systems, including the production of neutrons from fission, the slowing down process due to collisions with moderator material, and the capture and fission of neutrons in the thermal and resonance energy ranges
  – General principles for applying Monte Carlo methods to realistic models of nuclear reactor systems and criticality safety analyses

• Students should develop a basic understanding of how Monte Carlo methods are implemented into computer codes:
  – Demonstrate the ability to write simple Monte Carlo programs using a programming language they know (e.g., Matlab, C++, Fortran, etc.)
  – Demonstrate the ability to perform realistic Monte Carlo calculations using an industry-standard Monte Carlo code (e.g., MCNP6)
Course Information (4)

NE 462 - Monte Carlo Techniques for Nuclear Systems - Topics to be covered

- **Overview of Monte Carlo methods**

- **Theory:**
  - Basics - Nuclear Engineering & Monte Carlo
  - Random Numbers & Random Sampling
  - Computational Geometry
  - Collision Physics
  - Tallies & Statistics
  - Eigenvalue Calculations – Parts I, II
  - Variance Reduction

- **Practical:**
  - Introduction to MCNP
  - MCNP Basics
  - Criticality Calculations
  - Advanced Geometry
  - Sources
  - Tallies
  - Physics & Nuclear Data
  - Point Kinetics Parameters
  - Sensitivity-Uncertainty Analysis

- **General:**
  - Eigenvalue Calculations – Part III
  - Parallel Monte Carlo
  - Parameter Studies
  - Doppler Broadening
  - Monte Carlo Depletion
Preliminaries (1)

• **Monte Carlo Techniques for Nuclear Systems**
  – Introduction to Monte Carlo methods for particle transport simulation
  – Limited to neutrons & photons
  – Roughly 2/3 theory & 1/3 practical

• **Understand basic MC simulation techniques**
  – Random number generation, random sampling
  – 3D computational geometry
  – Basic physics simulation & collisions
  – Tallies

• **Application to nuclear systems**
  – Flux, cross-sections, reaction rates
  – Simple models
  – Reactor models
  – Criticality safety

• **Computing**
  – Simple DIY MC programs
  – Using MCNP for realistic models
Preliminaries (2)

• Useful references & websites
  
  – MCNP6 User Manual, part of MCNP6 distribution
  – MCNP5 Theory Manual (Volume I), on class DVD & MCNP website
  – Carter & Cashwell book, on class DVD & MCNP website
  
  – MCNP website - mcnp.lanl.gov
  – RSICC website - rsicc.ornl.gov
    For ordering MCNP, SCALE, …
  – MCNP Forum email-list - see MCNP website
  
  – National Nuclear Data Center, Brookhaven National Lab. - nndc.bnl.gov
    ENDF nuclear cross-sections & plotting
    Chart of the Nuclides
  – Wikipedia ??? Maybe for class, not for work or papers

• Note
  – In class or at work, you should always cite the sources for any physical data you use - cross-sections (ENDF/B-VI, -VII.0, VII.1, …), masses, isotopic abundances, etc.
Nuclear Engineering Review
&
Monte Carlo Basics
Outline

• Introduction

• Review of Nuclear Engineering basics
  – Flux
  – Cross-sections
  – Reaction rates

• Monte Carlo – intro
  – Linear Boltzmann Transport Equation
  – Monte Carlo Simulation
  – Criticality Calculations

• Review of statistics
  – Basic Random Sampling
  – Probability Density (PDF)
  – Cumulative Distribution (CDF)
  – Statistics
Introduction

• John Von Neumann invented scientific computing in the 1940s
  – Stored programs, "software"
  – Algorithms & flowcharts
  – Assisted with hardware design as well
  – "Ordinary" computers today are called "Von Neumann machines"

• Von Neumann invented Monte Carlo methods for particle transport in the 1940s (with Ulam, Fermi, Metropolis, & others at LANL)
  – Highly accurate - no essential approximations
  – Expensive - typically the "method of last resort"
  – First MC code for ENIAC in 1947
  – Monte Carlo codes for particle transport have been proven to work effectively on all types of computer architectures:

    SIMD, MIMD, vector, parallel, supercomputers, clusters, workstations, PCs, netbooks, tablets, CPUs, GPUs, MICs, ...
Introduction

- Two basic ways to approach the use of Monte Carlo methods for solving the transport equation:
  - Mathematical technique for numerical integration
  - Computer simulation of a physical process

→ Each is "correct"

- **Mathematical approach is useful for:**
  - Importance sampling, convergence, variance reduction,
  - random sampling techniques, eigenvalue calculation schemes, ...

- **Simulation approach is useful for:**
  - collision physics, tracking, tallying, ...

- Monte Carlo methods solve integral problems, so consider the integral form of the Boltzmann equation
Introduction

Simple Monte Carlo Example

Evaluate \( G = \int_0^1 g(x)dx \), with \( g(x) = \sqrt{1-x^2} \)

• **Mathematical approach:**
  For \( k = 1, \ldots, N \): choose \( \hat{x}_k \) randomly in \((0,1)\)

  \[
  G \approx \frac{1}{N} \cdot \sum_{k=1}^{N} g(\hat{x}_k) = \frac{1}{N} \cdot \sum_{k=1}^{N} \sqrt{1-\hat{x}_k^2}
  \]

• **Simulation approach, "darts game":**
  For \( k = 1, \ldots, N \): choose \( \hat{x}_k, \hat{y}_k \) randomly in \((0,1)\),
  if \( \hat{x}_k^2 + \hat{y}_k^2 \leq 1 \), tally a "hit"

  \[
  G \approx \frac{\text{number of hits}}{N}
  \]
Review of NE Basics

• Flux

\[ \Phi = n \cdot v \]

- \( n \) = particle density, \( \text{particles/cm}^3 \) \( \text{function of position} \)
- \( v \) = particle speed, \( \text{cm/sec} \)
- \( \Phi \) = scalar flux, \( \text{particles/cm}^2\text{-sec} \) \( \text{function of position & energy} \)  
  \( \text{but not direction} \)

- Most textbooks:
  Scalar flux is the total distance traveled by all particles, per \( \text{cm}^3 \), per sec

• Thought experiment:
  Suppose we could watch all particles as they fly around & have collisions

  • Keep track of all the portions of flight distances within a region \( G \), for all the particles, for a second

  • The total distance traveled divided by the volume & 1 second is the scalar flux in the region

⇒ This is how MC estimates the flux in a region. This is the basis for a “pathlength estimator”
Mean Free Path & Macroscopic Cross-section

- Mean Free Path, $\lambda$
  - $\lambda$ = average distance to collision

- Macroscopic Cross-section, $\Sigma$
  \[ \Sigma = \frac{1}{\lambda} = N \sigma \]
  - Probability of interaction with a material, per unit distance traveled
  - $N =$ nuclide density, atoms/barn-cm,
    \[ N = \frac{N_{av} \cdot m}{A \cdot V} = \frac{N_{av} \cdot \rho}{A} \]
  - $\sigma =$ microscopic cross-section, target area per nuclide
    units: bars, 1 barn = $10^{-24}$ cm$^2$
  - $\Sigma =$ [nuclides/barn-cm] * [barns/nuclide] = 1 / cm
  - For a mixture of nuclides, add $\Sigma$'s for each nuclide

\[ \Sigma_{UO2}^T = N_{U235}^T \sigma_{U235}^T + N_{U238}^T \sigma_{U238}^T + N_{O16}^T \sigma_{O16}^T + \ldots \]
Macroscopic Cross-sections

- **Material Cross-section**

  - Add $N\sigma$ for each nuclide, for a particular reaction $x$ ($x: \text{T,A,F,S,C,N,-2N, ...}$)

  \[
  \sum_{x}^{\text{mat}} = \sum_{i} N_i \cdot \sigma^i_x
  \]

  \[
  \sum_T^{\text{UO}_2} = \sum_T^{U^{235}} + \sum_T^{U^{238}} + \sum_T^{O^{16}} + \ldots
  \]

  \[
  = N^{U^{235}} \sigma_T^{U^{235}} + N^{U^{238}} \sigma_T^{U^{238}} + N^{O^{16}} \sigma_T^{O^{16}} + \ldots
  \]

  - Given that a collision occurs in a material, the probability that it will involve nuclide $J$ is:

  \[
  p^{\text{mat}}_J = \frac{\sum_T^\text{J}}{\sum_T^{\text{mat}}}
  \]

  $\Rightarrow$ This is how MC selects a particular nuclide, given that a collision occurs in a material
Macroscopic Cross-sections

- If a neutron collision occurs in UO$_2$ fuel, find the probability that the collision is with U$^{235}$, U$^{238}$, or O$^{16}$

$$\Sigma_{fuel}^{fuel} = N^{U^{235}} \sigma_T^{U^{235}} + N^{U^{238}} \sigma_T^{U^{238}} + N^{O^{16}} \sigma_T^{O^{16}} = \Sigma_{T}^{U^{235}} + \Sigma_{T}^{U^{238}} + \Sigma_{T}^{O^{16}}$$

Probability that collision is with

- U$^{235}$: \( P = \frac{\Sigma_{T}^{U^{235}}}{\Sigma_{fuel}} \),
- U$^{238}$: \( P = \frac{\Sigma_{T}^{U^{238}}}{\Sigma_{fuel}} \),
- O$^{16}$: \( P = \frac{\Sigma_{T}^{O^{16}}}{\Sigma_{fuel}} \)

- If a neutron collision occurs in UO$_2$ fuel, find the probability that the neutron is absorbed

$$\Sigma_{A}^{fuel} = N^{U^{235}} \sigma_A^{U^{235}} + N^{U^{238}} \sigma_A^{U^{238}} + N^{O^{16}} \sigma_A^{O^{16}} = \Sigma_{A}^{U^{235}} + \Sigma_{A}^{U^{238}} + \Sigma_{A}^{O^{16}}$$
$$\Sigma_{S}^{fuel} = N^{U^{235}} \sigma_S^{U^{235}} + N^{U^{238}} \sigma_S^{U^{238}} + N^{O^{16}} \sigma_S^{O^{16}} = \Sigma_{S}^{U^{235}} + \Sigma_{S}^{U^{238}} + \Sigma_{S}^{O^{16}}$$
$$\Sigma_{T}^{fuel} = \Sigma_{A}^{fuel} + \Sigma_{S}^{fuel}$$

Probability that neutron is

- Absorbed: \( P_A = \frac{\Sigma_{A}^{fuel}}{\Sigma_{T}^{fuel}} \),
- Scattered: \( P_S = \frac{\Sigma_{S}^{fuel}}{\Sigma_{T}^{fuel}} \),

\( P_A + P_S = 1 \)
Microscopic Cross-sections

- **Microscopic Cross-section, \( \sigma \)**
  - \( \sigma = \) Target area of single nuclide for an interaction
  - Units: barns, \( 1 \text{ barn} = 10^{-24} \text{ cm}^2 \)

- Examples: \( \sigma_{A^{235}}^{U}, \sigma_{S^1}^{H}, \sigma_{N-2N}^{U^{238}} \)

- Microscopic cross-section data:
  - Measured in experiments
  - Determined from theory (e.g., quantum mechanics)
  - Experiment + theory + judgement used by CSWEG to produce ENDF/B datasets

- **Absorption vs Capture**
  - For NE’s and this class:
    \[
    \sigma_A = \sigma_C + \sigma_F \\
    \sigma_T = \sigma_C + \sigma_F + \sigma_S = \sigma_A + \sigma_S
    \]
  - Absorption = fission + capture
  - Physicists often interchange “capture” & “absorption”
U^{235} Fission Cross-section, $\sigma_F^{U^{235}}$

- Thermal
- Resolved Resonances
- Unresolved Resonances

ENDF/B Data, Plotted by MCNP5

Experimental Data used by CSWEG
Nuclear Data - Where to Find It

• Use the MCNP built-in cross-section plotter
  – Generally, MCNP uses latest ENDF/B-VII.0 nuclear data
  – Can plot any isotope & reaction (if available) in a problem
  – Will be covered in Practical Lecture on Nuclear Data

• LANL T-2 Nuclear Information Service
  – t2.lanl.gov
  – Nuclear Data Viewer & ENDF data

• National Nuclear Data Center, Brookhaven National Lab.
  – nndc.bnl.gov - has ENDF data & plotting utilities
  – Isotopic abundances - AMDC
  – Chart of the Nuclides
Microscopic Cross-sections

- **Thought experiments:**
  - Given that a collision occurs with nuclide J, what is the probability of the incoming particle surviving the collision?
    - Note that \( \sigma_T^J = \sigma_A^J + \sigma_S^J \)
      - Absorption ⇒ particle disappears
      - Scatter ⇒ particle survives
    - Probability of surviving the collision = \( \frac{\sigma_S^J}{\sigma_T^J} \)
  
- Given that a collision occurs with nuclide J, what is the probability of fission?
  - Note that \( \sigma_T^J = \sigma_F^J + \sigma_C^J + \sigma_S^J \)
  - Probability of fission = \( \frac{\sigma_F^J}{\sigma_T^J} \)

⇒ In MC codes, “given that a collision occurs with nuclide J”, probabilities based on ratios of partial \( \sigma \)'s for nuclide J are used in sampling what reaction takes place with that nuclide
**Beam Attenuation**

- **For a beam of particles**
  - $\Psi(r, E, \Omega) = \text{angular flux at } r, \text{ direction } \Omega$
    - Directional quantity
    - Integrate over all $\Omega$ to get scalar flux $\Phi$
  - If a beam of particles is directed at a purely-absorbing infinite slab, what is the beam strength at penetration $x$?
    - $\Psi(x) = \Psi_0 \cdot e^{-\Sigma_T x}$
  - Probability of traveling distance $x$ **without** collision is $e^{-\Sigma_T x}$
  - Probability of **colliding** at distance $x$ is
    - $f(x) = \Sigma_T \cdot e^{-\Sigma_T x}$

$\Rightarrow$ MC codes use this relation to sample the distance to the next collision
**Reaction Rates**

- Most of the time, nuclear engineers don’t care about flux. The important & useful quantities desired include:
  - absorption in a region,
  - fission in a region,
  - heating in a region,
  - absorption in $\text{U}^{235}$ $\text{U}^{238}$ $\text{Pu}^{239}$ $\text{B}^{10}$ $\text{Hf}$ $\text{Xe}^{135}$ $\text{Zr}$ ..... etc.

- **Reaction rates**
  - Collision rate $\Sigma_T \Phi$
  - Fission rate $\Sigma_F \Phi$
  - Neutron production rate $\nu \Sigma_F \Phi$
  - Absorption rate in $\text{U}^{235}$ $\Sigma_A \text{U}^{235} \Phi$

- Reaction rate = [reactions / cm] * [total cm traveled / cm$^3$sec]
  = reactions / cm$^3$sec
- To get total reactions/sec for a region, integrate over the region volume

- For reaction rates in a material, use $\Sigma$ for the material
- For reaction rates for a nuclide, use $\Sigma$ for the nuclide (in the material)
Another Flux Estimator

- **Thought experiment:**
  - Suppose we could watch all particles as they fly around & have collisions
  - Keep track of all the collisions within a region $G$, for all the particles, for a second
  - $[\text{collision rate}] = \Sigma_T \Phi$
  - Solve for flux: $\Phi = [\text{collision rate}] / \Sigma_T$
  
  ⇒ MC uses this to estimate the flux in a region.
  This is the basis for a “collision estimator”

- We have seen that MC can estimate the flux in a region in several ways:
  - Pathlength estimator - $[\text{total distance traveled in region}] / (\text{Volume x time})$
  - Collision estimator - $[\text{total collisions in region}] / \Sigma_T / (\text{Volume x time})$

  - Which is correct? Both
  - Which is better? Depends on physics & geometry of problem
  - Are there other flux estimators? Yes (flux at point, flux on surface, …..)
  - Which one should be used? If available, all (statistically combined)
  - MCNP only provides pathlength flux estimators for a region.
    Some MC codes only provide collision estimators; some provide both.
Monte Carlo - Introduction

- **Goal:** Simulate nature, particles moving through physical objects

**Flight**
- Random sampling using $\Sigma_T$ & exponential PDF:
  - Free-flight distance to next collision, $s$
- Ray-tracing in 3D computational geometry

**Collision**
- Simulate absorption:
  - absorb, or
  - reduce weight
- Random sampling using nuclear data:
  - Collision isotope
  - Reaction type
  - Exit $E'$ & $\Omega'$
  - Secondary particles

During analysis of both flights & collisions, tally information about distances, collisions, etc. to use later in statistical analysis for results.
Monte Carlo Techniques for Nuclear Systems – Theory Lectures

Monte Carlo Simulation - Assumptions

Assume:

– Neutrons & photons are particles, not waves
– Particles move in a straight line between collisions (neutrons, photons)
– Collisions occur instantaneously, at a point in space

– Particle speeds are small enough to neglect relativistic effects
– Particle speeds are high enough to neglect quantum effects

– Particle collisions do not change the properties of a material (ie, no feedback, no material heating, no depletion)

– Material properties are fixed for the duration of the simulation (geometry, densities, temperatures, material compositions, …..)

Why?

– Want to solve the linear Boltzmann transport equation
– Want to apply the superposition principle
– Want the Central Limit Theorem to apply for computing statistics
  • Statisticians love the term “IID” - Independent, Identically Distributed

(Any or all of the above assumptions can be relaxed, with careful analysis & extra computing cost.)
Linear Boltzmann Transport Equation

- Time-dependent linear Boltzmann transport equation for neutrons, with prompt fission source & external source

\[
\frac{1}{v} \frac{\partial \psi(\vec{r}, E, \vec{\Omega}, t)}{\partial t} = \frac{Q(\vec{r}, E, \vec{\Omega}, t)}{v} + \int \int \psi(\vec{r}, E', \vec{\Omega}', t) \Sigma_S(\vec{r}, E' \rightarrow E, \vec{\Omega} \cdot \vec{\Omega}') \, d\vec{\Omega}' dE'
\]

- Scattering

\[
+ \frac{\chi(\vec{r}, E)}{4\pi} \int \int v \Sigma_F(\vec{r}, E') \psi(\vec{r}, E', \vec{\Omega}', t) \, d\vec{\Omega}' dE'
\]

- External source

- Multiplication

- Collision

- Leakage

- Gains

- Losses

\[
\frac{1}{v} \frac{\partial \psi(\vec{r}, E, \vec{\Omega}, t)}{\partial t} = \frac{Q}{v} + [S + M] \cdot \psi - [L + T] \cdot \psi
\]

- This equation can be solved directly by Monte Carlo, assuming:
  - Each neutron history is an IID trial (independent, identically distributed)
  - All neutrons must see same probability densities in all of phase space
  - Usual method: geometry & materials fixed over solution interval $\Delta t$
Time-dependent Diffusion

- Monte Carlo codes solve the transport equation

- For comparison, the multigroup diffusion equation is an approximation used by many codes & taught to undergrads:

\[
\frac{1}{v} \frac{\partial \Phi_G(\vec{r}, t)}{\partial t} = Q_G(\vec{r}, t) + \sum_{G' \neq G} \sum_{S,G' \rightarrow G} \Phi_{G'}(\vec{r}, t) + \frac{\chi_G(\vec{r})}{4\pi} \sum_{G'} v \sum_{F,G'} \Phi_{G'}(\vec{r}, t)
\]

<table>
<thead>
<tr>
<th>External source</th>
<th>Scattering</th>
<th>Multiplication</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ 1 \frac{\partial \Phi_G(\vec{r}, t)}{\partial t} = Q_G(\vec{r}, t) ]</td>
<td>[ \sum_{G' \neq G} \sum_{S,G' \rightarrow G} \Phi_{G'}(\vec{r}, t) ]</td>
<td>[ \frac{\chi_G(\vec{r})}{4\pi} \sum_{G'} ]</td>
</tr>
<tr>
<td>[\sum_{A,G}(\vec{r}) - \nabla \cdot D_G \nabla ] [ \Phi_G(\vec{r}, t) ]</td>
<td>[ \sum_{G'} ]</td>
<td>[ v \sum_{F,G'} ]</td>
</tr>
</tbody>
</table>

- Many approximations - angle, energy, D, .....
Monte Carlo Techniques for Nuclear Systems – Theory Lectures

Monte Carlo & Transport Equation

- **Derive integral equation, in kernel form**
  - Start with integro-differential equation
  - Use integrating factor
  \[
  \exp\left[-\int_{0}^{R} \Sigma_{T}(\vec{r} - R\hat{\Omega}, E) dR'\right], \quad \text{where } R\hat{\Omega} = \vec{r} - \vec{r}'
  \]
  - Define
  \[
  \vec{E} = E \cdot \hat{\Omega}
  \]

  **Collision density:**
  \[
  \Psi(\vec{r}, \vec{E}) = \Sigma_{T}(\vec{r}, E) \cdot \psi(\vec{r}, \vec{E})
  \]

  **Transport kernel:**
  \[
  T(\vec{r}' \rightarrow \vec{r}, \vec{E}) = \Sigma_{T}(\vec{r}, E) \cdot \exp\left[-\int_{0}^{[\vec{r}-\vec{r}']} \Sigma_{T}(\vec{r}', + s\hat{\Omega}, E) ds\right] \cdot \frac{\delta(\hat{\Omega} \cdot [\vec{r}-\vec{r}'] - 1)}{|\vec{r}-\vec{r}'|^2}
  \]

  **Collision kernel:**
  \[
  C(\vec{E}' \rightarrow \vec{E}, \vec{r}) = \frac{\Sigma_{S}(\vec{r}, \vec{E}' \rightarrow \vec{E})}{\Sigma_{T}(\vec{r}, E')} + \frac{\chi(\vec{r}, E)\nu\Sigma_{F}(\vec{r}, E')}{4\pi \cdot \Sigma_{T}(\vec{r}, E')}
  \]

  - Then
  \[
  \Psi(\vec{r}, \vec{E}) = \int \left[ \int \Psi(\vec{r}', \vec{E}') \cdot C(\vec{E}' \rightarrow \vec{E}, \vec{r}') d\vec{E}' + Q(\vec{r}', \vec{E}') \right] \cdot T(\vec{r}' \rightarrow \vec{r}, \vec{E}) d\vec{r}'
  \]

Reference: D.C. Irving, "The Adjoint Boltzmann Equation and Its Simulation by Monte Carlo"  
*Nuclear Engineering & Design* 15, 273-292 (1971)
Monte Carlo & Transport Equation

Basis for the Monte Carlo Solution Method

\[ \Psi(\vec{r},E) = \int \left[ \int \Psi(\vec{r}',E') \cdot C(\vec{E}' \rightarrow \vec{E},\vec{r}') d\vec{E}' + Q(\vec{r}',\vec{E}') \right] \cdot T(\vec{r}' \rightarrow \vec{r}, \vec{E}) d\vec{r}' \]

Let \( p = (\vec{r},E) \) and \( R(p' \rightarrow p) = C(\vec{E}' \rightarrow \vec{E},\vec{r}') \cdot T(\vec{r}' \rightarrow \vec{r},\vec{E}) \)

Expand \( \Psi \) into components having 0,1,2,...,k collisions

\[ \Psi(p) = \sum_{k=0}^{\infty} \Psi_k(p), \quad \text{with} \quad \Psi_0(p) = \int Q(\vec{r}',\vec{E}) \cdot T(\vec{r}' \rightarrow \vec{r},\vec{E}) d\vec{r}' \]

By definition,

\[ \Psi_k(p) = \int \Psi_{k-1}(p') \cdot R(p' \rightarrow p) dp' \]

Markovian: collision k depends only on the results of collision k-1, and not on any prior collisions k-2, k-3, ...
Monte Carlo & Transport Equation

Histories

• After repeated substitution for $\Psi_k$

$$\Psi_k(p) = \int \Psi_{k-1}(p') \cdot R(p' \rightarrow p) dp'$$

$$= \int \ldots \int \Psi_0(p_0) \cdot R(p_0 \rightarrow p_1) \cdot R(p_1 \rightarrow p_2) \ldots R(p_{k-1} \rightarrow p) dp_0 \ldots dp_{k-1}$$

• A "history" is a sequence of states $(p_0, p_1, p_2, p_3, \ldots)$

• For estimates in a given region, tally the occurrences for each collision of each "history" within a region
Monte Carlo & Transport Equation

\[ \Psi_k(p) = \int \cdots \int \Psi_0(p_0) \cdot R(p_0 \rightarrow p_1) \cdot R(p_1 \rightarrow p_2) \cdots R(p_{k-1} \rightarrow p) dp_0 \cdots dp_{k-1} \]

**Monte Carlo approach:**

- **Generate a sequence of states** \((p_0, p_1, p_2, p_3, \ldots)\) [i.e., a history] by:
  - Randomly sample from PDF for source: \(\Psi_0(p_0)\)
  - Randomly sample from PDF for \(k^{th}\) transition: \(R(p_{k-1} \rightarrow p_k)\)

- **Generate estimates of results** by averaging over states for \(M\) histories:
  \[
  A = \int A(p) \cdot \Psi(p) dp \approx \frac{1}{M} \cdot \sum_{m=1}^{M} \left( \sum_{k=1}^{\infty} A(p_{k,m}) \right)
  \]
Fixed-source Monte Carlo Calculation

Random Walk for a particle

Track through geometry,
- select collision site randomly
- tallies

Collision physics analysis,
- Select new E,Ω randomly
- tallies

Secondary Particles

Particle Histories

History 1

Source
- select r,E, Ω

Random Walk

Random Walk

Random Walk

Random Walk

History 2

Source
- select r,E, Ω

Random Walk

Random Walk

Random Walk

Random Walk

History 3

Source
- select r,E, Ω

Random Walk

Random Walk

Random Walk

Random Walk

Random Walk
**K_{eff} Eigenvalue Equation**

- For problems with fission multiplication, another approach is to create a static eigenvalue problem from the time-dependent transport equation (the asymptotic or steady-state solution)

- Introduce $K_{eff}$, a scaling factor on the multiplication ($\nu$)

- **Assume:**
  1. Fixed geometry & materials
  2. No external source: $Q(r,E,\Omega,t) = 0$
  3. $\frac{\partial \Psi}{\partial t} = 0$: $\nu \Rightarrow \nu / K_{eff}$

- Setting $\frac{\partial \Psi}{\partial t} = 0$ and introducing the $K_{eff}$ eigenvalue gives

  $$\left[ L + T \right] \Psi_k(\vec{r},E,\Omega) = \left[ S + \frac{1}{K_{eff}} M \right] \Psi_k$$

  - Steady-state equation, a static eigenvalue problem for $K_{eff}$ and $\Psi_k$
  - $K_{eff}$ = effective multiplication factor
  - Critical: $K=1$, subcritical: $k<1$, supercritical: $k >1$
  - $K_{eff}$ and $\Psi_k$ should never be used to model time-dependent problems.
The factor $1/k$ changes the relative level of the fission source, to balance the equation & permit a steady-state solution.

- **Criticality**
  - Supercritical: $K_{\text{eff}} > 1$
  - Critical: $K_{\text{eff}} = 1$
  - Subcritical: $K_{\text{eff}} < 1$
Monte Carlo Eigenvalue calculation

- Initial Guess
- Cycle 1 $K_{\text{eff}}^{(1)}$
- Cycle 2 $K_{\text{eff}}^{(2)}$
- Cycle 3 $K_{\text{eff}}^{(3)}$
- Cycle 4 $K_{\text{eff}}^{(4)}$

Source - select $r, E, \Omega$

Random Walk

Iterate (cycle) until converged, then more to accumulate tallies
**K\textsubscript{eff} Eigenvalue Equation**

- A common misinterpretation: $K\textsubscript{eff} > 1$ means “power is rising”
  - Not true!
  - The factor $1/K\textsubscript{eff}$ in the transport equation is the eigenvalue required to make the equation balance for a steady-state solution (stationary eigenfunction)
  - Reactor designers & analysts instead think of reactivity:
    \[
    \rho = \frac{k\textsubscript{eff} - 1}{k\textsubscript{eff}}
    \]
  - Think of reactivity as a “potential” for power to rise or fall
    - If it is not cancelled out by some control, then power would rise.
    - In practice, positive reactivity is cancelled out by negative reactivity from control rods, soluble boron, fission product poisoning, temperature changes, etc.
  - The job of reactor designers is:
    - Calculate the reactivity for a specific reactor configuration.
    - If the reactivity is positive, some changes need to be made to introduce negative reactivity. Conversely for negative reactivity.
Statistics Review

- Probability Density Function & Random Sampling
- Continuous PDF & CDF
- Discrete PDF & CDF
- Mean, Standard Deviation, Standard Deviation of the Mean
Introduction

Probability?

What are the odds of ..... 

- Being audited by the IRS this year 100 to 1
- Losing your luggage on a U.S. flight 176 to 1
- Being dealt 4 aces on an opening poker hand 4,164 to 1
- Being struck by lightning in your lifetime 9,100 to 1
- Being hit by a baseball at a major league game 300,000 to 1
- Drowning in your bathtub this year 685,000 to 1
- Winning the Powerball jackpot with 1 ticket 292,201,338 to 1

Yet we all still keep buying Powerball tickets, but don’t worry too much about lightning...
Simple Random Sampling (1)

- Suppose we have 3 items, A, B, and C
  - $P_A$ = probability of randomly picking item A = 25% = 0.25
  - $P_B$ = probability of randomly picking item B = 50% = 0.50
  - $P_C$ = probability of randomly picking item C = 25% = 0.25
  - $P_A + P_B + P_C = 1.0$

- Random sampling - pick A or B or C

Generate a random number $R$ in the range (0,1)

If $R < 0.25$ $$$ select A
Else if $0.25 < R < 0.75$ $$$ select B
Otherwise $$$ select C

Cumulative Probabilities
Simple Random Sampling (2)

- **Random sampling - pick A or B or C**
  - $P_A = \text{probability of randomly picking item A} = 25\% = 0.25$
  - $P_B = \text{probability of randomly picking item B} = 50\% = 0.50$
  - $P_C = \text{probability of randomly picking item C} = 25\% = 0.25$
  - $P_A + P_B + P_C = 1.0$

Generate a random number $R$ in the range (0,1),
Pick A, B, or C
Probability Density Functions

- **Continuous Probability Density**
  \[ f(x) = \text{probability density function (PDF)} \]
  \[ f(x) \geq 0 \]
  \[
  \text{Probability}\{a \leq x \leq b\} = \int_{a}^{b} f(x)\,dx \\
  \text{Normalization:} \quad \int_{-\infty}^{\infty} f(x)\,dx = 1
  \]

- **Discrete Probability Density**
  \[
  \{ f_k \}, \quad k = 1, \ldots, N, \quad \text{where} \quad f_k = f(x_k) \\
  f_k \geq 0 \\
  \text{Probability}\{ x = x_k \} = f_k \\
  \text{Normalization:} \quad \sum_{k=1}^{N} f_k = 1
  \]
Random Sampling

The key to Monte Carlo methods is the notion of random sampling.

- The problem can be stated this way:
  Given a probability density, \( f(x) \), produce a sequence of \( \hat{x} \)'s. The \( \hat{x} \)'s should be distributed in the same manner as \( f(x) \).

- Use of random sampling distinguishes Monte Carlo from other methods.

- When Monte Carlo is used to solve the Boltzmann transport equation:
  - Random sampling models the outcome of physical events (e.g., neutron collisions, fission process, sources, .....)
  - Computational geometry models the arrangement of materials.
Continuous PDF & CDF

- **Probability Density Function (PDF)**

  \[ f(x) = \text{probability density function (PDF)} \]
  \[ f(x) \geq 0 \]
  \[ \text{Probability}\{a \leq x \leq b\} = \int_{a}^{b} f(x) \, dx \]
  \[ \text{Normalization:} \int_{-\infty}^{\infty} f(x) \, dx = 1 \]

- **Cumulative Distribution Function (CDF)**

  \[ F(x) = \int_{-\infty}^{x} f(x') \, dx' \]
  \[ 0 \leq F(x) \leq 1 \]
  \[ \frac{dF(x)}{dx} \geq 0 \]
  \[ F(-\infty) = 0, \quad F(\infty) = 1 \]
Direct Sampling

• Direct solution of \( \hat{x} = F^{-1}(\xi) \)

Solve for \( \hat{x} \):

\[
\xi = \int_{-\infty}^{\hat{x}} f(x) \, dx
\]

• Sampling procedure
  – Generate \( \xi \)
  – Determine \( \hat{x} \) such that \( F(\hat{x}) = \xi \)

• Advantages
  – Straightforward mathematics & coding
  – "High-level" approach

• Disadvantages
  – Often involves complicated functions
  – In some cases, \( F(x) \) cannot be inverted (e.g., Klein-Nishina)
Rejection Sampling

- **Von Neumann:**
  
  " .......... it seems objectionable to compute a transcendental function of a random number. "

- **Select a bounding function,** $g(x)$, such that
  
  - $c \cdot g(x) > f(x)$ for all $x$
  - $g(x)$ is an easy-to-sample PDF

- **Sampling Procedure:**
  
  - sample $x'$ from $g(x)$: $x' \leftarrow G^{-1}(\xi_1)$
  - test: $\xi_2 \cdot c g(x') < f(x')$

    - if **true**, accept $x'$, done
    - if **false**, reject $x'$, try again

- **Advantages**
  - Simple computer operations

- **Disadvantages**
  - "Low-level" approach, sometimes hard to understand
Mean & Standard Deviation

- Given a set of random samples, $x_1, x_2, \ldots, x_N$,
  - Mean $\bar{x} = \frac{1}{N} \sum_{j=1}^{N} x_j$

- Population variance & standard deviation
  $$\sigma^2 = \frac{1}{N} \sum_{j=1}^{N} x_j^2 - \left( \frac{1}{N} \sum_{j=1}^{N} x_j \right)^2 = \frac{1}{N} \sum_{j=1}^{N} x_j^2 - \bar{x}^2$$
  $$\sigma = \sqrt{\frac{1}{N} \sum_{j=1}^{N} x_j^2 - \bar{x}^2}$$

- Variance & standard deviation of the mean
  $$\sigma_{\bar{x}}^2 = \frac{\sigma^2}{N}$$
  $$\sigma_{\bar{x}} = \frac{1}{\sqrt{N}} \cdot \sqrt{\frac{1}{N} \sum_{j=1}^{N} x_j^2 - \bar{x}^2}$$

Monte Carlo codes calculate mean values for tallies, & report the standard deviation of the mean

In the definitions above, some of the “$N$” terms should really be “$N-1$”. MCNP & many other codes ignore that, since $N$ is very large.
Random Numbers & Random Sampling
The key to Monte Carlo methods is the notion of *random sampling*.

- The problem can be stated this way:
  
  Given a probability density, \( f(x) \), produce a sequence of \( \hat{X} \)'s. The \( \hat{X} \)'s should be distributed in the same manner as \( f(x) \).

- Random sampling distinguishes Monte Carlo from other methods
- When Monte Carlo is used to solve the Boltzmann transport equation:
  - Random sampling models the outcome of physical events (e.g., neutron collisions, fission process, sources, ……)
  - Computational geometry models the arrangement of materials
Monte Carlo & Random Sampling

Categories of random sampling

• Random number generator - uniform PDF on (0,1)
• Sampling from analytic PDFs - normal, exponential, Maxwellian, …
• Sampling from tabulated PDFs - angular PDFs, spectrum, …

For Monte Carlo codes…

• Random numbers, $\xi$, are produced by the RN generator on (0,1)
• Non-uniform random variates are produced from the $\xi$’s by:
  – Direct inversion
  – Rejection methods
  – Transformations
  – Composition (mixtures)
  – Sums, products, ratios, …
  – Table lookup + interpolation
  – Lots (!) of other tricks
• Typically < 5 - 10% of total CPU time
Random Number Generators

"Randomness is a negative property; it is the absence of any pattern." Richard W. Hamming, 1991

- **Numbers are not random; a sequence of numbers can be.**

- **Truly random sequences are generally not desired on a computer.**

- **RNG**
  - Function which generates a sequence of numbers which appear to have been randomly sampled from a uniform distribution on (0,1)
  - Repeatable (deterministic)
  - Pass statistical tests for randomness
  - Typical usage in codes: \( r = \text{rang()} \)
  - Also called "pseudo-random number generators"

- All other random sampling is performed using this basic RNG

- Note that the probability of something occurring also varies between 0 & 1 ….
Random Number Generators

Most production-level Monte Carlo codes for particle transport use linear congruential random number generators:

\[ S_{i+1} = S_i \cdot g + c \mod 2^m \]

- \( S_i \) = seed, \( g \) = multiplier, \( c \) = adder, \( 2^m \) = modulus

- Simple, fast, robust, over 50 years of heavy-duty use
- Theory is well-understood (e.g., DE Knuth, Vol. 2, 177 pages)
- Not the "best" generators, but good enough - RN's are used in unpredictable ways during particle simulation
- To achieve reproducibility for vector or parallel calculation, there must be a fast, direct method for skipping ahead (or back) in the random sequence
Simple RNG - Example #1

\( S_{k+1} = [g \cdot S_k + C] \mod p, \quad \text{with } g=47, \; C=1, \; S_0=1, \; p=100 \)

\[

table
\begin{array}{ll}
   s(0) & = 1 \\
   s(1) & = (47 \times 1 + 1) \mod 100 = 48 \mod 100 = 48 \\
   s(2) & = (47 \times 48 + 1) \mod 100 = 2257 \mod 100 = 57 \\
   s(3) & = (47 \times 57 + 1) \mod 100 = 2680 \mod 100 = 80 \\
   s(4) & = (47 \times 80 + 1) \mod 100 = 3761 \mod 100 = 61 \\
   s(5) & = (47 \times 61 + 1) \mod 100 = 2868 \mod 100 = 68 \\
   s(6) & = (47 \times 68 + 1) \mod 100 = 3197 \mod 100 = 97 \\
   s(7) & = (47 \times 97 + 1) \mod 100 = 4560 \mod 100 = 60 \\
   s(8) & = (47 \times 60 + 1) \mod 100 = 2821 \mod 100 = 21 \\
   s(9) & = (47 \times 21 + 1) \mod 100 = 988 \mod 100 = 88 \\
   s(10) & = (47 \times 88 + 1) \mod 100 = 4137 \mod 100 = 37 \\
   s(11) & = (47 \times 37 + 1) \mod 100 = 1740 \mod 100 = 40 \\
   s(12) & = (47 \times 40 + 1) \mod 100 = 1881 \mod 100 = 81 \\
   s(13) & = (47 \times 81 + 1) \mod 100 = 3808 \mod 100 = 8 \\
   s(14) & = (47 \times 8 + 1) \mod 100 = 377 \mod 100 = 77 \\
   s(15) & = (47 \times 77 + 1) \mod 100 = 3620 \mod 100 = 20 \\
   s(16) & = (47 \times 20 + 1) \mod 100 = 941 \mod 100 = 41 \\
   s(17) & = (47 \times 41 + 1) \mod 100 = 1928 \mod 100 = 28 \\
   s(18) & = (47 \times 28 + 1) \mod 100 = 1317 \mod 100 = 17 \\
   s(19) & = (47 \times 17 + 1) \mod 100 = 800 \mod 100 = 0 \\
   s(20) & = (47 \times 0 + 1) \mod 100 = 1 \mod 100 = 1 \\
   s(21) & = (47 \times 1 + 1) \mod 100 = 48 \mod 100 = 48 \\
   s(22) & = (47 \times 48 + 1) \mod 100 = 2257 \mod 100 = 57 \\
\end{array}
\]
Simple RNG - Examples #2 & #3

Example #2: \[ S_{k+1} = \left[ g \cdot S_k + C \right] \mod p, \]
with \( g=5, \ c=1, \ S_0=1, \ p=100 \)

\[
\begin{align*}
s(0) &= 1 \\
s(1) &= (5 \times 1 + 1) \mod 100 = 6 \mod 100 = 6 \\
s(2) &= (5 \times 6 + 1) \mod 100 = 31 \mod 100 = 31 \\
s(3) &= (5 \times 31 + 1) \mod 100 = 156 \mod 100 = 56 \\
s(4) &= (5 \times 56 + 1) \mod 100 = 281 \mod 100 = 81 \\
s(5) &= (5 \times 81 + 1) \mod 100 = 406 \mod 100 = 6 \\
s(6) &= (5 \times 6 + 1) \mod 100 = 31 \mod 100 = 31 \\
\end{align*}
\]

Example #3: \[ S_{k+1} = \left[ g \cdot S_k + C \right] \mod p, \]
with \( g=5, \ c=0, \ S_0=1, \ p=100 \)

\[
\begin{align*}
s(0) &= 1 \\
s(1) &= (5 \times 1) \mod 100 = 5 \mod 100 = 5 \\
s(2) &= (5 \times 5) \mod 100 = 25 \mod 100 = 25 \\
s(3) &= (5 \times 25) \mod 100 = 125 \mod 100 = 25 \\
s(4) &= (5 \times 25) \mod 100 = 125 \mod 100 = 25 \\
\end{align*}
\]

etc.
Typical Linear Congruential RNGs

- **Multiplicative congruential method - Lehmer**

\[ S_k = g \cdot S_{k-1} + c \mod 2^m, \quad 0 < S_k < 2^m, \text{ integer} \]

\[ \xi_k = S_k / 2^m, \quad 0 \leq \xi_k < 1, \text{ real} \]

- **Typical parameters**

<table>
<thead>
<tr>
<th>RNG</th>
<th>(2^m)</th>
<th>Period</th>
<th>(g)</th>
<th>(c)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RACER (KAPL)</td>
<td>(2^{47})</td>
<td>(2^{45})</td>
<td>84,000,335,758,957</td>
<td>0</td>
</tr>
<tr>
<td>RCP (BAPL)</td>
<td>(2^{48})</td>
<td>(2^{48})</td>
<td>2^{9+1} 59,482,192,516,946</td>
<td></td>
</tr>
<tr>
<td>MORSE (ORNL)</td>
<td>(2^{47})</td>
<td>(2^{45})</td>
<td>5^{15}</td>
<td>0</td>
</tr>
<tr>
<td>VIM (ANL)</td>
<td>(2^{48})</td>
<td>(2^{46})</td>
<td>5^{19}</td>
<td>0</td>
</tr>
<tr>
<td>RANF (CRAY)</td>
<td>(2^{48})</td>
<td>(2^{46})</td>
<td>44,485,709,377,909</td>
<td>0</td>
</tr>
<tr>
<td>G. Marsaglia</td>
<td>(2^{32})</td>
<td>(2^{32})</td>
<td>69069</td>
<td>1</td>
</tr>
<tr>
<td>MCNP6 (default)</td>
<td>(2^{48})</td>
<td>(2^{46})</td>
<td>5^{19}</td>
<td>0</td>
</tr>
<tr>
<td>MCNP6 (opt)</td>
<td>(2^{63})</td>
<td>(2^{63})</td>
<td>(6 options)</td>
<td>1 or 0</td>
</tr>
</tbody>
</table>
MCNP5 & MCNP6 RNG

• MCNP5 & MCNP6 Linear congruential generator (LCG)

\[ S_{n+1} = g \ S_n + c \mod 2^m \]

  – See Knuth for rules for selecting g,c,m so that period is maximized & correlation minimized

  – 7 different LCGs are available -- chosen based on the spectral test, Knuth's tests, & Marsaglia's DIEHARD tests

  – LCG( g, c, 2^m ):
    • Traditional MCNP, period = \( 2^{46} \approx 7 \times 10^{14} \)
      – LCG( 5^{19}, 0, 2^{48} )
    • L'Ecuyer 63-bit Mixed LCGs, period = \( 2^{63} \approx 9 \times 10^{18} \)
      – LCG( 9219741426499971445, 1, 2^{63} )
      – LCG( 2806196910506780709, 1, 2^{63} )
      – LCG( 3249286849523012805, 1, 2^{63} )
    • L'Ecuyer 63-bit Multiplicative LCGs, period = \( 2^{61} \approx 2 \times 10^{18} \)
      – LCG( 3512401965023503517, 0, 2^{63} )
      – LCG( 2444805353187672469, 0, 2^{63} )
      – LCG( 1987591058829310733, 0, 2^{63} )

Using RNGs in Particle Transport MC Codes

- **Naïve use, in many older codes & student codes**

  ![Diagram of RN sequences for different particles]

  - **Problem:** Can't start Particle-2 until Particle-1 is finished, etc.
    Can't do parallel processing of different particles

- **MCNP, VIM, RACER, MC21, & many other production codes**
  - Partition RN sequence into equal-length subsequences, one for each particle

  ![Diagram of partitioned RN sequences for different particles]

  - Can process all particles in parallel
  - Length of each subsequence is called the **stride**
  - Must have a fast way to **skip-ahead** in the RN sequence
Using RNGs in Particle Transport MC Codes

- **Histories vs particles**
  - With splitting &/or secondary particle creation, the number of particles in a given history is not known in advance.
  - Need to partition RN sequence by history, not by particle.

- With this scheme, can process histories in parallel, but not particles in same history.
- Must have a predictable scheme for banking/unbanking particles in a given history (e.g., LIFO).
Random Sampling

"Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin."

John Von Neuman, 1951
Introduction

Probability?

What are the odds of …..

- Being audited by the IRS this year: 100 to 1
- Losing your luggage on a U.S. flight: 176 to 1
- Being dealt 4 aces on an opening poker hand: 4,164 to 1
- Being struck by lightning in your lifetime: 9,100 to 1
- Being hit by a baseball at a major league game: 300,000 to 1
- Drowning in your bathtub this year: 685,000 to 1
- Winning the Powerball jackpot with 1 ticket: 292,201,338 to 1

Yet we all still keep buying Powerball tickets, but don’t worry too much about lightning…
Simple Random Sampling (1)

- **Suppose we have 2 items, A and B**
  - \( P_A \) = probability of randomly picking item A = 25% = 0.25
  - \( P_B \) = probability of randomly picking item B = 75% = 0.75
  - \( P_A + P_B = 1.0 \)

- **Random sampling - pick A or B**

  Generate a random number \( R \) in the range (0,1)

  If \( R < 0.25 \) select A
  Otherwise select B
Simple Random Sampling (2)

- **Suppose we have 3 items, A, B, and C**
  - $P_A = \text{probability of randomly picking item A} = 25\% = 0.25$
  - $P_B = \text{probability of randomly picking item B} = 50\% = 0.50$
  - $P_C = \text{probability of randomly picking item C} = 25\% = 0.25$
  - $P_A + P_B + P_C = 1.0$

- **Random sampling - pick A or B or C**

  Generate a random number $R$ in the range $(0,1)$

  If $R < 0.25$ - select A
  Else If $0.25 < R < 0.75$ - select B
  Otherwise - select C

![Cumulative Probabilities](image)
Simple Random Sampling (3)

- **Random sampling - pick A or B or C**
  - $P_A$ = probability of randomly picking item A = 25% = 0.25
  - $P_B$ = probability of randomly picking item B = 50% = 0.50
  - $P_C$ = probability of randomly picking item C = 25% = 0.25
  - $P_A + P_B + P_C = 1.0$

Generate a random number $R$ in the range (0,1),
Pick A, B, or C
Probability Density Functions

- **Continuous Probability Density**
  \[ f(x) = \text{probability density function (PDF)} \]
  \[ f(x) \geq 0 \]
  \[ \text{Probability}\{a \leq x \leq b\} = \int_{a}^{b} f(x) \, dx \]
  \[ \text{Normalization: } \int_{-\infty}^{\infty} f(x) \, dx = 1 \]

- **Discrete Probability Density**
  \[ \{ f_k \}, \quad k = 1, \ldots, N, \quad \text{where } f_k = f(x_k) \]
  \[ f_k \geq 0 \]
  \[ \text{Probability}\{ x = x_k \} = f_k \]
  \[ \text{Normalization: } \sum_{k=1}^{N} f_k = 1 \]
Continuous PDF & CDF

- **Probability Density Function (PDF)**

  \[ f(x) = \text{probability density function (PDF)} \]

  \[ f(x) \geq 0 \]

  Probability \( \{a \leq x \leq b\} = \int_a^b f(x) \, dx \)

  Normalization: \( \int_{-\infty}^{\infty} f(x) \, dx = 1 \)

- **Cumulative Distribution Function (CDF)**

  \[ F(x) = \int_{-\infty}^{x} f(x') \, dx' \]

  \[ 0 \leq F(x) \leq 1 \]

  \[ \frac{dF(x)}{dx} \geq 0 \]

  \[ F(-\infty) = 0, \quad F(\infty) = 1 \]

  Note: convention is to use \( f(x) \) for PDF, \( F(x) \) for CDF
Discrete PDF & CDF

• Probability Density Function (PDF)

\[ \{ f_k \}, \text{ where } f_k = f(x_k), \quad k = 1, \ldots, N \]

\[ f_k \geq 0 \]

\[ \sum_{j=1}^{N} f_j = 1 \]

• Cumulative Distribution Function (CDF)

\[ \{ F_k \}, \text{ where } F_k = \sum_{j=1}^{k} f_j, \quad k = 1, \ldots, N-1 \]

and

\[ F_0 = 0, \]

\[ F_N = 1 \]

Note: convention is to use \( f_j \) for PDF, \( F_j \) for CDF
Fundamental RNG

- **Random Number Generator**
  - Not strictly "random", but good enough
  - Pass statistical tests for randomness
  - Reproducible sequence
  - Uniform PDF on (0,1)
  - Must be easy to compute

- **Linear Congruential Method**
  - Algorithm
    \[
    S_0 = \text{initial seed, odd integer, } < M \\
    S_k = G \cdot S_{k-1} + c \mod M, \quad k = 1, 2, \ldots \\
    \xi_k = S_k / M
    \]

- **Usage**
  - In algorithms, usually denote RN uniform on (0,1) by $\xi$ ("xye")
  - In codes, invoke basic RN generator by: $r = \text{rang}()$
  - Each new usage of $\xi$ or $\text{rang}()$ generates a new RN
Random Sampling

The key to Monte Carlo methods is the notion of random sampling.

• The problem can be stated this way:
  Given a probability density, f(x), produce a sequence of \( \hat{X} \)'s. The \( \hat{X} \)'s should be distributed in the same manner as f(x).

• The use of random sampling distinguishes Monte Carlo from other methods

• When Monte Carlo is used to solve the integral Boltzmann transport equation:
  – Random sampling models the outcome of physical events (e.g., neutron collisions, fission process, sources, .....)
  – Computational geometry models the arrangement of materials
Direct Sampling

- Direct solution of \( x = F^{-1}(\xi) \)

\[ \xi = \int_{-\infty}^{x} f(y) \, dy \]

- Sampling procedure
  - Generate \( \xi \)
  - Determine \( x \) such that \( F(x) = \xi \)

- Advantages
  - Straightforward mathematics & coding
  - "High-level" approach

- Disadvantages
  - Often involves complicated functions
  - In some cases, \( F(x) \) cannot be inverted (e.g., Klein-Nishina)
Discrete PDFs

- **Sampling from Discrete PDF's - Conventional Procedure**

Direct Solution of \( x' \leftarrow F^{-1}(\xi) \)

1. Generate \( \xi \)
2. Determine \( k \) such that \( F_{k-1} \leq \xi < F_k \)
3. Return \( x' = x_k \)

- **Step (2) requires a table search**
  - Linear table searches require \( O(N) \) time - use when \( N \) small
  - Binary table searches require \( O(\ln N) \) time - use when \( N \) large

- **For some discrete PDFs, \( F_k \)'s are not precomputed.**
  - Use linear search, with \( F_k \)'s computed on-the-fly as needed
Discrete Uniform PDF

- Example - Sampling from Discrete Uniform PDF

- Discrete Uniform PDF
  - \( f_k = \frac{1}{N}, \quad k = 1, \ldots, N \)
  - \( F_k = \frac{k}{N}, \quad F_0 = 0, \quad F_N = 1 \)

- Sampling procedure:
  - Could use table search method, ....
  - Easier, for this special case:
    
    \[
    k \leftarrow 1 + \left\lfloor N \xi \right\rfloor,
    \]
    
    where \( \left\lfloor y \right\rfloor \) is the "floor" function, largest integer < y

  - Fortran: \( k = 1 + \text{int}( N \times \text{rang}() ) \)
  - C: \( k = 1 + \text{floor}( N \times \text{rang}() ) \)

  - Note: must be sure that \( 1 \leq k \leq N \)
Discrete PDFs - Examples

• Example – Pick 1 Powerball number, uniform integer in [1,60]

  \[ k = \text{int}( 1 + 60 \times \text{rang}() ) \]

• Example - loaded die, faces show 2,2,3,4,5,5 – simulate 1 roll

  \[
  \begin{align*}
  \text{pdf}(1:6) & = \left[ 0./6., 2./6., 1./6., 1./6., 2./6., 0./6. \right] \\
  \text{cdf}(1:6) & = \left[ 0./6., 2./6., 3./6., 4./6., 6./6., 6./6. \right]
  \end{align*}
  \]

  \[
  \begin{align*}
  r & = \text{rang}() \\
  \text{do } j & = 1, 6 \\
  & \quad \text{if( } r < \text{cdf}(j) \text{ ) then} \\
  & \quad \quad k = j \\
  & \quad \quad \text{exit} \\
  & \quad \text{endif} \\
  \text{endo}
  \end{align*}
  \]

  \{\text{result is } k\}

This coding is a simple linear search to determine an integer \( k \) in the range [1,6].

Search for the first occurrence of \( \xi \leq \text{cdf}(j) \).
Random Sampling -- Discrete PDFs

- **Multigroup Scattering**
  
  - Scatter from group $g$ to group $g'$, where $1 \leq g' \leq G$
  
  $$f_{g'} = \frac{\sigma_{g \rightarrow g'}}{\sum_{k=1}^{G} \sigma_{g \rightarrow k}}$$

  - **Selection of scattering nuclide for a collision**

  - $K =$ number of nuclides in composition
  
  $$f_k = \frac{N^{(k)} \sigma_s^{(k)}}{\sum_{j=1}^{K} N^{(j)} \sigma_s^{(j)}}$$
★★★★★ Direct Sampling ★★★★★

- Direct solution of \( x = F^{-1}(\xi) \)

Solve for \( x \):
\[
\xi = \int_{-\infty}^{x} f(y) \, dy
\]

- Sampling procedure
  - Generate \( \xi \)
  - Determine \( x \) such that \( F(x) = \xi \)

- Advantages
  - Straightforward mathematics & coding
  - "High-level" approach

- Disadvantages
  - Often involves complicated functions
  - In some cases, \( F(x) \) cannot be inverted (e.g., Klein-Nishina)
Continuous PDFs - Exponential

Examples - Sampling from an Exponential PDF

PDF: \( f(x) = \sum \cdot e^{-\Sigma x}, \quad x > 0 \)

CDF: \( F(x) = \int_0^x f(y) \, dy = \int_0^x \sum \cdot e^{-\Sigma y} \, dy = -e^{-\Sigma y}\bigg|_0^x = 1 - e^{-\Sigma x} \)

Direct sampling:

Solve for \( x \): \( F(x) = \xi \)

Solving \( \xi = 1 - e^{-\Sigma x} \) gives: \( x \leftarrow -\ln(1 - \xi) / \Sigma \)

or

\( x \leftarrow -\ln \xi / \Sigma \)

Although \( (1 - \xi) \neq \xi \), both \( \xi \) and \( (1 - \xi) \) are uniformly distributed on \((0,1)\), so that we can use either in the random sampling procedure.

i.e., the numbers are different, but the distributions are the same
Continuous PDFs – Power Law on [0,1]

Example - Sampling from power law PDF in range [0,1],

PDF: \( f(x) = (n+1) \cdot x^n, \quad n>0, \quad 0 \leq x \leq 1 \)
\( = 0 \quad \text{for } x < 0, \text{ or } x > 1 \)

CDF: \( F(x) = \int_0^x f(y) \, dy = \int_0^x (n+1) \cdot y^n \, dy = \left. (n+1) \cdot \frac{y^{n+1}}{n+1} \right|_0^x = x^{n+1}, \quad 0 \leq x \leq 1 \)

Sampling scheme: \( F(x) = \xi, \quad \text{solve for } x \)
\( x^{n+1} = \xi \quad \Rightarrow \quad x \leftarrow \xi^{1/(n+1)} \)

For power laws on [0,1]:
\( n=1: \quad f(x) = 2x, \quad F(x) = x^2, \quad x \leftarrow \sqrt{\xi} \)
\( n=2: \quad f(x) = 3x^2, \quad F(x) = x^3, \quad x \leftarrow \sqrt[3]{\xi} \)
\( n=3: \quad f(x) = 4x^3, \quad F(x) = x^4, \quad x \leftarrow \sqrt[4]{\xi} \)
Continuous PDFs - Uniform

Example - Sampling from uniform PDF in range (a,b),
Histogram with 1 bin

PDF: \( f(x) = \frac{1}{b - a}, \quad a \leq x \leq b \)
\( = 0 \quad x < a, \) or \( x > b \)

CDF: \( F(x) = \frac{(x - a)}{(b - a)}, \quad a \leq x \leq b \)

Sampling scheme: \( F(x) = \xi, \) solve for \( x \)
\( \frac{(x - a)}{(b - a)} = \xi \)
\( x \leftarrow a + (b - a) \xi \)

Note: Often implemented as:
\( f = \xi \)
\( x \leftarrow (1-f) a + f b \)
Continuous PDFs - Histogram

Example - Sampling from histogram with 2 bins

\[ A_1 = (x_1 - x_0) \cdot f_1 \]
\[ A_2 = (x_2 - x_1) \cdot f_2 \]

\[ p_1 = \text{Prob}\{ x_0 < x < x_1 \} = \frac{A_1}{A_1 + A_2} \]
\[ p_2 = \text{Prob}\{ x_1 < x < x_2 \} = \frac{A_2}{A_1 + A_2} \]
\[ p_1 + p_2 = 1 \]

Two-step sampling procedure:

1. Select a bin, b:
   - If \( \xi_1 < p_1 \), select \( b = \text{bin 1} \)
   - Otherwise, select \( b = \text{bin 2} \)

2. Sample \( x \) uniformly within bin:
   \[ x \leftarrow x_{b-1} + \xi_2 \cdot (x_b - x_{b-1}) \]
Continuous PDFs - Histogram

Example - Sampling from Histogram PDF

Two-step sampling:
1. Sample from discrete PDF to select a bin
2. Sample from uniform PDF within bin

- Discrete PDF:
  \[ p_k = f_k \cdot (x_k - x_{k-1}), \quad k = 1, \ldots, N, \quad \Sigma p_k = 1 \]
  - Generate \( \xi_1 \)
  - Use table search to select \( k \)

- Uniform sampling within bin \( k \)
  - Generate \( \xi_2 \)
  - Then, \( x \leftarrow x_{k-1} + (x_k - x_{k-1}) \cdot \xi_2 \)
Continuous PDFs – Linear (1)

Example - Sampling from an increasing linear PDF in range [0,1]

PDF: \( f(x) = 2x, \ 0 \leq x \leq 1 \)

CDF: \( F(x) = x^2, \ 0 \leq x \leq 1 \)

Sampling scheme:

\[ F(x) = \xi, \quad \text{solve for } x \]

\[ x^2 = \xi \]

\[ x \leftarrow \sqrt{\xi} \]

While not obvious, 2 alternative schemes for sampling x are:

\[ x \leftarrow \max(\xi_1, \xi_2) \]

\[ x \leftarrow 1 - \text{abs}(\xi_1 - \xi_2) \]
Continuous PDFs – Linear \( (2) \)

**Example - Sampling from a decreasing linear PDF in range \([0, 1]\)**

**PDF:** \[ f(x) = 2 - 2x, \quad 0 \leq x \leq 1 \]

**CDF:** \[ F(x) = 2x - x^2, \quad 0 \leq x \leq 1 \]

**Sampling scheme:**

\[ F(x) = \xi, \quad \text{solve for } x \]

\[ 2x - x^2 = \xi \]
\[ x^2 - 2x + 1 = 1 - \xi \]
\[ (x-1)^2 = 1 - \xi \]
\[ x - 1 = \pm \sqrt{1 - \xi} \]

Choose the **minus sign** for correct range in \(x\):

\[ x \leftarrow 1 - \sqrt{1 - \xi} \]

Or, since \(\xi\) and \(1 - \xi\) have the same distribution:

\[ x \leftarrow 1 - \sqrt{\xi} \]
Continuous PDFs – Linear

Increasing linear PDF

Decreasing linear PDF

Random sampling can be done with a simple shifting & scaling of the unit PDF:

\[ x \leftarrow a + (b-a) \sqrt{\xi} \]

Random sampling can be done with a simple shifting & scaling of the unit PDF:

\[ x \leftarrow a + (b-a)(1 - \sqrt{\xi}) \]
Continuous PDFs – Linear (4)

Example - Sampling from linear PDF in range [a,b], 1 bin

PDF: \( f(x) = f_a + m (x-a), \quad m = (f_b-f_a)/(b-a), \quad a \leq x \leq b \)

CDF: \( F(x) = \frac{m}{2} x^2 + (f_a-ma) x + (ma^2/2 - f_a a) \)

\[ = A x^2 + B x + C \]

Sampling scheme: \( F(x) = \xi, \) solve for \( x \)
\[ x = \frac{\{-B \pm \sqrt{B^2 - 4AC}\}}{2A} \]

⇒ Awfully complicated, and sensitive to numerical roundoff
⇒ There must be a simpler scheme (there is …)
Continuous PDFs – Linear (5)

Example - Sampling from linear PDF in range [a,b], 1 bin

Composition method #1

Decompose the original PDF into the sum of 2 PDFs, uniform + linear:

\[ f(x) = p_u \ u(x) + p_l \ l(x) \]

\[ u(x) = \text{uniform on } a \leq x \leq b, \]

\[ p_u = \{ \min(f_a, f_b) \ (b-a) \} / \{ .5(f_a+f_b) \ (b-a) \} \]

\[ l(x) = \text{linear on } a \leq x \leq b, \]

\[ p_l = \{ .5 \abs(f_b-f_a) \ (b-a) \} / \{ .5(f_a+f_b) \ (b-a) \} \]

Sampling scheme:

if( \[ \xi_1 < p_u \] )

\[ x \leftarrow a + (b-a) \xi_2 \]

else

if( \[ f_b > f_a \] )

\[ x \leftarrow a + (b-a) \sqrt{\xi_2} \]

else

\[ x \leftarrow a + (b-a) (1 - \sqrt{\xi_2}) \]
Continuous PDFs – Linear \( (6) \)

Example - Sampling from linear PDF in range \([a,b]\), \(1\) bin

Composition method \#2

Decompose the original PDF into the sum of 2 PDFs, increasing + decreasing linear:

\[
f(x) = p_m \, m(x) + p_l \, l(x)
\]

\[
m(x) = \text{linear decreasing on } a \leq x \leq b,
\]
\[
p_m = \frac{.5 \, f_a \, (b-a)}{.5(f_a+f_b) \, (b-a)}
\]
\[
= \frac{f_a}{f_a+f_b}
\]

\[
l(x) = \text{linear increasing on } a \leq x \leq b,
\]
\[
p_l = \frac{.5 \, f_b \, (b-a)}{.5(f_a+f_b) \, (b-a)}
\]
\[
= \frac{f_b}{f_a+f_b}
\]

Sampling scheme:

\[
\text{if} \ ( \xi_1 < p_l )
\]
\[
x \leftarrow a + (b-a) \sqrt{\xi_2}
\]

\[
\text{else}
\]
\[
x \leftarrow a + (b-a) \left( 1 - \sqrt{\xi_2} \right)
\]
Continuous PDFs – Linear (7)

Examples — Sampling from Piecewise Linear PDF

Two-step sampling:
1. Sample from discrete PDF to select a bin
2. Sample from linear PDF within bin

- Discrete PDF:
  \[ p_k = \frac{(f_k + f_{k-1})}{2} \cdot (x_k - x_{k-1}), \quad k = 1, \ldots, N \]
  - generate \( \xi \)
  - use table search or alias method to select \( K \)

- Linear sampling within bin \( K \):
  - generate \( \xi \)
  - then, if \( \xi_1 < \frac{f_{k-1}}{f_k + f_{k-1}} \), \[ \hat{x} = x_k - (x_k - x_{k-1}) \sqrt{\xi_2} \]
  - otherwise, \[ \hat{x} = x_{k-1} + (x_k - x_{k-1}) \sqrt{\xi_2} \]
Continuous PDFs – Linear (8)

We have seen that a simple, increasing linear PDF in the range $[0,1]$ can be sampled directly by inverting the CDF to obtain:

PDF: \( f(x) = 2x, \quad 0 \leq x \leq 1 \)

CDF: \( F(x) = x^2, \quad 0 \leq x \leq 1 \)

Sampling scheme:

\[
F(x) = \xi, \quad \text{solve for } x
\]

\[
x \leftarrow \sqrt{\xi}
\]

While not obvious, some other schemes for sampling \( x \) are:

\[
x = \xi_1
\]

\[
r = \xi_2
\]

\[
\text{if( } r > x \text{ ) } x = r
\]

\[
x \leftarrow \max( \xi_1, \xi_2 )
\]

\[
x \leftarrow 1 - \text{abs}( \xi_1 - \xi_2 )
\]

Why consider these other schemes?

Sqrt() function used to be very expensive. The other schemes involve only simple non-arithmetic operations & were much faster.

Today, sqrt() operations & computers are very fast – sqrt() is as fast as generating a 2nd RN. We usually go with the more obvious direct method.

BUT, the older schemes are still commonly used in production MC codes. Learn to recognize them.
Rejection Sampling

- **Von Neumann:**
  
  "........ it seems objectionable to compute a transcendental function of a random number."

- **Select a bounding function, \( g(x) \), such that**
  - \( c \geq g(x) > f(x) \) for all \( x \)
  - \( g(x) \) is an easy-to-sample PDF

- **Sampling Procedure:**
  - sample \( x' \) from \( g(x) \): \( x' \leftarrow G^{-1}(\xi_1) \)
  - test: \( \xi_2 \leq c g(x') < f(x') \)
    - if *true*, accept \( x' \), done
    - if *false*, reject \( x' \), try again

- **Advantages**
  - Simple computer operations

- **Disadvantages**
  - "Low-level" approach, sometimes hard to understand
Rejection Sampling - Examples

- Sample from a PDF
  \[ f(x) = c \cdot \text{erf}(x), \quad 0 \leq x \leq 5. \]
  
  note: \( \text{erf}(\infty) = 1. \)

Do
  \[
  \begin{align*}
  x_{\text{try}} &= 5.*\text{rang}() \\
  f_{\text{try}} &= 1.*\text{rang}() \\
  \text{if}( f_{\text{try}} \leq \text{erf}(x_{\text{try}}) ) &\text{ exit}
  \end{align*}
  \]
Enddo

\( x = x_{\text{try}} \)

- Select \((x,y)\) points uniformly in a disk

Do
  \[
  \begin{align*}
  x &= 2.*\text{rang}() - 1. \\
  y &= 2.*\text{rang}() - 1. \\
  \text{if}( x^2 + y^2 < 1.0 ) &\text{ exit}
  \end{align*}
  \]
Enddo
Direct vs. Rejection - 2D Direction Cosines

Example — 2D Isotropic

\[ f(\hat{\mathbf{p}}) = \frac{1}{2\pi}, \quad \hat{\mathbf{p}} = (u, v) \]

Rejection \hspace{1cm} (old vim)

```fortran
SUBROUTINE AZIRN_VIM( S, C )
IMPLICIT DOUBLE PRECISION (A-H, O-Z)

100 R1=2.*RANF() - 1.
R1SQ=R1*R1
R2=RANF()
R2SQ=R2*R2
RSQ=R1SQ+R2SQ
IF(1.-RSQ)100,105,105
105 S=2.*R1*R2/RSQ
C=(R2SQ-R1SQ)/RSQ
RETURN
END
```

Direct \hspace{1cm} (racer, new vim)

```fortran
subroutine azirn_new( s, c )
imPLICIT double precision (a-h,o-z)
parameter ( twopi = 2.*3.14159265 )
phi = twopi*ranf()
C = cos(phi)
S = sin(phi)
RETURN
END
```
Isotropic Scatter – Sampling the Scattering Angle

• Consider isotropic scattering
  – Any direction is equally likely
  – Interpret as:
    "pick a random point on a unit sphere, then get direction-cosines"

• Rejection method for scatter angle sampling
  – Pick x,y,z randomly in unit cube
  – If x,y,z outside unit sphere, reject and try again
  – If x,y,z inside unit sphere, scale so that x^2+y^2+z^2 = 1
  – Get direction-cosines of angles, u,v,w

• Direct method for scatter angle sampling
  \[ f(\Omega) = \frac{1}{4\pi}, \quad d\Omega = \sin\theta \cdot d\theta \cdot d\phi \]
  \[ f(\theta,\phi) = \frac{\sin\theta \cdot d\theta \cdot d\phi}{2 \cdot 2\pi}, \quad 0 \leq \theta \leq \pi, \quad 0 \leq \phi \leq 2\pi \]
  \[ f(\theta) = \int_0^{2\pi} f(\theta,\phi) d\phi = \frac{\sin\theta}{2} \]
  \[ \mu = \cos\theta, \quad d\mu = -\sin\theta \cdot d\theta, \quad -1 \leq \mu \leq +1 \]
  \[ f(\mu) = f(\theta) \left| \frac{d\theta}{d\mu} \right| = \frac{\sin\theta}{2} \cdot \frac{1}{\sin\theta} = \frac{1}{2} \]

→ \( \mu \) is distributed uniformly in [-1,1]
→ \( \phi \) is distributed uniformly in [0,2\pi]

\[ \mu \leftarrow 2\xi_1 - 1 \]
\[ \phi \leftarrow \xi_2 \cdot 2\pi \]
## Direct Sampling – Common PDFs

<table>
<thead>
<tr>
<th>Probability Density Function</th>
<th>Direct Sampling Method</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Linear:</strong> $f(x) = 2x, \quad 0 &lt; x &lt; 1$</td>
<td>$x \leftarrow \sqrt{\xi}$</td>
</tr>
<tr>
<td><strong>Exponential:</strong> $f(x) = e^{-x}, \quad 0 &lt; x$</td>
<td>$x \leftarrow -\log\xi$</td>
</tr>
</tbody>
</table>
| **2D Isotropic:** $f(\mathbf{\hat{r}}) = \frac{1}{2\pi}, \quad \mathbf{\hat{r}} = (u, v)$ | $u \leftarrow \cos 2\pi \xi_1$  
$v \leftarrow \sin 2\pi \xi_1$ |
| **3D Isotropic:** $f(\mathbf{\hat{r}}) = \frac{1}{4\pi}, \quad \mathbf{\hat{r}} = (u, v, w)$ | $u \leftarrow 2\xi_1 - 1$  
$v \leftarrow \sqrt{1 - u^2} \cos 2\pi \xi_2$  
$w \leftarrow \sqrt{1 - u^2} \sin 2\pi \xi_2$ |
| **Maxwellian:** $f(x) = \frac{2}{T\sqrt{\pi N}} \sqrt{\frac{x}{T}} e^{-x/T}, \quad 0 < x$ | $x \leftarrow T\left(-\log\xi_1 - \log\xi_2 \cos^2 \frac{\pi \xi_3}{2}\right)$ |
| **Watt Spectrum:** $f(x) = \frac{2e^{-ab/4}}{\sqrt{\pi a^3b}} e^{-x/a} \sinh \sqrt{b}x, \quad 0 < x$ | $w \leftarrow a\left(-\log\xi_1 - \log\xi_2 \cos^2 \frac{\pi \xi_3}{2}\right)$  
$x \leftarrow w + \frac{a^2b}{4} + (2\xi_4 - 1) \sqrt{a^2b}w$ |
| **Normal:** $f(x) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2}$ | $x \leftarrow \mu + \sigma \sqrt{-2\log\xi_1} \cos 2\pi \xi_2$ |
Software Considerations

"Rules of thumb" for M.C. algorithm design have changed

- Never take the square root of a random number
- Avoid using sin, cos, log, exp, ....
- Use IF...GOTO... to avoid arithmetic
- Random numbers are cheap, arithmetic is expensive

Direct sampling methods have advantages

- Clear, succinct coding — easier to verify & maintain
- Cpu time is comparable to rejection
- Direct methods vectorize efficiently
Mean & Standard Deviation

• Given a set of random samples, \( x_1, x_2, \ldots, x_N \),

  - Mean \( \bar{x} = \frac{1}{N} \sum_{j=1}^{N} x_j \)

- Population variance & standard deviation

  \[ \sigma^2 = \frac{1}{N} \sum_{j=1}^{N} x_j^2 - \left( \frac{1}{N} \sum_{j=1}^{N} x_j \right)^2 = \frac{1}{N} \sum_{j=1}^{N} x_j^2 - \bar{x}^2 \]
  \[ \sigma = \sqrt{\frac{1}{N} \sum_{j=1}^{N} x_j^2 - \bar{x}^2} \]

- Variance & standard deviation of the mean

  \[ \sigma_{\bar{x}}^2 = \frac{\sigma^2}{N} \]
  \[ \sigma_{\bar{x}} = \frac{1}{\sqrt{N}} \cdot \sqrt{\frac{1}{N} \sum_{j=1}^{N} x_j^2 - \bar{x}^2} \]

Monte Carlo codes calculate mean values for tallies, & report the standard deviation of the mean

In the definitions above, some of the “\( N \)” terms should really be “\( N-1 \)”. MCNP & many other codes ignore that, since \( N \) is very large.
Random Sampling -- References

Every Monte Carlo code developer who works with random sampling should own & read these references:


★ = Included in References on class CD
Extra Topics
Direct vs. Rejection - Watt

Example — Watt Spectrum

\[ f(x) = \frac{2e^{-\frac{ab^4}{4}}}{\sqrt{\pi a^3 b}} x^a \sinh \sqrt{bx}, \quad 0 < x \]

Rejection *(mcnp)*

- Based on Algorithm R12 from 3rd Monte Carlo Sampler, Everett & Cashwell
- Define \( K = 1 + \frac{ab}{8} \), \( L = a \{ K + (K^2 - 1)/2 \} \), \( M = L/a - 1 \)
- Set \( x \leftarrow -\log \xi_1, \quad y \leftarrow -\log \xi_2 \)
- If \( \{ y - M(x + 1) \}^2 \leq bLx \), accept: return \( Lx \)
  otherwise, reject

Direct *(new vim)*

- Sample from Maxwellian in C-of-M, transform to lab
  \[ w \leftarrow a \left( -\log \xi_1 - \log \xi_2 \cos^2 \frac{\pi \xi_3}{2} \right) \]
  \[ x \leftarrow w + a^2 b + \frac{a^2 b x^2}{4} + (2\xi_4 - 1) \sqrt{a^2 b w} \quad \text{(assume isotropic emission from fission fragment moving with constant velocity in C-of-M)} \]
- Unpublished sampling scheme, based on original Watt spectrum derivation
Stratified Sampling

If a specific number of samples, M, is needed from a single distribution:

- Naive approach — repeat the sampling procedure M times

- **Stratified sampling** approach
  
  — partition the sample space into M disjoint regions of equal probability
  
  — produce 1 sample from each region

- Stratified sampling considerations
  
  — $F(x)$ must be known & easy to partition
  
  — The number of partitions, M, must be known in advance
  
  — Must be relatively easy to sample within each given partition
  
  — Stratification improves the "coverage"
  
  — Stratified sampling reduces variance, at little or no computing cost
Rejection Method

- Rejection sampling methods are useful when it is difficult or impossible to invert $F(x)$, or when $F(x)$ is not known.

- Example - Selection of initial source sites in a reactor, rejection method based on spatial coordinates:

  - Select a trial site:
    
    $$x' \leftarrow x_1 + (x_2 - x_1) \cdot \xi_1$$
    $$y' \leftarrow y_1 + (y_2 - y_1) \cdot \xi_2$$

  - If $(x',y')$ is inside a fuel pin (shaded region), then accept $(x',y')$.

  - Otherwise, reject $(x',y')$ and repeat.

  - Efficiency of rejection sampling
    $$\sim \frac{\text{(volume source region)}}{\text{(total volume)}}$$
Weighted Sampling

It is sometimes useful to sample from an alternate PDF

\[ f(x) \, dx = \left[ \frac{f(x)}{g(x)} \right] g(x) \, dx = h(x) \, g(x) \, dx \]

& then "correct" the result via either weight factors or a 2nd sampling stage

Weighted Sampling

- To sample \( \hat{x} \) from \( f(x) \),
  - first, sample \( \hat{x} \) from \( g(x) \)
  - then, multiply the "weight" assigned to \( \hat{x} \) by \( \frac{\text{right answer}}{\text{wrong answer}} = \frac{f(\hat{x})}{g(\hat{x})} \)

- Note that \( g(x) \) must be \( >0 \) whenever \( f(x) > 0 \).
- Also, \( g(x) \) must be normalized so that \( \int g(x) \, dx = 1 \)

- Example — survival biasing of collisions
  
  If a collision occurs, \( P_{\text{survive}} = \frac{\Sigma_S}{\Sigma_T} \) is the probability of surviving.
  
  Instead of sampling the outcome, always choose survival & multiply the "weight" by \( P_{\text{survive}} \).
Splitting & Russian Roulette

Combined Russian Rouletting & Splitting

• Russian Roulette — kill off some particles, but conserve total weight
• Splitting — create extra identical particles, but conserve total weight

• Definitions

\[ \text{wgt} = \text{ Particle weight} \]

For the region containing the particle:

\[ w_{\text{high}} = \text{ upper bound on weight, if wgt larger — split} \]
\[ w_{\text{low}} = \text{ lower bound on weight, if wgt lower — roulette} \]
\[ w_{\text{ave}} = \text{ weight to assign survivors, } w_{\text{low}} < w_{\text{ave}} < w_{\text{high}} \]

Then,

\[ \frac{\text{wgt}}{w_{\text{ave}}} = \text{ probability of surviving split/roulette} \]

• Combined game for split/roulette:

if \[ w_{\text{gt}} < w_{\text{low}} \text{ or } w_{\text{high}} < w_{\text{gt}} \],

create \( n \) particles of weight \( w_{\text{ave}} \), where

\[ n \leftarrow \left\lfloor \frac{w_{\text{gt}}}{w_{\text{ave}}} + \xi \right\rfloor \]
Random Number Generators - Reproducibility

Reproducibility of a Particle History

- use separate, distinct random sequence for each particle
- starting seeds for separate particles are separated by "stride"

- stride should be large enough to prevent overlap (for most histories)
  - 1000 is common for reactor analysis problems
    - splitting & variance reduction not needed for in-core physics
    - reduces total random number usage
  - 4,297 is the "old" default for MCNP & VIM
  - 152,917 is the default for MCNP & VIM
    - prepared for lots of splitting & variance reduction
    - potential for lots of secondary particles
Random Number Generators - Reproducibility

Parallel processing

• take "super-stride" in random sequence for particles on each processor

Eigenvalue Problems

• batches of particles, distributed among parallel processors
• seeds for each

processors for batch $m$     processors for batch $m+1$
Computational Geometry
Engineering Model vs. Computational Model

- **Model Generation**
  - Focus on engineering productivity
  - Describes “reality” to computer
  - Interactive, batch, or CAD

- **Large-scale Computation**
  - Focus on efficiency & capabilities
  - Data structures should be compact & regular
  - Computational model often hidden from user

- **Post-Processing**
  - Interpretation of results
  - Visualization
Modeling vs. Computation

- Element geometry
- Elements --> assemblies
- Assemblies --> core
- Core + peripherals --> 3D model

Geometry computation

Model construction
Monte Carlo Geometry

Geometry
- Which cell is particle in?
- What will it hit next?
- How far to boundary?
- What’s on other side?
- Survival?

Physics
- How far to collision?
- Which nuclide?
- New E, direction?
- Secondaries?
- Survival?

Tallies
- Tally events of interest
- Compute results
- Compute statistics
- Balances
- Performance stats

mcnp, rcp, vim, racer, sam, tart, morse, keno, tripoli, mcbend, monk, o5r, recap, andy, ...

Development of particular geometric capabilities is driven by applications:
- Shielding & experiment analysis
  - Irregular geometry
  - Moderate number of regions & compositions
- Reactor core analysis
  - Regular geometry
  - Very many regions & compositions
Comments on 3D Computational Geometry  (1)

- At the most fundamental level, 3D computational geometry is an exercise in Set Theory, the same concepts we all learned as kids.

Set of all x,y,z points inside box A

Set of all x,y,z points inside box B

Intersection of sets A & B

Union of sets A & B
Comments on 3D Computational Geometry (2)

- Some codes use primitive bodies (box, sphere, etc.) in defining the sets of points to consider

- Other codes use half-spaces – all the points on one side of a surface

- A little thought should convince you that either scheme can be used in set theory for defining objects in space
Computational Algorithm - Geometric View

Repeat for all $k_{\text{eff}}$ cycles
.    Repeat for all histories in cycle
.    .    Repeat while particles are left in history
.    .    .    While particle wgt > 0
.    .    .    .    Repeat until collision
.    .    .    .    .    Repeat for each universe level
.    .    .    .    .    .    Repeat for surfaces of 3D region
.    .    .    .    .    .    .    Distance calculation
.    .    .    .    .    .    .    ...  
.    .    .    .    .    .    ...  
.    .    .    .    .    .    Boundary crossing
.    .    .    .    .    .    Neighbor search
.    .    .    .    .    .    Roulette/split
.    .    .    .    .    ...  
.    .    .    .    .    .    Collision analysis
.    .    .    .    .    .    Roulette/split
.    .    .    .    ...  
.    .    ...  
.    .    Update tallies
.    ...  
.    Update tallies, source, & $k_{\text{eff}}$
...
Computational Geometry

• Every point in space that a particle could possibly reach must be defined in the geometry model – no gaps or overlaps of regions

• Cells (3D volumes) are defined by their bounding surfaces
  – Boundary representation
  – Combinatorial geometry, with either surfaces or primitive bodies
  – CSG - constructive solid geometry, tree structure with boolean operators
  – Mesh geometry
  – For some codes, disjoint volumes must have different cell numbers
  – For MCNP & others, disjoint volumes may have the same cell number

• Properties are assigned to each cell
  – Material, density, temperature, importance, etc.

• Tallies are defined for particular cells or surfaces, reaction types, & estimator types
Basic Geometry Operations

- **Locate**
  Given a point in space, determine what cell it is in

- **Distance to surface**
  Given a point & direction in a particular cell, determine the distance to the next surface of that cell

- **Neighbor search**
  For a particle which has hit a bounding surface of a cell, determine the cell to be entered next

- **Boundary conditions**
  For a particle which has hit a cell bounding surface declared to be periodic or reflecting, determine the new position & direction and cell to be entered next
Simple Case - Mesh Geometry

- **Particle**
  Position = (x,y,z),  Direction = (u,v,w)

- **Cell number**
  (i,j,k),  indices in mesh

- **Locate**
  i:  binary search to find x-interval containing x
  j:  binary search to find y-interval containing y
  k:  binary search to find z-interval containing z

- **Distance**
  - Use signs of (u,v,w) to select surfaces, then compute 3 distances:
    if u>0,  \( d_x = \frac{(x_{i+1}-x)}{u} \),  otherwise  \( d_x = \frac{(x_i-x)}{u} \)
    \( \ldots \)  similar for \( d_y \) & \( d_z \)
  - Distance:  \( d = \min( d_x, d_y, d_z ) \)

- **Neighbor search**
- **Boundary conditions**
MCNP Geometry

- **MCNP uses a "combinatorial geometry" based on surfaces**
  - Define surfaces, specify sense (which side of surface)
  - Define cells using surfaces & operators (intersection, union, complement)
  - Can also group cells together into a universe, and embed that universe inside another cell
  - Can also group cells together into a universe, repeat that universe in a lattice arrangement, and embed that universe inside another cell
  - Assign materials to cells
  - Assign other properties to cells (e.g., density, temperature, importance)
  - Define tallies using cell or surface numbers
Surfaces

• In MCNP, surface types include:
  1st order: planes
  2nd order: spheres, cylinders, cones, ellipsoid, hyperboloid, paraboloid, general quadric
  4th order: elliptical & circular torus (axes parallel to x, y, or z)

• Quadratic polynomial for surface:

\[ F(x,y,z) = ax^2 + by^2 + cz^2 + dxy + eyz + fzx + gx + hy + jz + k \]

  – Surface is defined by: \( F(x,y,z) = 0 \)
  – Surface is either infinite or closed
  – Normalization convention: factor of leading 2nd order term is positive
# MCNP Surfaces

## Table 3.1: MCNP Surface Cards

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>Type</th>
<th>Description</th>
<th>Equation</th>
<th>Card Entries</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>Plane</td>
<td>General</td>
<td>$Ax + By + Cz - D = 0$</td>
<td>ABCD</td>
</tr>
<tr>
<td>PX</td>
<td>Plane</td>
<td>Normal to $X$-axis</td>
<td>$x - D = 0$</td>
<td>D</td>
</tr>
<tr>
<td>PY</td>
<td>Plane</td>
<td>Normal to $Y$-axis</td>
<td>$y - D = 0$</td>
<td>D</td>
</tr>
<tr>
<td>PZ</td>
<td>Plane</td>
<td>Normal to $Z$-axis</td>
<td>$z - D = 0$</td>
<td>D</td>
</tr>
<tr>
<td>SO</td>
<td>Sphere</td>
<td>Centered at Origin</td>
<td>$x^2 + y^2 + z^2 - R^2 = 0$</td>
<td>$R$</td>
</tr>
<tr>
<td>S</td>
<td>Sphere</td>
<td>General</td>
<td>$(x - \bar{x})^2 + (y - \bar{y})^2 + (z - \bar{z})^2 - R^2 = 0$</td>
<td>$\bar{x} \bar{y} \bar{z} R$</td>
</tr>
<tr>
<td>SX</td>
<td>Cylinder</td>
<td>Centered on $X$-axis</td>
<td>$(x - \bar{x})^2 + y^2 + z^2 - R^2 = 0$</td>
<td>$\bar{x} R$</td>
</tr>
<tr>
<td>SY</td>
<td>Cylinder</td>
<td>Centered on $Y$-axis</td>
<td>$x^2 + (y - \bar{y})^2 + z^2 - R^2 = 0$</td>
<td>$\bar{y} R$</td>
</tr>
<tr>
<td>SZ</td>
<td>Cylinder</td>
<td>Centered on $Z$-axis</td>
<td>$y^2 + y^2 + (z - \bar{z})^2 - R^2 = 0$</td>
<td>$\bar{z} R$</td>
</tr>
<tr>
<td>C/X</td>
<td>Cylinder</td>
<td>Parallel to $X$-axis</td>
<td>$(y - \bar{y})^2 + (z - \bar{z})^2 - R^2 = 0$</td>
<td>$\bar{y} \bar{z} R$</td>
</tr>
<tr>
<td>C/Y</td>
<td>Cylinder</td>
<td>Parallel to $Y$-axis</td>
<td>$(x - \bar{x})^2 + (z - \bar{z})^2 - R^2 = 0$</td>
<td>$\bar{x} \bar{z} R$</td>
</tr>
<tr>
<td>C/Z</td>
<td>Cylinder</td>
<td>Parallel to $Z$-axis</td>
<td>$(x - \bar{x})^2 + (y - \bar{y})^2 - R^2 = 0$</td>
<td>$\bar{x} \bar{y} R$</td>
</tr>
<tr>
<td>CX</td>
<td></td>
<td>On $X$-axis</td>
<td>$y^2 + z^2 - R^2 = 0$</td>
<td>$R$</td>
</tr>
<tr>
<td>CY</td>
<td></td>
<td>On $Y$-axis</td>
<td>$x^2 + z^2 - R^2 = 0$</td>
<td>$R$</td>
</tr>
<tr>
<td>CZ</td>
<td></td>
<td>On $Z$-axis</td>
<td>$x^2 + y^2 - R^2 = 0$</td>
<td>$R$</td>
</tr>
</tbody>
</table>
## MCNP Surfaces

<table>
<thead>
<tr>
<th>K/X</th>
<th>Cone</th>
<th>Parallel to X-axis</th>
<th>(\sqrt{(y - y')^2 + (z - z')^2 - t(x - x')} = 0)</th>
<th>(\bar{x} , \bar{y} , \bar{z} , t^2 \pm 1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>K/Y</td>
<td>Parallel to Y-axis</td>
<td>(\sqrt{(x - x')^2 + (z - z')^2 - t(y - y')} = 0)</td>
<td>(\bar{x} , \bar{y} , z , t^2 \pm 1)</td>
<td></td>
</tr>
<tr>
<td>K/Z</td>
<td>Parallel to Z-axis</td>
<td>(\sqrt{(x - x')^2 + (y - y')^2 - t(z - z')} = 0)</td>
<td>(\bar{x} , \bar{y} , \bar{z} , t^2 \pm 1)</td>
<td></td>
</tr>
<tr>
<td>KX</td>
<td>On X-axis</td>
<td>(\sqrt{y^2 + z^2 - t(x - x')} = 0)</td>
<td>(\bar{x} , t^2 \pm 1)</td>
<td></td>
</tr>
<tr>
<td>KY</td>
<td>On Y-axis</td>
<td>(\sqrt{x^2 + z^2 - t(y - y')} = 0)</td>
<td>(\bar{y} , t^2 \pm 1)</td>
<td></td>
</tr>
<tr>
<td>KZ</td>
<td>On Z-axis</td>
<td>(\sqrt{x^2 + y^2 - t(z - z')} = 0)</td>
<td>(\bar{z} , t^2 \pm 1)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SQ</th>
<th>Ellipsoid</th>
<th>(A(x - x')^2 + B(y - y')^2 + C(z - z')^2 + 2D(x - x') + 2E(y - y') + 2F(z - z') + G = 0)</th>
<th>A B C D E</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Hyperboloid</td>
<td></td>
<td>F G</td>
</tr>
<tr>
<td></td>
<td>Paraboloid</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GQ</th>
<th>Cylinder</th>
<th>(Ax^2 + By^2 + Cz^2 + Dxy + Eyz + Fzx + Gx + Hy + Jz + K = 0)</th>
<th>A B C D E</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Cone</td>
<td></td>
<td>F G H J K</td>
</tr>
<tr>
<td></td>
<td>Ellipsoid</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Hyperboloid</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Paraboloid</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TX</th>
<th>Elliptical torus.</th>
<th>(\frac{(x - x')^2}{B^2} + \frac{(y - y')^2 + (z - z')^2 - A}{C^2} - 1 = 0)</th>
<th>(\bar{x} , \bar{y} , \bar{z} , A , B , C)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Axis is parallel to X-, Y-, or Z-axis</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TY</th>
<th>(\frac{(y - y')^2}{B^2} + \frac{(x - x')^2 + (z - z')^2 - A}{C^2} - 1 = 0)</th>
<th>(\bar{x} , \bar{y} , \bar{z} , A , B , C)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Axis is parallel to X-, Y-, or Z-axis</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TZ</th>
<th>(\frac{(z - z')^2}{B^2} + \frac{(x - x')^2 + (y - y')^2 - A}{C^2} - 1 = 0)</th>
<th>(\bar{x} , \bar{y} , \bar{z} , A , B , C)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Axis is parallel to X-, Y-, or Z-axis</td>
<td></td>
</tr>
</tbody>
</table>

| X Y Z P | Surfaces defined by points | See pages 3–15 and 3–17 |
Sense

- For a given point in space, \((x,y,z)\), and surface equation, \(F(x',y',z')=0\), the sense of the point with respect to the surface is defined as:

  Inside the surface,  \(\text{sense} < 0\), if  \(F(x,y,z) < 0\)

  Outside the surface, \(\text{sense} > 0\), if  \(F(x,y,z) > 0\)

  On the surface, \(\text{sense} = 0\), if  \(F(x,y,z) = 0\)

  [Must be careful to consider computer roundoff!]

  ![Diagram showing sense notation]

  Note: Outward surface normal points in direction of  + sense
Sides

- **A surface divides space into positive & negative sides**

  - **MCNP convention:**
    - +1 = positive side of surface 1
    - -1 = negative side of surface 1

  - If not sure which side is + or -, pick a point & substitute into surface function, \( F(x,y,z) \) -- see if result is + or -

\[
\begin{array}{c|c|c|c}
\text{-1} & \text{+1} & \text{+2} & \text{+3} \\
\hline
\text{-2} & & & \\
\end{array}
\]
Intersection & Union of Sides

MCNP convention:
Blank signifies intersection

\[ +1 \, -2 = \text{intersection of} \]
\[ +\text{side of surface 1 and} \]
\[ -\text{side of surface 2} \]

MCNP convention:
Colon signifies union

\[ -1 : 2 = \text{union of} \]
\[ -\text{side of surface 1 with} \]
\[ +\text{side of surface 2} \]
Cells

- **A cell is defined to be the**
  - Intersection of half-spaces defined by a list of signed surface numbers
    
    Example: 
    
    \[
    \begin{align*}
    \text{cell 10} & : -5 \\
    \text{cell 20} & : +1 -2 +3 -4 +5 \\
    \end{align*}
    \]

- **Union of half-spaces defined by signed surface numbers**
  
  Example: 
  
  \[
  \text{cell 43} \ : \ +1 : -2
  \]

- **The complement of another cell** (i.e., volume NOT in other cell)
  
  Example: 
  
  \[
  \text{cell 30} \ : \ #50
  \]

- **A combination of the above**
  
  Example: 
  
  \[
  \text{cell 57} \ : \ (+1 -2) : 3 \ : \ #50
  \]
Cells

- Cells do not have to be convex, & can be reentrant

\(-3 +1 : -3 -2 -1\)

or

\(-3 +1 : -3 -2\)

or

\(-3 (+1 : -2)\)

- Cells may involve discontiguous regions

Cell 10: -1 : -2 : -3

Cell 20: #10

or 1 2 3
Locate Operations

Given point \((x,y,z)\), determine which cell it is contained in:

\[
\text{For( } \text{cell } = 1 \ldots \text{n}\_\text{cells} \text{ ) } \{ \\
\quad \text{Foreach } \text{surf in cell } \{ \\
\quad \quad \text{Evaluate } S_{\text{surf}} = \text{sign}\{ F_{\text{surf}}(x,y,z) \} \\
\quad \quad \text{Does } S_{\text{surf}} \text{ match the sense from the cell definition?} \\
\quad \} \\
\quad \text{If all surface-senses for } (x,y,z) \text{ matched the cell definition,} \\
\quad \quad \text{then exit & return cell as the result} \\
\} 
\]
Distance Calculation

Given point \((x,y,z)\) in cell I, determine the distance to the cell boundary

\[
d \leftarrow \text{infinity}
\]

Foreach surf in cell I {
    If surf is part of the external boundary of cell I {
        Evaluate \(d_{\text{surf}} = \text{smallest positive root of } F_{\text{surf}}(x+du, y+dv, z+dw) = 0\)
        \[
d = \min(d, d_{\text{surf}})
        \]
    }
}
return the value of \(d\)
Neighbor Search

• When a cell boundary is reached, what's on the other side?

<table>
<thead>
<tr>
<th>Easy case</th>
<th>Hard case</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

• Most codes build "neighbor lists" during tracking
  – For each bounding surface of cell, remember list of neighbors
  – Initially, neighbor lists are empty
  – Check all cells having surface in common, until one is found satisfying all sense conditions for the particle position
  – Save it
  – Later, check neighbor lists first, only do search if necessary

• Neighbor search is expensive at first, cheap later

• Tracking speeds up as calculation progresses
Embedded Geometry - Universes

- In most real-world applications, there is a need for modeling detailed geometry with many repeating units

- All production Monte Carlo codes provide capabilities for multiple levels of nested geometry
  - Called "universes" in MCNP
  - A collection of cells may be grouped into a "universe"
  - Universe may be embedded in another cell, with the universe 'clipped' by the cell boundaries
Universes & Lattices

- **Universe 1** - cells for detailed fuel pin
- **Universe 2** - lattice of cells for fuel assembly
- **Universe 2, with cells filled by Universe 1**
- **Universe 3** - lattice of cells for reactor
- "Real world" - final geometry
Body Geometry

- Some Monte Carlo codes use primitive bodies rather than surfaces for defining cells (e.g., MORSE, KENO, ITS, VIM)

  - SPH - sphere
  - BOX - box
  - RPP - box
  - RCC - cylinder
  - WED - wedge
  - ELL - ellipsoid
  - REC - right elliptic cylinder
  - RHP - hexagonal prism
  - HEX - hexagonal prism
  - ARB - arbitrary polyhedron
  - TRC - truncated cone

- Usually called "combinatorial geometry"
  - Invented by MAGI in ~1956, used in SAM-CE & other codes
  - Space inside the body has a negative sense, outside a positive sense
  - Boolean operators AND, OR, NOT may be used to combine bodies (like MCNP's intersection, union, & complement operators)
  - MCNP allows body geometry input (calls them "macrobodies"), but internally converts them to lists of surfaces
Special Topics - Simple Cells

- **Simple cells** are those which can be constructed using only intersections, with no union operators.

- Some Monte Carlo codes require that all cells be simple cells. Union operators are not allowed.

- Tracking through simple cells is fast, at the expense of more complex geometry input & setup.
  - For simple cells, the logic to find the distance to boundary is simple - check the distance to each of the cell surfaces & keep only the smallest positive distance.
Consider the example at the left.

Using the union operator, the cell is described by: \[ +1 : -2 \]

Without the union operator, separate cells must be defined & then assigned the same material properties:

\[ +1, -1 -2 \]

or \[ -2, +1 +2 \]

or \[ +1 -2, +1 +2, -1 -2 \]
Special Topics - Distance Calculations

- **3D Surface**
  - \( F(x,y,z) = 0 \)
    - Linear: \( \nabla F = \text{constant} \)
    - Quadratic: \( \nabla F = f(x,y,z), \quad \nabla^2 F = \text{constant} \)

- **Distance calculation**
  - \( S = \) directed distance from \((x_0,y_0,z_0)\) along \((u,v,w)\) to \(F(x,y,z)=0\)
    = smallest positive root of \( F(x_0+su, y_0+sv, z_0+sw) = 0 \)
  - **General form:** \( As^2 + 2Bs + C = 0, \quad D = B^2 - AC \)
    - 27 combinations of \( A, B, C >0, =0, <0 \)
    - Only 12 yield valid solutions:
      - \( s = -C/(2B) \) if \((A=0, C<0, B>0)\) or \((A=0, C>0, B<0)\)
      - \( s = (-B-\sqrt{D})/A \) if \((A>0, C>0, B<0, D>0)\) or \((A<0, C>0, B>0, D>0)\) or \((A<0, C>0, B=0, D>0)\) or \((A>0, C<0, B>0, D>0)\) or \((A>0, C=0, B<0, D>0)\) or \((A<0, C=0, B>0, D>0)\)
      - \( s = (-B+\sqrt{D})/A \) if \((A>0, C<0, B>0, D>0)\) or \((A>0, C<0, B<0, D>0)\) or \((A<0, C<0, B>0, D>0)\)
      - \( s = \infty \) otherwise
Special Topics - Distance Calculations

- Noting that \( C = F(x_0, y_0, z_0) \) = sense at \( (x_0, y_0, z_0) \), the valid solutions can be simplified using the known surface sense \( \mathcal{S} \):

\[
\begin{align*}
\text{s'} &= -C/(2B) \quad \text{if} \quad (A=0, \ D>0) \\
\text{s'} &= (-B - \sqrt{D})/A \quad \text{if} \quad (A \neq 0, \ D>0, \ \mathcal{S} >0) \\
\text{s'} &= (-B + \sqrt{D})/A \quad \text{if} \quad (A \neq 0, \ D>0, \ \mathcal{S} <0) \\
\text{s'} &= \infty \quad \text{otherwise}
\end{align*}
\]

And

\[
\begin{align*}
\text{s} &= \text{s'} \quad \text{if} \ s'>0 \\
&= \infty \quad \text{otherwise}
\end{align*}
\]
Special Topics - Common Surfaces

• If 2 surfaces coincide, neighbor searches become more complicated & tracking can slow down significantly.

• Most MC codes check for coincident surfaces & eliminate one of them (replacing it by the other).

• The tolerance for coincident surfaces usually defaults to a small separation distance (e.g., 1.e-4 cm). For problems with unusual geometry (very small or very large), this may have to be changed in the code or code input.
Stochastic Geometry & HTGR Modeling (optional)
Introduction

• Much interest lately in analyzing HTGRs
  – Fuel kernels with several layers of coatings
  – Very high temperatures
  – Contain fission products
  – Safety aspects …

• Double heterogeneity problem
  – Fuel kernels randomly located within fuel elements
  – Fuel elements may be "compacts" or "pebbles" (maybe random)
  – Challenging computational problem

• Monte Carlo codes can faithfully model HTGRs
  – Full 3D geometry
  – Multiple levels of geometry, including embedded lattices
  – Random geometry ?????
Example - Very High Temperature Gas Cooled Reactor

TRISO Coated fuel particles (left) are formed into fuel rods (center) and inserted into graphite fuel elements (right).

Example - GT-MHR Modeling


Fig. 2. A cross-section of the GT-MHR homogeneous core with hexagonal structure (HTR1).

Fig. 3. Fragments of single-heterogeneous GT-MHR (HTR2): (A) an active core structure: three rings of hexagonal fuel columns; (B) magnified view of a separate fuel assembly. Fuel compacts are presented in small grey circles, burnable poison compacts in light grey. Bigger diameter holes stand for He channels, while the rest material represents the graphite matrix.

Fig. 4. Fragments of double-heterogeneous GT-MHR (HTR3): (A) fuel element (compact) cross section with coated fuel particles; (B) magnified view of coated fuel particles: spherical kernels of PuO$_2$ are surrounded by protective coatings made of PyC$_{n+}$, PyC I, SiC and PyC II layers correspondingly. The same structure is valid for particles containing burnable poison—natural Eu$_2$O$_3$. 
Example - Pebble Bed Experiments at Proteus Facility


Fig. 3. Cross section of the odd layers of the hex configurations, case 4 with polyethylene rods. The fit with the visualization tools of the MCNP program.

Fig. 5. Case 4, vertical cross section. The black pebble (fuel type) appears magnified in Figs. 6, 7, and 8 to show the heterogeneous details of the fuel pebble.

Fig. 6. Magnified fuel pebble of Fig. 5.

Fig. 7. Details of the cubic lattice of fuel kernels inside the pebble of Fig. 5.

Fig. 8. Fuel kernel with the four coatings at each location of the cubic lattice and inside the fuel region of the pebble shown in Fig. 6.
MCNP Models for HTGRs

• Existing MCNP geometry can handle:
  – 3D description of core
  – Fuel compacts or lattice of pebbles
    • Typically, hexagonal lattice with close-packing of spherical pebbles
    • Proteus experiments: ~ 5,000 fuel pebbles
      ~ 2,500 moderator pebbles
  – Lattice of fuel kernels within compact or pebble
    • Typically, cubic lattice with kernel at center of lattice element
    • Proteus experiments: ~ 10,000 fuel kernels per pebble
      ~ 50 M fuel kernels, total
  – Could introduce random variations in locations of a few thousand cells in MCNP input, but not a few million.
  – See papers by: Difilippo, Plukiene et al, Ji-Conlin-Martin-Lee, etc.
MCNP5 Stochastic Geometry

• When a neutron enters a new lattice element, a transformation is made to the neutron's position & direction to the local coordinates of the universe embedded in that lattice element. [standard MCNP]

• Users can flag selected universes as "stochastic" [new]
  – A neutron entering a lattice element containing a stochastic universe undergoes the normal transformations.

  – Then, additional random translations are made:

    \[
    \begin{align*}
    x & \leftarrow x + (2\xi_1 - 1) \cdot \delta_x \\
    y & \leftarrow y + (2\xi_2 - 1) \cdot \delta_y \\
    z & \leftarrow z + (2\xi_3 - 1) \cdot \delta_z
    \end{align*}
    \]

  – Then, tracking proceeds normally, with the universe coordinates fixed until the neutron exits that lattice element
MCNP5 Stochastic Geometry

- Neutron on lattice edge, about to enter embedded universe

- Embedded universe,
  before random translation   after random translation

- Track normally, until neutron exits the lattice element
MCNP5 Stochastic Geometry

- On-the-fly random translations of embedded universes in lattice
  - Does not require any extra memory storage
  - Very little extra computing cost - only 3 random numbers for each entry into a stochastic universe

- For K-effective calculations (KCODE problems)
  - If fission occurred within fuel kernel, should have source site in next cycle be at same position within fuel kernel
  - Need to save $\delta_x$, $\delta_y$, $\delta_z$ along with neutron coordinates in fission bank
  - On source for next cycle, apply $\delta_x$, $\delta_y$, $\delta_z$ after neutron pulled from bank

- To preserve mass exactly, rather than on the average stochastically, must choose $\delta_x$, $\delta_y$, $\delta_z$ so that fuel kernels are not displaced out of a lattice element
Numerical Results -- HTGR Fuel Kernels

- Infinite array of TRISO fuel kernels in graphite matrix
  - Fuel kernel geometry & composition taken from the NGNP Point Design (MacDonald et al. 2003)

TRISO Fuel Kernel Geometry and Composition

<table>
<thead>
<tr>
<th>Region #</th>
<th>Name</th>
<th>Outer radius $\mu$m</th>
<th>Composition</th>
<th>Density (g/cc)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Uranium oxycarbide</td>
<td>175</td>
<td>UCO ($\text{UC}^{0.5}\text{O}^{1.5}$)</td>
<td>10.5</td>
</tr>
<tr>
<td>2</td>
<td>Porous carbon buffer</td>
<td>275</td>
<td>C</td>
<td>1.0</td>
</tr>
<tr>
<td>3</td>
<td>Inner pyrolytic carbon</td>
<td>315</td>
<td>C</td>
<td>1.9</td>
</tr>
<tr>
<td>4</td>
<td>Silicon carbide</td>
<td>350</td>
<td>SiC</td>
<td>3.2</td>
</tr>
<tr>
<td>5</td>
<td>Outer pyrolytic carbon</td>
<td>390</td>
<td>C</td>
<td>1.9</td>
</tr>
</tbody>
</table>

- Calculations run 4 ways:
  1. Fixed lattice with centered kernels
  2. Fixed lattice with random kernels  [MCNP stochastic geometry]
  3. Multiple lattice realizations
  4. Box of randomly place kernels
Calculations - Case #1

- **Fixed lattice with centered kernels**
  - 5x5x5 cubical lattice
  - Lattice edge chosen to preserve the specified packing fraction.
  - Fuel kernels centered within the cubical cells
  - Reflecting boundaries on the outer surfaces
  - Essentially same as Difilipo, Plukiene et al, Ji-Conlin-Martin-Lee
  - No random geometry, standard MCNP5 calculations
Calculations - Case #2

- Fixed lattice with random kernels [MCNP stochastic geometry]
  - 5x5x5 cubical lattice
  - Lattice edge chosen to preserve the specified packing fraction.
  - Fuel kernels randomly placed on-the-fly within the cubical cells
  - Reflecting boundaries on the outer surfaces
  - Uses new MCNP5 stochastic geometry

Fuel kernel displaced randomly within lattice element each time that neutron enters
Calculations - Case #3

- **Multiple lattice realizations**
  - 5x5x5 cubical lattice
  - Lattice edge chosen to preserve the specified packing fraction.
  - Fuel kernels randomly placed in job input within the cubical cells
  - Reflecting boundaries on the outer surfaces
  - Uses standard MCNP5
  - 25 separate calculations, each with different location of kernels in the input files

1 realization, fixed lattice with kernel locations chosen randomly in problem input & held constant during each MCNP calculation
Calculations - Case #4

• **Box of randomly placed fuel kernels**
  – Single box with 125 fuel kernels
  – Box size chosen to preserve the specified packing fraction.
  – Fuel kernels randomly placed in job input within the box (using RSA algorithm, Random Sequential Addition)
  – Reflecting boundaries on the outer surfaces
  – Uses standard MCNP5
  – 25 separate calculations, each with different location of kernels in the input files

2 different realizations of "truly random" cases:
# Numerical Results

## MCNP5 Results for Infinite Lattices of Fuel Kernels

<table>
<thead>
<tr>
<th>#</th>
<th>Method</th>
<th>K-effective</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Fixed 5x5x5 lattice with centered spheres</td>
<td>1.1531 ± 0.0004</td>
</tr>
<tr>
<td>2</td>
<td>Fixed 5x5x5 lattice with randomly located spheres (&quot;on the fly&quot;)</td>
<td>1.1515 ± 0.0004</td>
</tr>
<tr>
<td>3</td>
<td>Multiple (25) realizations of 5x5x5 lattice with randomly located spheres</td>
<td>1.1513 ± 0.0004</td>
</tr>
<tr>
<td>4</td>
<td>Multiple (25) realizations of randomly packed (RSA) fuel &quot;box&quot;</td>
<td>1.1510 ± 0.0003</td>
</tr>
</tbody>
</table>
Conclusions

• The new stochastic geometry treatment for MCNP5 provides an accurate and effective means of modeling the particle heterogeneity in TRISOL particle fuel
  – Same results as (brute-force) multiple realizations of random geometry input with standard MCNP
  – Negligible difference from "truly random" multiple realizations

• The results indicate that:
  – The neutronic effect of using a fixed lattice is negligible
  – The effect of choosing either a centered spheres or randomly located spheres is also small, at least for the specific fuel geometry that was analyzed during this study

• Future work
  – Examination of finite geometries, including cylindrical fuel compacts, hexagonal fuel blocks, and full core configurations.
  – We will also consider lattices other than simple cubic lattices, such as BCC, FCC, and HCP lattices.
References - HTGR Models & Stochastic Geometry

References - HTGR Models & Stochastic Geometry


Collision Physics
Monte Carlo Calculations

- **Geometry**
  - Which cell is particle in?
  - What will it hit next?
  - How far to boundary?
  - What’s on other side?
  - Survival?

- **Physics**
  - How far to collision?
  - Which nuclide?
  - New E, direction?
  - Secondaries?
  - Survival?

- **Tallies**
  - Tally events of interest
  - Compute results
  - Compute statistics
  - Balances
  - Performance stats

- **mcnp, rcp, vim, racer, sam-ce, tart, morse, keno, tripoli, mcbend, monk, o5r, recap, andy,…**

- **Geometry routines** determine the cell & material in that cell

- **Collision routines** model the physical interactions with the material
  - Random sampling from PDFs determined by cross-section data
    - Continuous: flight distance, exit E & direction, …
    - Discrete: select nuclide, select interaction type, secondaries, …
Monte Carlo Calculations

- After a particle emerges from source or collision, or if the particle is on a cell bounding surface:
  - Randomly sample the free-flight distance to the next interaction
  - If the distance-to-interaction is less than the distance to cell boundary, then move the particle to the interaction point

- Collision physics at the interaction point:
  - Determine which isotope the interaction is with
  - Determine which reaction type for that isotope
  - Determine the exit energy & direction of the particle
  - Determine if secondary particles were produced
  - Biasing + weight adjustments
  - Tallies of quantities of interest
Collision Physics

Free-flight distance to next collision, s

Collision isotope, Reaction type, Exit E' & (u',v',w'), Secondary particles
Sampling the Flight Distance

- Given a particle at \((x_0,y_0,z_0)\) with direction \((u,v,w)\) in cell I containing material M, sample the free-flight distance to the next interaction

\[ \Sigma_T = \text{total macroscopic cross-section in material M} \]
\[ = \text{sum} \{ N^k \sigma^k \}, \quad \text{where } k = \text{isotopes in material M} \]
\[ = \text{probability of any interaction per unit distance, units cm}^{-1} \]

- PDF for flight distance \(s\), where \(0 \leq s \leq \infty\),

\[ f(s) = \{\text{prob interaction p.u.d}\} \cdot \{\text{prob travelling dist s w/o interact}\} \]
\[ f(s) = \Sigma_T \exp(-\Sigma_T s) \]

- Sampling procedure

\[ F(s) = 1 - \exp(-\Sigma_T s) \quad \rightarrow \quad s = -\ln(\xi) / \Sigma_T \]
Selecting the Collision Isotope

- \[ \Sigma_T = \sum_j N^{(j)} \sigma^{(j)}_T \] where \( j = \) isotopes in material \( M \)

- Probability that collision is with isotope \( k \)

\[ p_k = \frac{N^{(k)} \sigma^{(k)}_T}{\Sigma T} \]

- \( \{ p_k \} = \) set of discrete probabilities for selecting collision isotope

- \( \{ P_k \} = \) discrete CDF, \( P_k = \text{sum}\{ p_i, i=1,k \}, \quad P_0=0 \)

- **Discrete sampling** for collision isotope \( k \)
  - table search to determine \( k \) such that \( P_{k-1} \leq \xi < P_k \)
Selecting the Reaction Type

• For collision isotope k,

\[ \sigma_T = \sigma_{\text{elastic}} + \sigma_{\text{inelastic}} + \sigma_{\text{capture}} + \sigma_{\text{fission}} + \ldots. \]

• \( p_j = \frac{\sigma_j}{\sigma_T} = \text{probability of reaction type } j \text{ for isotope } k \)

• \{ p_j \} = \text{set of discrete probabilities for selecting reaction type } j

• \{ P_j \} = \text{discrete CDF, } P_j = \text{sum} \{ p_i, \text{ i=1,j } \}, \quad P_0=0

• Discrete sampling for reaction type \( j \)
  
  table search to determine \( j \) such that \( P_{j-1} \leq \xi < P_j \)
Selecting the Reaction Type - Modified

• In many applications, survival biasing is an effective variance reduction technique
  – Survival biasing is also called implicit absorption, nonabsorption weighting, or (loosely) implicit capture
  – $\Sigma_T = \Sigma_{\text{absorption}} + \Sigma_{\text{scatter}}$ (absorption = disappearance)
  – Probability that particle survives collision = $P_{\text{surv}} = \Sigma_{\text{scatter}}/\Sigma_T$
  – Probability that particle is absorbed (killed) = $1 - P_{\text{surv}}$

• Disallow absorption of particle, & then adjust particle weight to ensure a fair game
  – Tally absorption of $wgt \cdot (1 - P_{\text{surv}})$
  – Multiply particle weight by $P_{\text{surv}}$
  – When selecting collision isotope: use $\Sigma_S$'s, not $\Sigma_T$'s for isotopes
  – When selecting reaction type: don't include $\sigma_A$
Sampling Exit Energy & Direction

- Given a collision isotope $k$ & reaction type $j$, the random sampling techniques used to determine the exit energy and direction, $E'$ and $(u',v',w')$, depend on
  - Conservation of energy & momentum
  - Scattering laws - either equations or tabulated data

- **Examples**
  - Isotropic scattering in lab system [example on next slides]
  - Multigroup scattering [example on next slides]
  - Elastic scattering, target-at-rest [example on next slides]
  - Inelastic scattering, MCNP
  - Other collision physics, MCNP
Isotropic Scatter in Lab System

- Elastic scattering from infinite-mass target nucleus
  - No change in energy:
    \[ E' = E \]
  - Sample direction from isotropic scattering PDF, \( f(u',v',w') = \frac{1}{4\pi} \):
    \[ \varphi = 2\pi \xi_1 \leftarrow \text{uniform azimuthal} \]
    \[ u' = 2\xi_2 - 1 \leftarrow \text{isotropic, uniform in } u' \]
    \[ v' = \sqrt{1-u'^2} \cos(\varphi) \]
    \[ w' = \sqrt{1-u'^2} \sin(\varphi) \]
Isotropic Scatter – Sampling the Scattering Angle

• **Consider isotropic scattering**
  – Any direction is equally likely
  – Interpret as:
    "pick a random point on a unit sphere, then get direction-cosines"

• **Rejection method for scatter angle sampling**
  – Pick x,y,z randomly in unit cube
  – If x,y,z outside unit sphere, reject and try again
  – If x,y,z inside unit sphere, scale so that \( x^2 + y^2 + z^2 = 1 \)
  – Get direction-cosines of angles, u,v,w

• **Direct method for scatter angle sampling**

  \[
  f(\hat{\Omega}) = \frac{1}{4\pi}, \quad \frac{d\hat{\Omega}}{4\pi} = \frac{\sin \theta \cdot d\theta \cdot d\phi}{2 \cdot 2\pi} \]

  \[
  f(\theta,\phi) = \frac{\sin \theta \cdot d\theta \cdot d\phi}{2 \cdot 2\pi}, \quad 0 \leq \theta \leq \pi, \quad 0 \leq \phi \leq 2\pi \]

  \[
  f(\theta) = \int_0^{2\pi} f(\theta,\phi) d\phi = \frac{\sin \theta}{2} \]

  \[
  \mu = \cos \theta, \quad d\mu = -\sin \theta \cdot d\theta, \quad -1 \leq \mu \leq +1 \]

  \[
  f(\mu) = f(\theta) \left| \frac{d\theta}{d\mu} \right| = \frac{\sin \theta}{2} \cdot \frac{1}{\sin \theta} = \frac{1}{2} \]

  \[ \Rightarrow \mu \text{ is distributed uniformly in } [-1,1] \]

  \[ \Rightarrow \phi \text{ is distributed uniformly in } [0,2\pi] \]

  \[
  \mu \leftarrow 2\xi_1 - 1 \\
  \phi \leftarrow \xi_2 \cdot 2\pi
  \]
Multigroup Scattering

• **Multigroup approach**
  – Divide energy range into intervals (groups)
  – Use average cross-sections for each group: \( \Sigma_{Tg}, \Sigma_{Sg}, \Sigma_{Ag}, \nu\Sigma_{Fg} \)
  – Use discrete transfer matrix for group-to-group scatter, 
    \( \Sigma_{gg'} = \text{cross-section for scatter from group } g \text{ to group } g' \)
    \[
    \Sigma_{Sg} = \sum_{k=1}^{G} \Sigma_{g\rightarrow k}
    \]

• **Multigroup scattering**
  – For particle with energy \( E \), determine initial energy group \( g \)
  – Select exit energy group \( g' \) by discrete sampling from \( \Sigma_{gg'} \)
    \[
    p_{g'} = \frac{\Sigma_{g\rightarrow g'}}{\Sigma_{Sg}}
    \]
  – Sample exit energy uniformly within bound of group \( g' \)
  – **Direction**
    • For \( P_0 \) scattering - use procedure for isotropic lab scatter
    • For \( P_1 \) scattering - sample \( \mu \) from linear PDF, then select new direction
      (see next section on elastic scatter)
Elastic Scattering, Target-at-rest

- Sample $\mu_{cm}$ from tabulated PDF data, $f(\mu_{cm})$
- Use kinematics to get $E'_{lab}$ & $\mu_{lab}$
- Sample azimuthal angle $\phi$ uniformly on $(0,2\pi)$
- Rotate particle direction using $\mu_{lab}$ & $\phi$

$\mu_{cm} = \cos \theta_{cm}$
$\mu_{lab} = \cos \theta_{lab}$

$\theta = \text{scattering angle, cm or lab}$
Sampling the Scattering Direction-cosine, $\mu_{cm}$

- Typical representations for $f(\mu_{cm})$
  - Histogram or Equiprobable Histogram PDF
  
  ![Histogram PDF](image1)
  
  - Piecewise linear PDF
  
  ![Piecewise linear PDF](image2)
Elastic Scatter - $E'$ & $\mu_{\text{lab}}$

- **Target-at-rest elastic scatter in lab system – kinematics**
  (from conservation of energy & momentum)

\[
E' = E \cdot \frac{A^2 + 2A\mu_{\text{cm}} + 1}{(A+1)^2}
\]

\[
\mu_{\text{lab}} = \frac{1 + A\mu_{\text{cm}}}{\sqrt{A^2 + 2A\mu_{\text{cm}} + 1}}
\]

Where $A = \frac{\text{(mass target)}}{\text{(mass particle)}}$
Exit Direction

- Rotation from $(u,v,w)$ to $(u',v',w')$ using $\mu_{\text{lab}}$ & $\varphi$

$$\mu = \mu_{\text{lab}}$$

$$\phi = 2\pi \xi$$

$$u' = \mu u + \frac{\sqrt{1-\mu^2} (uw \cos \phi - v \sin \phi)}{\sqrt{1-w^2}}$$

$$v' = \mu v + \frac{\sqrt{1-\mu^2} (vw \cos \phi + u \sin \phi)}{\sqrt{1-w^2}}$$

$$w' = \mu w - \sqrt{1-\mu^2} \sqrt{1-w^2} \cos \phi$$

If $\mu$ close to 1, special coding may be used to avoid roundoff
Inelastic Scattering - MCNP

- Law 1  ENDF law 1 - Equiprobable energy bins
- Law 2  Discrete photon energies
- Law 3  ENDF law 3 - Inelastic scatter from nuclear levels
- Law 4  ENDF law 4 - Tabular distribution
- Law 5  ENDF law 5 - General evaporation spectrum
- Law 7  ENDF law 7 - Simple Maxwell fission spectrum
- Law 9  ENDF law 9 - Evaporation spectrum
- Law 11 ENDF law 11 - Energy dependent Watt spectrum
- Law 22 UK law 2 - Tabular linear functions of incident energy out
- Law 24 UK law 6 - Equiprobable energy multipliers
- Law 44 ENDF law 1, lang 2, Kalbach-87 correlated energy-angle scatter
- Law 61 ENDF law 11, lang 0,12, or 14 - correlated energy-angle scatter
- Law 66 ENDF law 6 - N-body phase space distribution
- Law 67 ENDF law 7 - correlated energy-angle scatter
Other Collision Physics - MCNP

- Emission from fission
- Delayed neutron emission
- S(α,β) scattering for thermal neutrons
- Free-gas scattering for neutrons
- Probability tables for the unresolved resonance energy range for neutrons
- Photoelectric effect
- Pair production
- Compton scattering (incoherent)
- Thomson scattering (coherent)
- Fluorescent emission
- Photonuclear reactions

- Electron interactions - condensed history approach
  - Stopping power, straggling, angular deflections
  - Bremsstrahlung
  - K-shell impact ionization & Auger transitions
  - Knock-on electrons
Secondary Particle Creation

• Consider a collision which results in fission
  \[ \text{wgt} \cdot \nu \Sigma_F / \Sigma_T \] = expected number of fission neutrons produced per collision

• To sample the number of neutrons produced in the collision
  - Let \( r = \text{wgt} \cdot \nu \Sigma_F / \Sigma_T \)
    \[ n = \text{int}[\ r \ ] \]
  - Then, Produce \( n \) fission neutrons with probability 1 and an additional fission neutron with probability \( r - n \)
  - Example: \( \text{wgt} \cdot \nu \Sigma_F / \Sigma_T = 1.75 \)
    If \( \xi < .75 \), produce 2 neutrons, otherwise produce 1 or
    Produce \( \text{int}[1.75 + \xi] \) neutrons
Alternative Schemes for Flights/Collisions

• **Conventional scheme**
  - Particle weight constant during flight
  - Use $\Sigma_T$ to determine distance-to-collision, $s = -\ln \xi / \Sigma_T$
  - Change weight only on collisions
  - For pathlength absorption estimator, tally $\text{wgt} \cdot s \cdot \Sigma_A$
  - Most common scheme for reactors & shielding applications

• **Continuous absorption**
  - Particle weight decreases continuously during flight, due to absorption
    \[ \text{wgt}(s) = \text{wgt}_0 \cdot e^{-\Sigma_A s} \]
  - Use $\Sigma_S$ to determine distance-to-scattering, $s = -\ln \xi / \Sigma_s$
  - For pathlength absorption estimator, tally $\text{wgt}_0 \cdot (1 - e^{-\Sigma_A s})$
  - No absorption in collision
  - Typical use in astrophysics (Implicit Monte Carlo codes)
Tallies & Statistics
Monte Carlo Calculations

- **Geometry**
  - Which cell is particle in?
  - What will it hit next?
  - How far to boundary?
  - What’s on other side?
  - Survival?

- **Physics**
  - How far to collision?
  - Which nuclide?
  - New E, direction?
  - Secondaries?
  - Survival?

- **Tallies**
  - Tally events of interest
  - Compute results
  - Compute statistics
  - Balances
  - Performance stats

- **mcnp, rcp, vim, racer, sam-ce, tart, morse, keno, tripoli, mcbend, monk, o5r, recap, andy…..**

- During a history, tally the events of interest

- Upon completing a history, accumulate total scores & squares

- After completing all histories, compute mean scores & standard deviations
Monte Carlo Estimates of Integrals

Given a function $R(x)$, where $x$ is a random variable with PDF $f(x)$,

- Expected value of $R(x)$ is
  $$\bar{R} = \int R(x) \, f(x) \, dx$$

- Variance of $R(x)$ is
  $$\sigma^2 = \int R^2(x) \, f(x) \, dx - \mu^2$$

Monte Carlo method for estimating $\bar{R}$:

- make $N$ random samples $\hat{x}_j$ from $f(x)$
- Then
  $$\bar{R} \approx \frac{1}{N} \sum_{j=1}^{N} R(\hat{x}_j)$$

- Central Limit Theorem states that for large $N$,
  the PDF of $\bar{R}$ approaches a Gaussian distribution

- That is, if the Monte Carlo problem is repeated,
  $\bar{R}$ will be normally distributed
Laws of Large Numbers

Let \( x_1, x_2, \ldots, x_N \) be a sequence of independent, identically distributed random variables each with a finite mean \( \text{E}[x_j] = \mu \) and let

\[
\bar{x}_N = \frac{1}{N} \sum_{j=1}^{N} x_j
\]

- **Weak Law of Large Numbers**
  For any \( \varepsilon > 0 \)
  \[
  \lim_{N \to \infty} \text{P}(|\bar{x}_N - \mu| > \varepsilon) = 0
  \]
  Tells how a sequence of probabilities converges

- **Strong Law of Large Numbers**
  \[
  \text{P}\left( \lim_{N \to \infty} |\bar{x}_N - \mu| > \varepsilon \right) = 0
  \]
  Tells how the sequence of IID random variables behaves in the limit
Central Limit Theorem

- Central Limit Theorem

\[
\lim_{N \to \infty} \text{Prob}\left\{ \mu - a \frac{\sigma}{\sqrt{N}} \leq \bar{x} \leq \mu + b \frac{\sigma}{\sqrt{N}} \right\} = \frac{1}{\sqrt{2\pi}} \int_{-a}^{b} e^{-t^2} dt
\]

± 1 σ:

\[
\text{Prob}\left\{ \mu - \frac{\sigma}{\sqrt{N}} \leq \bar{x} \leq \mu + \frac{\sigma}{\sqrt{N}} \right\} = 68\%
\]

Note: 32% of the time, \( \bar{x} \) should be outside range \( \mu \pm \frac{\sigma}{\sqrt{N}} \)

± 2 σ:

\[
\text{Prob}\left\{ \mu - \frac{2\sigma}{\sqrt{N}} \leq \bar{x} \leq \mu + \frac{2\sigma}{\sqrt{N}} \right\} = 95\%
\]

Note: 5% of the time, \( \bar{x} \) should be outside range \( \mu \pm \frac{2\sigma}{\sqrt{N}} \)
Tallies & Statistics

- **For a given history, tally events of interest**
  - Example - surface crossings
    - For each particle crossing surface A, accumulate the weight each time a particle crosses that surface
    - A particular particle may cross the surface more than once
    - Progeny of that particle (e.g., another particle created by splitting) may also cross that surface one or more times

- **When the history is complete, add the score & score\(^2\) to accumulators for the problem**
  \[
  S_{1\text{problem}} = S_{1\text{problem}} + (S_{\text{history}}) \\
  S_{2\text{problem}} = S_{2\text{problem}} + (S_{\text{history}})^2
  \]

- **When all N histories are complete, compute final mean score & standard deviation**
  \[
  \text{mean score} = \frac{1}{N} \cdot S_1 \\
  \text{std dev of mean} = \sqrt{\frac{1}{N-1} \left[ \frac{S_2}{N} - \left( \frac{S_1}{N} \right)^2 \right]}
  \]
Variance of the Population vs. Mean

- Given a set of random samples, \( x_1, x_2, \ldots, x_N \),
  
  - Mean\[ \bar{x} = \frac{1}{N} \sum_{j=1}^{N} x_j \]

- Population variance [what you normally see in statistics textbooks]
  \[ \sigma^2 = \frac{1}{N} \sum_{j=1}^{N} x_j^2 - \left( \frac{1}{N} \sum_{j=1}^{N} x_j \right)^2 = \frac{1}{N} \sum_{j=1}^{N} x_j^2 - \bar{x}^2 \]

- Variance of the mean [what you normally find in MC codes]
  \[ \sigma_{\bar{x}}^2 = \frac{\sigma^2}{N} = \frac{1}{N} \cdot \left( \frac{1}{N} \sum_{j=1}^{N} x_j^2 - \bar{x}^2 \right) \]
  \[ \sigma_{\bar{x}} \sim \frac{1}{\sqrt{N}} \]
Tally Bins

- Tallies can be made for selected events & portions of phase space:
  - Range of energies, \( E_1 - E_2 \)
  - Range of particle times, \( t_1 - t_2 \)
  - Specified cells
  - Specified surfaces
  - Specified range of \( n \cdot \Omega \) for surface crossings
  - Specified reaction cross-sections \( \Sigma_x \)
  - Secondary particle production
  - Energy deposited in cell
  - Conditional events, e.g., absorption in cell B due to source in cell A
  - Energy of neutrons causing fission
  - Scattering from energy range \( E_1 - E_2 \) to range \( E_3 - E_4 \)
  - Etc.
Flux & Current

- **Angular flux**  \( \Psi(r, E, \Omega) \)

- **Flux**  \( \phi(r) = \int_{E_1}^{E_2} dE \int_{4\pi} d\Omega \Psi(r, E, \Omega) \)
  - Scalar quantity
  - Total distance traveled by all particles in a cm\(^3\) per second
  - Units: distance / cm\(^3\)-sec = 1 / cm\(^2\)-sec

- **Current**
  - Number of particles crossing surface per second per unit area
  - Units: 1 / cm\(^2\)-sec
  - Partial current: in + or - direction only, \( J^+ \) or \( J^- \)
  - Net current = \( J = J^+ - J^- \)

\[
J(r) = \int_{E_1}^{E_2} dE \int_{4\pi} d\Omega \, \tilde{n} \cdot \Omega \, \Psi(r, E, \Omega)
\]

\[
J^+(r) = \int_{E_1}^{E_2} dE \int_{\Omega > 0} d\Omega \, \tilde{n} \cdot \Omega \, \Psi(r, E, \Omega) \quad \quad J^-(r) = \int_{E_1}^{E_2} dE \int_{\Omega < 0} d\Omega \, \tilde{n} \cdot \Omega \, \Psi(r, E, \Omega)
\]
Reaction Rates

- For a particular reaction "x"
  \[ R_x(r) = \frac{1}{4\pi} \int_{E_1}^{E_2} \int d\Omega \int dE \Psi(r,E,\Omega) \Sigma_x(r,E) \]
  - Reactions per cm\(^3\) per sec

- Collision density
  \[ C(r) = \frac{1}{4\pi} \int_{E_1}^{E_2} \int d\Omega \int dE \Psi(r,E,\Omega) \Sigma_T(r,E) \]

- Energy deposition (average per collision)
  \[ E_{\text{deposited}}(r) = \frac{1}{4\pi} \int_{E_1}^{E_2} \int d\Omega \int dE \Psi(r,E,\Omega) \Sigma_T(r,E) \ K(r,E) \]

  where \( K(r,E) = \text{average E deposited per collision} \)
Analog vs. Weighted Monte Carlo

• **Analog Monte Carlo**
  – Faithful simulation of particle histories
  – No alteration of PDFs (i.e., no biasing or variance reduction)
  – At collision, particle is killed if absorption occurs
  – Particle is born with weight = 1.0
  – Weight unchanged throughout history until particle is killed
  – Score 1.0 when tallying events of interest

• **Weighted Monte Carlo (non-analog)**
  – Alter the PDFs to favor events of interest
  – Particle is born with weight = 1.0
  – Weight is altered if biased PDF is used
  – Typically, particle always survives collision & weight is reduced by $P_{\text{surv}}$
  – Weight can also be changed by Russian roulette/splitting & other variance reduction techniques
  – Score $\text{wgt}$ when tallying events of interest
Tally Types

- **Current tallies**
  - Surface crossing estimator

- **Flux tallies**
  - Pathlength estimator
  - Collision estimator
  - Surface crossing estimator
  - Next event estimator (point detector)

- **Reaction rate tallies**
  - Any of the above flux estimators times a cross-section

- **Energy deposition tallies**
  - Any of the above flux estimators times $\Sigma_T$ times energy deposited per collision
Basic Tallies

Current across surface

\[ J = \frac{1}{W} \sum_{\text{all flights}} wgt \]

\( W = \text{total source weight} \)

Flux on surface

\[ \phi = \frac{1}{A \cdot W} \sum_{\text{all flights}} \frac{wgt}{|\mu|} \]

\( A = \text{surface area} \)
\( W = \text{total source weight} \)
\( \mu = \Omega \cdot [\text{surface normal}] \)

Flux in a cell

\[ \phi = \frac{1}{V \cdot W} \sum_{\text{all flights in cell}} wgt \cdot \text{dist} \]

\( V = \text{cell volume} \)
\( W = \text{total source weight} \)

Flux at a point

\[ \phi = \frac{1}{W} \sum_{\text{all collisions}} wgt \cdot \frac{p(\mu)e^{-\Sigma R}}{2\pi R^2} \]
Current Tallies

- For each particle crossing surface, tally the particle weight

- Divide by total starting weight & surface area to get current

\[ J = \frac{1}{W \cdot A} \sum_{\text{all particles crossing surface}} \text{wgt}_j \]

W = total starting weight
A = surface area

- Typically, keep separate tally for outward partial current for each surface of a cell

- Can get net current by combining partial currents
Flux Tally - Pathlength

- For each particle flight within a cell, tally \((\text{pathlength} \times \text{weight})\)

- Divide by cell volume & total starting weight to get flux estimate

\[
\phi = \frac{1}{W V} \sum_{\text{all particle flights in cell}} d_j \cdot \text{wgt}_j
\]

- \(W = \text{total starting weight}\)
- \(V = \text{cell volume}\)
Flux Tally - Collisions

- Since \( (\Sigma_T \varphi) \) is collision rate, for each collision, tally \( \left( \frac{wgt}{\Sigma_T} \right) \) to estimate flux

- Divide by total starting weight & cell volume

\[
\phi = \frac{1}{W \cdot V} \cdot \sum_{\text{all collisions in cell}} \frac{wgt_j}{\Sigma_T (E_j)}
\]

- \( wgt_j = \) weight of particle entering collision

- \( W = \) total starting weight

- \( V = \) cell volume
Flux Tally - Surface Crossing

• Consider particles crossing a surface
  – Put a "box" of thickness \( a \) around the surface
  – Pathlength estimate of flux in the box

\[
\phi = \frac{1}{W \ a A} \sum \text{ all particles crossing surface} \ wgt_j \ a \left| \mu_j \right|
\]

where \( \mu_j = \cos \theta_j \)

– Note that \( a \) cancels out
– Take the limit as \( a \to 0 \)

• Surface crossing estimate of flux

\[
\phi = \frac{1}{W \ A} \sum \text{ all particles crossing surface} \ \frac{wgt_j}{\left| \mu_j \right|}
\]

where \( \mu_j = \Omega_j \cdot \vec{S} \)
Flux Tally - Surface Crossing

- **Complication:** \( \text{wgt}_j/\mu_j \) can be very large for small \( \mu_j \)
  - Usual solution, based on theory from FH Clark, "Variance of Certain Flux Estimators Used in Monte Carlo Calculations", Nucl. Sci. Eng. 27, 235-239 (1967)
  - For small \(|\mu|\), that is, \(-\varepsilon < \mu < \varepsilon\), (where \(\varepsilon\) is small), if it is assumed that the flux is only isotropic or linearly anisotropic, then the expected value of \(1/|\mu|\) is \(2/\varepsilon\).

- **Actual tally procedure:**
  - If \(|\mu|<\varepsilon\), then replace \(|\mu|\) by \(\varepsilon/2\) to score an expected flux.
  - This results in a reliable variance, without affecting the flux estimate significantly.

- **MCNP uses** \(\varepsilon=.1\). Many other codes use \(\varepsilon=.001\)
Flux at a Point

- Instead of estimating flux for a cell or surface, it may be useful to estimate flux at a point
  - Probability of a history trajectory going through a particular point is zero

- Use a "next event estimator" to get flux at a point
  - Regardless of the actual outcome of simulating a collision, estimate what would happen if the particle scattered exactly in the direction of a point detector

\[
\text{Expected } \phi \text{ score} = \text{wgt}' \cdot \frac{p_{sc}(\mu)}{2\pi R^2} \cdot \exp \left\{ -\int_0^R \Sigma_T(E') ds \right\}
\]

where
- \( \text{wgt}' = \text{weight after collision} \)
- \( p_{sc}(\mu) = \text{scatter PDF evaluated at } \mu \)
- \( E' = \text{energy corresponding to } \mu \)
**Flux at a Point**

- **Expected score has** $1/R^2$ **singularity** - collisions close to detector can result in large scores
  - Point detector estimator has finite mean, but infinite variance due to $1/R^2$ singularity

- **To keep variance finite:**
  - For collisions within radius $\mathcal{R}$ of detector, replace the factor
    \[
    \exp\left\{-\int_0^R \Sigma_T(E') ds\right\}
    \frac{1}{R^2}
    \]
    by volume average assuming uniform collisions inside sphere
    \[
    \int_0^\mathcal{R} s^2 ds = \frac{1}{3} \mathcal{R}^3 \Sigma_T(E')
    \]
    - Typically choose $\mathcal{R}$ to be $\sim$ half a mean free path

\[
\int_0^\mathcal{R} e^{-\Sigma_T(E') s} ds = \frac{1 - e^{\Sigma_T(E') \mathcal{R}}}{\mathcal{R}^3 \Sigma_T(E')}
\]
Reaction Rate Tallies

- Tally \((\text{flux-estimator}) \cdot (\text{cross-section})\)
- Example - pathlength tallies

After each flight, tally

- Flux \(\text{wgt} \cdot d_j\)
- Total absorption \(\text{wgt} \cdot d_j \cdot \Sigma_A\)
- Nu-fission \(\text{wgt} \cdot d_j \cdot \nu \Sigma_F\)
- U235 absorption \(\text{wgt} \cdot d_j \cdot N_{U^{235}} \cdot \Sigma_{A}^{U^{235}}\)
Mesh Tallies & Fission Matrix

• **Mesh Tallies**
  – Impose a grid over the problem & tally flux or reaction rates in each grid cell

• **Fission matrix**
  – Impose a grid over problem
  – Tally $F(I \rightarrow J)$ for source in cell $I$ causing fission in cell $J$
  – For $N$ cells in grid, $N^2$ tallies
Cautions

• Some codes (e.g., MCNP) report the mean score & relative error; 
  \[ \text{RE} = \frac{\sigma_x}{\bar{x}} \]
  - RE should decrease smoothly with \(1/\sqrt{N}\) dependence as more histories are run

• Tallies are reliable only if "enough" histories traverse the portions of problem phase space being tallied
  - Undersampling can lead to questionable or erroneous values of the mean score & relative error
  - Indicators of undersampling:
    • Large RE, \( \text{RE} > 0.1 \)
    • RE does not decrease smoothly as \(1/\sqrt{N}\)
    • A few histories have very large scores

• MCNP performs statistical checks on selected tallies to try to detect undersampling effects
  - Large RE
  - Variance of the variance (VOV)
  - Tally fluctuation charts (distribution of scores)
  - Slope of tails in tally fluctuation charts
  - Etc.
RE & FOM

• Some codes (e.g., MCNP) report the mean score & relative error

\[
RE = \frac{\sigma_x}{\bar{x}}
\]

• Some codes report a Figure-of-Merit for selected tallies

\[
FOM = \frac{1}{RE^2 \cdot T}
\]

Where \( T \) = computer time used

– \( RE^2 \sim 1/N \), where \( N \) is the total number of histories
– \( T \sim N \)
– Therefore, FOM should be roughly constant
– Used for comparing effectiveness of different variance reduction schemes
Combining Independent MC Results

Given N sets of (mean, std-dev) for independent Monte Carlo calculations, \((x_1, \sigma_1), (x_2, \sigma_2), \ldots\), how should the results be combined?

\[ w_j = \frac{1}{\sigma_j^2} \quad \quad \quad W = \sum_{j=1}^{N} \frac{1}{\sigma_j^2} \]

\[ \bar{x} = \sum_{j=1}^{N} \frac{w_j}{W} x_j \]

\[ \sigma_{\bar{x}}^2 = \sum_{j=1}^{N} \frac{w_j}{W^2} = \frac{1}{W} \]

Weighting factors \(\sim 1/\sigma^2\)
Combining Correlated Tallies

- Suppose 2 estimators, x and y, are correlated, such as the path & collision estimator for Keff

\[
\bar{x} = \frac{1}{N} \sum_{j=1}^{N} x_j \quad \bar{y} = \frac{1}{N} \sum_{j=1}^{N} y_j
\]

\[
\sigma_x^2 = \frac{1}{N} \sum_{j=1}^{N} x_j^2 - \bar{x}^2 \quad \sigma_y^2 = \frac{1}{N} \sum_{j=1}^{N} y_j^2 - \bar{y}^2 \quad \sigma_{xy}^2 = \frac{1}{N} \sum_{j=1}^{N} x_j y_j - \bar{x} \cdot \bar{y}
\]

Minimum variance combination of x & y

\[
\alpha = \frac{\sigma_y^2 - \sigma_{xy}^2}{\sigma_x^2 - 2\sigma_{xy}^2 + \sigma_y^2}
\]

\[
\text{mean}_{x,y} = \alpha \bar{x} + (1-\alpha) \bar{y}
\]

\[
\text{std-dev}_{x,y} = \sqrt{\frac{\alpha^2 \sigma_x^2 + 2\alpha (1-\alpha) \sigma_{xy}^2 + (1-\alpha)^2 \sigma_y^2}{N-1}}
\]
Eigenvalue Calculations - I

- $k$- and $\alpha$- Eigenvalue Equations
- Power Iteration
- Convergence
Reactor Analysis with Monte Carlo

Geometry Model (1/4)

K vs cycle

H_{src} vs cycle

Assembly Powers

Fast Flux

Thermal Flux
The Challenge

For Monte Carlo calculations of just K-effective, plots of $k_{cycle}$ vs cycle are adequate to judge convergence.

To compute power distributions, heating distributions, dose rates, production/depletion, & local reaction rates, new tools are needed to judge convergence of the source distribution.

- The source distribution takes longer to converge than K-effective.

- How do you tell if a 3D distribution has converged?

- For the past 40 years, people calculating power distributions or production/depletion with Monte Carlo were "flying blind" -- no tools were available to assess source convergence.
**Source Distribution Convergence**

**Fuel Storage Vault**

**K vs cycle**

**\( H_{\text{src}} \) vs cycle**

For this calculation,

- Should discard ~20 cycles if calculating \( K_{\text{eff}} \) only
- Should discard ~2000 cycles if calculating heating distribution
K- and $\alpha$- Eigenvalue Equations
Time-dependent Transport

- Time-dependent linear Boltzmann transport equation for neutrons, with prompt fission source & external source

\[
\frac{1}{v} \frac{\partial \psi(\bar{r}, E, \bar{\Omega}, t)}{\partial t} = Q(\bar{r}, E, \bar{\Omega}, t) + \int \int \psi(\bar{r}, E', \bar{\Omega}', t) \sum_s (\bar{r}, E' \rightarrow E, \bar{\Omega} \cdot \bar{\Omega}') d\bar{\Omega}' dE' + \frac{\chi(\bar{r}, E)}{4\pi} \int \int \nu \sum_f (\bar{r}, E') \psi(\bar{r}, E', \bar{\Omega}', t) d\bar{\Omega}' dE' - \left[ \bar{\Omega} \cdot \nabla + \sum_t (\bar{r}, E) \right] \cdot \psi(\bar{r}, E, \bar{\Omega}, t)
\]

\[
\frac{1}{v} \frac{\partial \psi(\bar{r}, E, \bar{\Omega}, t)}{\partial t} = Q + \left[ S + M \right] \cdot \psi - \left[ L + T \right] \cdot \psi
\]

- This equation can be solved directly by Monte Carlo, assuming:
  - Each neutron history is an IID trial (independent, identically distributed)
  - All neutrons must see same probability densities in all of phase space
  - Usual method: geometry & materials fixed over solution interval \( \Delta t \)
Time-dependent Transport

\[ \frac{1}{v} \frac{\partial \psi(\vec{r},E,\hat{\Omega},t)}{\partial t} = Q + [S+M] \cdot \psi - [L+T] \cdot \psi \]

- Monte Carlo solution (over \( \Delta t \), with fixed geometry & materials)
  - Simulate time-dependent transport for a neutron history
  - If fission occurs, bank any secondary neutrons.
  - When original particle is finished, simulate secondaries till done.
  - Tallies for time bins, energy bins, cells, ...

- At time \( t \), the overall neutron level is
  \[ N(t) = \iiint_\Omega \frac{\psi(\vec{r},E,\hat{\Omega},t)}{v} d\vec{r} dE d\hat{\Omega} \]

- Alpha & T (reactor period) can be defined by:
  \[ N(t) = N_0 e^{\alpha t} = N_0 e^{t/T} \]
  \[ \alpha = \frac{1}{T} \approx \frac{\ln N(t) - \ln N_0}{t - t_0} \]

This is the "dynamic alpha", NOT an eigenvalue!
Particle Histories

- **Random Walk for particle**
  - Track through geometry,
    - select collision site randomly
    - tallies
  - Collision physics analysis,
    - Select new E,Ω randomly
    - tallies
  - Secondary Particles

- **Particle History**
  - Source
    - select r,E,Ω
  - Random Walk
  - Random Walk
  - Random Walk
  - Random Walk
  - Random Walk
  - Random Walk
Fixed-source Monte Carlo Calculation

**History 1**

Source - select r,E,Ω → Random Walk → Random Walk

**History 2**

Source - select r,E,Ω → Random Walk → Random Walk

**History 3**

Source - select r,E,Ω → Random Walk → Random Walk
Alpha Eigenvalue Equations

- For problems which are separable in space & time, it may be advantageous to solve a static eigenvalue problem, rather than a fully time-dependent problem

- Assume:
  1. Fixed geometry & materials
  2. No external source: \( Q(r,E,\Omega,t) = 0 \)
  3. Separability: \( \Psi(r,E,\Omega,t) = \Psi_\alpha(r,E,\Omega) e^{\alpha t} \)

- Substituting \( \Psi \) into the time-dependent transport equation yields

\[
\left[ L + T + \frac{\alpha}{V} \right] \Psi_\alpha (\vec{r},E,\vec{\Omega}) = [S + M] \Psi_\alpha
\]

- This is a static equation, an eigenvalue problem for \( \alpha \) and \( \Psi_\alpha \) without time-dependence
- \( \alpha \) is often called the time-eigenvalue or time-absorption
- \( \alpha \)-eigenvalue problems can be solved by Monte Carlo methods
**$K_{\text{eff}}$ Eigenvalue Equations**

- Another approach to creating a static eigenvalue problem from the time-dependent transport equation is to introduce $K_{\text{eff}}$, a scaling factor on the multiplication ($\nu$)

- Assume:
  1. Fixed geometry & materials
  2. No external source: $Q(r,E,\Omega,t) = 0$
  3. $\partial\Psi/\partial t = 0$: $\nu \rightarrow \nu / k_{\text{eff}}$

- Setting $\partial\Psi/\partial t = 0$ and introducing the $K_{\text{eff}}$ eigenvalue gives

  \[
  \left[ L + T \right] \Psi_k(\vec{r},E,\vec{\Omega}) = \left[ S + \frac{1}{K_{\text{eff}}} M \right] \Psi_k
  \]

  - This is a static equation, an eigenvalue problem for $K_{\text{eff}}$ and $\Psi_k$ without time-dependence
  - $K_{\text{eff}}$ is called the effective multiplication factor
  - $K_{\text{eff}}$ and $\Psi_k$ should never be used to model time-dependent problems.
  - $K_{\text{eff}}$-eigenvalue problems can be solved by Monte Carlo methods
Comments on $K_{\text{eff}}$ and $\alpha$ Equations

- **Criticality**
  - Supercritical: $\alpha > 0$ or $K_{\text{eff}} > 1$
  - Critical: $\alpha = 0$ or $K_{\text{eff}} = 1$
  - Subcritical: $\alpha < 0$ or $K_{\text{eff}} < 1$

- **$K_{\text{eff}}$ vs. $\alpha$ eigenvalue equations**
  - $\Psi_k(r,E,\Omega) \neq \Psi_\alpha(r,E,\Omega)$, except for a critical system
  - $\alpha$ eigenvalue & $\Psi_\alpha$ eigenfunction used for time-dependent problems
  - $K_{\text{eff}}$ eigenvalue & $\Psi_k$ eigenfunction used for reactor design & analysis
  - Although $\alpha = (K_{\text{eff}}-1)/\lambda$, where $\lambda$ = lifetime,
    there is no direct relationship between $\Psi_k(r,E,\Omega)$ and $\Psi_\alpha(r,E,\Omega)$

- $K_{\text{eff}}$ eigenvalue problems can be solved **directly** using Monte Carlo

- $\alpha$ eigenvalue problems are solved by Monte Carlo **indirectly**
  using a series of $K_{\text{eff}}$ calculations
Comments on $K_{\text{eff}}$ and $\alpha$ Equations

**K equation**

$\left[ L + T \right] \Psi_k = \left[ S + \frac{1}{k} M \right] \Psi_k$

**$\alpha$ equation**

$\left[ L + T + \frac{\alpha}{\nu} \right] \Psi_\alpha = \left[ S + M \right] \Psi_\alpha$

- The factor $1/k$ changes the relative level of the fission source

- The factor $\alpha/\nu$ changes the absorption & neutron spectrum
  - For $\alpha > 0$, more absorption at low E $\Rightarrow$ harder spectrum
  - Double-density Godiva, average neutron energy causing fission:
    - $k$ calculation: $1.30$ MeV
    - $\alpha$ calculation: $1.68$ MeV

- For separable problems, $\Psi(r,E,\Omega,t) = \Psi_\alpha(r,E,\Omega) e^{\alpha t}$

- No similar equation for $k$, since not used for time-dependence
Power Iteration & Convergence
K-eigenvalue equation

\[(L + T)\Psi = S\Psi + \frac{1}{K_{\text{eff}}} M\Psi\]

where

- \(L\) = leakage operator
- \(S\) = scatter-in operator
- \(T\) = collision operator
- \(M\) = fission multiplication operator

• Rearrange

\[(L + T - S)\Psi = \frac{1}{K_{\text{eff}}} M\Psi\]

\[\Psi = \frac{1}{K_{\text{eff}}} \cdot (L + T - S)^{-1} M\Psi\]

\[\Psi = \frac{1}{K_{\text{eff}}} \cdot F\Psi\]

→ This eigenvalue equation will be solved by power iteration

\[\Psi^{(n+1)} = \frac{1}{K_{\text{eff}}^{(n)}} \cdot F\Psi^{(n)}\]
Power Iteration

Diffusion Theory or Discrete-ordinates Transport

1. Initial guess for $K_{\text{eff}}$ and $\Psi$
   
   $K_{\text{eff}}^{(0)}, \Psi^{(0)}$

2. Solve for $\Psi^{(n+1)}$
   
   Inner iterations over space or space/angle to solve for $\Psi^{(n+1)}$
   
   \[(L + T - S)\Psi^{(n+1)} = \frac{1}{K_{\text{eff}}^{(n)}} M\Psi^{(n)}\]

3. Compute new $K_{\text{eff}}$
   
   $K_{\text{eff}}^{(n+1)} = K_{\text{eff}}^{(n)} \cdot \frac{1}{K_{\text{eff}}^{(n)}} \int M\Psi^{(n+1)} d\vec{r}$

4. Repeat 1-3 until both $K_{\text{eff}}^{(n+1)}$ and $\Psi^{(n+1)}$ have converged

Monte Carlo

1. Initial guess for $K_{\text{eff}}$ and $\Psi$

   $K_{\text{eff}}^{(0)}, \Psi^{(0)}$

2. Solve for $\Psi^{(n+1)}$
   
   Follow particle histories to solve for $\Psi^{(n+1)}$
   
   \[(L + T - S)\Psi^{(n+1)} = \frac{1}{K_{\text{eff}}^{(n)}} M\Psi^{(n)}\]

   During histories, save fission sites to use for source in next iteration

3. Compute new $K_{\text{eff}}$
   
   During histories for iteration (n+1), estimate $K_{\text{eff}}^{(n+1)}$
   
   \[K_{\text{eff}}^{(n+1)} = K_{\text{eff}}^{(n)} \cdot \frac{\int M\Psi^{(n+1)} d\vec{r}}{\int M\Psi^{(n)} d\vec{r}}\]

4. Repeat 1-3 until both $K_{\text{eff}}^{(n+1)}$ and $\Psi^{(n+1)}$ have converged

5. Continue iterating, to compute tallies
• Power iteration for Monte Carlo $k$-effective calculation

- **Initial Guess**
- **Batch 1** $K_{\text{eff}}^{(1)}$
- **Batch 2** $K_{\text{eff}}^{(2)}$
- **Batch 3** $K_{\text{eff}}^{(3)}$
- **Batch 4** $K_{\text{eff}}^{(4)}$

![Diagram showing power iteration process]

- **Source particle generation**
- **Monte Carlo random walk**
- **Neutron**
α-Eigenvalue Calculations

- **Eigenvalue equation with both $K_{\text{eff}}$ & $\alpha$**
  - $\alpha$ is a fixed number, not a variable or eigenvalue
  - Find the k-eigenvalue as function of $\alpha$, $K(\alpha)$

\[
\begin{bmatrix}
L + T + \frac{\alpha}{v}
\end{bmatrix} \Psi_\alpha (\vec{r},E,\vec{\Omega}) = \begin{bmatrix}
S + \frac{1}{K_{\text{eff}}}
\end{bmatrix} \Psi_\alpha
\]

- **Note:** If $\alpha < 0$
  - Real absorption plus time absorption could be negative
  - Move $\alpha/v$ to right side to prevent negative absorption,
  - $-\alpha/v$ term on right side is treated as a delta-function scatter

- Select a fixed value for $\alpha$
- Solve the K-eigenvalue equations, with fixed time-absorption $\alpha/v$
- Select a different $\alpha$ and solve for a new $K_{\text{eff}}$
- Repeat, searching for value of $\alpha$ which results in $K_{\text{eff}} = 1$
K- and $\alpha$-Eigenvalue Calculations

- **$K$-eigenvalue solution**

  Loop for Power Iteration for $K$
  
  - Loop over neutrons in cycle
  - neutron history
  - 
  
  - 
  
  - 
  
  - 
  
  - 
  
  - 
  
  - 

- **$\alpha$-eigenvalue solution**

  Loop for $\alpha$ search iterations
  
  - Loop for Power Iteration for $K$
  
  - Loop over neutrons in cycle
  
  - neutron history
  
  - 
  
  - 
  
  - 
  
  - 
  
  - 

  $\Rightarrow$ Find $K(\alpha)$, then search for $\alpha$ that gives $K(\alpha)=1$
K-Calculations

- Guess an initial source distribution
- Iterate until converged (How do you know ???)
- Then
  - For $S_n$ code: done, print the results
  - For Monte Carlo: start tallies, keep running until uncertainties small enough

- Details? Bias? Statistics?
Monte Carlo Solution of $K_{\text{eff}}$ Problems

Note: batch = cycle = iteration = generation

• Initialize
  – Assume a value for the initial $K_{\text{eff}}$ (usually, $K_0 = 1$)
  – Sample $M$ fission sites from the initial source distribution

• For each cycle $n$, $n = 1 \ldots N+D$
  – Follow histories for all source particles in cycle
    • If fissions occur, bank the sites for use as source in next cycle
    • Make tallies for $K_{\text{cycle}}^{(n)}$ using path, collision, & absorption estimators
    • If $n \leq D$, discard any tallies
    • If $n > D$, accumulate tallies
  – Estimate $K_{\text{cycle}}^{(n)}$

• Compute final results & statistics using last $N$ cycles
K-Calculations -- Power Iteration

- **Power iteration procedure:**
  
  1. Initial guess for $K_{\text{eff}}$ and $\Psi$
     
     $K_{\text{eff}}^{(0)}, \Psi^{(0)}$
  
  2. Solve for $\Psi^{(n+1)}$ [Monte Carlo random walk for N particles]
     
     $\Psi^{(n+1)} = \frac{1}{K_{\text{eff}}^{(n)}} \cdot F \Psi^{(n)}$
  
  3. Compute new $K_{\text{eff}}$
     
     $K_{\text{eff}}^{(n+1)} = K_{\text{eff}}^{(n)} \cdot \frac{\int M \Psi^{(n+1)} d\vec{r}}{\int M \Psi^{(n)} d\vec{r}}$
  
  4. Repeat 1-3 until both $K_{\text{eff}}^{(n+1)}$ and $\Psi^{(n+1)}$ have converged
K-Calculations -- Banking Fission Sites

- During a particle random walk,

\[ wgt \cdot \frac{v \Sigma_F}{\Sigma_T} = \text{expected number of fission neutrons created at collision point} \]

- Averaged over all collisions for all histories, the expected value for \( wgt \cdot \frac{v \Sigma_F}{\Sigma_T} \) is \( K_{\text{eff}} \).

- In order to bank approximately the same number of fission sites in each cycle, the current value of \( K_{\text{eff}} \) is used to bias the selection of fission sites at a collision:

\[ R = wgt \cdot \frac{v \Sigma_F}{\Sigma_T} \cdot \frac{1}{K}, \quad n = \left\lfloor R \right\rfloor \]

If \( \xi < R - n \), store \( n + 1 \) sites in bank with \( wgt' = K \)

Otherwise, store \( n \) sites in bank with \( wgt' = K \)
K-Calculations -- Renormalization

- \( N_J = \) number of particles starting cycle \( J \),
  \( N'_J = \) number of particles created by fission in cycle \( J \)
  (number of particles stored in fission bank)
- The expected value for \( N'_J \) is: \( E[ N'_J ] = K_{\text{eff}} \cdot N_J \)
- \( (N'_J/N_J) \) is a single-cycle estimator for \( K_{\text{eff}} \)

- To prevent the number of particles per cycle from growing exponentially (for \( K>1 \)) or decreasing to 0 (for \( K<1 \)), the particle population is renormalized at the end of each cycle:
  - In some Monte Carlo codes, the number of particles starting each cycle is a constant \( N \). Russian roulette or splitting are used to sample \( N \) particles from the \( N' \) particles in the fission bank. (All particles in fission bank have a weight of 1.0)
  - In other codes, the total weight \( W \) starting each cycle is constant. The particle weights in the fission bank are renormalized so that the total weight is changed from \( W' \) to \( W \). (Particles in fission bank have equal weights, but not necessarily 1.0)
Single-cycle $K_{\text{eff}}$ Estimators

- **Pathlength estimator for $K_{\text{eff}}$**
  \[
  K_{\text{path}} = \left( \sum_{\text{all flights}} \text{wgt}_j \cdot d_j \cdot \nu \Sigma_F \right) / W
  \]
  \(W = \text{total weight starting each cycle}\)

- **Collision estimator for $K_{\text{eff}}$**
  \[
  K_{\text{collision}} = \left( \sum_{\text{all collisions}} \text{wgt}_j \cdot \frac{\nu \Sigma_F}{\Sigma_T} \right) / W
  \]

- **Absorption estimator for $K_{\text{eff}}$**
  \[
  K_{\text{absorption}} = \left( \sum_{\text{all absorptions}} \text{wgt}_j \cdot \frac{\nu \Sigma_F}{\Sigma_A} \right) / W
  \]
K-Calculations -- Bias

• The renormalization procedure used at the end of each cycle introduces a small bias into the computed $K_{\text{eff}}$
  – Renormalization involves multiplying particle weights by $(W/W')$, where $W =$ total weight starting a cycle, $W'$ = total weight at the end of a cycle.
  – $W'$ is a random variable, due to fluctuations in particle random walks.

• Theoretical analysis of the MC iteration process & propagation of history fluctuations gives

$$\text{bias in } K_{\text{eff}} = - \frac{\sigma_k^2}{K_{\text{eff}}} \left( \text{sum of correlation coeff's between batch } K\text{'s} \right)$$

  – $M =$ histories/cycle
  – Bias in $K_{\text{eff}} \sim 1/M$
    • Smaller $M \rightarrow$ larger cycle correlation $\rightarrow$ larger bias in $K_{\text{eff}}$ & source
    • Larger $M \rightarrow$ smaller cycle correlation $\rightarrow$ smaller bias

K-Calculations -- Bias

• For a simple Godiva reactor calculation:

![Graph showing Keff vs 1/M](image-url)
K-Calculations -- Bias

- Observed PDF for single-cycle $K_{\text{eff}}$, for varying $M$

- Bias in $K_{\text{eff}}$ is negative: $K_{\text{calc}} < K_{\text{true}}$

- Bias is significant for $M < 10$ particles/cycle
  - small for $M \sim 100$
  - negligible for $M > 1000$
  - 0 for $M \rightarrow \infty$

- Recommendation: Always use 10,000 or more particles/cycle
Power Iteration & Convergence

- Guess an initial source distribution
- Iterate until converged (How do you know ???)
- Then
  - For $S_n$ code: done, print the results
  - For Monte Carlo: start tallies, keep running until uncertainties small enough

- Convergence? Stationarity?
K-eigenvalue equation

• Use operator (or matrix) form to simplify notation

\[(L + T)\Psi = S\Psi \, + \, \frac{1}{K_{\text{eff}}} M\Psi\]

where

\[L = \text{leakage operator} \quad S = \text{scatter-in operator}\]

\[T = \text{collision operator} \quad M = \text{fission multiplication operator}\]

• Rearrange

\[(L + T - S)\Psi = \frac{1}{K_{\text{eff}}} M\Psi\]

\[\Psi = \frac{1}{K_{\text{eff}}} \cdot (L + T - S)^{-1} M\Psi\]

\[\Psi = \frac{1}{K_{\text{eff}}} \cdot F\Psi\]

→ This eigenvalue equation will be solved by power iteration
Power Iteration - Convergence

- Expand $\Psi$ in terms of eigenfunctions $u_j(r,E,\Omega)$

$$\Psi = \sum_{j=0}^{\infty} a_j \tilde{u}_j = a_0 \tilde{u}_0 + a_1 \tilde{u}_1 + a_2 \tilde{u}_2 + a_3 \tilde{u}_3 + \ldots$$

$$\int \tilde{u}_j \tilde{u}_k dV = \delta_{jk} \quad a_j = \int \Psi \cdot \tilde{u}_j dV$$

$$\tilde{u}_j = \frac{1}{k_j} F \cdot \tilde{u}_j \quad k_0 > k_1 > k_2 > \ldots \quad k_0 \equiv k_{\text{effective}}$$

- Expand the initial guess in terms of the eigenmodes

$$\Psi^{(0)} = \sum_{j=0} \tilde{a}_j^{(0)} \tilde{u}_j$$

- Substitute the expansion for $\Psi^{(0)}$ into power iteration equation

$$\Psi^{(n+1)} = \frac{1}{K^{(n)}} F \cdot \Psi^{(n)} = \frac{1}{k^{(n)}} \cdot \frac{1}{k^{(n-1)}} \ldots \frac{1}{k^{(0)}} \cdot F^n \cdot \Psi^{(0)}$$

$$= \left[ \prod_{m=0}^{n} \frac{k_0}{k^{(m)}} \right] \cdot a_0^{(0)} \cdot \tilde{u}_0 + \sum_{j=1} \left( \frac{a_j^{(0)}}{a_0^{(0)}} \right) \cdot \left( \frac{k_j}{k_0} \right)^{n+1} \cdot \tilde{u}_j$$
Power Iteration - Convergence

\[ \Psi^{(n+1)} \approx [\text{constant}] \cdot \left[ \tilde{u}_0 + \left( \frac{a_1^{(0)}}{a_0^{(0)}} \right) \cdot \left( \frac{k_1}{k_0} \right)^{n+1} \tilde{u}_1 + \ldots \right] \]

\[ K^{(n+1)} \approx k_0 \cdot \left[ 1 + \left( \frac{a_1^{(0)}}{a_0^{(0)}} \right) \cdot \left( \frac{k_1}{k_0} \right)^n \left( \frac{k_1}{k_0} - 1 \right) \cdot G_1 + \ldots \right] \]

- Because \( k_0 > k_1 > k_2 > \ldots \), all of the red terms vanish as \( n \to \infty \)
  - \( \Psi^{(n+1)} \to \text{constant} \cdot u_0 \)
  - \( K^{(n+1)} \to k_0 \)

- After the initial transient, error in \( \Psi^{(n)} \) is dominated by first mode
  - \( \left( \frac{k_1}{k_0} \right) \) is called the dominance ratio, DR or \( \rho \)
  - Errors in \( \Psi^{(n)} \) die off as \( \sim (DR)^n \)

- For problems with a high dominance ratio (e.g., \( DR \sim .99 \)), the error in \( K_{\text{eff}} \) may be small, since the factor \( \left( \frac{k_1}{k_0} - 1 \right) \) is small.
  - \( K_{\text{eff}} \) may appear converged, even if the source distribution is not converged
Power Iteration - Convergence

- After \( n \) iterations, \( J^{th} \) mode error component is reduced by the factor \( \left( \frac{k_J}{k_0} \right)^n \)

- Since \( 1 > \frac{k_1}{k_0} > \frac{k_2}{k_0} > \frac{k_3}{k_0} > \ldots \), after the initial transient, error in \( \Psi^{(n)} \) is dominated by first mode:

\[
\Psi^{(n)} \approx \text{[constant]} \cdot \left[ \tilde{u}_0 + \left( \frac{a_1^{(0)}}{a_0^{(0)}} \right) \cdot \left( \frac{k_1}{k_0} \right)^n \cdot \tilde{u}_1 + \ldots \right]
\]

- \( \left( \frac{k_1}{k_0} \right) \) is called the dominance ratio, \( DR \) or \( \rho \)

  - Errors die off as \( \sim (DR)^n \)

  - To reduce 10% error \( \rightarrow .1\% \) error

    \[
    \begin{align*}
    DR \sim .9 & \rightarrow 44 \text{ iterations} \\
    DR \sim .99 & \rightarrow 458 \text{ iterations} \\
    DR \sim .999 & \rightarrow 2301 \text{ iterations}
    \end{align*}
    \]
Power Iteration - Convergence

Typical K-effective convergence patterns

- Higher mode error terms die out as $(\frac{k_J}{k_0})^n$, for $n$ iterations

- When initial guess is concentrated in center of reactor, initial $K_{\text{eff}}$ is too high (underestimates leakage)

- When initial guess is uniformly distributed, initial $K_{\text{eff}}$ is too low (overestimates leakage)

- The Sandwich Method uses 2 $K_{\text{eff}}$ calculations - one starting too high & one starting too low. Both calculations should converge to the same result.
Power Iteration - Convergence

- Keff is an integral quantity - converges faster than source shape

Keff calculation for 2 nearly symmetric slabs, with Dominance Ratio = .9925
Power Iteration - Convergence

- For Monte Carlo power iteration, statistical fluctuations in source shape die out gradually over a number of successive iterations.
  - Persistence of the noise over successive iterations gives correlation among source distributions in successive iterations. (Positive correlation)
  - Correlation directly affects confidence intervals: Serial correlation in the source distribution ➔ larger confidence intervals

➔ Most Monte Carlo codes ignore these correlation effects & incorrectly underestimate the confidence intervals
Power Iteration - Convergence

Summary

• **Local errors in the source distribution** decay as \((k_j/k_0)^n\)
  – Higher eigenmodes die out rapidly, convergence dominated by \(k_1/k_0\)
  – High DR \(\rightarrow\) slow convergence
  – High DR \(\rightarrow\) large correlation \(\rightarrow\) large error in computed variances

• **Errors in \(K_{\text{eff}}\)** decay as \((k_j/k_0 - 1) \times (k_j/k_0)^n\)
  – High DR \(\rightarrow\) \(k_j/k_0 \sim 1\) \(\rightarrow\) small error

• **\(K_{\text{eff}}\)** errors die out faster than local source errors
  – \(K_{\text{eff}}\) is an integral quantity - positive & negative fluctuations cancel

• **High DR is common for**
  – Large reactors, with small leakage
  – Heavy-water moderated or reflected reactors
  – Loosely-coupled systems

\(\Rightarrow\) **If local tallies are important** (e.g., assembly power, pin power, ...),
examine their convergence - not just \(K_{\text{eff}}\) convergence
Eigenvalue Calculations - II

• Stationarity Diagnostics

• Wielandt & Superhistory Methods

• Dominance Ratio
Keff Calculations

- Plots of single-cycle Keff or cumulative Keff are sometimes difficult to interpret when assessing convergence

**Cycle \( k_{\text{eff}} \)**

50000 histories per cycle

- uniform initial source
- initial source at (1,3) lattice
Introduction

- Monte Carlo codes use **power iteration** to solve for $K_{\text{eff}}$ & $\Psi$ for eigenvalue problems

$$ (L + T - S)\Psi^{(n)} = \frac{1}{K^{(n-1)}} M \Psi^{(n-1)} $$

$L = \text{loss to leakage}$  
$T = \text{loss to collisions}$  
$S = \text{gain from scatter-in}$  
$M = \text{gain from fission multiplication}$

- **Power iteration convergence** is well-understood

  $$ \Psi^{(n)}(\vec{r}) = \tilde{u}_0(\vec{r}) + a_1 \cdot \rho^n \cdot \tilde{u}_1(\vec{r}) + ... $$

  $$ k_{\text{eff}}^{(n)} = k_0 \cdot \left[ 1 - \rho^{n-1} (1 - \rho) \cdot g_1 + ... \right] $$

  - First-harmonic source errors die out as $\rho^n,$  
    $\rho = k_1 / k_0 < 1$

  - First-harmonic $K_{\text{eff}}$ errors die out as $\rho^{n-1} (1 - \rho)$

  - Source converges slower than $K_{\text{eff}}$

- Most codes only provide tools for assessing $K_{\text{eff}}$ convergence.

→ MCNP5 also looks at Shannon entropy of the source distribution, $H_{\text{src}}$. 
Source Convergence & Shannon Entropy
Keff Calculations

• Initial cycles of a Monte Carlo K-effective calculation should be discarded, to avoid contaminating results with errors from initial guess
  – How many cycles should be discarded?
  – How do you know if you discarded enough cycles?

• Analysis of the power iteration method shows that Keff is not a reliable indicator of convergence -- Keff can converge faster than the source shape

• Based on concepts from information theory, Shannon entropy of the source distribution is useful for characterizing the convergence of the source distribution
**Keff Calculations - Stationarity Diagnostics**

- Divide the fissionable regions of the problem into \( N_S \) spatial bins
  - Spatial bins should be consistent with problem symmetry
  - Typical choices: -- 1 bin for each assembly
    -- regular grid superimposed on core

  - Rule-of-thumb for number of spatial bins:
    \[ N_S \sim \frac{\text{(histories/batch)}}{25} \text{ or less} \]

**Why?**
- Would like to have >25 fission source sites per bin to get good statistics
- If source distribution were uniform, \( \sim 25 \) sites would be in each bin

- **Shannon entropy of the source distribution**

\[
H(S) = - \sum_{J=1}^{N_S} p_J \cdot \ln_2(p_J), \quad \text{where} \quad p_J = \frac{\text{(\# source particles in bin J)}}{\text{(total \# source particles in all bins)}}
\]
**K_{eff} Calculations - Stationarity Diagnostics**

- **Shannon entropy of the source distribution**

\[
H(S) = - \sum_{J=1}^{N_S} p_J \cdot \ln_2(p_J), \quad \text{where} \quad p_J = \frac{\text{(\# source particles in bin } J)}{\text{(total \# source particles in all bins)}}
\]

- \( 0 \leq H(S) \leq \ln_2( N_S ) \)

- Note that \( p_J \ln_2(p_J) = 0 \) if \( p_J = 0 \)

- For a **uniform** source distribution, \( H(S) = \ln_2( N_S ) \)
- For a **point** source (in a single bin), \( H(S) = 0 \)
- For any **general** source, \( 0 \leq H(S) \leq \ln_2( N_S ) \)

- \( H(S^{(n)}) \) provides a single number to characterize the source distribution for iteration \( n \) (no physics!)

\[\rightarrow\] As the source distribution converges in 3D space, a line plot of \( H(S^{(n)}) \) vs. \( n \) (the iteration number) converge
Criticality Calculations - Convergence

- Reactor core (Problem inp24)

\[ K^{(n)} \text{ vs cycle} \]

\[ H(\text{fission source}) \]

\[ K_{\text{eff}} = 80 \]

\[ \text{DR} = 0.98 \]
Critically Calculations - Convergence

- PWR 1/4-Core  (Napolitano)

DR = .95

\[ K^{(n)} \text{ vs cycle} \]

\[ H(\text{fission source}) \]
Criticality Calculations - Convergence

- 2D PWR (Ueki)

\[ \text{DR} = .97 \]
Criticality Calculations - Convergence

- Loosely-coupled array of spheres (Problem test4s)

\[ K^{(n)} \text{ vs cycle} \]

\[ H(\text{fission source}) \]

\[ K_{\text{eff}} \]

DR = .91
**Criticality Calculations - Convergence**

- **Fuel Storage Vault (Problem OECD_bench1)**

![Fuel Storage Vault Diagram]

![Graph of K(n) vs cycle]

- **K(n) vs cycle**

- **H( fission source )**

- **DR = .99+**
**$H_{src}$ Convergence vs Number of Spatial Bins**

- For large number of bins, $H_{src}$ approaches uniform upper limit
- Use 10s or 100s of bins, not 1000s or more

![Graph showing convergence vs number of spatial bins](image-url)
**H_{src} and 2D vs 3D Spatial Bins**

- For 3D problems, using a 2D bin layout for $H_{src}$ may incorrectly assess convergence.

- Important to use 3D bin layout for 3D problems.

![Graph showing 3D and 2D bin layouts](image)
H_{src} Convergence vs Neutrons per Cycle

- Problems converge at the same rate, for any number of neutrons/cycle. (The rate depends on dominance ratio, ie, physics & geom)

- More neutrons/cycle does not make problems converge faster

- More neutrons/cycle $\Rightarrow$ less noise in convergence plots

Shannon Entropy
Of the Fission Source
for different neutrons / cycle

1000 - black
5000 - blue
20000 - red
Conclusions - $H_{src}$

- Shannon entropy is a highly effective means of characterizing convergence of the fission distribution

- If you are computing more than just $K_{eff}$ (eg, local reaction rates, dose fields, fission distributions, heating distributions, etc.):

  Should check both $k_{eff}$ and $H_{src}$ for convergence

- MCNP6 will compute & plot $H_{src}$ as an important new tool for assessing problem convergence.

- The recommended MCNP6 procedures for defining spatial tally bins and computing $H_{src}$ are effective for a variety of typical criticality problems.
Wielandt Acceleration

(For future versions of MCNP)

Wielandt Method

- Basic transport equation for static eigenvalue problems
  \[(L + T - S)\Psi = \frac{1}{k_{\text{eff}}} M\Psi\]

  \(L = \) loss to leakage  \(S = \) gain from scatter-in
  \(T = \) loss to collisions  \(M = \) gain from fission multiplication

- Define a fixed parameter \(k_e\) such that \(k_e > k_0\) \((k_0 = \) exact eigenvalue\)
  \[k_e = k_0 + \Delta, \quad \Delta > 0\]

- Subtract \(\frac{1}{k_e} M\Psi\) from each side of the transport equation

- Solve the modified transport equation by power iteration
  \[(L + T - S - \frac{1}{k_e} M)\Psi^{(n)} = (\frac{1}{k_{\text{eff}}}^{(n-1)} - \frac{1}{k_e})M\Psi^{(n-1)}\]
Wielandt Method

• **Power iteration for modified transport equation**

\[
(L + T - S - \frac{1}{k_e} M)\Psi^{(n+1)} = (\frac{1}{K_{\text{eff}}^{(n)}} - \frac{1}{k_e})M\Psi^{(n)}
\]

\[
\Psi^{(n+1)} = (\frac{1}{K_{\text{eff}}^{(n)}} - \frac{1}{k_e}) \cdot (L + T - S - \frac{1}{k_e} M)^{-1} M\Psi^{(n)}
\]

\[
\Psi^{(n+1)} = \frac{1}{\tilde{K}^{(n)}} \cdot \tilde{F}\Psi^{(n)}
\]

where \( \tilde{K}^{(n)} = (\frac{1}{K_{\text{eff}}^{(n)}} - \frac{1}{k_e})^{-1} \) or \( K_{\text{eff}}^{(n)} = (\frac{1}{\tilde{K}^{(n)}} + \frac{1}{k_e})^{-1} \)

• **How to choose \( k_e \)**
  - \( k_e \) must be larger than \( k_0 \) (but, don't know \( k_0 \)!)  
  - \( k_e \) must be held constant for all of the histories in a batch, but can be adjusted between batches
    - Typically, guess a large initial value for \( k_e \), such as \( k_e=5 \) or \( k_e=2 \)
    - Run a few batches, keeping \( k_e \) fixed, to get an initial estimate of \( K_{\text{eff}} \)
    - Adjust \( k_e \) to a value slightly larger than the estimated \( K_{\text{eff}} \)
    - Run more batches, possibly adjusting \( k_e \) if the estimated \( K_{\text{eff}} \) changes
Wielandt Method

- **Convergence**
  - Eigenfunctions for the Wielandt method are same as for basic power iteration
  - Eigenvalues are shifted:
    \[
    \tilde{k}_j = \left[ \frac{1}{k_j} - \frac{1}{k_e} \right]^{-1} \quad k_e > k_0 > k_1 > ...
    \]

  - Expand the initial guess, substitute into Wielandt method, rearrange to:
    \[
    \Psi^{(n+1)} \approx \text{constant} \cdot \left[ \tilde{u}_0 + \left( \frac{a_1^{(0)}}{a_0^{(0)}} \right) \cdot \left( \frac{k_e - k_0}{k_e - k_1} \cdot \frac{k_1}{k_0} \right)^{n+1} \tilde{u}_1 + ... \right]
    \]

    \[
    K^{(n+1)} \approx k_0 \cdot \left[ 1 + \left( \frac{a_1^{(0)}}{a_0^{(0)}} \right) \cdot \left( \frac{k_e - k_0}{k_e - k_1} \cdot \frac{k_1}{k_0} \right)^n \left( \frac{k_e - k_0}{k_e - k_1} \cdot \frac{k_1}{k_0} - 1 \right) \cdot G_1 + ... \right]
    \]

  - Additional factor \( \frac{(k_e-k_0)}{(k_e-k_1)} \) is less than 1 and positive, so that the red terms die out faster than for standard power iteration
Convergence

- Eigenfunctions for Wielandt method are same as for basic power iteration, but the eigenvalues are shifted

- The dominance ratio for Wielandt method is always smaller than for power iteration

\[ \rho_{\text{Wielandt}} = \frac{k_e - k_0}{k_e - k_1} \cdot \rho_{\text{Power}} \]

\[ \rho = \frac{k_1}{k_0} < 1, \quad k_e > k_0 > k_1 > \ldots \]

Wielandt method will converge in fewer iterations
Monte Carlo Interpretation

- **Power iteration with Wielandt acceleration**

\[(L + T - S - \frac{1}{k_e} M) \Psi^{(n)} = \left( \frac{1}{k_{\text{eff}}}^{(n-1)} - \frac{1}{k_e} \right) M \Psi^{(n-1)}\]

Fission neutrons to follow in **current** iteration

Fission neutron source from **previous** iteration

- **During neutron random walk, at each collision in fissile material:**

Create these neutrons in the **current** iteration

\[n'_e = \left[ \text{wgt} \cdot \frac{\nu \Sigma_F}{\Sigma_T} \cdot \frac{1}{k_e} + \xi \right] \]

Save these neutrons as the source for the **next** iteration

\[n'_F = \left[ \text{wgt} \cdot \frac{\nu \Sigma_F}{\Sigma_T} \cdot \left( \frac{1}{K^{(n-1)}} - \frac{1}{k_e} \right) + \xi \right] \]
Generations vs Iterations

- **Power method**: one neutron generation per iteration
- **Wielandt method**: multiple neutron generations per iteration, varies for each starting neutron

**Standard power iteration (generation model)**
- Initial Guess
- Batch 1 $K_{\text{eff}}^{(1)}$
- Batch 2 $K_{\text{eff}}^{(2)}$
- Batch 3 $K_{\text{eff}}^{(3)}$
- Batch 4 Source

**Wielandt iteration (chain model)**
- Initial Guess
- Batch 1 $K_{\text{eff}}^{(1)}$
- Batch 2 $K_{\text{eff}}^{(2)}$
- Batch 3 $K_{\text{eff}}^{(3)}$
- Batch 4 Source

- **Source particle generation**
- **Monte Carlo random walk**
- **Additional Monte Carlo random walks within batch due to Wielandt method**

- **Neutron**
Choosing \( k_e = k + \Delta \)

- In MCNP, the collision estimator is used for \( k_{\text{eff}}^{(n-1)} \), so that
  \[
  k_e^{(n)} = k_{\text{col}}^{(n-1)} + \Delta
  \]

- For cycle \( n \), average number of fission generations per source neutron

\[
\text{Cycle } n \\
L = 1 + k/\Delta
\]

Neutron generations

1 neutron ➔ 1 neutron

For \( k \sim 1 \):
- \( \Delta = \infty, \ L = 1 \)
- \( \Delta = .5, \ L = 3 \)
- \( \Delta = .05, \ L = 21 \)
- \( \Delta = 1, \ L = 2 \)
- \( \Delta = .1, \ L = 11 \)
- \( \Delta = .01, \ L = 101 \)

Typical:
- \( \Delta = .1, \ .05, \ or \ .025 \)

Smaller \( \Delta \) ➔ larger average chain length, \( L \)
➔ more spread in fission sites each cycle
➔ faster convergence
**Numerical Testing**

2D whole-core PWR test problem (Ueki)

- Ran with different shifts, $\Delta$: $\infty, 1.0, 0.5, 0.2, 0.1$

- Examined convergence of source entropy, $\mathcal{H}_{src} \text{ vs } \Delta$ (plots next page)

- Examined $\text{FOM} = 1 / \sigma^2 T \text{ vs } \Delta$

<table>
<thead>
<tr>
<th>$\Delta$</th>
<th>FOM</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\infty$</td>
<td>168 K</td>
</tr>
<tr>
<td>1.0</td>
<td>188 K</td>
</tr>
<tr>
<td>0.5</td>
<td>212 K</td>
</tr>
<tr>
<td>0.2</td>
<td>188 K</td>
</tr>
<tr>
<td>0.1</td>
<td>184 K</td>
</tr>
</tbody>
</table>

For this problem, FOM was about the same for a range of $\Delta$'s

$\text{DR} = .97$
Numerical Testing

2D PWR test problem

– Wielandt shift parameter: \( K_e^{(n)} = K^{(n-1)}_{\text{collision}} + \Delta \)

Convergence of \( H_{sr} \) vs \( \Delta \)

Iterations for convergence vs

\[ \Delta = \infty \text{ -- black} \]
\[ \Delta = 1 \text{ -- red} \]
\[ \Delta = 0.1 \text{ -- blue} \]
Numerical Testing

- Fuel Storage Vault (Problem OECD_bench1)

DR = .99+
Wielandt Method - Summary

• **Wielandt Method:**
  - Faster convergence rate than power iteration ➔ fewer iterations
  - Some of the particle random walks are moved from the next generation into the current generation ➔ more work per iteration
  - Same total number of random walks ➔ no reduction in CPU time

• **Advantages**
  - Reduced chance of false convergence for very slowly converging problems
  - Reduced inter-generation correlation effects on variance
  - Fission source distribution spreads more widely in a generation (due to the additional particle random walks), which should result in more interactions for loosely-coupled problems

➔ Wielandt method will be included in future versions of MCNP
Superhistory Method

- **Standard generation model, solved by power iteration**
  \[ \Psi^{(n+1)} = \frac{1}{K_{\text{eff}}^{(n)}} \cdot F \Psi^{(n)} \]

- **Superhistory method**
  - Follow several generations (L) before recomputing \( K_{\text{eff}} \) and renormalizing
  \[ \Psi^{(n+1)} = \frac{1}{\tilde{K}^{(n)}} \cdot \tilde{F} \Psi^{(n)}, \quad \text{with} \quad \tilde{F} = F^L, \quad \tilde{K}^{(n)} = (K_{\text{eff}}^{(n)})^L \]

- **Convergence**
  - Same eigenfunctions as standard power iteration
  - Eigenvalues are \( k_0^L, k_1^L, k_2^L, \ldots \)
  - \( DR' = DR^L \), where \( DR \) = dominance ratio for power iteration
  - Fewer iterations, but L generations per iteration \( \Rightarrow \) same work as power iteration
  - Same convergence rate as power iteration

- **Advantages**
  - Reduced correlation between iterations
  - Fewer renormalizations
Superhistory Method

- **Superhistory Method** for Monte Carlo $k$-effective calculation

Example with $L = 2$ generations/batch

- **Initial Guess**
- **Batch 1** $K_{\text{eff}}^{(1)}$
- **Batch 2** $K_{\text{eff}}^{(4)}$
- **Batch 1 Source**
- **Batch 2 Source**
- **Batch 3 Source**

- **Source particle generation**
- **Monte Carlo random walk**

**Neutron**
Dominance Ratio Calculations

(For future versions of MCNP5)
DR - Overview

**Time-series methods for computing DR**

- **Ueki developed a method**
  

- **Recent extensions by Nease & Ueki provide a practical method**
  

- **Recent work by Nease & Brown for MCNP5**
  

  - **Accurate**, regardless of mesh used for collecting statistics
  - Can be used only after source has converged

**Fission Matrix method**

- \( \overline{S} = \frac{1}{k} \overline{F} \cdot \overline{S} \)
  
  \( F_{ij} = \text{prob fission in cell } j, \text{ given fission in cell } i \)

- Tally \( F_{ij} \), then find eigenvalues & eigenvectors of \( F \)

  - **Very old** - used by dozens of researchers


  - **Approximate**, results are very sensitive to mesh
  - Can be used before source has converged
Comparison of Methods

• **Fission matrix method:**
  – Accurate only for fine mesh, large number of mesh cells, N
  – Fission matrix storage varies as $N^2$
    - 10 x 10 x 10 mesh - pretty coarse, but requires $10^6$ storage locations
  – Need to find eigenvalues/vectors of nonsymmetric $N \times N$ matrix

• **Godiva problem example**
  – Fission matrix

<table>
<thead>
<tr>
<th>mesh size</th>
<th>F-matrix size</th>
<th>DR</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 x 2 x 2</td>
<td>8 x 8</td>
<td>.560</td>
</tr>
<tr>
<td>4 x 4 x 4</td>
<td>64 x 64</td>
<td>.602</td>
</tr>
<tr>
<td>8 x 8 x 8</td>
<td>512 x 512</td>
<td>.646</td>
</tr>
</tbody>
</table>

  – Coarse Mesh & time series: .677 +- .033
Example

- From Nease & Ueki (NSE, Sept 2007)
  - 1-group, 2D problem
  - DR from previous work (Ueki) = 0.9993 ± 0.0004
  - DR fission matrix using (4x4x1)^2 = 0.988
    using (9x9x1)^2 = 0.993
    using (18x18x1)^2 = 0.997
  - DR using CMM + time series method = 0.998 ± 0.002
MCNP5 Implementation

• Both methods for DR computation were added to test version of MCNP5
• Negligible extra CPU time for either method

• Fission matrix DR
  – Can be determined early, before convergence
  – Sensitive to mesh size
  – Provides approximate DR
  – Useful for characterizing problem convergence
  – May be useful for automated convergence tests

• Coarse Mesh Method with time series analysis for DR
  – Can only be used after convergence
  – Independent of mesh size
  – Provides accurate DR
Conclusions

• Shannon entropy of the fission distribution is a highly effective means of characterizing convergence of the fission distribution
  – MCNP5 (version 1.40) computes & plots Shannon entropy as an important new tool for assessing problem convergence.
  – It is highly recommended that both $k_{\text{eff}}$ and $H_{\text{src}}$ be carefully checked for convergence in all Monte Carlo criticality calculations.

• Wielandt's method improves convergence & reduces inter-cycle correlation, without significant changes in CPU time
  – Helps to prevent false convergence assessment
  – Eliminates nonconservative underprediction of confidence intervals

• Can now reliably calculate the dominance ratio using Monte Carlo
  – Fission matrix for early, approximate DR
  – Coarse-mesh method with time-series for later, accurate DR
References

Monte Carlo Methods


Monte Carlo k-effective Calculations


Superhistory Method


Wielandt Method


References

Source convergence, Shannon entropy, & dominance ratio

References

Source convergence, Shannon entropy, & dominance ratio

Eigenvalue Calculations – III
Case Studies

Concerns for MC Eigenvalue Calculations

- **Monte Carlo Criticality Calculations**
  - Methodology
  - Concerns

- **Numerical Results**
  - $K_{\text{eff}}$ of the World Problem
  - 1/4-Core PWR Problem
  - Criticality Safety Problem

- **Best Practices**
  - Discussion
  - Conclusions
MC Criticality Calculations - Methodology & Concerns
Introduction

• **Several fundamental problems with MC criticality calculations were identified in the 1960s - 1980s:**
  – Convergence of $K_{eff}$ & source distribution
  – Bias in $K_{eff}$ & tallies
  – Underprediction bias in tally statistics

  (see Lieberoth, Gelbard & Prael, Gast & Candelore, Brissenden & Garlick)

• **These problems are well-understood & can be readily avoided, if some simple "best practices" guidelines are followed**

• **Previous discussion of details:**
  – 2008 - PHYSOR - Monte Carlo workshop
  – 2009 - M&C - Monte Carlo workshop
  – 2010 - PHYSOR - Monte Carlo workshop

Presentations available at
http://mcnp.lanl.gov/publication/mcnp_publications.html
Concerns

Power Iteration for MC Criticality Calculations

Initial Guess

Generation 1 $K_{\text{eff}}^{(1)}$

Generation 2 $K_{\text{eff}}^{(2)}$

Generation 3 $K_{\text{eff}}^{(3)}$

Generation 4 $K_{\text{eff}}^{(4)}$

Convergence of $K_{\text{eff}}$ & fission distribution

Bias in average $K_{\text{eff}}$ & tallies

Bias in statistics for tallies

$K_{\text{eff}}^{(n)}$

Iteration, n

Monte Carlo Deterministic ($S_n$)
Convergence

- Monte Carlo codes use power iteration to solve for $K_{\text{eff}}$ & $\Psi$ for eigenvalue problems

- Power iteration convergence is well-understood:

  $n = \text{cycle number, } k_0 u_0 - \text{fundamental, } k_1 u_1 - 1\text{st higher mode}$

  $$\Psi^{(n)}(\vec{r}) = \tilde{u}_0(\vec{r}) + a_1 \cdot \rho^n \cdot \tilde{u}_1(\vec{r}) + \ldots$$

  $$K_{\text{eff}}^{(n)} = k_0 \cdot \left[ 1 - \rho^{n-1}(1 - \rho) \cdot g_1 + \ldots \right]$$

  - First-harmonic source errors die out as $\rho^n$, $\rho = k_1 / k_0 < 1$
  - First-harmonic $K_{\text{eff}}$ errors die out as $\rho^{n-1}(1 - \rho)$
  - Source converges slower than $K_{\text{eff}}$

- Most codes only provide tools for assessing $K_{\text{eff}}$ convergence.

→ MCNP5 also looks at Shannon entropy of the source distribution, $H_{\text{src}}$. 
Bias in $K_{\text{eff}}$ & Tallies

- **Power iteration is used for Monte Carlo $K_{\text{eff}}$ calculations**
  - For one cycle (iteration):
    - $M_0$ neutrons start
    - $M_1$ neutrons produced, $E[M_1] = K_{\text{eff}} \cdot M_0$
  - At end of each cycle, must renormalize by factor $M_0 / M_1$
  - Dividing by stochastic quantity ($M_1$) introduces bias in $K_{\text{eff}}$ & tallies

- **Bias in $K_{\text{eff}}$, due to renormalization**

  $$\text{Bias in } K_{\text{eff}} \propto \frac{1}{M}$$

  - Power & other tally distributions are also biased, produces “tilt”
Bias in Statistics

- MC eigenvalue calculations are solved by power iteration
  - Tallies for one generation are spatially correlated with tallies in successive generations
  - The correlation is positive
  - MCNP & other MC codes ignore this correlation, so computed statistics are smaller than the real statistics
  - Errors in statistics are small/negligible for $K_{\text{eff}}$, may be significant for local tallies (eg, fission distribution)
  - Running more cycles or more neutrons/cycle does not reduce the underprediction bias in statistics
  - $(\text{True } \sigma^2) > (\text{computed } \sigma^2)$, since correlations are positive

\[
\frac{\text{True } \sigma_X^2}{\text{Computed } \sigma_X^2} = \frac{\sigma_X^2}{\hat{\sigma}_X^2} \approx 1 + 2 \cdot \left( \text{sum of lag-i correlation} \right) \left( \text{coeff's between tallies} \right)
\]
Numerical Results

$K_{\text{eff}}$ of the World Problem
Introduction

Elliot Whitesides, 1971:

… if one attempts to calculate the $k_{\text{eff}}$ of the world using a Monte Carlo calculation, what $k_{\text{eff}}$ would be computed assuming that there are several critical assemblies located around the world?

The answer would likely be the $k_{\text{eff}}$ of the world with no critical assemblies present. …

… The erroneous results for these types of problems are the result of the failure of the calculation to converge the source to the fundamental source mode. …

… unless the correct fission distribution is achieved, the results will most likely be nonconservative.
Whitesides' Model Problem

9 x 9 x 9 array of Pu-239 spheres
- 729 spheres
- Void between spheres
- Surrounded by 30 cm water
- Sphere radii ~ 4 cm
- Pitch = 60 cm

- MCNP5-1.60 + ENDF/B-VII.0 data
- For uniform array of identical spheres with surrounding water, sphere radii adjusted to \( r = 3.9 \) cm, so that
  \[ K_{eff} = 0.9328 \pm 0.0002 \]
- Single bare sphere, \( r=4.928 \) cm,
  \[ K_{eff} = 1.0001 \pm 0.0002 \]
- Whitesides' model problem:
  Replace center sphere of array by larger (critical) sphere

Should be supercritical - is it?
Whitesides' Problem, circa 1971

- Due to severe computer limitations ~1971, KENO defaults were:
  - 300 neutrons/cycle
  - Discard first 3 cycles
  - Run 100 more cycles

- If MCNP5 is run using the 1971 KENO defaults, 200 independent replica calculations give:
  - Average of 200 replicas: \( K_{\text{eff}} = 0.9431 \pm 0.0010 \)
  - None of the 200 calculations produced \( K_{\text{eff}} > 1 \)
  - Distribution of replica results:

  ![Distribution of \( K_{\text{eff}} \) results](image_url)
Convergence

$K_{\text{eff}}$ vs cycle, various $M$

$M = \text{neutrons/cycle}$

$K_{\text{eff}}$ converges in 75-100 cycles

$H_{\text{src}}$ vs cycle, various $M$

$M = \text{neutrons/cycle}$

$H_{\text{src}}$ converges in 100-150 cycles

Must discard 150 or more initial cycles

Convergence depends on the dominance ratio & source guess, NOT on neutrons/cycle

Initial source guess = uniform sampling of points at sphere centers
**K_{eff} Bias**

![Graph showing K_{eff} vs 1/M](image)

**Notes:**
- All cases discarded the first 150 cycles
- All cases used 10M neutrons in active cycles
- All cases: $\sigma \sim 0.00025$, smaller than plot markers

**Historical note:**
When this problem was first proposed in 1971, the default batch size for KENO was 300 neutrons/cycle
$K_{\text{eff}}$ Bias

Distribution of $K_{\text{eff}}$ for 200 replicas, various $M = \text{neuts/cycle}$

- $M=200$
- $M=300$
- $M=500$
- $M=10K$

Frequency of $K_{\text{eff}}$, for replicas vs. $K_{\text{eff}}$
Discussion & Conclusions

• The original 1971 version suffered from:
  – Computers: small memory & slow
  – Discard only 3 cycles: not converged
  – 300 neutrons/cycle: $K_{\text{eff}}$ bias - too low, nonconservative
  – 300 neutrons/cycle: undersampled the source (729 spheres)

  – No tools were available for diagnosing fission distribution convergence (today, we have Shannon entropy & other diagnostics)

• If (1) enough initial cycles are discarded (150 or more), and (2) enough neutrons/cycle are used (10K or more),

  then the "K-effective of the World" problem is actually not a difficult problem to solve
Numerical Results
- 1/4-Core PWR
Example Problem - Reactor

2D quarter-core PWR  (Nakagawa & Mori model)

- **48 1/4 fuel assemblies:**
  - 12,738 fuel pins with cladding
  - 1206 1/4 water tubes for control rods or detectors

- **Each assembly:**
  - Explicit fuel pins & rod channels
  - 17x17 lattice
  - Enrichments: 2.1%, 2.6%, 3.1%

- **Dominance ratio ~ .96**

- 125 M active neutrons for each calculation
- ENDF/B-VII data, continuous-energy
- Tally fission rates in each quarter-assembly
Convergence

\[ K_{\text{eff}} \text{ converges sooner than the fission distribution} \]

\[ H_{\text{src}} = \]

- Shannon entropy of fission source distribution
- A metric for assessing convergence of the distribution
- Computed/plotted by MCNP
Bias in $K_{\text{eff}}$

$N = \# \text{ cycles}$

$M = \text{ neutrons/cycle}$

$N \cdot M = \text{ constant for all calculations}$

$0.0003 \Delta k$
### Bias in Tallies

<table>
<thead>
<tr>
<th></th>
<th>0.0</th>
<th>-0.5</th>
<th>-0.6</th>
<th>-0.2</th>
<th>-0.3</th>
<th>0.5</th>
<th>0.8</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.2</td>
<td>-0.7</td>
<td>-0.8</td>
<td>0.1</td>
<td>0.3</td>
<td>0.7</td>
<td>0.6</td>
<td></td>
</tr>
<tr>
<td>-0.5</td>
<td>-0.7</td>
<td>-0.7</td>
<td>0.0</td>
<td>0.3</td>
<td>0.7</td>
<td>1.0</td>
<td>1.3</td>
</tr>
<tr>
<td>-0.1</td>
<td>-0.7</td>
<td>-0.8</td>
<td>0.2</td>
<td>0.3</td>
<td>0.8</td>
<td>1.1</td>
<td>1.2</td>
</tr>
<tr>
<td>-0.4</td>
<td>-0.6</td>
<td>-0.5</td>
<td>0.0</td>
<td>-0.1</td>
<td>0.2</td>
<td>0.7</td>
<td>0.6</td>
</tr>
<tr>
<td>-0.7</td>
<td>-0.9</td>
<td>-0.8</td>
<td>-0.7</td>
<td>0.2</td>
<td>0.5</td>
<td>0.4</td>
<td>1.0</td>
</tr>
<tr>
<td>-0.6</td>
<td>-0.3</td>
<td>-0.7</td>
<td>-0.6</td>
<td>0.3</td>
<td>0.8</td>
<td>1.1</td>
<td>1.2</td>
</tr>
<tr>
<td>-0.5</td>
<td>-0.8</td>
<td>-1.0</td>
<td>-0.8</td>
<td>0.2</td>
<td>0.8</td>
<td>0.9</td>
<td>1.2</td>
</tr>
<tr>
<td>-0.5</td>
<td>-0.9</td>
<td>-0.8</td>
<td>-1.0</td>
<td>-0.6</td>
<td>0.2</td>
<td>0.2</td>
<td>0.6</td>
</tr>
<tr>
<td>-0.9</td>
<td>-0.9</td>
<td>-1.1</td>
<td>-1.0</td>
<td>-0.9</td>
<td>-0.1</td>
<td>0.2</td>
<td>0.6</td>
</tr>
<tr>
<td>-1.2</td>
<td>-1.3</td>
<td>-1.2</td>
<td>-1.0</td>
<td>-0.6</td>
<td>-0.5</td>
<td>-0.3</td>
<td>0.2</td>
</tr>
<tr>
<td>-1.3</td>
<td>-1.5</td>
<td>-1.0</td>
<td>-0.9</td>
<td>-0.7</td>
<td>-0.5</td>
<td>-0.6</td>
<td>0.3</td>
</tr>
<tr>
<td>-1.7</td>
<td>-1.5</td>
<td>-1.1</td>
<td>-1.1</td>
<td>-0.6</td>
<td>-0.5</td>
<td>-0.2</td>
<td>-0.1</td>
</tr>
<tr>
<td>-1.5</td>
<td>-1.5</td>
<td>-1.4</td>
<td>-1.0</td>
<td>-1.1</td>
<td>-0.8</td>
<td>0.0</td>
<td>0.1</td>
</tr>
<tr>
<td>-1.6</td>
<td>-1.6</td>
<td>-1.2</td>
<td>-1.2</td>
<td>-0.6</td>
<td>-0.7</td>
<td>-0.4</td>
<td>-0.2</td>
</tr>
</tbody>
</table>

Percent errors in 1/4-assembly fission rates using 500 neutrons/cycle

Errors of -1.7% to +3.2%

Statistics ~ .1% to .3%

Reference: ensemble-average of 25 independent calculations, with 25 M neutrons each & 20K neutrons/cycle
Bias in Tallies

Percent error in fission rates along diagonal
## Bias in $\sigma$'s

<table>
<thead>
<tr>
<th></th>
<th>3.4</th>
<th>3.1</th>
<th>2.7</th>
<th>2.6</th>
<th>2.3</th>
<th>2.7</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.3</td>
<td>3.7</td>
<td>3.6</td>
<td>3.7</td>
<td>3.7</td>
<td>2.7</td>
<td>2.9</td>
</tr>
<tr>
<td>3.8</td>
<td>3.8</td>
<td>3.9</td>
<td>4.0</td>
<td>3.6</td>
<td>3.3</td>
<td>3.0</td>
</tr>
<tr>
<td>3.8</td>
<td>3.9</td>
<td>4.2</td>
<td>3.3</td>
<td>3.5</td>
<td>3.4</td>
<td>3.2</td>
</tr>
<tr>
<td></td>
<td>3.6</td>
<td>3.5</td>
<td>3.4</td>
<td>4.0</td>
<td>3.9</td>
<td>3.5</td>
</tr>
<tr>
<td>3.9</td>
<td>3.6</td>
<td>3.5</td>
<td>3.3</td>
<td>4.0</td>
<td>3.4</td>
<td>3.0</td>
</tr>
<tr>
<td>3.4</td>
<td>3.4</td>
<td>3.2</td>
<td>3.5</td>
<td>2.6</td>
<td>3.0</td>
<td>2.9</td>
</tr>
<tr>
<td>4.2</td>
<td>3.5</td>
<td>3.4</td>
<td>3.1</td>
<td>2.7</td>
<td>2.3</td>
<td>2.0</td>
</tr>
<tr>
<td></td>
<td>3.6</td>
<td>3.1</td>
<td>2.9</td>
<td>2.3</td>
<td>2.1</td>
<td>2.3</td>
</tr>
<tr>
<td>3.7</td>
<td>3.3</td>
<td>3.6</td>
<td>2.4</td>
<td>2.2</td>
<td>2.2</td>
<td>2.5</td>
</tr>
<tr>
<td></td>
<td>3.0</td>
<td>3.1</td>
<td>2.2</td>
<td>2.2</td>
<td>2.1</td>
<td>2.4</td>
</tr>
<tr>
<td>2.9</td>
<td>3.7</td>
<td>3.3</td>
<td>2.6</td>
<td>2.5</td>
<td>2.8</td>
<td>3.0</td>
</tr>
<tr>
<td></td>
<td>3.2</td>
<td>3.1</td>
<td>2.9</td>
<td>3.1</td>
<td>3.5</td>
<td>3.2</td>
</tr>
<tr>
<td>3.4</td>
<td>3.0</td>
<td>3.1</td>
<td>3.6</td>
<td>3.4</td>
<td>3.5</td>
<td>3.7</td>
</tr>
<tr>
<td></td>
<td>3.5</td>
<td>3.2</td>
<td>2.8</td>
<td>3.5</td>
<td>3.9</td>
<td>4.1</td>
</tr>
<tr>
<td>3.5</td>
<td>3.2</td>
<td>2.8</td>
<td>3.5</td>
<td>3.8</td>
<td>3.9</td>
<td>4.1</td>
</tr>
</tbody>
</table>

True relative errors in 1/4-assembly fission rates, as multiples of calculated relative errors, $\sigma_{\text{TRUE}} / \sigma_{\text{MCNP}}$

Calculated uncertainties are 1.7 to 4.7 times smaller than true uncertainties.

Average factor = 3.1
Numerical Results
- Crit-Safety Problem
Example Problem - Criticality Safety

2 x 3 array of steel cans containing plutonium nitrate solution

From MCNP Criticality Primer (chap 5) & MCNP Criticality Classes
Convergence

\[ H_{\text{src}}, \text{ initial source in center of solution in 1 can} \]
\[ H_{\text{src}}, \text{ initial source in center of solution in all cans} \]
\[ H_{\text{src}}, \text{ initial source uniform in solution in all cans} \]

\[ k_{\text{eff}}, \text{ initial source in center of solution in 1 can} \]
\[ k_{\text{eff}}, \text{ initial source in center of solution in all cans} \]
\[ k_{\text{eff}}, \text{ initial source uniform in solution in all cans} \]
Bias in $K_{\text{eff}}$

Note: Bias in green point is a convergence problem due to using Keno default - discard 3 cycles, 203 cycles total

Using ENDF/B-VI+T16 data
Best Practices For
MC Criticality Problems
Convergence - Guidance

- **Plot** $K_{\text{eff}}$ vs cycle to check convergence of $K_{\text{eff}}$

- **If computing any tallies (flux, fissions, dose, foils, heating, ...)** plot $H_{\text{src}}$ vs cycle to check convergence of fission distribution

- **Dominance ratio** $\rho = k_1 / k_0$ determines the rate of convergence
  - Smaller dominance ratio $\Rightarrow$ fewer cycles to converge
  - To reduce the dominance ratio, use problem symmetry & reflecting boundary, to eliminate some higher modes

    PWR example:
    
    |          | full core | 1/2 core | 1/4 core | 1/8 core |
    |----------|-----------|----------|----------|----------|
    | $\rho$:  | .98       | .97      | .96      | .94      |

- **Better initial source guess** $\Rightarrow$ fewer cycles to converge
  - Reactor: good guess - uniform in core region
  - Criticality Safety: good guess - points in each fissionable region, good guess - uniform in each fissionable region

- **Convergence does not depend on number of neutrons/cycle ($M$)**
Bias in $K_{\text{eff}}$ & Tallies - Guidance

- Using too few neutrons/cycle leads to bias in $K_{\text{eff}}$ & the fission distribution

- Bias in $K_{\text{eff}}$ is usually small, but always negative (nonconservative)

- Bias in the fission distribution is generally larger than for $K_{\text{eff}}$ & shows a significant tilt

- Practical solution - use large M (neutrons/cycle)
  - Using 10K neutrons/cycle or more $\Rightarrow$ bias negligible (100K or more for large models)
  - More neutrons/cycle $\Rightarrow$ more efficient parallel calculations
Bias in $\sigma'$s - Guidance

- Uncertainties computed by MC codes exhibit a bias due to inter-cycle correlation effects that are neglected.

- Primarily affects local tally statistics, not K-effective statistics.

- Computed uncertainties are always smaller than the true uncertainties for a tally.

- Running more cycles or more neutrons/cycle does not reduce the biases.

- Wielandt’s method can reduce or eliminate the underprediction bias in uncertainties (coming soon in MCNP5...).
Best Practices - Summary

• To avoid bias in $K_{eff}$ & tally distributions:
  - Use 10K or more neutrons/cycle (maybe 100K+ for full-core)
  - Discard sufficient initial cycles
  - Always check convergence of both $K_{eff}$ & the fission distribution

• To help with convergence:
  - Take advantage of problem symmetry, if possible
  - Use good initial source guess, cover fissionable regions

• Run at least a few 100 active cycles to allow codes to compute reliable statistics

• Statistics on tallies from codes are underestimated, often by 2-5x; possibly make multiple independent runs
References


• Previous discussion of details concerning bias, convergence, & statistics and "Best Practices" previously presented at
  – 2008 - PHYSOR Monte Carlo workshop
  – 2009 - M&C Monte Carlo workshop
  – 2009 - Paper at NCSD topical meeting
  – 2010 PHYSOR Monte Carlo Workshop

References

Shannon entropy & convergence


Bias in Keff & Tallies

E.M. Gelbard and R.E. Prael, "Monte carlo Work at Argonne National Laboratory", in Proc. NEACRP Meeting of a Monte Carlo Study Group, ANL-75-2, Argonne National Laboratory, Argonne, IL (1974).


Correlation & Bias in Uncertainties


E.M. Gelbard and R.E. Prael, "Monte carlo Work at Argonne National Laboratory", in Proc. NEACRP Meeting of a Monte Carlo Study Group, ANL-75-2, Argonne National Laboratory, Argonne, IL (1974).


References

Monte Carlo Methods


Monte Carlo k-effective Calculations


Wielandt Method


Variance Reduction
Monte Carlo Calculations

**Geometry**
- Which cell is particle in?
- What will it hit next?
- How far to boundary?
- What’s on other side?
- Survival?

**Physics**
- How far to collision?
- Which nuclide?
- New E, direction?
- Secondaries?
- Survival?

**Tallies**
- Tally events of interest
- Compute results
- Compute statistics
- Balances
- Performance stats

- **Variance reduction**
  - Modify the PDFs for physics interactions to favor events of interest
  - Use splitting/rouletting to increase particles in certain geometric regions
  - Kill particles in uninteresting parts of problem
- **May be necessary in order to sample rare events**
- **More samples (with less weight each) ➔ smaller variance in tallies**

mcnp, rcp, vim, racer, sam-ce, tart, morse, keno, tripoli, mcbend, monk, o5r, recap, andy,…..
Monte Carlo Estimates of Integrals

Given a function $R(x)$, where $x$ is a random variable with PDF $f(x)$,

- Expected value of $R(x)$ is  
  $$\mu = \int R(x) \, f(x) \, dx$$

- Variance of $R(x)$ is  
  $$\sigma^2 = \int R^2(x) \, f(x) \, dx - \mu^2$$

Monte Carlo method for estimating $\mu = \langle R \rangle$

- make $N$ random samples $\hat{x}_j$ from $f(x)$
- Then

  $$\bar{R} \approx \frac{1}{N} \sum_{j=1}^{N} R(\hat{x}_j)$$

  $$\sigma_R^2 \approx \frac{1}{N-1} \left( \frac{1}{N} \sum_{j=1}^{N} R^2(\hat{x}_j) - \bar{R}^2 \right)$$
Variance Reduction - Basic Idea

- Expected mean score is not changed by variance reduction

\[ \mu = \int R(x) f(x) \, dx = \int R(x) \left( \frac{f(x)}{g(x)} \right) \cdot g(x) \, dx \]

- Sample \( x' \) from \( f(x) \)
- Tally \( R(x') \)

- Sample \( x' \) from \( g(x) \)
- Tally \( R(x') \cdot f(x')/g(x') \)

- Variance is changed due to altered sampling scheme

\[ \sigma^2 = \int [R(x)]^2 f(x) \, dx - \mu^2 \]

\[ \sigma^2 = \int \left[ R(x) \frac{f(x)}{g(x)} \right]^2 g(x) \, dx - \mu^2 \]

Goal: Choose \( g(x) \) such that variance is reduced
Review

• Given a set of random samples, \( x_1, x_2, \ldots, x_N \),
  
  – Mean
  \[
  \bar{x} = \frac{1}{N} \sum_{j=1}^{N} x_j
  \]

  – Variance of the mean
  \[
  \sigma_{\bar{x}}^2 = \frac{1}{N-1} \left( \frac{1}{N} \sum_{j=1}^{N} x_j^2 - \bar{x}^2 \right)
  \]

  – Relative Error
  \[
  \text{RE} = \frac{\sigma_{\bar{x}}}{\bar{x}}
  \]

  – Figure of Merit
  \[
  \text{FOM} = \frac{1}{\text{RE}^2 \cdot T}
  \]

• Variance reduction: Reduce RE or T, to increase FOM
Analog vs. Weighted Monte Carlo

- **Analog Monte Carlo**
  - Faithful simulation of particle histories
  - No alteration of PDFs (i.e., no biasing or variance reduction)
  - Particle is born with weight = 1.0
  - Weight unchanged throughout history until particle is killed
  - Scores are weighted by 1.0 when tallying events of interest

- **Weighted Monte Carlo (non-analog)**
  - Alter the PDFs to favor events of interest
  - Particle is born with weight = 1.0
  - Weight, \( wgt \), is altered if biased PDF is used
  - Weight can also be changed by Russian roulette/splitting & other variance reduction techniques
  - Scores are weighted by \( wgt \) when tallying events of interest
Variance Reduction - General Approaches

- **Truncation**
  - Remove particles from parts of phase space that do not contribute significantly to the tallies

- **Population control**
  - Use particle splitting and Russian rouletting to control the number of samples taken in various regions of phase space

- **Modified sampling**
  - Modify the PDFs representing problem physics, to favor tallies of interest

- **Deterministic methods**
  - Replace portions of a particle random walk by the expected results obtained from a deterministic calculation
Typical Variance Reduction Techniques

- **MCNP has 14 variance reduction techniques**
  1. Time and energy cutoffs
  2. Geometry splitting & roulette
  3. Weight windows
  4. Exponential transform
  5. Forced collisions
  6. Energy splitting & roulette
  7. Time splitting & roulette
  8. Point and ring detectors
  9. DXTRAN
  10. Implicit capture
  11. Weight cutoff
  12. General source biasing
  13. Secondary particle biasing
  14. Bremsstrahlung energy biasing
Survival Biasing

- Also called implicit absorption or non-absorption weighting
- Modify collision process according to expected outcome
- Particle always survives collision
  - Tally expected absorption, \( \text{wgt} \cdot \left( \frac{\Sigma_A}{\Sigma_T} \right) \)
  - Reduce weight of surviving particle, \( \text{wgt}' = \text{wgt} \cdot \left( 1 - \frac{\Sigma_A}{\Sigma_T} \right) \)
- Extends particle history so that more particles reach events which occur after many collisions
- Most effective for thermal reactor problems, but doesn't hurt in other types of problems
- Must also use some form of low-weight cutoff to eliminate particles with very low weight
Geometry Splitting & Russian Roulette

• Increase the number of particles in "important" regions, decrease the number of particles in "unimportant" regions

• Assign each cell an importance, $I_{\text{cell}}$
  – Arbitrary, use best guess or adjoint fluxes from deterministic calculation
  – Could use one value for all energies or separate values for different energy ranges
  – Higher value --> more important
  – $I_{\text{cell}} > 0$
  – $I_{\text{cell}}=0$ is a way to declare regions as not in physical problem
  – Values of $I_{\text{cell}}$ must not change during Monte Carlo calculation

• Modify random walk simulation at surface crossings:
  – If $(I_{\text{enter}}/I_{\text{leave}}) > 1$, perform splitting
  – If $(I_{\text{enter}}/I_{\text{leave}}) < 1$, perform Russian roulette
Geometry Splitting & Russian Roulette

- Let \( r = \frac{I_B}{I_A} \)
  \[ n = \text{floor}(r) \]

- If \( n > 1 \), split into \( n \) particles with weight \((\text{wgt}/n)\)
  - All of the \( n \) particles emerging from splitting have identical attributes (e.g., \( x, y, z, u, v, w, E \)) including \( \text{wgt}' = \text{wgt}/n \)
  - All of the \( n \) particles from a splitting are part of the same history, and their tallies must be combined
  - Typically, \((n-1)\) particles are banked, 1 particle is followed until its death, then a particle is removed from the bank & followed, etc.

- Avoid over-splitting
  - Splitting into a large number of particles can increase CPU-time & lead to (apparent) bias in results
  - Typically, choose cell importances to split 2-for-1 or 3-for-1
  - Typically, can limit the splitting to \( n \)-for-1 or less

- Total particle weight is exactly conserved in splitting
Geometry Splitting & Russian Roulette

- Let \( r = \frac{I_B}{I_A} \)

- If \( r < 1 \), play Russian roulette
  - With probability \( r \), keep the particle & alter its weight to \( \frac{\text{wgt}}{r} \)
  - With probability \( 1 - r \), kill the particle (set its weight to 0)

\[
\begin{align*}
\text{if } \xi < r, & \quad \text{wgt}' = \frac{\text{wgt}}{r} \\
\text{else} & \quad \text{wgt}' = 0
\end{align*}
\]

- Russian roulette effectively merges a number of low-weight particles into one with higher weight

- Total particle weight is only conserved statistically (expected value)
Weight Cutoff

- Specify a cutoff weight, $W_{\text{low}}$, and a survival weight, $W_{\text{ave}}$

- If particle weight drops below $W_{\text{low}}$, play Russian roulette with weight of $W_{\text{ave}}$ for survivors
  - Probability of surviving RR = $\frac{\text{wgt}}{W_{\text{ave}}}$
  - Probability of being killed = $1 - \frac{\text{wgt}}{W_{\text{ave}}}$

If $\text{wgt} < W_{\text{low}}$,

  if $\xi < \frac{\text{wgt}}{W_{\text{ave}}}$,
    \[ \text{wgt}' = W_{\text{ave}} \]
  else
    \[ \text{wgt}' = 0 \]

- Expected value of surviving weight is conserved, $(\frac{\text{wgt}}{W_{\text{ave}}}) \cdot W_{\text{ave}}$
Weight Cutoff

• In some codes (e.g., MCNP), the weight cutoff parameters are functions of cell importance

  – Let \( R_j = \frac{\text{importance of source cell}}{\text{importance of cell } j} \)

  – Then, \( W_{\text{ave}}(j) = W_{\text{ave}} \cdot R_j \)
  \( W_{\text{low}}(j) = W_{\text{low}} \cdot R_j \)

• Weight cutoffs reduce computing time, not variance

• Weight cutoffs can be applied anytime the particle weight changes - after collisions, after boundary crossings, ...
Weight Windows

- Prevent particle weights from getting too large or too small
  - Weight too large ➜ splitting
  - Weight too small ➜ Russian Roulette
Weight Windows

- Large fluctuations in particle weights contributing to a tally lead to larger variance

- Weight windows eliminate large or small weights (outside the window) by creating or destroying particles

- Weight windows can be applied any time - after collisions, after surface crossings, ...

\[
\begin{align*}
\text{If} & \quad wgt > W_{\text{hi}} \\
& \quad \text{splitting} \\
\text{Elseif} & \quad wgt < W_{\text{low}} \\
& \quad \text{roulette}
\end{align*}
\]
Weight Windows

- MCNP weight window scheme
  
  **Input:** \( W_{\text{low}} \) for each cell (can be energy or time dependent), 
  \( [W_{\text{ave}}/W_{\text{low}}], [W_{\text{hi}}/W_{\text{low}}], \text{mxspln} \)

  If \( \text{wgt} > W_{\text{hi}} \)
  \[
  n = \min( \text{mxspln}, 1 + \text{wgt}/W_{\text{hi}} ) \quad \text{ <-- max splitting is mxspln-to-1}
  \]
  \( \text{wgt} = \text{wgt}/n \)
  bank \( n-1 \) copies of particle \quad \text{ <-- n-to-1 splitting}

  Elseif \( \text{wgt} < W_{\text{low}} \)
  \[ P = \max( 1/\text{mxspln}, \text{wgt}/W_{\text{ave}} ) \quad \text{ <-- limits survivor to mxspln*wgt} \]
  if \( \xi < P \)
  \( \text{wgt} = \text{wgt}/P \) \quad \text{ <-- particle survives}
  else
  \( \text{wgt} = 0 \) \quad \text{ <-- particle killed}
Source Biasing

- Bias the PDFs used to select the angle, energy, or position or source particles
  - Produce more source particles (with lower weights) in desired parts of phase space

True source: \( f(R, E, \Omega) \)

Sample \( (R', E', \Omega') \) from \( g(R, E, \Omega) \)

& assign weight \( f(R', E', \Omega')/g(R', E', \Omega') \) to source particle

Choose \( g(R, E, \Omega) \) to favor directions more important to tallies
Forced Collisions

- Particles entering specified cells are split into collided & uncollided parts
  - For distance-to-boundary $d$
    \[
    \text{Prob(}\text{no collision}) = \exp(-\Sigma_T d) \\
    \text{Prob(}\text{collision}) = 1 - \exp(-\Sigma_T d)
    \]
Forced Collisions

- **Sampling the flight distance** $s$ **for a forced collision with max flight distance** $d$

**Sampling from a truncated exponential PDF:**

\[
f(s) = \Sigma_T \cdot \frac{e^{-\Sigma Ts}}{1 - e^{-\Sigma Td}}, \quad 0 \leq s \leq d
\]

\[
F(s) = \frac{1 - e^{-\Sigma Ts}}{1 - e^{-\Sigma Td}}
\]

**Solve for** $s$ : \[\xi = F(s)\]

\[
s = \frac{-\ln\left[1 - (1 - e^{-\Sigma Td})\xi\right]}{\Sigma T}
\]
Exponential Transform

- Encourage particles to head in a certain preferred direction, $\Omega_0$

- Replace $\Sigma_T$ by $\Sigma^* = \Sigma_T [1 - p \Omega \cdot \Omega_0]$
  
  $p$ = a parameter, $0 < p < 1$
  
  $\Omega_0$ = unit vector from particle position to detector
  
  $\Omega$ = actual particle direction

- Sample flight distance $s'$ from $g(s) = \Sigma^* \exp(-\Sigma^* s)$

- Adjust weight by factor:
  
  $f(s')/g(s') = \exp(-p \Omega \cdot \Omega_0 \Sigma_T s')/[1 - p \Omega \cdot \Omega_0]$

- Paths toward detector are stretched ($\Sigma^* < \Sigma_T$)

- Paths away from detector are shortened ($\Sigma^* > \Sigma_T$)
Variance Reduction Goals & Cautions

- Maximize FOM - either reduce RE or T
- Keep the number of particles per cell roughly constant from source to detector
- Reduce the number of particles in unimportant regions
- Achieve adequate sampling of all portions of phase space
- Avoid over-biasing (e.g., over-splitting)
- Ensure that tallies pass statistical checks
References


- L.L. Carter & E.D. Cashwell, "Particle Transport Simulation with the Monte Carlo Method", TID-26607, National Technical Information Service (1975)
Parallel Monte Carlo
Perspective

• Fast desktop computers
  1980s super: 200 MHz 16 MB 10 GB $20 M
  Today, Mac Pro: 8 x 3000 MHz 8000 MB 250 GB $5 K

• Linux clusters + MPI + multi-core
  – Cheap parallel computing
  – Everyone can do parallel computing, not just national labs

• Mature Monte Carlo codes
  – MCNP, VIM, KENO, MCBEND, MONK, COG, TART, RACER, RCP, …

• New generation of engineers/scientists
  – Less patience for esoteric theory & tedious computing procedures
  – Computers are tools, not to be worshipped
  – What's a slide rule ???

⇒ More calculations with Monte Carlo codes
Trends in Computing Technology

• **Commodity chips**
   *Through early-2000s (little change since then):*
   - Microprocessor speed ➔ ~2x gain / 18 months
   - Memory size ➔ ~2x gain / 18 months
   - Memory latency ➔ ~ no change (getting worse)

• **High-end scientific computing**
  - Key driver (or limit) ➔ economics: mass production of desktop PCs & commercial servers
  - Architecture ➔ clusters: with moderate number of commodity microprocessors on each node
    - **multicore:** multiple CPUs per processor permits threading within each node

• **Operating systems**
  - Desktop & server ➔ Windows, Linux
  - Supercomputers ➔ Unix, Linux

**CPU performance on supercomputer ➔ same as desktop PC**
**High-performance scientific computing ➔ parallel computing**
Parallel Computers

1 GFLOP = $10^9$ FLOP
1 TFLOP = $10^{12}$ FLOP
1 PFLOP = $10^{15}$ FLOP
1 EFLOP = $10^{18}$ FLOP

Teraflop computers $\rightarrow \sim 10^3$ processors
Petaflop computers $\rightarrow \sim 10^5$ processors
Exaflop computers $\rightarrow \sim 10^7$ processors (?)
Parallel Computers

• Characterize computers by:
  – CPU: scalar, vector, superscalar, RISC, ..... 
  – Memory: shared, distributed, cache, banks, bandwidth, ..... 
  – Interconnects: bus, switch, ring, grid, ..... 

• Basic types:
Approaches to Parallel Processing

High-level
- Independent programs + message-passing
- Distribute work among processors
- Loosely-coupled
- Programmer must modify high-level algorithms

Mid-level
- Threads (task-level)
- Independent tasks (subprograms) + shared memory
- For shared memory access, use locks on critical regions
- Compiler directives by programmers

Low-level
- Threads (loop-level)
- Split DO-loop into pieces, compute, synchronize
- Compiler directives by programmers

Low-level
- Pipelining or vectorization
- Pipelined execution of DO-loops
- Automatic vectorization by compilers &/or hardware, or compiler directives by programmers
Message-passing

- Independent programs
- Separate memory address space for each program (private memory)
- All control information & data must be passed between programs by explicit messages (SENDs & RECEIVEs)
- Can run on distributed or shared memory systems
- Efficient only when $T_{\text{computation}} \gg T_{\text{messages}}$
- Standard message-passing:
  - MPI
Threading (task-level)

- Single program, independent sections or subprograms
- Each thread executes a portion of the program
- Common address space, must distinguish private & shared data
- Critical sections must be "locked"
- Can run only on shared memory systems, not distributed memory
- Thread control by means of compiler directives
- Standard threading:
  - OpenMP

```
program A
    ...
    !$omp parallel
    call trnspt
    !$omp end parallel
    ...
end program A

subroutine trnspt
    ...
    return
end subroutine trnspt
```

```
subroutine trnspt
    ...
    return
end subroutine trnspt
```

Address space for Program A
Amdahl's Law

If a computation has fast (parallel) and slow (scalar) components, the overall calculation time will be dominated by the slower component.

\[
\text{Overall System} = \text{Single CPU} \times \frac{1}{1 - F + \frac{F}{N}}
\]

where \( F \) = fraction of work performed in parallel
\( N \) = number of parallel processors

\[
\text{Speedup} = \frac{1}{1 - F + \frac{F}{N}}
\]

For \( N=10 \)

<table>
<thead>
<tr>
<th>( F )</th>
<th>( S )</th>
<th>( F )</th>
<th>( S )</th>
</tr>
</thead>
<tbody>
<tr>
<td>20%</td>
<td>1.2</td>
<td>90%</td>
<td>5.3</td>
</tr>
<tr>
<td>40%</td>
<td>1.6</td>
<td>95%</td>
<td>6.9</td>
</tr>
<tr>
<td>60%</td>
<td>2.2</td>
<td>99%</td>
<td>9.2</td>
</tr>
<tr>
<td>80%</td>
<td>3.6</td>
<td>99.5%</td>
<td>9.6</td>
</tr>
</tbody>
</table>

For \( N=\infty \)

<table>
<thead>
<tr>
<th>( F )</th>
<th>( S )</th>
<th>( F )</th>
<th>( S )</th>
</tr>
</thead>
<tbody>
<tr>
<td>20%</td>
<td>1.3</td>
<td>90%</td>
<td>10</td>
</tr>
<tr>
<td>40%</td>
<td>1.7</td>
<td>95%</td>
<td>20</td>
</tr>
<tr>
<td>60%</td>
<td>2.5</td>
<td>99%</td>
<td>100</td>
</tr>
<tr>
<td>80%</td>
<td>5</td>
<td>99.5%</td>
<td>200</td>
</tr>
</tbody>
</table>
Amdahl's Law

My favorite example .....  

Which system is faster?

System A:  (16 processors)•(1 GFLOP each)  =  16 GFLOP total

System B:  (10,000 procs)•(100 MFLOP each)  =  1,000 GFLOP total

Apply Amdahl's law, solve for F:

\[
\frac{1}{(1-F + F/16)} = \frac{0.1}{(1-F + F/10000)}
\]

→ System A is faster, unless >99.3% of work is parallel

• In general, a smaller number of fatter nodes is better
• For effective parallel speedups, must parallelize everything
Parallel Monte Carlo
Parallel Algorithms

- **Possible parallel schemes:**
  - Jobs run many sequential MC calculations, combine results
  - Functional sources, tallies, geometry, collisions, .....
  - Phase space space, angle, energy
  - Histories Divide total number of histories among processors

- **All successful parallel Monte Carlo algorithms to date have been history-based.**
  - Parallel jobs always works, variation on parallel histories
  - Some limited success with spatial domain decomposition
Boss / Worker Algorithm (Simple)

- **Boss task:** control + combine tallies from each Worker
- **Worker tasks:** Run histories, tallies in private memory

- **Initialize:**
  Boss sends problem description to each Worker (geometry, tally specs, material definitions, …)

- **Compute, on each of N Workers:**
  Each Worker task runs 1/N of total histories.
  Tallies in private memory.
  Send tally results back to Boss.

- **Combine tallies:**
  Boss receives tallies from each Worker & combines them into overall results.

- **Concerns:**
  - Random number usage
  - Load-balancing
  - Fault tolerance (rendezvous for checkpoint)
  - Scaling
Boss / Worker Algorithm (Simple)

Control + Bookkeeping

Boss

! initialize
do n=1,nworkers
  send_info( n )

! Compute
  nchunk = nhistories / nworkers
  do n=1,nslaves
    k1 = 1 + (n-1)*nchunk
    k2 = min( k1+nchunk, nhistories )
    send_control( n, k1,k2 )

! Collect & combine results
  totals(:) = 0
  do n=1,nworkers
    recv_tallies( n )
    add_tallies_to_totals()

! Done
  print_results()
  save_files()

Computation

Worker 1

! Initialize
  recv_info()

! Compute
  recv_control( k1, k2 )
  do k=k1,k2
    run_history( k )
  ! Send tallies to Boss
  send_tallies()

! Done
  stop

Worker 2

! Initialize
  recv_info()

! Compute
  recv_control( k1, k2 )
  do k=k1,k2
    run_history( k )
  ! Send tallies to m
  send_tallies()

! Done
  stop

Worker 3

! Initialize
  recv_info()

! Compute
  recv_control( k1, k2 )
  do k=k1,k2
    run_history( k )
  ! Send tallies to Worker 2
  send_tallies()

! Done
  stop
Random Number Usage

- **Linear Congruential RN Generator**
  \[ S_{k+1} = g \cdot S_k + C \mod 2^M \]

- **RN Sequence & Particle Histories**
  
  \[
  \begin{array}{ccc}
  \_\_\_\_\_\_\_\_\_\_\_\_\_ & \_\_\_\_\_\_\_\_\_\_\_\_\_\_ & \_\_\_\_\_\_\_\_\_\_\_\_\_\_ \\
  1 & 2 & 3 & \text{etc.} \\
  \end{array}
  \]

  MCNP stride for new history: 152,917

- **To skip ahead** \( k \) **steps in the RN sequence:**
  \[ S_k = g \cdot S_{k-1} + C \mod 2^M = g^k \cdot S_0 + C \frac{(g^k-1)}{(g-1)} \mod 2^M \]

- **Initial seed for n-th history**
  \[ S_0^{(n)} = g^{n\cdot152917} \cdot S_0 + C \frac{(g^{n\cdot152917}-1)}{(g-1)} \mod 2^M \]

  This is easy to compute quickly using exact integer arithmetic

- **Each history has a unique number**
  - Initial problem seed → initial seed for \( n^{th} \) particle on \( m^{th} \) processor
  - If Worker knows initial problem seed & unique history number, can initialize RN generator for that history
Fault Tolerance

• On parallel systems with complex system software & many CPUs, interconnects, disks, memory, MTBF for system is a major concern.

• Simplest approach to fault tolerance:
  – Dump checkpoint files every M histories (or XX minutes)
  – If system crashes, restart problem from last checkpoint

• Algorithm considerations
  – Rendezvous every M histories.
  – Workers send current state to Boss, Boss saves checkpoint files
  – Parallel efficiency affected by M.
Fault Tolerance

- For efficiency, want (compute time) \(>>\) (rendezvous time)
  - Compute time: Proportional to #histories/task
  - Rendezvous time: Depends on amount of tally data & latency+bandwidth for message-passing
Boss / Worker Algorithm, with Rendezvous

- **Initialize:**
  Boss sends problem description to each Worker (geometry, tally specs, material definitions, ...)

- **For** rendezvous = 1, L
  
  - **Compute**, on each of N Workers:
    Each Worker task runs 1/N of (total histories)/L.
    Tallies in private memory.
    Send tally results back to Boss.

  - **Combine tallies:**
    Boss receives tallies from each Worker & combines them into overall results.

  - **Checkpoint:**
    Boss saves current tallies & restart info in file(s)
Load Balancing

• **Time per history may vary significantly**
  – **For problems using variance reduction:**
    • Particles headed in "wrong" direction may be killed quickly, leading to a short history.
    • Particles headed in "right" direction may be split repeatedly. Since the split particles created are part of the same history, may give a very long history.
  – **For problems run on a workstation cluster:**
    • Workstation nodes in the cluster may have different CPU speeds
    • Workstations in the cluster may be simultaneously used for interactive work, with highly variable CPU usage on that node.
    • Node performance effectively varies continuously over time.

• **Naïve solution**
  – Monitor performance per node (e.g., histories/minute)
  – Periodically adjust number of histories assigned to each node, according to node performance

\[
\text{# histories assigned to node } n \sim \text{ measured speed of node } n
\]

• **Better solution:** self-scheduling
Load Balancing – Self-Scheduling

• For a problem with N Worker processors, divide histories into more than N chunks.
  – Let $L = \text{number of chunks, } L > N$
  – Typically, $L \sim 20N$ or $L \sim 30N$
  – Histories/chunk = (total histories) / $L$

  – Worker: If idle, ask Boss for work. Repeat until no more work.
  – Boss: Send chunk of work to idle Worker. Repeat until no more work.

  – On average, imbalance in workload should be < $1/L$

• Additional gains:
  – Naïve Boss/Worker algorithm is synchronous
  – Self-scheduling Boss/Worker algorithm is asynchronous. More overlap of communication & computation $\Rightarrow$ reduced wait times & better performance
Load Balancing – Self-Scheduling

- Much more communication with Boss, but only minimal amount of control info needed (1st & last history in chunk)

- Need to handle stopping condition carefully – avoid "dangling" messages
Load Balancing – Self-Scheduling

**Boss**

```fortran
! Initialize
do n=1,nworkers
    send_info( n )
enddo

! Compute
nchunks = nworkers*20
nh = nhistories / nchunks
do n=1,nchunks
    k1 = 1 + (n-1)*nh
    k2 = min( k1+nh, nhistories)
    recv_idle_proc( M )
    send_control( M, k1,k2 )
endo

! Collect & combine results
totals(:) = 0
do n=1,nworkers
    recv_idle_proc( M )
    send_control( M, 0,-1 )
    recv_tallies( M )
    add_tallies_to_totals()
endo

! Done
print_results()
save_files()
```

**Worker 1**

```fortran
! Initialize
recv_info()

! Compute
do
    send_idle_proc()
    recv_control( k1, k2 )
    if( k1>k2 ) exit
    do k=k1,k2
        run_history( k )
endo
dendo

! Send tallies to Boss
send_tallies()

! Done
stop
```

**Worker 2**

```fortran
! Initialize
recv_info()

! Compute
do
    send_idle_proc()
    recv_control( k1, k2 )
    if( k1>k2 ) exit
    do k=k1,k2
        run_history( k )
endo
dendo

! Send tallies to Boss
send_tallies()

! Done
stop
```

**Worker 3**

```fortran
! Initialize
recv_info()

! Compute
recv_control( k1, k2 )
if( k1>k2 ) exit
do k=k1,k2
    run_history( k )
endo
dendo

! Send tallies to Boss
send_tallies()

! Done
stop
```
Hierarchical Parallelism

- For clustered SMPs,
  - Use message-passing to distribute work among Workers ("boxes")
  - Use threading to distribute histories among individual processors on box

- Only the Master thread (thread 0) on each Worker uses MPI send/recv's
Monte Carlo Techniques for Nuclear Systems – Theory Lectures

Boss / Worker Algorithm, threaded & self-scheduling

**Boss**

```fortran
! initialize
do n=1,nworkers
    send_info( n )
enddo

! Compute
nchunks = nworkers*20
nh = nhistories / nchunks
do n=1,nchunks
    k1 = 1 + (n-1)*nh
    k2 = min( k1+nh, nhistories )
    recv_idle_proc( M )
    send_control( M, k1,k2 )
enddo

! Collect & combine results
totals(:) = 0
do n=1,nworkers
    recv_idle_proc( M )
    send_control( M, 0, -1 )
    recv_tallies( M )
    add_tallies_to_totals()
enddo

! Done
print_results()
save_files()
```

**Worker 3**

```fortran```

```fortran
! Initialize
recv_info()

! Compute
do k=k1,k2
    run_history( k )
enddo

! Send tallies to Boss
send_tallies()

! Done
stop```

**Worker 2**

```fortran```

```fortran
! Initialize
recv_info()

! Compute
do k=k1,k2
    run_history( k )
enddo

! Send tallies to master
send_tallies()

! Done
stop```

**Worker 1**

```fortran```

```fortran
! Initialize
recv_info()

! Compute
do send_idle_proc()
    recv_control( k1, k2 )
    if( k1>k2 )  exit
    !$OMP PARALLEL DO
do k=k1,k2
        run_history( k )
    enddo
enddo

!$OMP PARALLEL
!$ combine_thread_tallies()

! Send tallies to Boss
send_tallies()

! Done
stop```

```fortran```
Parallel Monte Carlo Performance
Parallel MC Computational Characteristics

- **For Boss/Worker algorithms (with self-scheduling, fault tolerance, & threads):**
  - No communication among Worker tasks
  - Occasional communication between Boss & Workers (rendezvous)
  - **Worker tasks are compute-intensive**
    - Few DO-loops
    - 40% of ops are test+branch (IF… GOTO…)
    - Irregular memory access, no repetitive patterns

- **For fixed-source problems:**
  - Only 1 rendezvous is strictly necessary, at end of calculation
  - More rendezvous used in practice, for fault tolerance

- **For eigenvalue problems (K-effective):**
  - Must have a rendezvous every cycle (cycle = batch = generation)
  - Boss controls iteration & source sampling

- **Common-sense approach to performance:**
  Fewer rendezvous ➞ better parallel performance
Parallel MC Performance Measures

- **Metrics**
  - Speedup \( S_N = \frac{T_1}{T_N} \) \( N = \) # Worker processors
  - Efficiency \( E_N = \frac{S_N}{N} \)

- **Fixed overall work** (fixed problem size)
  - Efficiency decreases with \( N \)
  - Speedup (eventually) drops as \( N \) increases
  - Why?
    As \( N \) increases, same communication/processor, but less work/processor (fewer histories/processor) \( \Rightarrow \) (computation/communication) decreases

- **Fixed work per processor** (scaled problem size)
  - Efficiency approx. constant with \( N \)
  - Speedup approx. linear with \( N \)
  - Why?
    As \( N \) increases, same communication/processor, same work/processor
    (# histories ~ \( N \)) \( \Rightarrow \) (computation/communication) stays approx. same
  - Called scaled speedup
Parallel MC Performance Limits

• Another way to determine efficiency

Parallel Efficiency = \frac{T_C}{T_C + T_M}

\[ T_C = \text{computing time} \]
\[ T_M = \text{time for messages, not overlapped with computing} \]

• Workers can send messages in parallel

• Boss receives & processes messages serially

If enough messages are sent to Boss, extra wait time will limit performance
Parallel MC Performance Scaling

N = # processors

$T_1 = \text{CPU time for } M \text{ histories using 1 processor}$

( Depends on physics, geometry, compiler, CPU speed, memory, etc. )

$L = \text{amount of data sent from 1 Worker each rendezvous}$

$T_S = 0$   negligible, time to distribute control info

$T_R = s + L/r$   $s = \text{latency for message, } r = \text{streaming rate}$

$T_C^{\text{fix}} = T_1 / N$   fixed problem size, $M$ histories/rendezvous

$T_C^{\text{scale}} = T_1$   scaled problem size, $NM$ histories/rendezvous
Parallel MC Performance Scaling

- **Scaling models, for Boss/Worker with serial rendezvous**
  - "fixed" = constant number of histories/rendezvous, \( M \) \hspace{1cm} (constant work)
  - "scaled" = \( M \) histories/Worker per rendezvous, \( NM \) total \hspace{1cm} (constant time)

<table>
<thead>
<tr>
<th>Histories/rendezvous</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>fixed</td>
<td>( S = N / (1 + cN^2) )</td>
</tr>
<tr>
<td>scaled</td>
<td>( S = N / (1 + cN) )</td>
</tr>
</tbody>
</table>

\( N \) = number of Workers
\( c = (s + L/r) / T_1 \)

\( T_1 \sim M, \) more histories/rendezvous \( \rightarrow \) larger \( T_1 \), smaller \( c \)
\( S+L/r, \) fixed, determined by number of tallies, ....

As \( M \rightarrow \infty \), \( c \rightarrow 0 \), \( S \rightarrow N \) \hspace{1cm} (limit for 1 rendezvous)
Parallel MC Performance Scaling

Fixed size, serial messages

\[ S = \frac{N}{1 + cN^2} \]

Scaled size, serial messages

\[ S = \frac{N}{1 + cN} \]

N = number of Workers

\[ c = \frac{s + L/r}{M_1t_h} \]
Parallel MC Performance Scaling

VIM Monte Carlo — Measured Performance on SP1 — TREAT reactor

- SP1 — P4 + EUIH \[ S = N / (1 + 0.0056 \times N) \]
- SP1 — P4 + ethernet \[ S = N / (1 + 0.028 \times N) \]

Measured message passing on SP1 is 2-3 times slower than specs (busy machine; experimental software; flaky hardware)
Parallel MC Summary

• **Boss/Worker algorithms work well**
  - Load-balancing: Self-scheduling
  - Fault-tolerance: Periodic rendezvous
  - Random numbers: Easy, with LCG & fast skip-ahead algorithm
  - Tallies: Use OpenMP "critical sections"
  - Scaling: Simple model, more histories/Worker + fewer rendezvous
  - Hierarchical: Boss/Worker MPI, OpenMP threaded Workers
  - Portability: MPI/OpenMP, clusters of anything

• **Remaining difficulties**
  - Memory size: Entire problem must fit on each Worker

  • Domain-decomposition has had limited success
    - Should be OK for reactor problems
    - May not scale well for shielding or time-dependent problems
    - For general 3D geometry, effective domain-decomposition is unsolved problem

  • Random access to memory distributed across nodes gives huge slowdown
    - May need functional parallelism with "data servers"
MCNP Parallel Calculations
DOE Advanced Simulation & Computing – ASC

Blue Mountain – 3 TeraOps (R.I.P.)

Q – 20 TeraOps (R.I.P.)

Red Storm
Blue Gene/L
Hurricane
Moonlight
Cielo

Lightning – 30 TeraOps (R.I.P.)

Roadrunner – 1.3 PetaOps

DOE Advanced Simulation & Computing – ASC
Hierarchical Parallelism

- Use **message-passing** to distribute work among Workers ("boxes")

- Use **threading** to distribute histories among individual cpus on box

• 1,000 processor jobs are "routine"
MCNP Parallel Calculations

\[ N = \text{total number of MPI tasks, Boss + (N-1) Workers} \]
\[ M = \text{number of OpenMP threads/Worker} \]

- **Running on parallel systems with MPI only**
  
  \[ \text{mpirun -np N -bynode mcnp6.mpi i=inp01 ...} \]

- **Running with threads only**
  
  \[ \text{mcnp6 tasks M i=inp01 ...} \]

- **Running on parallel systems with MPI & threads**
  
  \[ \text{mpirun -np N -bynode mcnp6.mpi tasks M i=inp01 ...} \]

If submitting jobs through a batch system (e.g., LSF, Moab, ...),
N & M must be consistent with LSF requested resources
MCNP Parallel Calculations

- **How many threads?**
  - Max number of threads = # CPU-cores per node
    - ASCI Bluemountain: 128 cpus / node
    - ASCI Q: 4 cpus / node
    - Laptop PC cluster: 1 cpu / node
  
  - Experience on many systems has shown that a moderate number of threads per Worker is efficient; using too many degrades performance
    - ASCI Bluemountain: 4-12 threads/Worker usually effective
      >16 threads/Worker usually has bad performance
    - ASCI Q: 4 threads/Worker is effective

- **Rules-of-thumb vary for each system**
  - Thread efficiency is strongly affected by operating system design
  - Scheduling algorithm for threads used by operating system is generally designed to be efficient for small number of threads (<16)
  - For large number of threads, context-switching & cache management may take excessive time, giving poor performance
  - Other jobs on system (& their priority) affect thread performance
  - No definite rules – need to experiment with different numbers of threads
**MCNP Parallel Calculations**

- **Parallel performance is sensitive to number of rendezvous**
  - Can't control number of rendezvous directly
  - The following things cause a rendezvous:
    - Printing tallies
    - Dumping to the RUNTPE file
    - Tally Fluctuation Chart (TFC) entries
    - Each cycle of eigenvalue problem

- **Use PRDMP card to minimize print/dump/TFC**

  **PRDMP**
  
<table>
<thead>
<tr>
<th>ndp</th>
<th>ndm</th>
<th>mct</th>
<th>ndmp</th>
<th>dmmp</th>
</tr>
</thead>
<tbody>
<tr>
<td>ndp</td>
<td>= increment for printing tallies</td>
<td>➡️ use large number</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ndm</td>
<td>= increment for dump to RUNTPE</td>
<td>➡️ use large number</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mct</td>
<td>= flag to suppress time/date info in MCTAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ndmp</td>
<td>= max number of dumps in RUNTPE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>dmmp</td>
<td>= increment for TFC &amp; rendezvous</td>
<td>➡️ use large number</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For fixed-source problems, increments are in **particles**
For eigenvalue problems, increments are in **cycles**
MCNP Parallel Calculations

- **Keff calculations**: Use KCODE card for hist/cycle
  
  - Want to reduce the number of cycles
  - More histories in each cycle
  - Should run hundreds of cycles or more for good results

```
KCODE    nsrck  rkk     ikz     kct     ..... 
```

- **nsrck** = histories / cycle ➔ use a large number
- **rkk** = initial guess for Keff
- **ikz** = number of initial cycles to discard
- **kct** = total number of cycles to run

Suggested: **nsrck** ~ (thousands) x (number of processors)
MCNP5 Parallel Scaled Speedup

ASCI Q system, using MPI+OpenMP, 4 threads/MPI-task
Fixed-source calculation
MCNP5 Parallel Calculations

MCNP Speed vs. Number of Processors
BNCT Model w/ NPS=100,000 on a Linux Cluster w/ MPICH

- Blue line: No Load Balancing
- Orange line: Load Balancing
- Red dashed line: Linear Increase

Linux cluster
MCNP - Threading with OpenMP

• **MCNP performance - both serial & parallel - depends strongly on the Fortran-90 compiler & options used**
  - Runtime factors of 2-4x with different compilers on same hardware
  - Runtime factors of 2-4x with different options on same hardware & compiler

• **Parallel performance**
  - MCNP5 & MCNP6 have always supported parallel calculations with message-passing (MPI) & threading (OpenMP)
  - Prior to mid-2006, Fortran compilers for Windows/Linux/Mac did a terrible job at threading. We recommended using only MPI.
  - Recently, using OpenMP threading with Intel compilers on Windows/Linux/Mac shows excellent speedups -- nearly 2x on dual-core, 3-4x on quad-core
MCNP - Threading on the Mac Pro

Hardware
- Mac Pro
- 2 x Quad-core Xeon
- 3GHz
- 8 GB memory

Software
- Mac OS X 10.4.11
- Intel F90, 10.0.017
  -O1 -openmp
- MCNP5 / 1.50

MCNP Calculations
- KCODE
  - BAWXI2 benchmark
  - kcode 5000 10 204
- Fixed-source
  - oil-well log, mode n
  - nps 500000
MCNP - Threading

Hardware
- **Lobo**
  - 4 x Quad-core AMD Opteron
  - 2.2 GHz, 32 GB memory
- **Mac Pro**
  - 2 x Quad-core Intel Xeon
  - 3GHz, 8 GB memory

Software
- MCNP5-1.51
- Intel-10 F90, "-O1 -openmp"

MCNP Calculations

Criticality Calculation
BAWXI2 benchmark
kcode 25000 1 10 204
MCNP - Threading

Hardware
- Lobo
  - 4 x Quad-core AMD Opteron
  - 2.2 GHz, 32 GB memory
- Mac Pro
  - 2 x Quad-core Intel Xeon
  - 3GHz, 8 GB memory

Software
- MCNP5-1.51
- Intel-10 F90, "-O1 -openmp"

MCNP Calculations
Oil Well Logging Calculation
inp12 benchmark
Nps 500000
Parallel Processing
For Large
Monte Carlo Calculations
Domain Decomposition

If a Monte Carlo problem is too large to fit into memory of a single processor

- Need periodic synchronization to interchange particles among nodes
- Use message-passing (MPI) to interchange particles

→ Domain decomposition is often used when the entire problem will not fit in the memory of a single SMP node
Parallel Monte Carlo

- **Inherent parallelism is on particles**
  - Scales well for all problems

- **Domain decomposition**
  - Spatial domains on different processors
  - Scales OK for $K_{eff}$ or $\alpha$ calculations, where particle distribution among domains is roughly uniform
  - Does not scale for time-dependent problems due to severe load imbalances among domains

- **Domain decomposition - scaling with N processors**
  - Best: performance $\sim N$ (uniform distribution of particles)
  - Worst: performance $\sim 1$ (localized distribution of particles)
Parallel Monte Carlo

• Data is distributed by domain decomposition, but parallelism is on particles

• Solution?

Parallel on particles + distributed data

• Particle parallelism + Data Decomposition
  – Existing parallel algorithm for particles
  – Distribute data among processor nodes
  – Fetch the data to the particles as needed (dynamic)

  – Essentially same approach as used many years ago for CDC (LCM) or CRAY (SSD) machines

  – Scales well for all problems (but slower)
Parallel Monte Carlo

- Particle parallelism + data decomposition -- logical view:
  - Mapping of logical processes onto compute nodes is flexible:
    - Could map particle & data processes to different compute nodes
    - Could map particle & data processes to same compute nodes
  - Can replicate data nodes if contention arises
Parallel Monte Carlo

- Particle parallelism + data decomposition

![Diagram showing parallel Monte Carlo](image)

Entire physical problem

Local copies of data for particle neighborhood

Particle Node

Data Node
Parallel Monte Carlo

- **History modifications for data decomposition**
  - source
  - while \( \text{wgt} > \text{cutoff} \)
    - compute distances & keep minimum:
      - dist-to-boundary
      - dist-to-time-cutoff
      - dist-to-collision
      - dist-to-data-domain-boundary
    - move particle
    - pathlength tallies
    - if \( \text{distance} == \text{dist-to-data-domain-boundary} \)
      - fetch new data
    - collision physics
    - roulette & split
    ...

Parallel Monte Carlo

- **Data windows & algorithm tuning**
  - Defining the "particle neighborhood" is an art
  - Anticipating the flight path can guide the pre-fetching of blocks of data
  - Tuning parameters:
    - How much data to fetch?
    - Data extent vs. particle direction?

- **Examples**
Conclusions

For Monte Carlo problems which can fit in memory:

• Concurrent scalar jobs - ideal for Linux clusters

• Boss/Worker parallel algorithm (replication) works well
  – Load-balancing: Self-scheduling
  – Fault-tolerance: Periodic rendezvous
  – Random numbers: Easy, with LCG & fast skip-ahead algorithm
  – Tallies: Use OpenMP "critical sections"
  – Scaling: Simple model, more histories/Worker + fewer rendezvous
  – Hierarchical: Boss/Worker MPI, OpenMP threaded Workers
  – Portability: MPI/OpenMP, clusters of anything

For Monte Carlo problems too large to fit in memory:

• Spatial domain decomposition (with some replication) can work for some problems

• Particle parallelism + data decomposition is a promising approach which should scale for all problems
References

10. OpenMPI - [http://www.open-mpi.org](http://www.open-mpi.org)
Monte Carlo
Parameter Studies & Uncertainty Analyses
With MCNP6
Outline

• Introduction
• mcnp_pstudy
• Examples
• Usage
  – Parameter definition
  – Parameter expansion
  – Constraints
  – Case setup & execution
  – Collecting & combining results
• Statistics
• Practical Examples from Criticality Safety
• Advanced Topics
Frequent Questions

How are calculated results affected by:

- **Nominal dimensions**
  - With minimum & maximum values?
  - With as-built tolerances?
  - With uncertainties?

- **Material densities**
  - With uncertainties?

- **Data issues**
  - Different cross-section sets?

- **Stochastic materials**
  - Distribution of materials?

Monte Carlo perturbation theory can handle the case of independent variations in material density, but does not apply to other cases.

**Brute force approach:**

Run many independent Monte Carlo calculations, varying the input parameters.
mcnp_pstudy

• To simplify & streamline the setup, running, & analysis of Monte Carlo parameter studies & total uncertainty analyses, a new tool has been developed: mcnp_pstudy

• Control directives are inserted into a standard MCNP input file
  – Define lists of parameters to be substituted into the input file
  – Define parameters to be sampled from distributions & then substituted
  – Define arbitrary relations between parameters
  – Specify constraints on parameters, even in terms of other parameters
  – Specify repetitions of calculations
  – Combine parameters as outer-product for parameter studies
  – Combine parameters as inner-product for total uncertainty analysis

• Sets up separate calculations
• Submits or runs all jobs
• Collects results
mcnp_pstudy

- Completely automates the setup/running/collection for parameter studies & total uncertainty analyses
  - Painless for users
  - 1 input file & run command can spawn 100s or 1000s of jobs
  - Fast & easy way to become the #1 user on a system
    (Added bonus: make lots of new friends in computer ops & program management.)

- Ideal for Linux clusters & parallel ASC computers:
  - Can run many independent concurrent jobs, serial or parallel
  - Faster turnaround: Easier to get many single-cpu jobs through the queues, rather than wait for scheduling a big parallel job
  - Clusters always have some idle nodes
mcnp_pstudy

- **mcnp_pstudy** is written in **perl**
  - 640 lines of **perl** (plus 210 lines of comments)
  - Would have taken many thousands of lines of Fortran or C

- **Portable to any computer system**
  - Tested on Unix, Linux, Mac OS X, Windows
  - For Windows PCs, need to have **perl** installed
    (ActivePerl is free at [activestate.com/activeperl](http://activestate.com/activeperl), easy to install)

- **Can be modified easily if needed**
  - To add extra features
  - To accommodate local computer configuration
    - Node naming conventions for parallel cluster
    - Batch queueing system for cluster
    - Names & configuration of disk file systems (ie, local or shared)
    - Location of MCNP6 and MCNP6.mpi
Examples

MCNP input for simple Godiva calculation

\[
gdv
\]
\[
c
\]
1 100 -18.74 -1 imp:n=1
2 0 1 imp:n=0
1 so 8.741
\[
\text{kcode 10000 1.0 15 115}
\text{ksrc 0 0 0}
\text{m100 92235 -94.73 92238 -5.27}
\text{prdmp 0 0 1 1 0}
\]

MCNP input using \textit{mcnp} \textit{pstudy}, Run 3 different cases - Each with a different radius

\[
gdv-A
\]
\[
c
\]
1 100 -18.74 -1 imp:n=1
2 0 1 imp:n=0
1 so RADIUS
\[
\text{kcode 10000 1.0 15 115}
\text{ksrc 0 0 0}
\text{m100 92235 -94.73 92238 -5.27}
\text{prdmp 0 0 1 1 0}
\]
Basics

- Within an MCNP input file, all directives to mcnp_pstudy must begin with
  
  \[ C @@ @ \]

- To continue a line, use "\" as the last character
  
  \[ C @@ @ XXX = 1 2 3 4 5 6 \ \]
  
  \[ C @@ @ 7 8 9 10 \]

- Parameter definitions have the form
  
  \[ C @@ @ P = value or list \]
  
  \[ C @@ @ P = ( arithmetic-expression ) \]

- Constraints have the form
  
  \[ C @@ @ CONSTRAINT = ( expression ) \]

- Control directives have the form
  
  \[ C @@ @ OPTIONS = list-of-options \]
Parameter Definition

• **Parameters**
  – Like C or Fortran variables
  – Start with a letter, contain only letters, integers, underscore
  – Case sensitive
  – Parameters are assigned values, either number(s) or string(s)
  – Examples: \( R_1, r_1, U\_density, U\_den \)

• **Single value**
  \[ C \quad P_1 = \text{value} \]

• **List of values**
  \[ C \quad P_2 = \text{value1 value2 ... valueN} \]

• **List of N random samples from Probability Densities:**
  – **Uniform**
    \[ C \quad P_3 = \text{uniform } N \quad \text{min } \quad \text{max} \]
  – **Normal**
    \[ C \quad P_4 = \text{normal } N \quad \text{ave } \quad \text{dev} \]
  – **Lognormal**
    \[ C \quad P_5 = \text{lognormal } N \quad \text{ave } \quad \text{dev} \]
  – **Beta**
    \[ C \quad P_6 = \text{beta } N \quad a \quad b \quad [a, b \text{ are integers}] \]
Parameter Definition

- **Arithmetic expression**
  
  \[ c \quad P5 = ( \text{arithmetic-statement} ) \]

  - Can use numbers & previously defined parameters
  - Can use arithmetic operators +, -, *, /, % (mod), ** (exponentiation)
  - Can use parentheses ( )
  - Can use functions: \( \sin() \), \( \cos() \), \( \log() \), \( \exp() \), \( \text{int()} \), \( \text{abs()} \), \( \text{sqrt()} \)
  - Can generate random number in (0,N): \( \text{rand}(N) \)
  - Can use \( \text{rn_seed}() \) to get odd seed for mcnp RN generator in \([1,2^{48}-1]\)
  - Must evaluate to a single value
  - Examples:
    
    \[
    c \quad SEED = \ ( \text{rn_seed}() \ )
    \]
    \[
    c \quad FACT = \normal 1 1.0 .05
    \]
    \[
    c \quad UDEN = \ ( 18.74 * FACT )
    \]
    \[
    c \quad URAD = \ ( 8.741 * (18.74/UDEN)**.333333 )
    \]

- **Repetition** (list of integers, 1..N)
  
  \[ c \quad P6 = \text{repeat} N \]
Parameter Definition

• Examples
  C  rod height in inches, for search
  C  @@@  HROD = 5  10  15  20  25  30  35  40  45  50

  C  nominal dimension, with uncertainty
  C  @@@  X1 = normal  25   1.234   .002

  C  dimension, with min & max
  C  @@@  X2 = uniform 25   1.232  1.236

  C  try different cross-sections
  C  @@@  U235 = 92235.42c  92235.49c  92235.52c  \ 
  C  @@@  92235.60c  92235.66c

  C  different random number seeds (odd)
  C  @@@  SEED = (  rn_seed()  )
Parameter Expansion

- After all parameters are defined, `mcnp_pstudy` expands them into sets to be used for each separate MCNP calculation

  - Outer product expansion: All possible combinations. Parameters specified first vary fastest.
  - Inner product expansion: Corresponding parameters in sequence. If not enough entries, last is repeated.

Example:

- Outer:
  - Case 1: A=1, B=3, C=5
  - Case 2: A=2, B=3, C=5
  - Case 3: A=1, B=4, C=5
  - Case 4: A=2, B=4, C=5

- Inner:
  - Case 1: A=1, B=3, C=5
  - Case 2: A=2, B=4, C=5
Constraint Conditions

- After all parameters are defined & expanded, constraint conditions are evaluated.
- Constraints involve comparison operators ( >, <, >=, <=, ==, != ) or logical operators ( && (and), || (or), ! (not) ), and may involve arithmetic or functions.
- Constraints must evaluate to True or False.
- If any constraint is not met, the parameters for that case are discarded & re-evaluated until all of the constraints are satisfied.

Example

C pick dimensions between min & max
C
C @@ X1 = uniform 1 3.9 4.1
C @@ X2 = uniform 1 5.9 6.1
C
C keep x1 & x2 if x1+x2 <= 10.0, otherwise reject & try again
C
C @@ CONSTRAINT = ( X1 + X2 <= 10.0 )
Creating INP Files & Job Directories

• Directory structure for MCNP5 jobs

```
JOBDIR
   ├── case001
   │    └── inp
   ├── case002
   │    └── inp
   ├── case003
   │    └── inp
   └── ....
```

– Unix filesystem conventions followed
  JOBDIR/case001/inp, JOBDIR/case002/inp, etc.

• Values of parameters are substituted into the original MCNP5 input file to create the input files for each case
  – Parameters substituted only when exact matches are found
  – Example: UDEN matches UDEN, and not UDEN1, UDENS, uden
Job Options

• Specifying options for running jobs
  – Can be specified on the mcnp_pstudey command-line
    mcnp_pstudey -inner -setup -i inp01
  – Within the INP file
    c @@ OPTIONS = -inner

• Common options
  -i str
  -jobdir str
  -case str
  -mcnp_opts str
  -bsub_opts str
  -inner
  -outer
  -setup
  -run
  -submit
  -collect
  
  The INP filename is str, default = inp
  Use str as the name of the job directory
  Use str as the name for case directories
  Append str to the MCNP5 run command, may be a string such as ‘o=outx tasks 4’
  str is appended to the LSF bsub command
  Inner product approach to case parameter substitution
  Outer product approach to case parameter substitution
  Create the cases & INP files for each
  Run the MCNP5 jobs on this computer
  Submit the MCNP5 jobs using LSF bsub command
  Collect results from the MCNP5 jobs
Running or Submitting Jobs

• Jobs can be run on the current system, or can be submitted to a batch queueing system (e.g., LSF)
• Tally results & K-effective can be collected when jobs finish

Examples:

bash: mcnp_pstudy -inner -i inp01 -setup
bash: mcnp_pstudy -inner -i inp01 -run
bash: mcnp_pstudy -inner -i inp01 -collect

bash: mcnp_pstudy -inner -i inp01 -setup -run -collect

bash: mcnp_pstudy -inner -i inp01 -setup -submit

... wait till all jobs complete...

bash: mcnp_pstudy -inner -i inp01 -collect
Creating Input Files ONLY

• To bypass the creation of job directories, and running/submitting problems:

  – A special command line option is available:  -inponly

  – Invoking this option performs the parsing & setup of the input files for each case, but the resulting mcnp input files are placed in the current directory with default names of the form

    inp_case001,   inp_case002, etc.

  – Using -case study01a -inponly would result in files with names

    inp_study01a001,  inp_study01a002, etc.

  – Other options -run, -submit cannot be used if -inponly is present

  – The option -whisper can be used, and is equivalent to -inponly
Combining Results

• Tally results & K-effective from separate cases can be combined using batch statistics:

\[ \bar{X} = \frac{1}{M} \cdot \sum_{k=1}^{M} X_k \]
\[ \sigma_{\bar{X}} = \sqrt{\frac{1}{M-1} \cdot \left[ \frac{1}{M} \sum_{k=1}^{M} X_k^2 - \bar{X}^2 \right]} \]

where \( M \) is the number of cases & \( X_k \) is some tally or Keff for case \( k \)

• Variance due to randomness in histories decreases as \( 1/M \), but variance due to randomness in input parameters is constant

\[ \sigma_{\bar{X}}^2 \approx \sigma_{\bar{X}, \text{Monte Carlo}}^2 + \sigma_{\bar{X}, \text{Initial Conditions}}^2 \]

Varies as \( 1/M \) \hspace{1cm} \sim \text{Constant}
# Examples

## Vary the fuel density randomly & adjust radius for constant mass, for 50 cases

```
gdv-E
c vary fuel density - normal, 5%sd,
c adjust the radius to keep constant mass
c
@@@ FACT= normal 50 1.0 .05
c @@ OPTIONS = -inner
@@@ UDEN= ( 18.74*FACT )
c @@ URAD= ( 8.741*(18.74/UDEN)**.333333 )
c
1 100 -UDEN -1 imp:n=1
2 0 1 imp:n=0

1 so URAD
```

```
kcode 10000 1.0 15 115
ksrc 0. 0. 0.
m100 92235 -94.73 92238 -5.27
prdmp 0 0 1 1 0
```

## Vary fuel density & mass independently, for 50 cases

```
gdv-F
c vary fuel radius - normal, 5%sd
c vary fuel density- normal, 5%sd
c
@@@ OPTIONS = -inner
@@@ DFACT = normal 50 1.0 .05
@@@ UDEN = ( DFACT * 18.74 )
c
@@@ UFACT = normal 50 1.0 .05
@@@ URAD = ( UFACT * 8.741 )
c
1 100 -UDEN -1 imp:n=1
2 0 1 imp:n=0

1 so URAD
```

```
kcode 10000 1.0 15 115
ksrc 0. 0. 0.
m100 92235 -94.73 92238 -5.27
prdmp 0 0 1 1 0
```

Vary the fuel density randomly & adjust radius for constant mass, for 50 cases

Vary fuel density & mass independently, for 50 cases
## Examples

### Table 1. Results from varying parameters in the Godiva problem

<table>
<thead>
<tr>
<th>Problem</th>
<th>Description</th>
<th>K-effective</th>
<th>$\sigma_{K-eff}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>base</td>
<td><strong>Base case</strong>, discard 15 initial cycles, retain 100 cycles with 10K histories/cycle, <strong>1M total histories</strong></td>
<td>0.9970</td>
<td>0.0005</td>
</tr>
<tr>
<td>A</td>
<td>Repeat the base problem 50 times, <strong>50M total histories</strong></td>
<td>0.9972</td>
<td>0.0001</td>
</tr>
<tr>
<td>B</td>
<td><strong>Vary the fuel density only</strong>: sample from a normal distribution with 5% std.dev, <strong>50M total histories</strong></td>
<td>0.9961</td>
<td>0.0061</td>
</tr>
<tr>
<td>C</td>
<td><strong>Vary the fuel radius only</strong>: sample from a normal distribution with 5% std.dev, <strong>50M total histories</strong></td>
<td>1.0057</td>
<td>0.0051</td>
</tr>
<tr>
<td>D</td>
<td><strong>Vary the enrichment only</strong>, sample from a normal distribution with 5% std.dev, <strong>50M total histories</strong></td>
<td>0.9890</td>
<td>0.0027</td>
</tr>
<tr>
<td>E</td>
<td>Sample the fuel density from a normal distribution with 5% std.dev, and adjust the fuel radius to keep constant fuel mass, <strong>50M total histories</strong></td>
<td>0.9966</td>
<td>0.0042</td>
</tr>
<tr>
<td>F</td>
<td>Sample the fuel density from a normal distribution with 5% std.dev, and independently sample the radius from a normal distribution with 5% std.dev, <strong>50M total histories</strong></td>
<td>1.0073</td>
<td>0.0076</td>
</tr>
</tbody>
</table>
Applications

• **Parameter studies**
  – Run a series of cases with different control rod positions
  – Run a series of cases with different soluble boron concentrations
  – Run a series of cases sampling certain dimensions from a Uniform or Normal probability density
  – Run a series of cases substituting different versions of a cross-section

• **Total uncertainty analysis**
  – Run a series of cases varying all input parameters according to their uncertainties

• **Parallel processing using a "parallel jobs" approach**
  – Running N separate jobs with 1 cpu each will be more efficient than running 1 job with N cpus
  – Eliminates queue waiting times while cpus are reserved
  – Take advantage of cheap Linux clusters

• **Simulation of stochastic geometry**
  – Run a series of cases with portions of geometry sampled randomly, with a different realization in each case
Conclusions

• **mcnp_pstudy works**
  – In use regularly at LANL for a variety of real applications
  – Developed on Mac & PC, runs anywhere
  – Easy to customize, if you have special needs

• **To get it:**
  – Included with MCNP6 distribution

Examples

- wval4: 4.5 kg Pu Sphere, Ta-reflected with varying reflector thickness
- wval1: 4.5 kg Pu Ingot, solid cylinder with varying H/D
- wval2: 4.5 kg Pu Ring, hollow cylinder with varying H & R_{in}
Example

wval4,
4.5 kg Pu Sphere,
Ta-reflected
Example wval4: 4.5 kg Pu Sphere, Ta-reflected (1)

- 4.5 kg Pu-239 sphere
- Pu density = 19.8 g/cm$^3$
- Reflected radially with Ta

- Vary the Ta-reflector thickness over the range 0.1 – 30. cm

  - Start with wval4.txt, input for thickness=7.62
    mcnp6 i=wval4.txt

  - Copy wval4.txt to wval4p.txt, then insert directives for mcnp_pstudy
    • Define list for thickness:
      c @@ THICK = 0.01 5. 10. 15. 20. 25. 30.
    • For a given THICK, compute reflector Rin & Rout
    • Use parameters for dimensions & location of KSRC point
    • Run:
      mcnp_pstudy -i wval4.txt -mcnp_opts ‘tasks 4’ -setup
      ...... examine files case*/inp
      mcnp_pstudy -i wval4.txt -mcnp_opts ‘tasks 4’ -run
Example wval4: 4.5 kg Pu Sphere, Ta-reflected (2)

wval4: Study of Pu reflected with Ta

Pu mass = 4500 g
Pu density = 19.8 g/cc
Pu volume = 227.272727

reflector definition:

<table>
<thead>
<tr>
<th>Thickness</th>
<th>Inner Radius</th>
<th>Outer Radius</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1 cm</td>
<td>3.7857584 cm</td>
<td>5.4857584 cm</td>
</tr>
<tr>
<td>0.5 cm</td>
<td>3.7857584 cm</td>
<td>4.2857584 cm</td>
</tr>
<tr>
<td>1.0 cm</td>
<td>3.7857584 cm</td>
<td>4.7857584 cm</td>
</tr>
<tr>
<td>1.5 cm</td>
<td>3.7857584 cm</td>
<td>5.2857584 cm</td>
</tr>
<tr>
<td>2.0 cm</td>
<td>3.7857584 cm</td>
<td>5.7857584 cm</td>
</tr>
<tr>
<td>2.5 cm</td>
<td>3.7857584 cm</td>
<td>6.2857584 cm</td>
</tr>
<tr>
<td>3.0 cm</td>
<td>3.7857584 cm</td>
<td>6.7857584 cm</td>
</tr>
<tr>
<td>3.5 cm</td>
<td>3.7857584 cm</td>
<td>7.2857584 cm</td>
</tr>
<tr>
<td>4.0 cm</td>
<td>3.7857584 cm</td>
<td>7.7857584 cm</td>
</tr>
<tr>
<td>4.5 cm</td>
<td>3.7857584 cm</td>
<td>8.2857584 cm</td>
</tr>
</tbody>
</table>

kcode 10000 1.0 50 250
sdef pos=0 0 0 rad=d1
si1 0 3.78
sp1 -21 2
m1 73180.80c 0.00012 73181.80c 0.99988
m4 94239.80c 1
prdmp 9e9 9e9 1 9e9

wval4p: Study of Pu reflected with Ta

Pu mass = 4500 g
Pu density = 19.8 g/cc
Pu volume = 227.272727

reflector definition:

<table>
<thead>
<tr>
<th>Thickness</th>
<th>Inner Radius</th>
<th>Outer Radius</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1 cm</td>
<td>3.7857584 cm</td>
<td>5.4857584 cm</td>
</tr>
<tr>
<td>0.5 cm</td>
<td>3.7857584 cm</td>
<td>4.2857584 cm</td>
</tr>
<tr>
<td>1.0 cm</td>
<td>3.7857584 cm</td>
<td>4.7857584 cm</td>
</tr>
<tr>
<td>1.5 cm</td>
<td>3.7857584 cm</td>
<td>5.2857584 cm</td>
</tr>
<tr>
<td>2.0 cm</td>
<td>3.7857584 cm</td>
<td>5.7857584 cm</td>
</tr>
<tr>
<td>2.5 cm</td>
<td>3.7857584 cm</td>
<td>6.2857584 cm</td>
</tr>
<tr>
<td>3.0 cm</td>
<td>3.7857584 cm</td>
<td>6.7857584 cm</td>
</tr>
<tr>
<td>3.5 cm</td>
<td>3.7857584 cm</td>
<td>7.2857584 cm</td>
</tr>
<tr>
<td>4.0 cm</td>
<td>3.7857584 cm</td>
<td>7.7857584 cm</td>
</tr>
<tr>
<td>4.5 cm</td>
<td>3.7857584 cm</td>
<td>8.2857584 cm</td>
</tr>
</tbody>
</table>

kcode 10000 1.0 50 250
sdef pos=0 0 0 rad=d1
si1 0 R_INNER
sp1 -21 2
m1 73180.80c 0.00012 73181.80c 0.99988
m4 94239.80c 1
prdmp 9e9 9e9 1 9e9
Example wval4: 4.5 kg Pu Sphere, Ta-reflected

wval4, thick=7.62
mcnp6 i=wval4.txt

wval4p, varying thick
mcnp_pstudy -i wval4p.txt -setup -run

k = 0.94638 (41)

\[
\begin{array}{cccc}
T & case & KEFF & KSIG \\
0.01 & 001 & 7.91693E-01 & 3.14948E-04 \\
5.0 & 002 & 9.27157E-01 & 4.47334E-04 \\
10.0 & 003 & 9.54775E-01 & 4.11031E-04 \\
15.0 & 004 & 9.61644E-01 & 4.34033E-04 \\
20.0 & 005 & 9.62867E-01 & 4.37235E-04 \\
25.0 & 006 & 9.63899E-01 & 4.04508E-04 \\
30.0 & 007 & 9.63160E-01 & 4.27633E-04 \\
\end{array}
\]

4.5 kg Pu with Ta Reflection
Example

wval1,
4.5 kg Pu Ingot,
varying H/D
Example wval1: 4.5 kg Pu Ingot, varying H/D (1)

- 4.5 kg Pu-239 right-circular cylinder
- Pu density = 19.86 g/cm$^3$
- Reflected radially with 1 inch of water
- Reflected on the bottom with ¼ inch steel

- Vary the height-to-diameter (H/D) over the range 0.5 – 3.0

  - Start with wval1.txt, input for H/D = 1
    
    mcnp6 i=wval1.txt

  - Copy wval1.txt to wval1p.txt, then insert directives for mcnp_pstudy
    
    - Define list for HD:
      
      `c @@ HD = 0.5 1.0 1.5 2.0 2.5 3.0`

    - For a given H/D, compute Pu radius, then other dimensions
      
      \[
      V = \frac{\text{Pu mass}}{\text{Pu density}} \\
      V = H\pi R^2 = (H/D) \cdot 2\pi R^3 \\
      R = \left[\frac{V}{2\pi(H/D)}\right]^{1/3}
      \]

    - Use parameters for dimensions & location of KSRC point
### Example wval1: 4.5 kg Pu Ingot, varying H/D (2)

<table>
<thead>
<tr>
<th>wval1: 4500 g Pu metal, H/D = 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>c reflected 1 inch water radially, c 0.25 in steel bottom</td>
</tr>
<tr>
<td>1 1 -19.860000 -1</td>
</tr>
<tr>
<td>11 3 -1.0 +1 -11</td>
</tr>
<tr>
<td>14 6 -7.92 -30</td>
</tr>
<tr>
<td>15 0 +11 +30 -20</td>
</tr>
<tr>
<td>20 0 +20</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>wval1p: 4500 g Pu metal, various H/D</th>
</tr>
</thead>
<tbody>
<tr>
<td>c reflected 1 inch water radially, c 0.25 in steel bottom</td>
</tr>
<tr>
<td>c V = H pi R<strong>2 = (H/D) 2pi R</strong>3</td>
</tr>
<tr>
<td>c R = (V/(2pi H/D)**1/3</td>
</tr>
<tr>
<td>c @@@ PI = 3.141592654</td>
</tr>
<tr>
<td>c @@@ VOL_PU = ( 4500. / 19.86 )</td>
</tr>
<tr>
<td>c @@@ HD = 0.5 1.0 1.5 2.0 2.5 3.0</td>
</tr>
<tr>
<td>c @@@ R_PU = ( (VOL_PU/(2<em>PI</em>HD)))**(1/3)</td>
</tr>
<tr>
<td>c @@@ H_PU = ( 2<em>R_PU</em>HD )</td>
</tr>
<tr>
<td>c @@@ R_H2O = ( R_PU + 2.54 )</td>
</tr>
<tr>
<td>c @@@ KSRC_Z = ( H_PU * 0.5 )</td>
</tr>
</tbody>
</table>

Pu cylinder:
- mass = 4500 g
- density = 19.86 g/cc
- volume = VOL_PU
- radius Pu = R_PU
- height Pu = H_PU
- H/D = HD

H2O outer radius = R_H2O

| 1 1 -19.860000 -1 |
| 11 3 -1.0 +1 -11 |
| 14 6 -7.92 -30 |
| 15 0 +11 +30 -20 |
| 20 0 +20 |

| prdmp 9e9 9e9 1 9e9 |

| kcode 10000 1.0 50 250 |
| ksrc 0 0 3.303831 |
| m1 94239.80c 1 |
| m3 1001.80c 0.66667 8016.80c 0.33333 |
| mt3 lwtr.20t |
| m6 24050.80c 0.000757334 |
| 24052.80c 0.014604423 |
| 24053.80c 0.001656024 |
| 24054.80c 0.000412220 |
| 26054.80c 0.003469592 |
| 26056.80c 0.054465174 |
| 26057.80c 0.001257838 |
| 26058.80c 0.000167395 |
| 25055.80c 0.00174 |
| 28058.80c 0.005255537 |
| 28060.80c 0.002024423 |
| 28061.80c 0.00088000 |
| 28062.80c 0.000280583 |
| 28064.80c 0.00071456 |

| prdmp 9e9 9e9 1 9e9 |

kcode 10000 1.0 50 250
ksrc 0. 0. KSRC_Z
C ...................... etc.
Example \textit{wval1}: 4.5 kg Pu Ingot, varying H/D (3)

\texttt{wval1}, H/D = 1
\texttt{mcnp6 i=wval1.txt}

\texttt{wval1p, varying H/D}
\texttt{mcnp_pstudy -i wval1p.txt -setup -run}

\begin{verbatim}
HD=0.5  case001  KEFF  7.87229E-01  KSIG  4.09191E-04
HD=1.0  case002  KEFF  8.34430E-01  KSIG  4.20175E-04
HD=1.5  case003  KEFF  8.29652E-01  KSIG  4.19130E-04
HD=2.0  case004  KEFF  8.11958E-01  KSIG  4.18723E-04
HD=2.5  case005  KEFF  7.93676E-01  KSIG  4.63720E-04
HD=3.0  case006  KEFF  7.73434E-01  KSIG  4.19664E-04
\end{verbatim}

4.5 kg Pu Ingot k-effective and USL
Example

wval2,
4.5 kg Pu Annulus,
varying H & $R_{in}$
Example  wval2:  4.5 kg Pu Annulus, varying H & R_{in}  (1)

- 4.5 kg Pu-239 right-circular cylinder, hollow
- Pu density = 19.86 g/cm³
- Reflected radially with 1 inch of water
- Reflected on the bottom with ¼ inch steel

- Set the height to be same as solid cylinder with height-to-diameter (H/D) = 1.0, 2.0, 3.0
- For given height, vary inner radius over 0+ - 2 cm

- Start with wval2.txt input
  mcnp6  i=wval2.txt

- Copy wval2.txt to wval2p.txt, then insert directives for mcnp_pstudy
  - Define list for solid HD:
    c @@ HD =  1.0  2.0  3.0
  - For a given H/D, compute Pu height
  - Define list for inner radius RIN_PU
    c @@ RIN_PU =  0.001  0.5  1.0  2.0
  - Then other dimensions & source

Solid cylinder
\[ V = \frac{\text{Pu mass}}{\text{Pu density}} \]
\[ V = H\pi R^2 = \frac{H}{D} \cdot 2\pi R^3 \]
\[ H = \left[ \frac{4V(H/D)^2}{\pi} \right]^{1/3} \]

Hollow cylinder
\[ V = H\pi(R_{out}^2 - R_{in}^2) \]
\[ R_{out} = \left[ R_{in}^2 + V/\pi H \right]^{1/2} \]
Example  wval2:  4.5 kg Pu Annulus, varying H & $R_{in}$  (2)

wval2: 4500 g Pu metal ring, fixed $R_{in}$

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>3 -1.0</th>
<th>-1</th>
<th>imp:n=1</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1</td>
<td>-19.860000</td>
<td>+1 -2</td>
<td>imp:n=1</td>
</tr>
<tr>
<td>11</td>
<td>3</td>
<td>-1.0</td>
<td>+2 -11</td>
<td>imp:n=1</td>
</tr>
<tr>
<td>14</td>
<td>6</td>
<td>-7.92</td>
<td>-30</td>
<td>imp:n=1</td>
</tr>
<tr>
<td>15</td>
<td>0</td>
<td>+11 +30 -20</td>
<td>imp:n=1</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>0</td>
<td>+20</td>
<td></td>
<td>imp:n=0</td>
</tr>
</tbody>
</table>

wval2p: 4500 g Pu metal ring, various H & $R_{in}$

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>c @@@ PI = 3.141592654</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>c @@@ VOL_PU = ( 4500. / 19.86 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>Pu mass = 4500 g</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>Pu density = 19.86 g/cc</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>Pu volume = VOL_PU</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>set height to match ingot with various H/D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>c @@@ HD = 1.0 2.0 3.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>c @@@ HEIGHT = ( (4<em>VOL_PU</em>(HD<strong>2)/PI)</strong>(1/3) )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>c for hollow cylinder:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>use same height as for solid ingot</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>set various inner radii</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>set Rout for given height, mass, Rin</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>c @@@ RIN_PU = .001 0.5 1.0 2.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>c @@@ ROUT_PU=(sqrt(RIN_PU**2+VOL PU/(PI*HEIGHT)))</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>c @@@ ROUT_H2O = ( OUTER_PU + 2.54 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>kcode</td>
<td>10000 1.0 50 250</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sdef</td>
<td>pos=0 0 0 rad=d1 axs=0 0 1 ext=d2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>si1</td>
<td>0.100 3.305259</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sp1</td>
<td>-21 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>si2</td>
<td>0.0 6.60800</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sp2</td>
<td>0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ml</td>
<td>94239.80c 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>m3</td>
<td>1001.80c 0.66667 8016.80c 0.33333</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mt3</td>
<td>lwtr.20t</td>
<td></td>
<td></td>
</tr>
<tr>
<td>m6</td>
<td>24050.80c 0.000757334</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>24052.80c 0.014604423</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>24053.80c 0.001656024</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>24054.80c 0.000412220</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>26054.80c 0.003469592</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>26056.80c 0.054465174</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>26057.80c 0.001257838</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>26058.80c 0.000167395</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>25055.80c 0.00174</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>28058.80c 0.000255537</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>28060.80c 0.002024423</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>28061.80c 0.000088000</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>28062.80c 0.000280583</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>28064.80c 0.000071456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>prdmp</td>
<td>9e9 9e9 1 9e9</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

--------------- etc.
Example *wval2*: 4.5 kg Pu Annulus, varying H & $R_{in}$ (3)

**wval2**

mcnp6 i=wval2.txt

$k = 0.83413$ (42)

**wval2p, varying H & $R_{in}$**

mcnp_pstudy -i wval2p.txt -setup -run

<table>
<thead>
<tr>
<th>HD</th>
<th>Rin</th>
<th>case</th>
<th>KEFF</th>
<th>426</th>
<th>425</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.001</td>
<td>001</td>
<td>8.34752E-01</td>
<td>4.35668E-04</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.001</td>
<td>002</td>
<td>8.12612E-01</td>
<td>4.09516E-04</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.001</td>
<td>003</td>
<td>7.72725E-01</td>
<td>3.82627E-04</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.5</td>
<td>004</td>
<td>8.20432E-01</td>
<td>4.01135E-04</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.5</td>
<td>005</td>
<td>7.95375E-01</td>
<td>4.60388E-04</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.5</td>
<td>006</td>
<td>7.54174E-01</td>
<td>3.96580E-04</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1.0</td>
<td>007</td>
<td>7.88497E-01</td>
<td>3.95026E-04</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1.0</td>
<td>008</td>
<td>7.62394E-01</td>
<td>3.90299E-04</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1.0</td>
<td>009</td>
<td>7.2081E-01</td>
<td>4.27354E-04</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2.0</td>
<td>010</td>
<td>7.21523E-01</td>
<td>4.02775E-04</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2.0</td>
<td>011</td>
<td>6.97954E-01</td>
<td>4.88269E-04</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2.0</td>
<td>012</td>
<td>6.64037E-01</td>
<td>4.88326E-04</td>
<td></td>
</tr>
</tbody>
</table>

Comparison of 4.5 kg Pu Ingot and Rings
Advanced Topics

Tied parameters

Concurrent jobs
Parameter Expansion (1)

• Standard inner & outer schemes for determining job parameters

Example:

\[
\begin{align*}
&c @@ A = 1 \ 2 \\
&c @@ B = 3 \ 4 \\
&c @@ C = 5 \ 6 \\
&c @@ D = 7 \ 8 \\
&c @@ E = 9 \\
\end{align*}
\]

**Outer:** all combinations, 16 cases

\[
\{1,3,5,7,9\}, \{2,3,5,7,9\}, \{1,4,5,7,9\}, \{2,4,5,7,9\}, \\
\{1,3,6,7,9\}, \{2,3,6,7,9\}, \{1,4,6,7,9\}, \{2,4,6,7,9\}, \\
\{1,3,5,8,9\}, \{2,3,5,8,9\}, \{1,4,5,8,9\}, \{2,4,5,8,9\}, \\
\{1,3,6,8,9\}, \{2,3,6,8,9\}, \{1,4,6,8,9\}, \{2,4,6,8,9\},
\]

**Inner:** 2 cases

\[
\{1,3,5,7,9\}, \{2,4,6,8,9\}
\]

• The inner & outer schemes for determining job parameters can be modified
  – Often desirable to deal with groups of parameters that are varied
  – 2 or more parameters can be “tied” together, to vary in an inner manner
  – Tied parameter lists must have the same lengths
Parameter Expansion  (2)

These examples assume that the -outer option is in effect for all parameter combinations.

Example:
```
c @@ tied = A B
```
```
c @@ A = 1 2  
c @@ B = 3 4  
c @@ C = 5 6  
c @@ D = 7 8  
c @@ E = 9
```

Cases, \{A,B,C,D,E\}:
{1,3,5,7,9},  {1,3,6,7,9},
{1,3,5,8,9},  {1,3,6,8,9},
{2,4,5,7,9},  {2,4,6,7,9},
{2,4,5,8,9},  {2,4,6,8,9}

Example:
```
c @@ tied = A B C
```
```
c @@ A = 1 2  
c @@ B = 3 4  
c @@ C = 5 6  
c @@ D = 7 8  
c @@ E = 9
```

Cases, \{A,B,C,D,E\}:
{1,3,5,7,9},  {1,3,5,8,9},
{1,3,5,8,9},  {2,4,5,7,9},
{2,4,5,8,9},  {2,4,6,7,9},
{2,4,6,8,9}

Example:
```
c @@ tied = A B C D
```
```
c @@ A = 1 2  
c @@ B = 3 4  
c @@ C = 5 6  
c @@ D = 7 8  
c @@ E = 9
```

Cases, \{A,B,C,D,E\}:
{1,3,5,7,9},  {2,4,6,8,9}
Parameter Expansion (3)

The -inner & -outer options can be varied for different parameters, and mixed with tied parameters

Example:
```
c @@ options = -inner
  c @@ A = 1 2
  c @@ B = 3 4
  c @@ C = 5 6
  c @@ D = 7 8
  c @@ E = 9
```

Cases:
{1,3,5,7, 9},  {2,4,6,8, 9},

Example:
```
c @@ options = -inner
  c @@ A = 1 2
  c @@ B = 3 4
c @@ options = -outer
  c @@ C = 5 6
  c @@ D = 7 8
  c @@ E = 9
```

Cases:
{1,3,5,7, 9},  {1,3, 6,8, 9},  
{2,4, 5,7, 9},  {2,4, 6,8, 9}

Example:
```
c @@ options = -outer
  c @@ tied = A B
  c @@ A = 1 2
  c @@ B = 3 4
  c @@ tied = C D
  c @@ C = 5 6
  c @@ D = 7 8
  c @@ E = 9
```

Cases:
{1,3, 5,7, 9},  {1,3, 6,8, 9},  
{2,4, 5,7, 9},  {2,4, 6,8, 9}

Example:
```
c @@ tied = A B C D
  c @@ A = 1 2
  c @@ B = 3 4
  c @@ C = 5 6
  c @@ D = 7 8
  c @@ E = 9
```

Cases:
{1,3,5,7, 9},  {2,4,6,8, 9}
Concurrent Jobs  (1)

• **By default, jobs for the different cases are run sequentially**
  - For `--run`: jobs for each case are run on the current computer, sequentially (one-at-a-time)
  - For `--submit`: separate batch jobs are submitted for each case,
  - For either `--run` or `--submit`, multiple threads can be used for the mcnp6 runs in each case, by using the option `--mcnp_opts 'tasks 8'`

• **For Linux & Mac systems, not Windows:**
  - Multiple concurrent cases can be run, even when threads are used
  - The `--ppn n` option specifies the number of processes per node (ie, cases to be run concurrently)

• **Examples:**
  - On a system with 24 hyperthreads, could run 6 cases at a time with 4 threads each:
    ```
    mcnp_pstudy --i inp.txt --mcnp_opts 'tasks 4' --ppn 6 --setup --run
    ```
  - For a cluster with 16 cores/node, can submit jobs with 16 cases each:
    ```
    mcnp_pstudy --i inp.txt --ppn 16 --setup --submit
    ```
Fission Matrix & Higher Eigenmodes

Forrest Brown
Senior Scientist, Monte Carlo Codes Group, LANL
National Lab Professor, University of New Mexico
Fission Matrix Capability for MCNP Monte Carlo

• Introduction
  – MCNP
  – Higher Eigenmodes
  – Green's Functions & Transport

• Fission Matrix
  – Theory
  – Sparse Storage
  – Transport Theory

• Examples
  – Homogeneous 2D Reactor
  – Whole-core PWR, 2D
  – Whole-core PWR, 3D (Kord Smith)
  – Advanced Test Reactor
  – Spent Fuel Storage Vault

• Conclusions

Introduction - Higher Eigenmodes

Vibrating strings:

- Higher modes add "tone", but die away quickly
- Fundamental mode persists
- Feedback, instability, nonlinear effects, ..., may excite higher modes
Introduction - Green's Functions & Transport Theory

- **F(A→B)**
  - Green's function, "here-to-there" function
  - Probability that source at point A produces source at point B

- **Transport theory - Peierl's equation for multiplying system**

\[
S(\mathbf{r}) = \frac{1}{k_{\text{eff}}} \cdot \int \text{d}\mathbf{r}' \cdot S(\mathbf{r}') \cdot F(\mathbf{r}' \rightarrow \mathbf{r})
\]

  - Discretize space into blocks, or mesh regions
  - Compute \( F(\mathbf{r}' \rightarrow \mathbf{r}) \) with Monte Carlo
  - Solve matrix eigenvalue problem for sources:

\[
\hat{S} = \frac{1}{k_{\text{eff}}} \cdot \mathbf{F} \cdot \hat{S}
\]

  - Can also solve for higher modes
Fission Matrix
In
MCNP Monte Carlo
Fission Matrix - Theory

- **Transport equation, k-eigenvalue form**

\[
M \cdot \Psi(\bar{r}, E, \hat{\Omega}) = \frac{\chi(E)}{4\pi} \cdot S(\bar{r}),
\]

\[
M \cdot \Psi(\bar{r}, E, \hat{\Omega}) = \frac{1}{k} \cdot \frac{\chi(E)}{4\pi} \cdot S(\bar{r}),
\]

\[
\begin{align*}
M \cdot \Psi(\bar{r}, E, \hat{\Omega}) &= \hat{\Omega} \cdot \nabla \Psi(\bar{r}, E, \hat{\Omega}) + \Sigma_T(\bar{r}, E) \Psi(\bar{r}, E, \hat{\Omega}) \\
&\quad - \int \int dE' d\hat{\Omega}' \Sigma_S(\bar{r}, E' \rightarrow E, \hat{\Omega}' \rightarrow \hat{\Omega}) \Psi(\bar{r}, E', \hat{\Omega'}),
\end{align*}
\]

\[
S(\bar{r}) = \int \int dE' d\hat{\Omega}' \nu \Sigma_F(\bar{r}, E') \Psi(\bar{r}, E', \hat{\Omega'}),
\]

- **Define Green’s function & integral transport equation**

\[
M \cdot G(\bar{r}_0, E_0, \hat{\Omega}_0 \rightarrow \bar{r}, E, \hat{\Omega}) = \delta(\bar{r} - \bar{r}_0) \cdot \delta(E - E_0) \cdot \delta(\hat{\Omega} - \hat{\Omega}_0),
\]

\[
\Psi(\bar{r}, E, \hat{\Omega}) = \frac{1}{k} \cdot \int \int \int d\bar{r}_0 \ dE_0 \ d\hat{\Omega}_0 \ \frac{\chi(E_0)}{4\pi} \cdot S(\bar{r}_0) \cdot G(\bar{r}_0, E_0, \hat{\Omega}_0 \rightarrow \bar{r}, E, \hat{\Omega})
\]

- **Multiply by \( \nu \Sigma_F \), integrate over \( E, \Omega, \) & initial regions \( (r_0) \) & final regions \( (r) \)**

\[
S_i = \frac{1}{k} \cdot \sum_{J=1}^{N} F_{i,J} \cdot S_J
\]

\[
F_{i,J} = \int_{\bar{r} \in V_i} \int_{\bar{r}_0 \in V_j} \frac{S(\bar{r}_0)}{S_j} \cdot \int \int \int dE d\hat{\Omega} dE_0 d\hat{\Omega}_0 \cdot \nu \Sigma_F(\bar{r}, E) \cdot \frac{\chi(E_0)}{4\pi} \cdot G(\bar{r}_0, E_0, \hat{\Omega}_0 \rightarrow \bar{r}, E, \hat{\Omega})
\]

\[
S_J = \int_{\bar{r} \in V_j} S(\bar{r}') d\bar{r}' = \int \int \int d\bar{r}' dE' d\hat{\Omega}' \nu \Sigma_F(\bar{r}', E') \Psi(\bar{r}', E', \hat{\Omega'}),
\]

**Exact equations for integral source \( S_i \), \( N = \# \) spatial regions, \( F \) is N\times N matrix**
Fission Matrix - Theory

- \( F_{i,j} = \) next-generation fission neutrons produced in region I, for each fission neutron starting in region J \( (J \rightarrow I) \)

- In the equation for \( F \),
  - \( S(r_0)/S_J \) is a local weighting function within region J
  - As \( V_J \rightarrow 0 \):
    - \( S(r_0)V_J/S_J \rightarrow 1 \)
    - Discretization errors \( \rightarrow 0 \)
    - Can accumulate tallies of \( F_{i,j} \) even if not converged

- \( F_{i,j} \) tallies:
  - Previous F-matrix work: tally during neutron random walks
  - Present F-matrix work: tally only point-to-point, using fission-bank in master proc (~free)
    - Eliminates excessive communications for parallel
    - Provides more consistency, \( F_{i,j} \) nonzero only in elements with actual sites
    - Analog-like treatment, better for preserving overall balance
MCNP Criticality Calculations

Monte Carlo K-effective Calculation
1. Start with fission source & k-eff guess
2. Repeat until converged:
   • Simulate neutrons in cycle
   • Save fission sites for next cycle
   • Calculate k-eff, renormalize source
3. Continue iterating & tally results

For Fission Matrix calculation
During standard k-eff calculation, at the end of each cycle:
   • Estimate $F_{i,J}$ tallies from start & end points in fission bank (~ free)
   • Accumulate $F_{i,J}$ tallies, over all cycles (even inactive cycles)
After the Monte Carlo is complete:
   • Normalize $F_{i,J}$ accumulators, divide by total sources in J regions
   • Find eigenvalues/vectors of F matrix (power iteration, with deflation)
Fission Matrix – Sparse Structure

• For a spatial mesh with N regions, F matrix is N x N
  – 100 x 100 x 100 mesh → F is $10^6 \times 10^6$ → 8,000 GB memory
  – In the past, memory storage was the major limitation

• Sparse storage for F matrix

  3D reactor with
  15x15 spatial mesh,
  225x225 F matrix

  – Don’t store zero elements, use sparse storage scheme

  – For 100x100x100 mesh, reduces F matrix storage to a few GB
K-eigenvalue Form of Transport Equation

- **Structure & properties**
  - 60 years ago:
    A single, non-negative, real, fundamental eigenfunction & eigenvalue exist
  - 50 years ago:
    For 1-speed or 1-group: A complete set of self-adjoint, real eigenfunctions & discrete eigenvalues exist
    - Energy-dependent transport equation is bi-orthogonal, forward & adjoint modes are orthogonal
    - Nothing else proven, always assumed that higher-mode solutions exist

- In the present work based on the Fission Matrix:
  - We provide evidence that higher modes exist, are real, have discrete eigenvalues, and are very nearly self-adjoint (for reactor-like problems)
  - Approach is similar to Birkhoff’s original proof for fundamental mode
  - This has never been done before using continuous-energy Monte Carlo

\[
M \cdot \Psi(\vec{r},E,\hat{\Omega}) = \frac{1}{K} \cdot \frac{\chi(E)}{4\pi} \cdot S(\vec{r})
\]
Homogeneous 2D Reactor Eigenmodes for:

Whole-core Model
Half-core Model
Quarter-core Model
Eigenmodes for Homogeneous 2D Reactor

Whole Core Model

Half Core Model

Quarter Core Model

0 1 2 3 4 5

6 7 8 9 10 11
Whole-core 2D PWR

Eigenvalue spectrum

Spatial Eigenmodes
Whole-core 2D PWR Model

2D PWR (Nakagawa & Mori model)

- **48 1/4 fuel assemblies:**
  - 12,738 fuel pins with cladding
  - 1206 1/4 water tubes for control rods or detectors

- **Each assembly:**
  - Explicit fuel pins & rod channels
  - 17x17 lattice
  - Enrichments: 2.1%, 2.6%, 3.1%

- **Dominance ratio ~ .98**

- **Calculations used whole-core model, symmetric quarter-core shown at right**

- ENDF/B-VII data, continuous-energy

- Tally fission rates in each quarter-assembly
PWR – Eigenvalue Spectrum & Fundamental Mode

500 K neutrons / cycle
Fission matrix tallies for cycles 4-55

- Fission matrix computed during MCNP k-effective inactive cycles
- Fundamental eigenmode of the fission matrix for a 2D whole-core PWR model, for various spatial meshes used to tally the fission matrix
PWR – Eigenmodes for 120x120x1 Spatial Mesh

\[
\begin{array}{c|c}
 n & K_n \\
\hline
0 & 1.29480 \\
1 & 1.27664 \\
2 & 1.27657 \\
3 & 1.25476 \\
4 & 1.24847 \\
5 & 1.24075 \\
6 & 1.22160 \\
7 & 1.22141 \\
8 & 1.19745 \\
9 & 1.19743 \\
10 & 1.18825 \\
11 & 1.18305 \\
12 & 1.15619 \\
13 & 1.14633 \\
14 & 1.14617 \\
15 & 1.14584 \\
\end{array}
\]
PWR – First 100 Eigenmodes
PWR – First 100 Eigenmodes, with More Neutrons
Eigenvalue Spectra with Varying Meshes

Real( $k_i$ )

$N = \text{number of mesh regions}$

( Fission matrix size = $N \times N$ )

$K_i$
For fine-enough spatial mesh, eigenvalue spectrum converges
Are the Eigenvalues Real or Complex?

Real( $k_i$):

The appearance of complex eigenvalues appears to be strictly an artifact of Monte Carlo statistical noise.

When more neutrons/cycle are used to decrease statistical noise, complex components diminish or vanish.

Imag( $k_i$):

The first few 100s or 1000s of discrete eigenvalues are real, and presumably all would be with sufficiently large neutrons/cycle.

120 by 120 Spectrum, Varying Neutrons/cycle
Strictly, eigenfunctions of the transport equation are bi-orthogonal. As shown above, forward eigenfunctions are very nearly orthogonal.
Kord Smith
Challenge Problem
-
3D Whole-Core PWR
MCNP & the "Kord Smith Challenge"

Full core, 3D benchmark for assessing MC computer performance
- Specified by Hoogenboom & Martin for OECD/NEA (2010)
- LWR model: 241 assemblies, 264 fuel pins/assembly
- Fuel contains 17 actinides + 16 fission products; borated water
- Detailed 3D MCNP model
  - Mesh tallies for pin powers, (63,624 pins) x (100 axial) = 6.3M pin powers
  - Runs easily on deskside computer (Mac Pro, 2 quad-core, 8 GB)
Standard MCNP & the "Kord Smith Challenge"

Pin Powers & Std.Dev

Assembly Power & Std.Dev

Axial  Mid  Top

$K_{eff}$ & $H_{src}$ Convergence

200M neutrons
Mac Pro, 8 cpu
Kord Smith Challenge Eigenvalue Spectrum

First 15 eigenvalues for 21x21x20 and 42x42x20 mesh

21x21x20 mesh, Real(K) spectrum
Eigenfunctions from Fission Matrix

XY plots of eigenfunctions at various Z elevations

55 cycles, 1 M neutrons/cycle, 42x42x20 mesh, 35280x4913 fission matrix
Eigenvalues & Inner Products of Eigenfunctions

<table>
<thead>
<tr>
<th>n</th>
<th>$K_n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.99919</td>
</tr>
<tr>
<td>1</td>
<td>0.98483</td>
</tr>
<tr>
<td>2</td>
<td>0.98362</td>
</tr>
<tr>
<td>3</td>
<td>0.98469</td>
</tr>
<tr>
<td>4</td>
<td>0.96956</td>
</tr>
<tr>
<td>5</td>
<td>0.96950</td>
</tr>
<tr>
<td>6</td>
<td>0.96693</td>
</tr>
<tr>
<td>7</td>
<td>0.96591</td>
</tr>
<tr>
<td>8</td>
<td>0.96043</td>
</tr>
<tr>
<td>9</td>
<td>0.95671</td>
</tr>
<tr>
<td>10</td>
<td>0.95178</td>
</tr>
<tr>
<td>11</td>
<td>0.95078</td>
</tr>
<tr>
<td>12</td>
<td>0.94524</td>
</tr>
<tr>
<td>13</td>
<td>0.94497</td>
</tr>
<tr>
<td>14</td>
<td>0.94472</td>
</tr>
</tbody>
</table>

42 x 42 x 20 spatial mesh, 35280 x 4913 fission matrix
55 cycles, 1 M neutrons/cycle
fission matrix tallies for cycles 4-55

Inner products of eigenfunctions
Convergence Acceleration Using Fission Matrix

- Fission matrix can be used to accelerate convergence of the MCNP neutron source distribution during inactive cycles.

- Very impressive convergence improvement.

$\text{accelerated using F matrix}$

![Graph showing $k_{eff}$ and $H_{src}$ comparison between standard MC and accelerated using F matrix.](image-url)
Advanced Test Reactor

Idaho National Laboratory
Advanced Test Reactor

“Serpentine Arrangement of Highly Enrichment Water-Moderated Uranium-Aluminide Fuel Plates Reflected by Beryllium”

Figure 20. An XY View at x=0, y=0 of the Benchmark Model.

ATR - Fission Matrix Structure

Matrix structure
(50x50 spatial mesh)

Four matrix columns (100x100 spatial mesh)
ATR - Fundamental Eigenvector, Eigenvalues

Real\( (k_i) \)

Fundamental mode, 100x100 spatial mesh
ATR - Eigenmodes (100x100 spatial mesh)
# ATR - Fission Matrix Orthogonality

<table>
<thead>
<tr>
<th>$n$</th>
<th>$K_n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.99490</td>
</tr>
<tr>
<td>1</td>
<td>0.85630</td>
</tr>
<tr>
<td>2</td>
<td>0.84612</td>
</tr>
<tr>
<td>3</td>
<td>0.78265</td>
</tr>
<tr>
<td>4</td>
<td>0.64564</td>
</tr>
<tr>
<td>5</td>
<td>0.55461</td>
</tr>
<tr>
<td>6</td>
<td>0.55207</td>
</tr>
<tr>
<td>7</td>
<td>0.53659</td>
</tr>
<tr>
<td>8</td>
<td>0.47004</td>
</tr>
<tr>
<td>9</td>
<td>0.46173</td>
</tr>
<tr>
<td>10</td>
<td>0.45794</td>
</tr>
<tr>
<td>11</td>
<td>0.41144</td>
</tr>
<tr>
<td>12</td>
<td>0.32865</td>
</tr>
<tr>
<td>13</td>
<td>0.29454</td>
</tr>
<tr>
<td>14</td>
<td>0.28401</td>
</tr>
<tr>
<td>15</td>
<td>0.28327</td>
</tr>
</tbody>
</table>

100 x 100 x 1 spatial mesh, no sparsification
55 cycles, 1 M neutrons/cycle
fission matrix tallies for cycles 4-55

**Inner products of eigenfunctions**
MCNP Fission Matrix

Spent Fuel Storage Vault
(idealized benchmark)

Loosely-Coupled Problem
Fuel Vault Problem

Fuel Storage Vault

K vs cycle

H_{src} vs cycle

Assembly Heating Distribution

For this calculation,

- Should discard ~20 cycles if calculating $K_{eff}$ only
- Should discard ~2000 cycles if calculating heating distribution
Eigenvalue Spectrum for Fuel Vault Problem- First 360

Real( $k_i$), $i = 1, 2, ..360$

36 semi-coupled assemblies $\rightarrow$ Mini-groups of 36 in size
XY Eigenmodes of Fuel Vault Problem, 96 by 12 by 10

XY planes mid-height. Axial shape is cosine, #10,13,15 have change in sign in z.
Fuel Vault Problem - Convergence Acceleration

It takes ~2,000 cycles for standard MC to converge for this problem,

Using the fission matrix for source convergence acceleration, only ~20 cycles are needed

standard MC accelerated using F matrix

Standard MC decreases slowly, converges to same value as F matrix after ~2,000 cycles
Conclusions

• Fission matrix capability has been added to MCNP (R&D for now)

• Tested on variety of real problems (3D, continuous-energy)

• Can obtain fundamental & higher eigenmodes
  
  – Empirical evidence for: existence of higher modes, real, discrete eigenvalues, very nearly orthogonal eigenmodes (for reactor-like problems)

  – Higher eigenmodes are important for
    BWR void stability, higher-order perturbation theory, Xenon oscillations, quasi-static transient analysis, control rod worth, correlation effects on statistics, accident behavior, etc., etc., etc.

• Can provide very effective acceleration of source convergence
On-The-Fly Neutron Doppler Broadening for MCNP

Forrest Brown¹, William Martin², Gokhan Yesilyurt³, Scott Wilderman²

¹Monte Carlo Methods (XCP-3), LANL
²University of Michigan
³Argonne National Laboratory
Abstract

On-The-Fly Neutron Doppler Broadening for MCNP

Forrest Brown, William Martin, Gokhan Yesilyurt, Scott Wilderman

The University of Michigan, ANL, and LANL have been collaborating on a US-DOE-NE University Programs project “Implementation of On-the-Fly Doppler Broadening in MCNP5 for Multiphysics Simulation of Nuclear Reactors.” This talk describes the project and provides results from the initial implementation of On-The-Fly Doppler broadening (OTF) in MCNP and testing.

The OTF methodology involves high precision fitting of Doppler broadened cross-sections over a wide temperature range (the target for reactor calculations is 250-3200K). The temperature dependent fits are then used within MCNP during the neutron transport, for OTF broadening based on cell temperatures. It is straightforward to extend this capability to cover any temperature range of interest, allowing the Monte Carlo simulation to account for a continuous distribution of temperature ranges throughout the problem geometry.
On-The-Fly Neutron Doppler Broadening for MCNP

• Introduction
  – Doppler Broadening - Obvious Stuff
  – Methods for Handling Temperature Variations

• OTF Doppler Broadening in MCNP
  – OTF Methodology
  – Union Energy Mesh
  – Temperature Fitting
  – OTF Doppler in MCNP
  – Testing
  – Work-in-Progress
Introduction – Doppler Broadening

- Neutrons born in MeV range from fission
- Most fissions caused by thermal neutrons
- 1/3 of neutron losses are due to $^{238}$U capture in epithermal energy range during slowing down

UO$_2$ Fuel Pin

3.1% Enriched
293.6 °K
.01 eV – 20 MeV

$^{238}$U capture cross-section

Neutron Flux in Fuel per unit lethargy

Epithermal Range

Thermal Peak

Fission Peak
Introduction – Doppler Broadening

238U capture cross-section

UO₂ Fuel Pin
3.1% Enriched
293.6 °K

Detail for
1 eV – 1 KeV

1/3 of neutron losses are due to 238U capture at epithermal energies during slowing down
Introduction – Doppler Broadening

UO₂ Fuel Pin

3.1% Enriched
293.6 °K vs 900 °K
.01 eV – 20 MeV

Keeping same densities, but changing cross-sections:

\[ k_{\text{inf}} \text{ (cold)} = 1.34498 \ (8) \]
\[ k_{\text{inf}} \text{ (hot)} = 1.31167 \ (8) \]

At higher temperatures, Doppler broadening of resonance cross-sections increases resonance capture
Introduction

Doppler Broadening

Temperature Variation in Monte Carlo Codes
Neutron - Nucleus Interactions

- **Low neutron energies:**
  - $S(\alpha,\beta)$ interaction data is used in modeling collision physics
    - 2002 data: $10^{-5}$ eV - 4.46 eV neutron energies (15 nuclides)
    - 2012 data: $10^{-5}$ eV - 9.15 eV neutron energies (20 nuclides)
  - $S(\alpha,\beta)$ data accounts for target nucleus chemical binding, molecular binding, crystal structure, thermal motion, etc.
  - Nuclides without $S(\alpha,\beta)$ data: use free-gas model (see below)

- **High neutron energies:**
  - Target nucleus thermal motion neglected
  - Typical: $E_{\text{neutron}} > 400 \text{ kT}$ for $A>1$

- **Epithermal neutron energies:**
  - Target nucleus thermal motion important
  - Free-gas scattering model -- nuclides have Maxwell-Boltzmann energy distribution at temperature $T$, isotropic direction

\[ f(E_{\text{nuc}}) = \frac{2}{\sqrt{\pi}} \cdot \frac{1}{kT} \cdot \left( \frac{E_{\text{nuc}}}{kT} \right)^{1/2} \cdot e^{-E_{\text{nuc}}/kT} \]

Gamma($kT, 3/2$),
mean = 1.5 kT
mode = .5 kT
Doppler Broadening

- Detailed kinematics of collisions must include nucleus $E$ & $\Omega$

- For free-flight, selection of collision isotope, & tallies of overall reactions: must use effective cross-sections, averaged over $(E, \Omega)$ distribution of nuclides at temperature $T$

$$\sigma_{\text{eff}}(v) = \int \frac{|\vec{v} - \vec{V}|}{v} \sigma(|\vec{v} - \vec{V}|)P(\vec{V})d\vec{V}, \quad P(\vec{V}) = \left(\frac{M}{2\pi kT}\right)^{3/2} e^{-\frac{M\vec{V}^2}{2kT}}$$

Doppler broadening equation $\quad v = \text{neutron, } V=\text{nucleus}$

This is a convolution of the cross-section with the target energy or speed distribution. Smears out & smoothes the cross-section, reduces peak values.
$^{238}\text{U}$ Doppler Broadening Examples
Doppler Broadening - Numerics

- ENDF/B nuclear data is represented by piecewise-linear tabulation of $\sigma(E)$

\[
\sigma_{\text{eff}}(v) = \int \frac{|\vec{v} - \vec{V}|}{v} \sigma(|\vec{v} - \vec{V}|) P(\vec{V}) \, d\vec{V}, \quad P(\vec{V}) = \left( \frac{M}{2\pi kT} \right)^{3/2} e^{-\frac{M}{2kT}v^2}
\]

- Red Cullen (NSE, 1976) showed how to exactly perform this convolution of Maxwell Boltzmann PDF with piecewise-linear $\sigma(E)$, called sigma1 method

- NJOY code is similar & adaptively chooses energy points to meet 0.1% accuracy in $\sigma_{\text{eff}}$ at $T$

- $\sigma_{\text{eff}}(E)$ has different E-mesh at different $T$’s

- Very compute-intensive, typically performed prior to Monte Carlo in preparing nuclear data libraries
Doppler Broadening with Adaptive Energy Grid

<table>
<thead>
<tr>
<th>Temperature Range (K)</th>
<th>Field of Study</th>
</tr>
</thead>
<tbody>
<tr>
<td>77 - 293.6</td>
<td>Cold Neutron Physics</td>
</tr>
<tr>
<td>293.6 – 550</td>
<td>Benchmarking Calculations</td>
</tr>
<tr>
<td>550 – 1600</td>
<td>Reactor Operation</td>
</tr>
<tr>
<td>1600 – 3200</td>
<td>Accident Conditions</td>
</tr>
</tbody>
</table>

**NJOY – adaptive E grid for $^{238}$U Doppler broadening**

<table>
<thead>
<tr>
<th>Fractional Tolerance</th>
<th>0.1%</th>
<th>0.3%</th>
<th>0.5%</th>
<th>1.0%</th>
<th>2.0%</th>
<th>3.0%</th>
<th>4.0%</th>
<th>5.0%</th>
</tr>
</thead>
<tbody>
<tr>
<td>T (K)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>193131</td>
<td>122935</td>
<td>100646</td>
<td>76856</td>
<td>57347</td>
<td>49659</td>
<td>44955</td>
<td>41676</td>
</tr>
<tr>
<td>77</td>
<td>103600</td>
<td>70240</td>
<td>59900</td>
<td>50049</td>
<td>43716</td>
<td>41408</td>
<td>40250</td>
<td>39514</td>
</tr>
<tr>
<td>293.6</td>
<td>85247</td>
<td>60192</td>
<td>52352</td>
<td>44810</td>
<td>39965</td>
<td>38089</td>
<td>37104</td>
<td>36494</td>
</tr>
<tr>
<td>500</td>
<td>77676</td>
<td>55786</td>
<td>49097</td>
<td>42506</td>
<td>38188</td>
<td>36509</td>
<td>35565</td>
<td>35006</td>
</tr>
<tr>
<td>1000</td>
<td>67437</td>
<td>50226</td>
<td>44773</td>
<td>39625</td>
<td>35957</td>
<td>34593</td>
<td>33810</td>
<td>33282</td>
</tr>
<tr>
<td>1500</td>
<td>62302</td>
<td>47227</td>
<td>42557</td>
<td>38000</td>
<td>34881</td>
<td>33616</td>
<td>32956</td>
<td>32490</td>
</tr>
<tr>
<td>2000</td>
<td>58735</td>
<td>45153</td>
<td>41098</td>
<td>36957</td>
<td>34109</td>
<td>32999</td>
<td>32384</td>
<td>31918</td>
</tr>
<tr>
<td>2500</td>
<td>56248</td>
<td>43774</td>
<td>39933</td>
<td>36177</td>
<td>33586</td>
<td>32543</td>
<td>31948</td>
<td>31560</td>
</tr>
<tr>
<td>3000</td>
<td>54282</td>
<td>42707</td>
<td>39051</td>
<td>35557</td>
<td>33208</td>
<td>32192</td>
<td>31661</td>
<td>31314</td>
</tr>
</tbody>
</table>
Temperature Variation in Monte Carlo

What if there are 1000s of T's?

Six approaches:

1. **Traditional NJOY+MC (exact)**
   - NJOY data at specific problem T’s
   - Each MC region in MC uses specific pre-broadened data
   - Exact, very cumbersome, very large amount of xsec data

2. **Traditional NJOY+MC (approx.)**
   - Like (1), but round off T’s to nearest 10-20°
   - Approximate, very cumbersome, very large amount of xsec data

3. **Stochastic Mixing (approx)**
   - NJOY data at a few bounding T’s
   - Set up MC input with a mix of hot & cold data for a nuclide, such that average T for the mix matches problem T
   - Run MC, will sometimes get "hot" data, sometimes "cold", average is OK
   - Approximate, cumbersome, very large amount of xsec data

4. **OTF Sigma1 (Monk)**
   - Use only 1 set of NJOY datafiles
   - During MC, use sigma1 method to broaden data as needed
   - Exact, but very expensive, ~10x increase in computer time

5. **OTF Using Delta-Track (Serpent)**
   - Use only 1 set of NJOY datafiles
   - During MC, use delta-tracking rejection method to broaden data as needed
   - Exact, but complex & expensive, ~4x increase in computer time
   - Cannot do pathlength MC estimators or point-detector estimators

6. **OTF Temp. Fitted Data (MCNP)**
   - Use only 1 set of NJOY datafiles
   - Prior to MC, generate OTF datasets to handle temperature variation
   - During MC, Doppler broaden as needed using fitting data
   - Exact, extra data for T-fits, ~1.1x increase in computer time
(1)+(2) Traditional NJOY+MC

- **Conventional MCNP problem specification:**
  - Temperatures are assigned to cells (geometry regions)
  - Materials are assigned to cells
  - Doppler broadening for temperature $T$ is performed on nuclides
  - Materials are composed of nuclides

- Cumbersome for 1,000+ cells/materials/temperatures/nuclides
- Many GB of data

(1) Exact, number of datasets = number of $T$’s
(2) Approx., match cell $T$ to closest material with nuclides at $T$’

![Diagram](attachment:image.png)
(3) Stochastic Mixing

- Often loosely called "stochastic interpolation" or "interpolation"
- This is simply mixing, not interpolation

- MCNP input example:
  - Want this at 500 K: m1000 92235 -.93 92238 -.07
  - Have these datasets from NJOY:
    92235.91c at 300 K, 92238.91c at 300 K
    92235.92c at 600 K, 92238.92c at 600 K
  - For mixing linear in T, mix 1/3 of 300 K data + 2/3 of 600 K data
    m1000 92235.91c -.31 92238.91c -.0233333
    92235.92c -.62 92238.92c -.0466667

- Cumbersome for 1,000+ cells/materials/temperatures/nuclides (could be scripted.....)

- Many GB of data, 2x nuclides, complex input
(4) OTF Sigma1,  (5) OTF Delta,  (6) OTF for MCNP

(4) OTF Sigma1
- Recently implemented in MONK
- Numerical sigma1 method OTF during neutron tracking
- Increases overall runtime by ~10x
- See Davies paper from ICNC-2011

(5) OTF Delta-tracking
- Currently being tested in Serpent
- Very elegant & innovative, very promising
- Increases overall runtime by ~2-4x, may improve
- Does not fit with many conventional MC schemes:
  - No pathlength estimators
  - No point-detector (flux at a point) tallies
  - Requires radical revisions to codes such as MCNP
- See Viitanen & Leppanen paper from PHYSOR-2012

(6) OTF for MCNP  -- rest of talk
OTF Doppler Broadening
-
U. Michigan + ANL + LANL
DOE NE-UP Project

OTF Methodology
Union Energy Mesh
Temperature Fitting
OTF Doppler in MCNP
Testing
Work-in-Progress
On-The-Fly Neutron Doppler Broadening

- **OTF Methodology** (for each nuclide)
  - Create union energy grid for a range of temperatures
  - Create fits for $\sigma_{\text{eff}}(T,E)$, for range of temperatures, on union E-grid
  - MCNP – evaluate $\sigma_{\text{eff}}(T,E)$ OTF during simulation

- **Comments**
  - Target application, for now: reactors
  - Relies on NJOY methodology
    - Supplements & extends NJOY
    - Methodology consistent with NJOY
    - Fitting $\sigma$ vs temperature (at each E)
      - High precision, least squares with singular value decomposition
      - Adaptive (for each E, MT, & nuclide)
      - Explicit, direct error checking for fits - fit error < linearization tolerance
      - Threaded parallel, broadening routines called millions of times
      - Over temperature, maintains accuracy consistent with NJOY
OTF Methodology – Union Energy Grid (1)

- $^{238}$U energy grid, 35-36 eV, various temperatures (ENDF/B-VII.0)

NJOY adapts the energy grid (for each nuclide, at given T) to preserve linearization tolerance

<table>
<thead>
<tr>
<th>Temperature (K)</th>
<th>Number of E pts</th>
</tr>
</thead>
<tbody>
<tr>
<td>293.6</td>
<td>157754</td>
</tr>
<tr>
<td>600</td>
<td>133964</td>
</tr>
<tr>
<td>900</td>
<td>122581</td>
</tr>
<tr>
<td>1200</td>
<td>115361</td>
</tr>
<tr>
<td>2500</td>
<td>99631</td>
</tr>
</tbody>
</table>

Union E-grid:
Need to determine 1 energy grid (for each nuclide) that preserves linear interpolation tolerance in E over the entire T range
OTF Methodology – Union Energy Grid

**For 1 nuclide, determine:**
- MT numbers for reactions to be broadened
- Energy range for broadening, \( E_{\text{min}} - E_{\text{max}} \)
  - Up to start of unresolved data, or high-threshold reactions (whichever smaller)
- Temperature range \( T_{\text{min}} - T_{\text{max}} \) & interval \( \Delta T \) for tolerance testing (input)
- Base set of \( \sigma_x(e)'s \) from NJOY at \( T_{\text{base}} \)
  - “\( x \)” = any MT reaction that needs broadening
  - ACE data file from NJOY:
    - Yesilyurt: \( T_{\text{base}}=0 \) K, Brown: \( T_{\text{base}}=293.6 \) K
- Energy grid from NJOY at \( T_{\text{min}} \)

**For 1 nuclide & a set of \( T \)'s in range, at each \( T \):**
- Adaptively add E points so that 0.1% linear tolerance is maintained
  - Exact Doppler broadening from \( T_{\text{base}} \) to \( T \), using sigma1 method
  - Check all broadened MT reaction data for each E interval
  - Subdivide E interval until 0.1% linearization tolerance met for all MT’s
  - Add E points as needed, do not remove E points
- Compute-intensive – millions of calls to sigma1 routine, parallel threads
- Typically expands number of E points by \( \sim 10\% \), for 293-3200 K range
- **Result:** union E-grid for nuclide, 0.1% linear tolerance over entire T range
OTF Methodology – Doppler Broadening vs Temperature

Near resonance peaks:
\[ \sigma_{T,C,F}(T) \sim \sum_{k=0}^{\infty} \frac{d_k}{T^{k/2}} \]

Mid resonance:
\[ \sigma_{T,C,F}(T) \sim \sum_{k=0}^{\infty} e_k T^{k/2} \]

Wings of resonance:
\[ \sigma_{T,C,F}(T) \sim \sum_{k=0}^{\infty} f_k T^k \]

Combined functional form:
\[ \sigma_{T,C,F}(T) \sim \sum_{k=1}^{n} \frac{a_k}{T^{k/2}} + \sum_{k=1}^{n} b_k T^{k/2} + c \]

- for specific E, MT
- n varies for E, MT
- \(a_k, b_k, c\) tabulated for E, MT

Functional forms for temperature fitting based on multilevel Adler-Adler model, with expansions for peak, mid-res, wings
OTF Methodology – Fitting vs T

• **For 1 nuclide, determine:**
  - MT numbers for reactions to be broadened
  - Energy range for broadening, $E_{\text{min}} - E_{\text{max}}$
    - Up to start of unresolved data, or high-threshold reactions (whichever smaller)
  - Temperature range $T_{\text{min}} - T_{\text{max}}$ & interval $\Delta T$ for tolerance testing (input)
  - Base set of $\sigma_x(e)$’s from NJOY at $T_{\text{base}}$
    - “x” = any MT reaction that needs broadening
    - ACE data file from NJOY:
      - Yesilyurt: $T_{\text{base}}=0$ K, Brown: $T_{\text{base}}=293.6$ K
  - Union energy grid for this nuclide & T range
  - Maximum order for temperature fitting
    - Adler-Adler based functional form, using powers of $T^{1/2}$ and $1/T^{1/2}$

• **For 1 nuclide, at each point in the union E grid:**
  - Exact Doppler broadening from $T_{\text{base}}$ to all $T$’s in range, using sigma1 method
  - Least-squares fitting over T
    - Singular value decomposition, least squares for temperature dependence
    - Fitting order chosen adaptively for each energy & reaction so that fits accurate within 0.1% for all $T$’s and all $E$’s in range, for all MT’s
  - Coefficients saved in files for MCNP use
OTF Methodology – MCNP OTF

• At problem setup, read in OTF data for various nuclides
  – Each OTF nuclide set can have different fit orders & union E-grid & reactions

• During simulation, if neutron in E-T range of fits
  – Use OTF data for each nuclide to create on-the-fly Doppler broadened cross-sections at current cell temperature
  – If outside E-T range of OTF data, use standard ACE data
  – Collision physics (exit E & angles) uses standard ACE data

• Only need to generate OTF datasets once, & then use for any problems

• Cost
  – Extra storage for OTF data
  – Extra computing for evaluating OTF functions (typical <10% runtime)

• Benefit
  – Less storage for ACE data (no need for multiple temperatures)
  – Can solve problems with 1000s of T’s or more, no limit
  – Greatly simplifies problem setup
OTF Testing - Yesilyurt
$^{238}$U Capture – NJOY vs OTF at 2000 K

Cross-sections from NJOY & OTF match within linearization tolerance 0.1% at all energies
NJOY vs OTF $^{238}$U Capture Cross-Section

- **NJOY vs OTF at 1000 K** (curves with higher peak)
- **NJOY vs OTF at 2000 K** (curves with lower peak)

Cross-sections from NJOY & OTF match within linearization tolerance 0.1% at all energies

6.67 eV resonance

20.9 eV resonance

36.6 eV resonance
MCNP Test Results – Doppler Defect Benchmark

- **Doppler Reactivity Benchmark**
  - Compare $k$-effective for HZP (hot, zero power) and HFP (hot, full power) conditions for a unit fuel cell typical of a PWR
  - **Basic model:**
    - **PWR fuel pin cell** with reflecting BCs, various enrichments
    - **HZP cases:** fuel at 600K, clad/moderator at 600K
    - **HFP cases:** fuel at 900K, clad/moderator at 600K
    - Uniform temperature within each fuel, clad, moderator region.
    - Number densities and dimensions adjusted for the HFP thermal expansion
    - 5M active neutron histories per each of 28 MCNP runs

- **NJOY+MCNP:** NJOY-broadened data at exact temperatures
- **OTF+MCNP:** OTF data for $^{16}\text{O}$, $^{234}\text{U}$, $^{235}\text{U}$, $^{238}\text{U}$ in fuel

- **OTF details**
  - For union E-grid: $T_{\text{base}}=293.6$K, T range 300-1000K, $\Delta T=100$K
  - For OTF fitting: 8th order, T range 300-1000K, $\Delta T=10$K
  - For general production use, would use larger T range & smaller $\Delta T$’s
# Doppler Defect Benchmark Results

<table>
<thead>
<tr>
<th>UO2 fuel pin</th>
<th>HZP</th>
<th>HFP</th>
<th>Doppler Coef. pcm/K</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.711% enrichment</td>
<td>NJOY+MCNP</td>
<td>0.66556 (18)</td>
<td>0.65979 (19)</td>
</tr>
<tr>
<td></td>
<td>OTF+MCNP</td>
<td>0.66567 (18)</td>
<td>0.66022 (19)</td>
</tr>
<tr>
<td>1.60% enrichment</td>
<td>NJOY+MCNP</td>
<td>0.96094 (26)</td>
<td>0.95293 (25)</td>
</tr>
<tr>
<td></td>
<td>OTF+MCNP</td>
<td>0.96026 (24)</td>
<td>0.95283 (23)</td>
</tr>
<tr>
<td>2.40% enrichment</td>
<td>NJOY+MCNP</td>
<td>1.09912 (27)</td>
<td>1.08997 (26)</td>
</tr>
<tr>
<td></td>
<td>OTF+MCNP</td>
<td>1.09923 (27)</td>
<td>1.08975 (28)</td>
</tr>
<tr>
<td>3.10% enrichment</td>
<td>NJOY+MCNP</td>
<td>1.17718 (27)</td>
<td>1.16744 (27)</td>
</tr>
<tr>
<td></td>
<td>OTF+MCNP</td>
<td>1.17703 (30)</td>
<td>1.16767 (30)</td>
</tr>
<tr>
<td>3.90% enrichment</td>
<td>NJOY+MCNP</td>
<td>1.23967 (27)</td>
<td>1.22920 (30)</td>
</tr>
<tr>
<td></td>
<td>OTF+MCNP</td>
<td>1.23953 (29)</td>
<td>1.22979 (29)</td>
</tr>
<tr>
<td>4.50% enrichment</td>
<td>NJOY+MCNP</td>
<td>1.27501 (30)</td>
<td>1.26526 (27)</td>
</tr>
<tr>
<td></td>
<td>OTF+MCNP</td>
<td>1.27534 (29)</td>
<td>1.26552 (29)</td>
</tr>
<tr>
<td>5.00% enrichment</td>
<td>NJOY+MCNP</td>
<td>1.29901 (31)</td>
<td>1.28920 (29)</td>
</tr>
<tr>
<td></td>
<td>OTF+MCNP</td>
<td>1.29907 (28)</td>
<td>1.28938 (29)</td>
</tr>
</tbody>
</table>

$$\rho = (\frac{1}{K_{HZP}} - \frac{1}{K_{HFP}}) \times 10^5 / 300 \text{ pcm/K}$$
Results – Fuel Assembly 1

- **Simplified PWR 15 x 15 fuel assembly**
  - From OECD/NEA fuel storage vault benchmark
    - Fuel = 900 K
    - Clad & water = 600 K
    - Outer iron rack = 293.6K
  - **Standard NJOY+MCNP5:**
    - 900K ACE data for fuel,
    - 600K ACE data for clad & mod
    - 293.6K ACE data for iron
  - **OTF+MCNP5**
    - use 293.6K ACE data for all nuclides
    - OTF data for all nuclides (except iron)
  - **MCNP5**
    - 20,000 neutrons/cycle,
    - 10 inactive cycles, 1000 active cycle
    - Reflecting BCs
Results - Fuel Assembly 1

- **K-effective**
  - NJOY+MCNP5: 1.13891 (15)
  - OTF+MCNP5: 1.13892 (15)

- **Total Fission**
  - NJOY+MCNP5: 0.464506 (.02%)
  - OTF+MCNP5: 0.464499 (.02%)

- **Total Capture in fuel**
  - NJOY+MCNP5: 0.250912 (.02%)
  - OTF+MCNP5: 0.250918 (.02%)

- **U235 capture in fuel**
  - NJOY+MCNP5: 0.089478 (.02%)
  - OTF+MCNP5: 0.089475 (.02%)

- **U238 capture in fuel**
  - NJOY+MCNP5: 0.160302 (.03%)
  - OTF+MCNP5: 0.160311 (.03%)

- **O16 capture in fuel**
  - NJOY+MCNP5: 9.73621e-4 (.11%)
  - OTF+MCNP5: 9.73248e-4 (.11%)
Results – Fuel Assembly II

- **Simplified PWR 15 x 15 fuel assembly, with varying temperatures**
  - From OECD/NEA fuel storage vault benchmark
    - Fuel = 900 K, 600 K, 300 K
    - Clad = 900 K, 600 K, 300 K
    - Water = 600 K, 300 K
    - Outer iron rack = 293.6K
  - **Standard NJOY+MCNP5:**
    - ACE data at explicit temperatures
  - **OTF+MCNP5**
    - use 293.6K ACE data for all nuclides
    - OTF data for all nuclides (except iron)
  - **MCNP5**
    - 20,000 neutrons/cycle,
    - 10 inactive cycles, 1000 active cycle
    - Reflecting BCs
## Results – Fuel Assembly II

### k-effective:

<table>
<thead>
<tr>
<th></th>
<th>STD</th>
<th>OTF</th>
</tr>
</thead>
<tbody>
<tr>
<td>STD</td>
<td>1.11599 (15)</td>
<td>1.11592 (15)</td>
</tr>
<tr>
<td>OTF</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Total fission

<table>
<thead>
<tr>
<th></th>
<th>900K</th>
<th>600K</th>
<th>300K</th>
</tr>
</thead>
<tbody>
<tr>
<td>STD</td>
<td>.045140 (.08%)</td>
<td>.161186 (.04%)</td>
<td>.248782 (.03%)</td>
</tr>
<tr>
<td>OTF</td>
<td>.045081 (.08%)</td>
<td>.161329 (.04%)</td>
<td>.248731 (.03%)</td>
</tr>
</tbody>
</table>

### Total capture in fuel

<table>
<thead>
<tr>
<th></th>
<th>900K</th>
<th>600K</th>
<th>300K</th>
</tr>
</thead>
<tbody>
<tr>
<td>STD</td>
<td>.027672 (.09%)</td>
<td>.096276 (.05%)</td>
<td>.116745 (.04%)</td>
</tr>
<tr>
<td>OTF</td>
<td>.027667 (.09%)</td>
<td>.096268 (.05%)</td>
<td>.116829 (.04%)</td>
</tr>
</tbody>
</table>

### U235 capture in fuel

<table>
<thead>
<tr>
<th></th>
<th>900K</th>
<th>600K</th>
<th>300K</th>
</tr>
</thead>
<tbody>
<tr>
<td>STD</td>
<td>.008993 (.08%)</td>
<td>.031910 (.04%)</td>
<td>.045998 (.03%)</td>
</tr>
<tr>
<td>OTF</td>
<td>.008983 (.08%)</td>
<td>.031932 (.04%)</td>
<td>.045987 (.03%)</td>
</tr>
</tbody>
</table>

### U238 capture in fuel

<table>
<thead>
<tr>
<th></th>
<th>900K</th>
<th>600K</th>
<th>300K</th>
</tr>
</thead>
<tbody>
<tr>
<td>STD</td>
<td>.018547 (.11%)</td>
<td>.063887 (.06%)</td>
<td>.070236 (.05%)</td>
</tr>
<tr>
<td>OTF</td>
<td>.018551 (.11%)</td>
<td>.063858 (.06%)</td>
<td>.070332 (.05%)</td>
</tr>
</tbody>
</table>

### O16 capture in fuel

<table>
<thead>
<tr>
<th></th>
<th>900K</th>
<th>600K</th>
<th>300K</th>
</tr>
</thead>
<tbody>
<tr>
<td>STD</td>
<td>1.15E-04 (.23%)</td>
<td>4.18E-04 (.14%)</td>
<td>4.37E-04 (.13%)</td>
</tr>
<tr>
<td>OTF</td>
<td>1.15E-04 (.23%)</td>
<td>4.16E-04 (.14%)</td>
<td>4.37E-04 (.13%)</td>
</tr>
</tbody>
</table>

*Fuel=900K, clad=900K, mod=600K*
*Fuel=600K, clad=600K, mod=600K*
*Fuel=300K, clad=300K, mod=300K*
OTF Work in Progress

• Better integration into MCNP (optimization)

• FIT_OTF fitting program
  – Investigate scaling & Chebychev, for better numerical stability
  – Investigate regression, to vary fit order by energy & reaction [done]

• U. Michigan work
  – Create OTF libraries for all nuclides in ENDF/B-VII.0
  – Test various applications: fuel assemblies, 3D whole core, LWR, HTGR, ...

• Methodology for Unresolved Resonances & S(α,β) data
  – Probable 1st cut – tables with temperature interpolation
  – Possible thesis topic for PhD student

• Implement corrected free-gas scatter model
  – Demonstrated, needs robust implementation

• Easy to extend to any temperature range
  – Need to investigate broadening for high-threshold reactions
References


Fit_OTF Example (1)

====================================================================
=  Fit_OTF: Command-line options:
=
=    perform_fitting = T
=    zaid = 92238.70c
=    ace_file =
=    ugrid_file = ugrid_92238.70c.txt
=    otf_file = otf_file.txt
=    fit order min = 1
=    fit order max = 8
=    fit min temp = 293.6000000000000 (if > ACE temp)
=    fit max temp = 1000.0000000000000
=    fit inc temp = 10.0000000000000
=
=    print n-th lines = 20
=
=    create ugrid = F
=
=    testing? = T
=    test_emin = 5.500000000000000E-006
=    test_emax = 7.500000000000000E-006
====================================================================
Fit_OTF Example (2)

.....read ACE file = 92238.70c
xsdir = /Volumes/fbb/fbrown/LANL/MCNP_DATA/xsdir
file  = /Volumes/fbb/fbrown/LANL/MCNP_DATA/endf70j

Info from ACE data file for ZAID = 92238.70c

Number of energies  =   157754
Atomic weight ratio = 236.005800 amu
Temperature         =  2.530100E-08 MeV,     293.6 K
Date                = 08/25/07
Info                = 92-U -238 at 293.6K from endf/b-vii.0 njoy99.248
endf MAT            = mat9237

MT reactions (std+gpd+mtlist), n= 52
  1  101  2  301  202  16  17  18  37  51
  52  53  54  55  56  57  58  59  60  61
  62  63  64  65  66  67  68  69  70  71
  72  73  74  75  76  77  78  79  80  81
  82  83  84  85  86  87  88  89  90  91
  102  444

MT reactions for fission, n= 1
  18

URR-probability tables are present
energy range:  2.000001E-02 MeV - 1.490287E-01 MeV

Doppler broadening info:
energy range:  1.000000E-11 MeV - 2.000000E-02 MeV
MT reactions for Doppler broadening, n= 8
  1  101  2  301  202  18  102  444

........................................................................................
Fit_OTF Example (3)

..... read ugrid

..... ugrid e pts = 168626
..... ugrid min e = 9.999999999999999E-012
..... ugrid max e = 2.074926000000000E-002

Broadening & fitting info:
  number of ugrid pts = 168603
  min energy = 1.000000E-11 MeV
  max energy = 2.000001E-02 MeV
  number of temps = 71
  min temp = 293.6 K
  max temp = 993.6 K
  temp increment = 10.0 K
  number of reactions = 8
  MT numbers = 1 101 2 301 202 18 102 444
  MT for tot fission = 18

fitting order is variable, to meet tolerance
  min order = 1
  max order = 8
  max number coefs = 17
### Fit_OTF Example (4)

*MT-order for kprt lines. Errors given if >tolerance.*

<table>
<thead>
<tr>
<th>k</th>
<th>e</th>
<th>ev:</th>
<th>1-1</th>
<th>101-2</th>
<th>2-3</th>
<th>301-1</th>
<th>202-4</th>
<th>18-1</th>
<th>102-5</th>
<th>444-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>460</td>
<td>5.60999</td>
<td>1-2</td>
<td>101-3</td>
<td>2-4</td>
<td>301-1</td>
<td>202-5</td>
<td>18-1</td>
<td>102-6</td>
<td>444-1</td>
<td></td>
</tr>
<tr>
<td>480</td>
<td>6.03017</td>
<td>1-2</td>
<td>101-3</td>
<td>2-4</td>
<td>301-1</td>
<td>202-5</td>
<td>18-1</td>
<td>102-6</td>
<td>444-1</td>
<td></td>
</tr>
<tr>
<td>500</td>
<td>6.28893</td>
<td>1-4</td>
<td>101-5</td>
<td>2-6</td>
<td>301-1</td>
<td>202-7</td>
<td>18-1</td>
<td>102-8</td>
<td>444-1</td>
<td></td>
</tr>
<tr>
<td>520</td>
<td>6.41344</td>
<td>1-4</td>
<td>101-5</td>
<td>2-6</td>
<td>301-1</td>
<td>202-7</td>
<td>18-1</td>
<td>102-8</td>
<td>444-1</td>
<td></td>
</tr>
<tr>
<td>540</td>
<td>6.47134</td>
<td>1-4</td>
<td>101-5</td>
<td>2-6</td>
<td>301-1</td>
<td>202-7</td>
<td>18-1</td>
<td>102-8</td>
<td>444-1</td>
<td></td>
</tr>
<tr>
<td>560</td>
<td>6.50794</td>
<td>1-4</td>
<td>101-5</td>
<td>2-6</td>
<td>301-1</td>
<td>202-7</td>
<td>18-1</td>
<td>102-8</td>
<td>444-1</td>
<td></td>
</tr>
<tr>
<td>580</td>
<td>6.53025</td>
<td>1-4</td>
<td>101-5</td>
<td>2-6</td>
<td>301-1</td>
<td>202-7</td>
<td>18-1</td>
<td>102-8</td>
<td>444-1</td>
<td></td>
</tr>
<tr>
<td>600</td>
<td>6.55195</td>
<td>1-3</td>
<td>101-4</td>
<td>2-5</td>
<td>301-1</td>
<td>202-6</td>
<td>18-1</td>
<td>102-7</td>
<td>444-8</td>
<td></td>
</tr>
<tr>
<td>620</td>
<td>6.57375</td>
<td>1-4</td>
<td>101-5</td>
<td>2-6</td>
<td>301-1</td>
<td>202-7</td>
<td>18-1</td>
<td>102-8</td>
<td>444-8</td>
<td></td>
</tr>
<tr>
<td>640</td>
<td>6.59798</td>
<td>1-3</td>
<td>101-4</td>
<td>2-5</td>
<td>301-1</td>
<td>202-6</td>
<td>18-7</td>
<td>102-8</td>
<td>444-8</td>
<td></td>
</tr>
<tr>
<td>660</td>
<td>6.65659</td>
<td>1-3</td>
<td>101-4</td>
<td>2-5</td>
<td>301-1</td>
<td>202-6</td>
<td>18-7</td>
<td>102-8</td>
<td>444-8</td>
<td></td>
</tr>
<tr>
<td>680</td>
<td>6.70483</td>
<td>1-3</td>
<td>101-4</td>
<td>2-5</td>
<td>301-1</td>
<td>202-6</td>
<td>18-7</td>
<td>102-8</td>
<td>444-8</td>
<td></td>
</tr>
<tr>
<td>700</td>
<td>6.76322</td>
<td>1-4</td>
<td>101-5</td>
<td>2-6</td>
<td>301-1</td>
<td>202-7</td>
<td>18-1</td>
<td>102-8</td>
<td>444-8</td>
<td></td>
</tr>
<tr>
<td>720</td>
<td>6.79844</td>
<td>1-3</td>
<td>101-4</td>
<td>2-5</td>
<td>301-1</td>
<td>202-6</td>
<td>18-1</td>
<td>102-7</td>
<td>444-8</td>
<td></td>
</tr>
<tr>
<td>740</td>
<td>6.85025</td>
<td>1-4</td>
<td>101-5</td>
<td>2-6</td>
<td>301-1</td>
<td>202-7</td>
<td>18-1</td>
<td>102-8</td>
<td>444-8</td>
<td></td>
</tr>
<tr>
<td>760</td>
<td>6.89968</td>
<td>1-3</td>
<td>101-4</td>
<td>2-5</td>
<td>301-1</td>
<td>202-6</td>
<td>18-1</td>
<td>102-7</td>
<td>444-8</td>
<td></td>
</tr>
<tr>
<td>780</td>
<td>6.98755</td>
<td>1-3</td>
<td>101-4</td>
<td>2-5</td>
<td>301-1</td>
<td>202-6</td>
<td>18-1</td>
<td>102-7</td>
<td>444-8</td>
<td></td>
</tr>
<tr>
<td>800</td>
<td>7.18240</td>
<td>1-2</td>
<td>101-3</td>
<td>2-4</td>
<td>301-1</td>
<td>202-5</td>
<td>18-1</td>
<td>102-6</td>
<td>444-1</td>
<td></td>
</tr>
</tbody>
</table>

**e-points/minute = 3624.46836348410**
Fit_OTF Example (5)

Overall error checks:

<table>
<thead>
<tr>
<th>mt</th>
<th>max-err</th>
<th>for</th>
<th>e</th>
<th>eV</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.100%</td>
<td>e</td>
<td>2883.54</td>
<td>eV,</td>
<td>303.6 K</td>
</tr>
<tr>
<td>101</td>
<td>0.100%</td>
<td>e</td>
<td>5967.96</td>
<td>eV,</td>
<td>313.6 K</td>
</tr>
<tr>
<td>2</td>
<td>0.097%</td>
<td>e</td>
<td>20.2401</td>
<td>eV,</td>
<td>313.6 K</td>
</tr>
<tr>
<td>301</td>
<td>0.079%</td>
<td>e</td>
<td>7089.62</td>
<td>eV,</td>
<td>313.6 K</td>
</tr>
<tr>
<td>202</td>
<td>0.026%</td>
<td>e</td>
<td>2664.37</td>
<td>eV,</td>
<td>313.6 K</td>
</tr>
<tr>
<td>18</td>
<td>0.002%</td>
<td>e</td>
<td>723.161</td>
<td>eV,</td>
<td>333.6 K</td>
</tr>
<tr>
<td>102</td>
<td>0.005%</td>
<td>e</td>
<td>4264.86</td>
<td>eV,</td>
<td>333.6 K</td>
</tr>
<tr>
<td>444</td>
<td>0.000%</td>
<td>e</td>
<td>20.6344</td>
<td>eV,</td>
<td>333.6 K</td>
</tr>
</tbody>
</table>

Overall maximum error = 0.100%

Number of energies with err > 0.10% = 0

nctot_max = 22930008
nctot = 11074134
Monte Carlo Depletion Tutorial

- Overview
- Timesteps
- Geometry & Depletion
- Materials & Nuclide Setup
- Cross-section Treatment
- Criticality & Depletion
- Concerns - Accuracy
- Error Propagation

Introduction

• There are now many Monte Carlo depletion systems
  • MONTEBURNS - MCNP + ORIGEN
  • MCODE - MCNP + ORIGEN
  • MCOR - MCNP + ORIGEN
  • MCNPX - MCNPX with built-in CINDER90
  • MCNP-ACAB - MCNP + ACAB
  • ALEPH - MCNP + ORIGEN
  • BGCore - MCNP + SARAF
  • OCTOPUS - MCNP + ORIGEN or FISPACT
  • SCALE - KENO + ORIGEN
  • SERPENT, PSG - standalone, or with ABURN
  • MVP-BURN
  • McCARD
  • MCB
  • MC21, RCP, RACER

• This tutorial provides an overview of Monte Carlo depletion, to help researchers & code users interpret the details & differences in the different MC depletion codes
Monte Carlo Techniques for Nuclear Systems – Theory Lectures

Introduction

• Monte Carlo depletion papers at the PHYSOR-2008 conference

Christos Trakas, François Thibout, Sebastien Thareau, Bernard Verboomen, Gert Van den Eynde, "Benchmark of ALEPH and Monteburns on French post-irradiation experiments"

Hyung Jin Shim, Ho Jin Park, Han Gyu Joo, Yeong-il Kim, Chang Hyo Kim "Uncertainty Propagation in Monte Carlo Depletion Analysis"

Emil Fridman, Eugene Shwageraus, Alex Galperin, "Implementation of multi-group cross-section methodology in BGCore MC-depletion code"

Michael Fensin, John Hendricks, Samim Anghaie, "MCNPX 2.6 depletion method enhancements and testing"
Introduction

- Monte Carlo depletion calculations - basic idea

1. Monte Carlo calculation at a fixed time, $t_0$
   - All geometry, number densities, temperatures, cross-sections must be constant
   - Keff eigenvalue calculation, normalized to required power level
   - Determine absorption rates, fission rates, fluxes for all depletable regions

2. Depletion calculation for $\Delta t = t_1 - t_0$
   - Using number densities, absorption rates, fission rates, fluxes from (1), determine new number densities at time $t_1$
   - Must account for fission product & actinide buildup/burnout
   - May assume constant flux over $\Delta t$, or constant power

Repeat (1) & (2) for each time step

Sounds straightforward, but there are many, many subtleties & complications
Definitions

\[ N_k = N(t_k) \]

= vector of all the number densities for each isotope of every region in the problem at time \( t_k \)

\[ \varphi_k = \varphi(t_k) = \varphi(N_k) \]

= Monte Carlo Keff calculation of fluxes \( \varphi_k \), absorption rates \( A_k \), fission rates \( F_k \) for all isotopes in all regions of problem at time \( t_k \), normalized to a specified reactor power level

\[ B_k = B(t_k, \Delta t, N_k, \varphi, A, F, \lambda) \]

= burnup calculation from time \( t_k \) to \( t_k+\Delta t \), using \( N_k, \varphi_k, A_k, F_k, \lambda_k \)

Solve 1 region at a time, using \( \varphi, A, F \) for the region from MC

**ORIGEN:** \( N_{k+1} = \exp\{-D_k \Delta t \} N_k \), where \( D_k \) is a matrix of \( A, F, \lambda \) for each isotope in region at time \( t_k \)

**CINDER:** Coupled linear chains of ODE's involving \( A, F, \lambda \) for each isotope in region at time \( t_k \)
Simple MC Depletion

- During a timestep $\Delta t$, if fluxes are constant, then $N$, $A$, $F$ change during the step.

- Need very small $\Delta t$ to get accurate results.
**Predictor-Corrector Scheme for MC Depletion**

\[ \phi_{0,C} = (\phi_0 + \phi_{1,P})/2 \quad \phi_{1,C} = (\phi_1 + \phi_{2,P})/2 \]

- **Predictor:** MC at start, deplete to end-of-step, MC at end-of-step
- **Corrector:** deplete again, using average beginning- & end- flux

- Better accuracy, can use much longer time steps
- 2 MC's & 2 depletions per timestep
- Other prescriptions could be used for corrector flux, \( \varphi_{J,C} \) (eg, linear, …)
- Could iterate until predictor-corrector N's are close

Note: For some depletion systems, computer time is reduced by \( \sim 50\% \) by assuming that \( \phi_J \approx \phi_{J,P} \)
Geometry & Depletion

- Must choose geometry regions fine enough to represent spatial detail need for accurate depletion
  - MC fluxes, absorption, fission are tallied for a region (uniform)
  - Material nuclides within a region are depleted uniformly

- Example - CASMO regions for a fuel assembly

- Most MC depletion codes can't handle this level of detail (yet) for the entire reactor
- If the depletion regions are too large, errors will be introduced
Materials & Nuclide Setup

**Material compositions**
- At BOL, fission products & actinides are not present
- Later timesteps must include them
- Generally, must specify trace amounts of all FPs & actinides at BOL
- Some MC depletion codes have built-in options, others don't

**Cross-sections**
- ENDF/B-VII has yield data for 1325 FPs
- ENDF/B-VII has datasets for only 390 nuclides
- Only nuclides with MC cross-sections can be included in the MC simulation
- All others must be treated outside of the MC
Cross-section Treatment

- **For depletion calculation, just need overall (1-group) absorption & fission in each nuclide**
  - These can be computed directly in the MC, if cross-sections are available

- **For nuclides without MC cross-sections**
  - Can tally multigroup fluxes in each material
  - Outside of the MC - can fold together multigroup MC fluxes & multigroup cross-sections

**Cinder90** has its own multigroup library with 3400 nuclides (63-group) & 1325 FP yields

**ORIGEN2** has 1-group xsecs for 1700 nuclides & 850 FP yields

**ORIGEN-S** has 1-group xsecs for 1946 nuclides & 1119 FP yields

---

**List of all nuclides**
- U235
- U238
- O
- ...
- Xe135
- Sm149
- ...

**Monte Carlo calculation**

**Nuclides with MC xsecs**

**Nuclides without MC xsecs**

**Collapse N*Φ**

**Multigroup Xsec library**

**MG φ's**

1-grp A, F, NF
- U235
- U238
- ...

---

**ORIGEN2** has 1-group xsecs for 1700 nuclides & 850 FP yields

**ORIGEN-S** has 1-group xsecs for 1946 nuclides & 1119 FP yields
Criticality & Depletion

- Depletion should be performed with a flux distribution corresponding to a critical system
  - Real reactors are critical & deplete with a flux distribution corresponding to $K_{\text{eff}}=1$
  - If $K_{\text{eff}} \neq 1$ in the Monte Carlo, subsequent depletion would use the wrong fluxes
  - Lattice physics codes (e.g., CASMO) perform a buckling search so that $K_{\text{eff}}=1$, & the depletion is performed with the critical fluxes
  - Not clear what to do for MC depletion

- Choices
  - Deplete anyway.
    - For comparisons, **turn off buckling search** in lattice codes for consistency.
      (wrong, but consistent)
  - For portions of the reactor (e.g., assemblies, unit cells), use albedo boundary conditions to get the correct leakage (in/out, energy-dependent) so that $K_{\text{eff}}=1$
    - Some MC codes don't allow albedo BCs (e.g., MCNP)
    - Getting the albedo BCs is a difficult computational problem

- This is an area that needs ideas & work ….
Concerns - Accuracy

• **Timesteps**
  - Should have short 1st timestep (~1 day), to allow Xe135 to build up to equilibrium
  - Should have short 2nd timestep (~4-5 days), to allow Sm149 to build up to equilibrium
  - Some codes avoid the 2 short steps by automatically handling equilibrium Xe & Sm

  – If timesteps are too long, results will not be accurate
    - Ideally, should run entire depletion lifestudy several times, reducing the timestep sizes until results show convergence
    - This is rarely done.
    - Adequate timestep sizes could be investigated using CASMO/SIMULATE or other codes, rather than with Monte Carlo

• **Geometry & depletion regions**
  - MC materials & tallies are constant within a region
  - Must subdivide depletable regions enough so that step-wise approximation to materials & fluxes is acceptable
  - May require 4-10 regions per fuel pin, or 10-40 regions per poison pin, rather than just 1
  - If the geometry of depletable materials is too coarse, results will not be accurate
Concerns - Accuracy

- **Fission products**
  - Need ~300 FPs in Monte Carlo
  - If that many FPs cannot be used, should consider some sort of lumped fission product approach for the "missing" FPs
    - Could assume residual FP xsecs have simple behavior (eg, 1/v in thermal range & constant in fast range) and lump them into 1, 2, or more lumps for the MC
    - Could use a multigroup background FP library, typically generated with a lattice physics code (eg, CASMO)
  - ..... 

- **Normalization**
  - Need to normalize the MC calculations to the correct power level
  - See other parts of this workshop regarding normalization
  - **Difficulties**
    - Straight neutron MC doesn't account for gamma transport & heating; must assume local fission energy deposition
    - MCNP only includes *prompt* energy from fission in Q values; need corrections
    - Should normalize total (prompt) fission energy from MC to total (prompt) fission energy of real problem

(note: MCNP manual suggests normalizing neutron source rate, rather than the resulting fission rate)
Concerns - Accuracy

• **Regardless of timesteps, geometry, & fission products**
  – Because of the many materials, nuclides per material, & tallies, the MC part of MC depletion runs much longer than normal, sometimes ~10x longer
  – While it is tempting to compensate by running fewer cycles & fewer neutrons/cycle in the MC, BEWARE:
    • Must discard enough initial cycles of each MC calculation to assure fission source distribution has converged before tallies start
    • Must run sufficient cycles after convergence to achieve acceptable statistics
    • Must run enough neutrons/cycle to assure that phase-space is reasonably covered by enough neutrons
Error Propagation

• Uncertainties in input for MC calcs:
  – Cross-sections (all calculations)
  – Number densities (depletion calculations)

• How do uncertainties in input affect results & std-dev's ?

• Three basic approaches:
  – Brute force - sample input params, run calc.; repeat many times
  – Sensitivity/Uncertainty analysis - needs adjoints
  – Perturbation theory approach

• Outstanding paper on error propagation in MC depletion:
Error Propagation

• From paper by Garcia-Herranz, et al.

To compare the impact of the statistical errors in the calculated flux with respect to the cross uncertainties, a simplified problem is considered, taking a constant neutron flux level and spectrum. It is shown that, provided that the flux statistical deviations in the Monte Carlo transport calculation do not exceed a given value, the effect of the flux errors in the calculated isotopic inventory are negligible (even at very high burn-up) compared to the effect of the large cross-section uncertainties available at present in the data files.

• My experience --
  – If you run many instances of an entire MC depletion lifestudy, the general trajectories of $K_{eff}$ & number densities are the same, with superimposed noise
  – Overall results & trajectories are not sensitive to the fluctuations in number densities - if something is too low in one step, it will recover in the next
  – Never observed any kind of nonlinear behavior
HTGR Modeling

HTGR Modeling with MCNP5
MCNP5 Stochastic Geometry
HTGR Physics & Modeling
Outline

• Introduction

• HTGR Modeling with MCNP5
  – MCNP Geometry - Universes, Lattices
  – HTGR Models - Fuel Kernels, Compacts, Pebbles, Core

• MCNP5 Stochastic Geometry
  – Random Translations for Stochastic Universes
  – Discussion & Limitations

• HTGR Physics & Modeling
  – Fuel Kernel Modeling
  – Stochastic Effects
  – Compact or Pebble Modeling
  – Full-core Calculations

• Conclusions
Introduction

• **High Temperature Gas-Cooled Reactor**
  – One of the Next Generation Reactors ("Gen-IV")
  – Coolant temperatures above 900 C, fuel temperatures above 1250 C
    • Higher energy conversion efficiency
    • Thermochemical hydrogen production
  – Fuel kernels with several layers of coatings
    • Contain fission products
    • Safety aspects ...

• **Two types of Reactor Design**
  – Prismatic fuel type reactor design, with fuel "compacts"
  – Pebble bed fuel type reactor design
  – Double Heterogeneity problem
    • Fuel kernels inside fuel compacts or pebbles in the core
    • Fuel kernels randomly located within fuel elements
    • Challenging computational problem

• **Monte Carlo codes can faithfully model HTGRs**
  – Full 3D geometry
  – Multiple levels of geometry, including embedded lattices
  – Random geometry ?????
TRISO Coated Fuel Particles:
- Lots of cladding - extremely strong
- Little fuel - fully encapsulated

Each fuel particle forms a separate pressure containment vessel for the fuel kernel

Fuel concept is same for block or pebble bed

(From INL)
TRISO Coated Fuel Particles - Burnup

Fresh TRISO Fuel

Very High Burnup

TRISO Fuel
- Fission product gases trapped within the layers of coatings
- Coatings remain intact, even with very high burnup

(From General Atomics)
HTGR Modeling With MCNP5
Embedded Geometry - Universes

- In most real-world applications, there is a need for modeling detailed geometry with many repeating units.

- All production Monte Carlo codes provide capabilities for multiple levels of nested geometry:
  - Called universes & lattices in MCNP
  - A collection of cells may be grouped into a universe
  - Universe may be embedded in another cell, with the universe 'clipped' by the cell boundaries

Fig. 2. A cross-section of the GT-AHR homogeneous core with hexagonal structure (HTR1).

Fig. 3. Fragments of single-heterogeneous GT-MHR (HTR2): (A) an active core structure: three rings of hexagonal fuel columns; (B) magnified view of a separate fuel assembly. Fuel compacts are presented in small grey circles, burnable poison compacts in light grey. Bigger diameter holes stand for He channels, while the rest material represents the graphite matrix.

Fig. 4. Fragments of double-heterogeneous GT-MHR (HTR3): (A) fuel element (compact) cross section with coated fuel particles; (B) magnified view of coated fuel particles. Spherical kernels of $\text{PuO}_2$, are surrounded by protective coatings made of $\text{PyC}_{\text{outer}}$, $\text{PyC}$ I, $\text{SiC}$ and $\text{PyC}$ II layers correspondingly. The same structure is valid for particles containing burnable poison—natural $\text{Er}_2\text{O}_3$. 
Pebble Bed HTGR

Very High Temperature Gas-cooled Reactor
MCNP5
Stochastic Geometry
MCNP Models for HTGRs

• Existing MCNP geometry can handle:
  – 3D description of core
  – Fuel compacts or lattice of pebbles
    • Typically, hexagonal lattice with close-packing of spherical pebbles
    • Proteus experiments: ~ 5,000 fuel pebbles
      ~ 2,500 moderator pebbles
  – Lattice of fuel kernels within compact or pebble
    • Typically, cubic lattice with kernel at center of lattice element
    • Proteus experiments: ~ 10,000 fuel kernels per pebble
      ~ 50 M fuel kernels, total
  – Could introduce random variations in locations of a few thousand cells in MCNP input, but not a few million.
  – See papers by: Difilipo, Plukiene et al, Ji-Conlin-Martin-Lee, etc.
MCNP5 Stochastic Geometry

• When a neutron enters a new lattice element, a transformation is made to the neutron's position & direction to the local coordinates of the universe embedded in that lattice element. [standard MCNP]

• Users can flag selected universes as "stochastic" [new]
  – A neutron entering a lattice element containing a stochastic universe undergoes the normal transformations.
  – Then, additional random translations are made:
    \[
    x \leftarrow x + (2\xi_1 - 1) \cdot \delta_x \\
y \leftarrow y + (2\xi_2 - 1) \cdot \delta_y \\
z \leftarrow z + (2\xi_3 - 1) \cdot \delta_z
    \]
  – Then, tracking proceeds normally, with the universe coordinates fixed until the neutron exits that lattice element
MCNP5 Stochastic Geometry

• Neutron on lattice edge, about to enter embedded universe

• Embedded universe,
  before random translation  after random translation

• Track normally, until neutron exits the lattice element
MCNP5 Stochastic Geometry

• On-the-fly random translations of embedded universes in lattice
  – Does not require any extra memory storage
  – Very little extra computing cost - only 3 random numbers for each entry into a stochastic universe

• For K-effective calculations (KCODE problems)
  – If fission occurred within fuel kernel, should have source site in next cycle be at same position within fuel kernel
  – Need to save \( \delta_x, \delta_y, \delta_z \) along with neutron coordinates in fission bank
  – On source for next cycle, apply \( \delta_x, \delta_y, \delta_z \) after neutron pulled from bank

• To preserve mass exactly, rather than on the average stochastically, must choose \( \delta_x, \delta_y, \delta_z \) so that fuel kernels are not displaced out of a lattice element
MCNP5 Stochastic Geometry - Testing

• MCNP5 stochastic geometry
  Fuel kernels displaced randomly on-the-fly within a lattice element each time that neutron enters

• RSA placement of fuel kernels
  Fuel kernels placed randomly in job input, using Random Sequential Addition
  Standard MCNP5 - geometry is fixed for entire calculation
  (Does not use stochastic geometry)
## MCNP5 Stochastic Geometry - Testing

### MCNP5 Results for Infinite Lattices of Fuel Kernels

<table>
<thead>
<tr>
<th>Method</th>
<th>K-effective</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed lattice with centered spheres</td>
<td>1.1531 ± 0.0004</td>
</tr>
<tr>
<td>Fixed lattice with randomly located spheres (&quot;on the fly&quot;)</td>
<td>1.1515 ± 0.0004</td>
</tr>
<tr>
<td>Multiple (25) realizations of lattice with randomly located spheres</td>
<td>1.1513 ± 0.0004</td>
</tr>
<tr>
<td>Multiple (25) realizations of randomly packed (RSA) fuel &quot;box&quot;</td>
<td>1.1510 ± 0.0003</td>
</tr>
</tbody>
</table>

- Small but significant effect from stochastic geometry, -.15% Δk
- New MCNP5 stochastic geometry matches multiple realizations
- New MCNP5 stochastic geometry matches true random (RSA)
HTGR Physics & Modeling
HTGR Modeling with MCNP5

• Modeling HTGR core & assembly geometry is relatively straightforward using standard MCNP5 features

• We have examined in detail some of the fine-points of modeling
  – Fuel kernels embedded in the graphite matrix
    • Lattice arrangements
    • Stochastic effects
  – Compacts or pebbles
    • Embedding a lattice of fuel kernels

• We have run full-core HTGR calculations to examine the effects of detailed modeling techniques
HTGR Modeling with MCNP5
Fuel Kernels

- TRISO fuel kernels in graphite matrix
  - Fuel kernel geometry & composition taken from the NGNP Point Design (MacDonald et al. 2003)

<table>
<thead>
<tr>
<th>Region #</th>
<th>Name</th>
<th>Outer radius ($\mu$)</th>
<th>Composition</th>
<th>Density (g/cc)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Uranium oxycarbide</td>
<td>175</td>
<td>UCO ($\text{UC}^{0.5}\text{O}^{1.5}$)</td>
<td>10.5</td>
</tr>
<tr>
<td>2</td>
<td>Porous carbon buffer</td>
<td>275</td>
<td>C</td>
<td>1.0</td>
</tr>
<tr>
<td>3</td>
<td>Inner pyrolytic carbon</td>
<td>315</td>
<td>C</td>
<td>1.9</td>
</tr>
<tr>
<td>4</td>
<td>Silicon carbide</td>
<td>350</td>
<td>SiC</td>
<td>3.2</td>
</tr>
<tr>
<td>5</td>
<td>Outer pyrolytic carbon</td>
<td>390</td>
<td>C</td>
<td>1.9</td>
</tr>
</tbody>
</table>

Fuel kernel packing fraction = .289
Fuel Kernel Modeling - Lattices

- **Simple cubic**
- **Body Centered Cubic**
- **Face Centered Cubic**

**Slice through base plane**
- Simple cubic: 0.09501 cm
- Body Centered Cubic: 0.119705 cm
- Face Centered Cubic: 0.150819 cm

**Slice through mid plane**
- Simple cubic: 0.09501 cm
- Body Centered Cubic: 0.119705 cm
- Face Centered Cubic: 0.150819 cm
Fuel Kernel Modeling - Results

- MCNP5 calculations for infinite geometry, fuel kernels in graphite matrix

<table>
<thead>
<tr>
<th>Configuration</th>
<th>K-effective ± 1σ</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homogenized matrix &amp; fuel kernel</td>
<td>1.0996 ± .0008</td>
</tr>
<tr>
<td>Simple cubic lattice</td>
<td>1.1531 ± .0004</td>
</tr>
<tr>
<td>pitch = 0.09501 cm</td>
<td></td>
</tr>
<tr>
<td>Body centered cubic lattice</td>
<td>1.1534 ± .0003</td>
</tr>
<tr>
<td>pitch = 0.119705 cm</td>
<td></td>
</tr>
<tr>
<td>Face centered cubic lattice</td>
<td>1.1526 ± .0003</td>
</tr>
<tr>
<td>pitch = 0.150819 cm</td>
<td></td>
</tr>
</tbody>
</table>

- Large errors for homogenized model
- Essentially same results for all lattice models
Fuel Kernels - Radial Flux Profiles

Six-Region Heterogeneous

Two-Region Heterogeneous

Reflecting b.c. on all sides of cubes
Radial Neutron Flux Profile in 6-region Kernel
Radial Neutron Flux in Resonance Range

- **Resonance Energy Range:** 6.57eV - 6.77eV
### MCNP5 Simulations of Fuel Kernels

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Kernel location</th>
<th>keff</th>
<th>std dev</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homogeneous microsphere cell</td>
<td>- - -</td>
<td>1.0995</td>
<td>.0004</td>
</tr>
<tr>
<td>Two-region heterogeneous microsphere cell</td>
<td>Centered</td>
<td>1.1535</td>
<td>.0004</td>
</tr>
<tr>
<td>Six-region heterogeneous microsphere cell</td>
<td>Centered</td>
<td>1.1533</td>
<td>.0003</td>
</tr>
</tbody>
</table>

- Essential to model the microsphere heterogeneity
- Homogenizing the coatings into the matrix does not introduce any significant errors, & can reduce model complexity
- Adequate to explicitly represent just the UCO spheres
Compact or Pebble Modeling

• If an infinite lattice of fuel kernels is embedded in a compact or pebble, clipping by the enclosing cylinder or sphere results in fragments of kernels. This is not correct modeling - does not preserve mass.

• A finite lattice of fuel kernels should be used, so that there are no fragments. Vertical pitch of the lattice should be adjusted to preserve total mass or packing fraction.

Infinite lattice of kernels, truncated by cylinder

Finite lattice of kernels, no intersections with cylinder
Full-core Calculations

- Heterogeneous core, with simple cubic infinite lattice of kernels (with partial kernels at cylinder boundary)
- Heterogeneous core, with simple cubic finite lattice of kernels (with no partial kernels at cylinder boundary)

<table>
<thead>
<tr>
<th>Case</th>
<th>Configuration</th>
<th>K-effective ± 1σ</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>Heterogeneous core, with simple cubic fixed infinite lattice (with partial kernels at cylinder boundary)</td>
<td>1.0948 ± .0002</td>
</tr>
<tr>
<td>F</td>
<td>Heterogeneous core, with simple cubic fixed finite lattice (with no partial kernels at cylinder boundary)</td>
<td>1.0974 ± .0002</td>
</tr>
</tbody>
</table>

→ Correct modeling of compacts, using a finite lattice to avoid partial kernels at boundary, is important
Full-core Calculations

- Heterogeneous core, simple cubic infinite lattice
- Heterogeneous core, simple cubic infinite lattice, with new MCNP5 stochastic geometry

<table>
<thead>
<tr>
<th>Case</th>
<th>Configuration</th>
<th>K-effective ± 1σ</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>Heterogeneous core, with simple cubic fixed finite lattice (with no partial kernels at cylinder boundary)</td>
<td>1.0974 ± .0002</td>
</tr>
<tr>
<td>G</td>
<td>New MCNP5 stochastic geometry, on-the-fly random location of kernels within simple cubic finite lattice elements (with no partial kernels at cylinder boundary)</td>
<td>1.0968 ± .0002</td>
</tr>
</tbody>
</table>

→ Stochastic effects are small for full-core calculations, may or may not be important
Conclusions

• The new stochastic geometry treatment for MCNP5 provides an accurate and effective means of modeling the particle heterogeneity in TRISO particle fuel

• Homogenizing the fuel or core introduces very large errors (~8%)

• Double heterogeneity important only in resonance energy range
  – Increased resonance self-shielding due to "particle shadowing".
  – Effect is more pronounced for fuel kernel or compact calculations, and decreases as one goes to full core due to increased moderation hence decreased effect of resonance absorption

• The effect of choosing either centered fixed spheres or randomly located spheres is small

• Can introduce significant errors by using an infinite lattice truncated by compact or pebble (clipped), rather than a finite lattice

• Homogenizing the fuel coatings into the graphite matrix is a reasonable approximation
References - HTGR Models & Stochastic Geometry

Coupled Monte Carlo & Thermal-Hydraulics Calculations

Temperature Dependence

• **Temperature effects on Monte Carlo**

• **Accounting for temperature effects in MCNP**
  – Generate NJOY libraries during NTH iterations
  – Generate NJOY libraries prior to the NTH iterations
  – Pseudo-materials approach

• **Applications**
  – Explicit coupling of MCNP5 and Star-CD for LWR configurations
  – Explicit coupling of MCNP5 and RELAP-Athena for full-core VHTR simulation
Acknowledgements

• Results for the STAR-CD/MCNP coupling were provided by Professor Tom Downar at the University of Michigan (UM). Also contributing to this work were Drs. Yunlin Xu and Volkan Seker, who are now at the UM.

• The following faculty and students at the UM contributed to the work on pseudo-materials and RELAP5/MCNP coupling:
  – John C. Lee, Professor
  – Wei Ji, former PhD candidate (now Professor at RPI)
  – Gokhan Yesilyurt, PhD candidate
  – Jeremy Conlin, PhD candidate
  – Kaushik Banerjee, PhD candidate
  – Etienne de Villèle, former exchange student at UM
Temperature Dependence

- **Temperature effects on Monte Carlo calculations**
  - Thermal expansion: changes in dimensions and densities
  - Cross-section data:
    - Need to Doppler broaden cross sections including resolved and unresolved resonances (probability tables)
    - Need to change \( S(\alpha,\beta) \) thermal scattering kernel

- **For most Monte Carlo codes, temperature effects must be handled explicitly by the code users**
  - Input changes are required to account for dimension & density changes
  - Must use cross-section data generated at the correct problem temperatures

- **MCNP**
  - Automatically Doppler broadens elastic scattering cross-sections
  - Does NOT adjust:
    - resolved resonance data
    - unresolved resonance data
    - thermal scattering kernels
Accounting for Temperature Effects in MCNP

Approaches to account for temperature changes:

A. Generate explicit temperature – dependent cross section libraries (NJOY)

B. Modify existing libraries (MAKXSF)

C. Approximate approach using pseudo-materials
A. Generate explicit temp-dependent datasets (NJOY)

- Use NJOY (or similar cross-section processing code) to generate nuclear cross-section datasets
  - Must generate a separate dataset for each nuclide at each region temperature
  - NJOY routines take care of Doppler broadening (resolved & unresolved) & thermal scattering kernels

- Two approaches:
  - Iterative NJOY updates: run NJOY during the neutronic-thermal/hydraulic (NTH) iterations for each temperature needed for the current T/H calculation.
  - Pregenerated NJOY libraries: run NJOY beforehand for a range of temperatures that adequately covers the temperatures expected for the NTH calculation, e.g., every 5K from 300K to 1200K for fuel nuclides.
Computational results (Downar, Monterey 2007)

- **Iterative NJOY updating** is very time-consuming
  - 95 s to prepare a dataset for U-235 on 3 Ghz Pentium P4.
  - Not practical for realistic reactor applications.

- **Pregenerated NJOY libraries** is a reasonable approach
  - Used to couple STAR-CD and MCNP
  - NJOY was run at 5K temperature increments over the temperature range. (Temperature increments of 1-2 K cause memory problems with MCNP.)
  - A Perl script was used to manage the NTH iterations.
  - The coupled code system (McStar) was applied to a 1/8 pin cell and a 3x3 array of pin cells.
  - Good agreement with DeCart/STAR-CD results
McSTAR

- Monte Carlo Neutron Transport: MCNP5
- Computational Fluid Dynamics: STAR-CD
- Cross Sections: NJOY
**Results:** coupled STAR-CD and MCNP results

1/8 pin cell

- **MCNP MODEL**
- **STAR-CD MODEL**

Keff agrees within 52 pcm with DeCart/STAR-CD

3x3 array of pin cells

- **MCNP**
- **STAR-CD**

Keff agrees within 66 pcm with DeCart/STAR-CD
Preliminary conclusions for McStar

• The preliminary results for two simple PWR test problems demonstrate the feasibility of coupling Monte Carlo to CFD for a potential audit tool.

• Validation of the cross section update methodology was performed to assess the accuracy of the 5K increment tables for these problems.

• McSTAR is now being applied to advanced BWR fuel assemblies with strong axial heterogeneities to verify the accuracy of the 2D/1D solution methods in DeCART
B. Modify existing MCNP library (MAKXSF)

- New version of MAKXSF

- Subset of NJOY routines, easy to use, part of MCNP5/1.51 distribution

- For ACE datasets (for MCNP), makxsf performs:
  - Doppler broadening of resolved resonance data (explicit profiles)
  - Interpolation of unresolved resonance data (probability tables) between ACE datasets at 2 different temperatures
  - Interpolation of thermal scattering kernels ($S(\alpha,\beta)$ data) between ACE datasets at 2 different temperatures

- For now, makxsf is run external to MCNP

- Long-term plan: put the makxsf routines in-line with the MCNP coding
C. Approximate method: pseudo-materials

• "Pseudo-materials" for temperature dependence

  – Equivalent to "stochastic interpolation"

  – To approximate the cross-sections for nuclide X at temperature T, use a weighted combination of nuclide X at lower temperature $T_1$ and higher temperature $T_2$

  – This weighted combination is input as an MCNP5 material with volume fractions given by the weights

\[
W_2 = \frac{\sqrt{T} - \sqrt{T_1}}{\sqrt{T_2} - \sqrt{T_1}}, \quad W_1 = 1 - W_2
\]

\[
\Sigma_i = \Sigma(T_i)
\]

\[
\Sigma(T) = W_1 \cdot \Sigma_1 + W_2 \cdot \Sigma_2
\]
Example: $^{235}\text{U}$ at 500 K

Existing datasets for MCNP:
(1) dataset for $^{235}\text{U}$ at 293.6 K: 92235.66c
(2) dataset for $^{235}\text{U}$ at 3000.1 K: 92235.65c

Weight the datasets using $T^{1/2}$ interpolation

$$w_2 = \frac{\sqrt{500} - \sqrt{293.6}}{\sqrt{3000.1} - \sqrt{293.6}} = .1389, \quad w_1 = .8611$$

In the MCNP input:

\begin{align*}
m1 & \quad 92235.66c & .8611 & \quad 92235.65c & .1389
\end{align*}
Application: VHTR geometry*

**Normal:** explicit NJOY at given temperature

**Pseudo:** interpolate between closest NJOY libraries (every 100K)

Application – LWR configuration

Results for LWR configuration with NJOY cross sections at 325K compared to pseudo-material approach using cross sections at 300K and 350K. Most deviations within statistics. (Downar, 2007 Monterey M&C)

<table>
<thead>
<tr>
<th></th>
<th>325 K (NJOY)</th>
<th>325 K Interpolated</th>
<th>Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k_{\text{eff}}$</td>
<td>1.40974 (± 0.00043)</td>
<td>1.41008 (± 0.00044)</td>
<td>34 pcm</td>
</tr>
<tr>
<td>$\phi$ in Fuel</td>
<td>1.37933 (± 0.0003)</td>
<td>1.37929 (± 0.0003)</td>
<td>0.00003</td>
</tr>
<tr>
<td>$\sigma_{aF\phi}$</td>
<td>3.67362e-03 (± 0.0006)</td>
<td>3.67648E-03 (± 0.0006)</td>
<td>0.0008</td>
</tr>
<tr>
<td>$\sigma_{f\phi}$</td>
<td>5.62964e-03 (± 0.0007)</td>
<td>5.63817E-03 (± 0.0007)</td>
<td>0.0010</td>
</tr>
<tr>
<td>$\gamma\sigma_{f\phi}$</td>
<td>1.38341e-02 (± 0.0007)</td>
<td>1.38548e-02 (± 0.0007)</td>
<td>0.0010</td>
</tr>
</tbody>
</table>
Application – full core VHTR with T/H feedback

• MCNP5 code was coupled with the RELAP5-3D/ATHENA code to analyze full core VHTR with temperature feedback (*pseudo-materials*) including explicit TRISO fuel

• Utilized a master process supervising independent computing platforms to automate coupled Nuclear-Thermal-Hydraulic (NTH) calculations.

• Axial power fractions determined for 10 axial zones for each of three rings by MCNP5 are input to RELAP5 to determine assembly-average temperature distributions.

• Updated RELAP5 temperature distributions are used for the next MCNP simulation to obtain updated power fractions. MCNP5 and RELAP5 iterations were performed in a cyclic fashion until convergence in temperature and power distributions were obtained.

• Totally automated with a Perl script that reads output files and generates input files.
Description of VHTR reactor

MCNP5 input decks were set up to represent the VHTR full core with homogeneous and heterogeneous fuel assemblies.

Each ring has 10 axial fuel segments and 30, 36, and 36 fuel assemblies, respectively, for the inner, middle, and outer core rings.

- Active Core Height: 7.93 m (10 blocks)
- Enrichment: 10.36%
- Natural Boron impurity: 6.9 ppm
- Total Number of Fuel Blocks: 1020
VHTR simulation - RELAP5 Methodology

- For RELAP5-3D/ATHENA calculations, the core was modeled consistent with the MCNP5 setup.

- Each annular region is axially discretized into ten segments and is represented as a cylindrical coolant channel comprising a central coolant hole, surrounded by three inner graphite rings, four fuel rings, and one outer graphite ring.

- An adiabatic boundary condition is imposed at the outer boundary of the coolant channel.
Based on the NGNP target for the helium outlet temperature of 1273 K, together with the inlet temperature of 763 K, helium mass flow rate was determined as 226 kg/s for rated power output of 600 MWt.
VHTR - Cross Platform NTH Architecture

- MCNP5 was run on a Mac G5 Unix cluster in parallel. 10K particles per cycle were used with a total of 140 active cycles for each MCNP5 calculation.
- RELAP5 was run on a remote Windows server.
VHTR - NTH Data Communication

- Data was communicated between MCNP5 and RELAP5 codes in a cyclic fashion until convergence in temperature and power distributions were obtained.

- Online monitoring of the RMSE was used to stop the iteration.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.016574</td>
<td>926.045</td>
<td>0.028488</td>
<td>1038.256</td>
</tr>
<tr>
<td>2</td>
<td>0.026557</td>
<td>1050.207</td>
<td>0.041420</td>
<td>1205.862</td>
</tr>
<tr>
<td>3</td>
<td>0.035137</td>
<td>1175.686</td>
<td>0.047680</td>
<td>1333.691</td>
</tr>
<tr>
<td>4</td>
<td>0.041325</td>
<td>1293.021</td>
<td>0.047675</td>
<td>1419.931</td>
</tr>
<tr>
<td>5</td>
<td>0.045263</td>
<td>1401.488</td>
<td>0.043716</td>
<td>1475.861</td>
</tr>
<tr>
<td>6</td>
<td>0.045369</td>
<td>1486.089</td>
<td>0.038116</td>
<td>1513.388</td>
</tr>
<tr>
<td>7</td>
<td>0.041935</td>
<td>1543.885</td>
<td>0.031252</td>
<td>1531.816</td>
</tr>
<tr>
<td>8</td>
<td>0.035221</td>
<td>1570.742</td>
<td>0.023775</td>
<td>1533.103</td>
</tr>
<tr>
<td>9</td>
<td>0.025824</td>
<td>1564.520</td>
<td>0.016720</td>
<td>1523.647</td>
</tr>
<tr>
<td>10</td>
<td>0.015604</td>
<td>1533.804</td>
<td>0.010089</td>
<td>1503.648</td>
</tr>
</tbody>
</table>
RMS Error in Temperature vs. NTH Iteration

- ** Inner Ring
- ** Middle Ring
- ** Outer Ring

**RMSE**

**Coupled NTH Iteration Number**
Converged Temperature/Power Distributions

Homogeneous Fuel

Temperature (K)

Power Fraction

Axial Height (m)
Converged Temperature/Power Distributions

Heterogeneous Fuel

Temperature (K)

Power Fraction

Axial Height (m)
Homogeneous vs Heterogeneous Fuel
VHTR Temperature Feedback - Conclusions

- A cross-platform computer architecture connecting Mac G5 Unix cluster and a Windows server was successfully developed to automate the coupled NTH calculations for the VHTR core.

- Online monitoring of RMSE shows that it converges rapidly (4-7 iterations)

- The converged power distributions are nearly independent of the double heterogeneity accounted for with MCNP5.

- We are now performing more highly resolved MCNP5 calculations with 100,000 histories per cycle and the effect of the heterogeneities appears to be more pronounced.

- The pseudo-material method works very well but the true test will be the above higher resolution cases.
Pseudo-materials – advantages/disadvantages

• **Advantages**
  – Libraries needed at fewer temperatures (e.g., every 100K)
  – Can interpolate to any temperature bounded by the library
    temperatures
  – No data preprocessing required

• **Disadvantages**
  – Approximate interpolation - *stochastic interpolation is not functional
    interpolation*: one of the two datasets is chosen randomly during the
    random walk
  – Finite error due to interpolation – seems to be small
  – Cannot be used for $S(\alpha,\beta)$ thermal scattering kernels
    • **MCNP limitation**: does not allow mixture of $S(\alpha,\beta)$ materials
    • Need to pick $S(\alpha,\beta)$ dataset at nearest temperature
Fission Energy Deposition

Estimation of the Spatial Distribution of Fission Energy Deposition in a VHTR with (only) MCNP5

How to perform fission energy deposition calculations with standard MCNP tallies with application to a full core VHTR configuration.
Outline

• Acknowledgements

• Motivation and summary

• Fission energy release and deposition

• Capabilities and limitations of MCNP5 fission energy deposition tallies for reactor applications

• Methodology to account for fission energy deposition with MCNP5

• Application to VHTR configurations

• Alternative approach – a simplified methodology
Acknowledgements

• This work is primarily the work of Etienne de Villèle, an exchange student from France who examined the coupling of MCNP5 and RELAP5 for temperature feedback in a VHTR. This work formed part of his MS thesis for CEA/INSTN.

• Other contributors, all from the Department of Nuclear Engineering and Radiological Sciences at the University of Michigan, include:
  – John C. Lee, Professor
  – Gokhan Yesilyurt, PhD Candidate
  – Wei Ji, former PhD Candidate (now Professor at RPI)
Motivation and summary

- **Motivation** – every few months there are conversations on the MCNP Forum regarding how MCNP handles fission energy deposition and how MCNP can be used to estimate the spatial distribution for a realistic reactor configuration.

- This talk is a summary of the process used at the University of Michigan to estimate the fission energy deposition in VHTR configurations. This is one approach that makes use of standard MCNP tallies and seemed to yield acceptable results. Comments or suggestions are welcome.
## Components of energy release in fission

<table>
<thead>
<tr>
<th>Quantity</th>
<th>Value(eV)</th>
<th>Uncertainty</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kinetic energy of the fragments</td>
<td>1.6912E+08</td>
<td>4.9000E+05</td>
</tr>
<tr>
<td>Kinetic energy of the prompt neutrons</td>
<td>4.7900E+06</td>
<td>7.0000E+04</td>
</tr>
<tr>
<td>Kinetic energy of the delayed neutrons</td>
<td>7.4000E+03</td>
<td>1.1100E+03</td>
</tr>
<tr>
<td>Kinetic energy of the prompt gammas</td>
<td>6.9700E+06</td>
<td>5.0000E+05</td>
</tr>
<tr>
<td>Kinetic energy of the delayed gammas</td>
<td>6.3300E+06</td>
<td>5.0000E+04</td>
</tr>
<tr>
<td>Total energy released by delayed betas</td>
<td>6.5000E+06</td>
<td>5.0000E+04</td>
</tr>
<tr>
<td>Energy carried away by the neutrinos</td>
<td>8.7500E+06</td>
<td>7.0000E+04</td>
</tr>
<tr>
<td>Total energy release per fission (sum)</td>
<td>2.0247E+08</td>
<td>1.3000E+05</td>
</tr>
<tr>
<td>Total energy less neutrino energy</td>
<td>1.9372E+08</td>
<td>1.5000E+05</td>
</tr>
</tbody>
</table>

Interpreted ENDF file for U-235e (ENDF/B-VI)

F7 tally includes items in red
Sources of Fission Energy (recoverable)

\[ Q_f = \text{kinetic energy of fission fragments} \]
\[ Q_n = \text{kinetic energy of fission neutrons} \]
\[ Q_\beta = \text{beta decay energy from fission} \]
\[ Q_{\gamma p} = \text{prompt gamma energy from fission} \]
\[ Q_{\gamma d} = \text{delayed gamma energy from fission} \]
\[ Q_{\gamma c} = \text{capture gamma energy from (n,\gamma) reactions} \]

\[ Q = Q_f + Q_n + Q_\beta + Q_{\gamma p} + Q_{\gamma d} + Q_{\gamma c} \]
## Energy Released per Fission (ENDF/B-VI)

<table>
<thead>
<tr>
<th>Particle</th>
<th>Notation</th>
<th>Energy released (MeV)</th>
<th>Deposition site</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fission products</td>
<td>$Q_f$</td>
<td>169.1</td>
<td>Local</td>
</tr>
<tr>
<td>Neutrons</td>
<td>$Q_n$</td>
<td>4.79</td>
<td>Global</td>
</tr>
<tr>
<td>Prompt gammas</td>
<td>$Q_{\gamma p}$</td>
<td>6.97</td>
<td>Global</td>
</tr>
<tr>
<td>Betas</td>
<td>$Q_\beta$</td>
<td>6.5</td>
<td>Local</td>
</tr>
<tr>
<td>Delayed gammas</td>
<td>$Q_{\gamma d}$</td>
<td>6.33</td>
<td>Global</td>
</tr>
<tr>
<td>Capture gammas</td>
<td>$Q_{\gamma c}$</td>
<td>$\sim 6 - 8$</td>
<td>Global</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>$Q$</td>
<td><strong>193.69</strong></td>
<td></td>
</tr>
</tbody>
</table>
Physical Assumptions

• Fission fragments and betas deposit their energy locally

• Prompt and delayed gammas (from fission product decay) deposit their energy globally and must be transported

• Fission neutrons must be transported and heat may be deposited during the neutron trajectory due to:
  – deposition of kinetic energy during moderation
  – emission of gammas as a result of neutron capture
  – energy release due to fission.

• Capture gammas are a distributed source of gammas throughout the reactor (including reflector) and they must be transported.
## MCNP5 Capabilities/Limitations

<table>
<thead>
<tr>
<th>Tally</th>
<th>Accounts for energy deposition due to</th>
</tr>
</thead>
<tbody>
<tr>
<td>F7:n</td>
<td>Fission products, prompt gammas, and neutrons</td>
</tr>
<tr>
<td>F6:n</td>
<td>Fission products and neutrons</td>
</tr>
<tr>
<td>F6:p</td>
<td>Prompt gammas and capture gammas</td>
</tr>
<tr>
<td>F6:np</td>
<td>Fission products, neutrons, prompt gammas, and capture gammas (F6:n + F6:p)</td>
</tr>
</tbody>
</table>

- F7 tally does not account for delayed gammas, betas, or capture gammas.
- No tallies account for betas or delayed gammas which comprise 6-7% of the fission energy release.
- This is not a problem if one assumes all fission energy is locally deposited because the power normalization is arbitrary.
- An accurate prediction of the spatial distribution of fission energy deposition, including neutron and gamma transport effects, should include contributions of the betas and delayed gammas.
Overall Approach to Compute Spatial Deposition

- **Goal:** Compute $H(r)$, where $H(r)dr = \text{amount of energy} / \text{s deposited in } dr \text{ about } r \text{ in a reactor (including reflector) at power } P$, accounting for all sources of fission energy.

- **Use standard F6 / F7 tallies in MCNP5**

- **Use reasonable models for those quantities that are unknown or not treated by MCNP5**
  - Beta energy is deposited locally and can be scaled from the conventional F7 tally.
  - Delayed gamma energy is deposited with the same spatial distribution as the prompt gamma energy.
Overall methodology to estimate $H(r)$

- Run multiple MCNP runs to get all contributions to the overall fission energy deposition.
  
  - **Prompt gammas** ($H_{\gamma p}$): F6:p tally with PIKMT card.
  - **Capture gammas** ($H_{\gamma c}$): F6:p tally with PIKMT card.
  - **Delayed gammas** ($H_{\gamma d}$): Scale $H_{\gamma p}$ by $Q_{\gamma d}/Q_{\gamma p}$.
  - **Fission products + neutrons** ($H_{fn}$): regular F6:n tally.
  - **Betas** ($H_{\beta}$): Scale regular F7:n tally by $Q_{\beta}/Q_{F7}$, where $Q_{F7} = 180.88$ MeV for U-235 fuel.

- Each run, scaled as indicated, yields a spatially dependent contribution to $H(r)$. The total is a simple sum of the individual contributions since they are scaled properly.

$$H(r) = H_{fn} + H_{\gamma p} + H_{\gamma c} + H_{\gamma d} + H_{\beta}$$

- Scale $H(r)$ to get correct total power $P$. 
## Summary of overall methodology

<table>
<thead>
<tr>
<th>Particle</th>
<th>Notation</th>
<th>Energy released (MeV)</th>
<th>MCNP Tally</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fission products</td>
<td>$Q_f$</td>
<td>169.1</td>
<td>F6:n</td>
<td>Normal</td>
</tr>
<tr>
<td>Neutrons</td>
<td>$Q_n$</td>
<td>4.79</td>
<td>F6:n</td>
<td>Normal</td>
</tr>
<tr>
<td>Prompt gammas</td>
<td>$Q_{\gamma p}$</td>
<td>6.97</td>
<td>F6:p</td>
<td>PIKMT</td>
</tr>
<tr>
<td>Betas</td>
<td>$Q_\beta$</td>
<td>6.5</td>
<td>F7:n</td>
<td>Scaled</td>
</tr>
<tr>
<td>Delayed gammas</td>
<td>$Q_{\gamma d}$</td>
<td>6.33</td>
<td>F6:p</td>
<td>Scaled</td>
</tr>
<tr>
<td>Capture gammas</td>
<td>$Q_{\gamma c}$</td>
<td>$\sim 6 - 8$</td>
<td>F6:p</td>
<td>PIKMT</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>$Q$</td>
<td><strong>193.69</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Tallied depositions for VHTR Configurations

<table>
<thead>
<tr>
<th></th>
<th>ENDF/B-VI</th>
<th>TRISO particle</th>
<th>Hom full core</th>
<th>Het full core</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fission products and neutrons</strong></td>
<td>173.89</td>
<td>171.59</td>
<td>173.45</td>
<td>173.26</td>
</tr>
<tr>
<td><strong>Prompt gammas</strong></td>
<td>6.97</td>
<td>6.7</td>
<td>6.71</td>
<td>6.78</td>
</tr>
<tr>
<td><strong>Delayed gammas</strong></td>
<td>6.33</td>
<td>6.33</td>
<td>6.44</td>
<td>6.50</td>
</tr>
<tr>
<td><strong>Betas</strong></td>
<td>6.5</td>
<td>6.5</td>
<td>6.5</td>
<td>6.5</td>
</tr>
<tr>
<td><strong>Subtotal</strong></td>
<td><strong>193.69</strong></td>
<td><strong>191.12</strong></td>
<td><strong>193.1</strong></td>
<td><strong>193.05</strong></td>
</tr>
<tr>
<td><strong>Capture gammas</strong></td>
<td>-</td>
<td>6.96</td>
<td>4.36</td>
<td>3.70</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>198.02</strong></td>
<td><strong>197.46</strong></td>
<td><strong>196.74</strong></td>
<td></td>
</tr>
</tbody>
</table>
Power distribution* for full core VHTR (het fuel)

<table>
<thead>
<tr>
<th>Bottom reflector</th>
<th>Inner reflector</th>
<th>Inner core</th>
<th>Middle core</th>
<th>Outer core</th>
<th>Outer reflector</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.06</td>
<td>2.72</td>
<td>2.92</td>
<td>2.87</td>
<td>0.08</td>
<td>8.65</td>
</tr>
<tr>
<td>2</td>
<td>0.10</td>
<td>4.59</td>
<td>4.89</td>
<td>4.88</td>
<td>0.14</td>
<td>14.59</td>
</tr>
<tr>
<td>3</td>
<td>0.13</td>
<td>6.42</td>
<td>6.85</td>
<td>6.82</td>
<td>0.19</td>
<td>20.41</td>
</tr>
<tr>
<td>4</td>
<td>0.16</td>
<td>7.81</td>
<td>8.11</td>
<td>8.17</td>
<td>0.24</td>
<td>24.49</td>
</tr>
<tr>
<td>5</td>
<td>0.18</td>
<td>8.50</td>
<td>8.97</td>
<td>9.00</td>
<td>0.26</td>
<td>26.91</td>
</tr>
<tr>
<td>6</td>
<td>0.19</td>
<td>8.62</td>
<td>9.10</td>
<td>9.06</td>
<td>0.25</td>
<td>27.22</td>
</tr>
<tr>
<td>7</td>
<td>0.17</td>
<td>7.97</td>
<td>8.41</td>
<td>8.49</td>
<td>0.24</td>
<td>25.29</td>
</tr>
<tr>
<td>8</td>
<td>0.15</td>
<td>6.87</td>
<td>7.33</td>
<td>7.35</td>
<td>0.21</td>
<td>21.92</td>
</tr>
<tr>
<td>9</td>
<td>0.11</td>
<td>5.25</td>
<td>5.58</td>
<td>5.57</td>
<td>0.16</td>
<td>16.67</td>
</tr>
<tr>
<td>10</td>
<td>0.06</td>
<td>3.23</td>
<td>3.43</td>
<td>3.38</td>
<td>0.09</td>
<td>10.19</td>
</tr>
<tr>
<td>Top reflector</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>1.31</td>
<td>61.96</td>
<td>65.56</td>
<td>65.57</td>
<td>1.86</td>
<td>196.60</td>
</tr>
</tbody>
</table>

*Error in simulation resulted in zero top and bottom reflector deposition rates
Alternative approach – a simplified methodology

- Multiple MCNP5 runs, especially with PIKMT cards active, are very time-consuming. Not practical for coupled MCNP5/RELAP5 calculations.

- In principle, MCNP5 could be modified to tally these quantities directly. This is probably a low-priority change since work-arounds can yield acceptable results.

- An alternative approach is based on the observation that the F6:n tally accounts for global transport of neutrons and perhaps the spatial distribution of the neutron tally might approximate reasonably well the spatial distribution of the overall fission energy deposition.
Comparison of F6:n with overall heat deposition*

<table>
<thead>
<tr>
<th>Bottom reflector</th>
<th>Inner reflector</th>
<th>Inner core</th>
<th>Middle core</th>
<th>Outer core</th>
<th>Outer reflector</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.26</td>
<td>0.90</td>
<td>0.89</td>
<td>0.90</td>
<td>0.26</td>
</tr>
<tr>
<td>2</td>
<td>0.26</td>
<td>0.90</td>
<td>0.88</td>
<td>0.90</td>
<td>0.26</td>
</tr>
<tr>
<td>3</td>
<td>0.26</td>
<td>0.90</td>
<td>0.89</td>
<td>0.90</td>
<td>0.27</td>
</tr>
<tr>
<td>4</td>
<td>0.26</td>
<td>0.90</td>
<td>0.88</td>
<td>0.90</td>
<td>0.26</td>
</tr>
<tr>
<td>5</td>
<td>0.26</td>
<td>0.89</td>
<td>0.88</td>
<td>0.90</td>
<td>0.25</td>
</tr>
<tr>
<td>6</td>
<td>0.25</td>
<td>0.89</td>
<td>0.89</td>
<td>0.90</td>
<td>0.26</td>
</tr>
<tr>
<td>7</td>
<td>0.25</td>
<td>0.89</td>
<td>0.88</td>
<td>0.90</td>
<td>0.26</td>
</tr>
<tr>
<td>8</td>
<td>0.26</td>
<td>0.89</td>
<td>0.88</td>
<td>0.90</td>
<td>0.26</td>
</tr>
<tr>
<td>9</td>
<td>0.26</td>
<td>0.90</td>
<td>0.89</td>
<td>0.90</td>
<td>0.27</td>
</tr>
<tr>
<td>10</td>
<td>0.27</td>
<td>0.90</td>
<td>0.89</td>
<td>0.90</td>
<td>0.27</td>
</tr>
<tr>
<td>Top reflector</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Ratio of F6:n tally to benchmark fission energy deposition
Simplified methodology – preliminary thoughts

• The F6:n tally (arbitrary normalization) yields fractional energy depositions in the core regions which are 88-90% of the benchmark fission energy fractions and within 25-27% for the reflector regions.

• Although this ratio may change by ~10% in the reflector, only a few % of the fission energy is deposited in the reflector.

• Implication: the F6:n tally, with prior calculations to estimate ratios of the F6:n tally to the true heat deposition tally in the core and reflector regions, may provide a very efficient and reasonably accurate method to estimate the fission power distribution in a realistic reactor configuration.

• These are preliminary results and more work needs to be done to assess the sensitivity of these ratios and to examine the possibility of using other tallies.