Moltres: a MOOSE Application for Simulation of MSRs

Alex Lindsay and Kathryn Huff
University of Illinois
Presentation Overview

- Intro to Moltres/MOOSE
- Moltres goals
- Results:
  - 3D unstructured: temperature coupling issues
  - 2D axisymmetric: good steady-state profiles
  - 3D structured: steady-state neutron and precursor profiles; incomplete conduction between fuel and moderator
- Future work
  - Transients
Obtaining Moltres

- git clone https://github.com/arfc/moltres
- cd moltres
- git submodule init
- git submodule update
Moltres/MOOSE

- Moltres is built on top of the Multi-physics Object-Oriented Simulation Environment (MOOSE):
- MOOSE interfaces with libMesh to discretize simulation volume into finite elements
- Provides interface for coding residuals that correspond to weak form of governing PDEs; also interface for coding Jacobians → more accurate Jacobians → more efficient convergence
- Residuals and Jacobians handed off to PetSc which handles solution of resulting non-linear system of algebraic equations

**Diagram:**

- **Moltres:** Describe MSR governing equations with residual functions
- **MOOSE & LibMesh:** Discretize physics via the finite element method. PDEs → algebraic system of equations
- **PetSc:** Solve algebraic system of equations using Newton methods
Moltres Kernels

- In MOOSE jargon, kernels are individual pieces of governing equations
- Modular in nature; for example, a “Diffusion” kernel could be used equally well to describe conduction or viscous shear

CoupledFissionEigenKernel

\[-\frac{\chi_g^p}{k} \sum_{g'=1}^{G} (1 - \beta)\nu \Sigma_{g'}^f \phi_{g'}\]

CoupledFissionKernel

\[-\chi_g^p \sum_{g'=1}^{G} (1 - \beta)\nu \Sigma_{g'}^f \phi_{g'}\]

CoupledScalarAdvection

\[\nabla \cdot \bar{a}u\]

DelayedNeutronSource

\[-\chi_g^d \sum_{i=1}^{I} \lambda_i C_i\]

DivFreeCoupledScalarAdvection

\[\bar{a} \cdot \nabla u\]

FissionHeatSource

\[-\frac{P}{\int_{\partial V} \sum_{g'=1}^{G} \nu \Sigma_{g'}^f \phi_{g'} dV} \sum_{g'=1}^{G} \nu \Sigma_{g'}^f \phi_{g'}\]

where \(P\) is a representation of the total power of the reactor.
Moltres Kernels

\[ -\gamma Q_f \]

where \( \gamma \) is a factor representing heat dissipation by gamma and neutron irradiation in the moderator and \( Q_f \) is given by:

\[ \sum_{g=1}^{G} \epsilon_{f,g} \Sigma_{f,g} \phi_g \]

with \( \epsilon_{f,g} \) the amount of heat given off per fission event.

\[ -\nabla \cdot D_g \nabla \phi_g \]

**GroupDiffusion**

\[ -\sum_{g \neq g'}^{G} \Sigma_{g \to g'} \phi_{g'} \]

**InScatter**

\[ \frac{1}{v_g} \frac{\partial \phi_g}{\partial t} \]

**NtTimeDerivative**

\[ -\nu_f \Sigma_f \phi \]

**SelfFissionEigenKernel**

\[ -\delta \nabla \phi \]

**ScalarAdvectionArtDiff**

\[ \delta = \frac{|\vec{a}| h_{max}}{2} \]

where \( \delta \) is an artificial diffusion coefficient determined by:

\[ -\Sigma_t \phi_g \]

**SigmaR**

\[ \sum_{g=1}^{G} \epsilon_{f,g} \Sigma_{f,g} \phi_g \]

**TransientFissionHeatSource**

\[ \lambda_i C_i \]

**PrecursorDecay**

\[ -\sum_{g=1}^{G} \beta_i \nu \Sigma_{g,g'} \phi_{g'} \]

**PrecursorSource**
Governing Equations

• Piece together the kernels:
  - Multi-group diffusion
  - Precursor balance with drift
  - Heat conduction-convection with fission source in fuel
  - Heat conduction with option for irradiation source in moderator

\[
\frac{1}{v_g} \frac{\partial \phi_g}{\partial t} - \nabla \cdot D_g \nabla \phi_g + \Sigma^r_g \phi_g = \sum_{g \neq g'} G \Sigma^s_{g' \rightarrow g} \phi_{g'} + \chi_g^p \sum_{g' = 1}^G (1 - \beta) \nu \Sigma^f_{g'} \phi_{g'} + \chi_g^d \sum_i^I \lambda_i C_i
\]

\[
\frac{\partial C_i}{\partial t} = \sum_{g' = 1}^G \beta_i \nu \Sigma^f_{g'} \phi_{g'} - \lambda_i C_i - \frac{\partial}{\partial z} uC_i
\]

\[
\rho_f c_{p,f} \frac{\partial T_f}{\partial t} + \nabla \cdot (\rho_f c_{p,f} \vec{u} \cdot T_f - k_f \nabla T_f) = Q_f
\]

\[
\rho_g c_{p,g} \frac{\partial T_g}{\partial t} + \nabla \cdot (-k_g \nabla T_g) = Q_g
\]
Example Literature MSR Multi-Physics

- Single channel
- Reflective/insulating conditions at radial boundary for neutrons/temperature
- Variables: velocity ($u_r, u_z$), fluxes (2 group), temperature, precursor conc.

Concurrent efforts in Italy, Switzerland, DOE lab, UCB, and others
Goals

• Current:
  – demonstrate steady-state coupling of neutron fluxes, precursors, and temperature for thermal MSR design

• Future:
  – investigate reactor dynamics under transient accident scenarios to assess safety
  – explore salt processing and chemistry
Problem Set-Up

- Base current model on Molten Salt Reactor Experiment
- 22.5% volume fraction fuel
- Remainder graphite
- Fuel salt: Fluoride Lithium Beryllium (FLiBe) with 33% enriched Uranium
- Generate group constants with Serpent or Newt
- Moltres builds neutronics with action so accepts arbitrary number of groups; however, all results presented here will be for two groups

<table>
<thead>
<tr>
<th>Component</th>
<th>Mass Fraction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Li-7</td>
<td>.1090</td>
</tr>
<tr>
<td>Li-6</td>
<td>5E-6</td>
</tr>
<tr>
<td>F-19</td>
<td>.6680</td>
</tr>
<tr>
<td>Be-9</td>
<td>.0627</td>
</tr>
<tr>
<td>U-235</td>
<td>.0167</td>
</tr>
<tr>
<td>U-238</td>
<td>.0344</td>
</tr>
</tbody>
</table>
Kernels

# ./diff_group1
  type = GroupDiffusion
  variable = group1
  group_number = 1

# ./sigma_r_group1
  type = SigmaR
  variable = group1
  group_number = 1

# ./fission_source_group1
  type = CoupledFissionEigenKernel
  variable = group1
  group_number = 1

# ./inscatter_group1
  type = InScatter
  variable = group1
  group_number = 1

# ./diff_group2
  type = GroupDiffusion
  variable = group2
  group_number = 2

# ./sigma_r_group2
  type = SigmaR
  variable = group2
  group_number = 2

# ./fission_source_group2
  type = CoupledFissionEigenKernel
  variable = group2
  group_number = 2

[Variables]
  [. ./group1]
  [. ./group2]

[BCs]
  [. ./vacuum_group1]
    type = VacuumConcBC
    boundary = 'fuel_bottoms fuel_tops moder_bottoms moder_tops outer_wall'
    variable = group1

  [. ./vacuum_group2]
    type = VacuumConcBC
    boundary = 'fuel_bottoms fuel_tops moder_bottoms moder_tops outer_wall'
    variable = group2

[NT]
  num_groups = 2
  num_precursor_groups = 6
  var_name_base = group
  vacuum_boundaries = 'fuel_bottoms fuel_tops moder_bottoms moder_tops outer_wall'
  create_temperature_var = false
  eigen = true
3D Geometry

- Start with simple cuboid lattice
- Red regions fuel
- 5 cm pitch
Mesh Construction

- Repeating structure, block and boundary IDs, generated using gmsh
k-eigenvalue simulation; reactor axial mid-plane

\[ T = 922 \text{ K}; \ k = 1.002 \]
Issues when Coupling in Temperature

Moderator temperatures next to highest fissioning fuel regions drop below inlet temperature of 922 K
Boundary Shocks

- Over 1 million elements in simulation, but only one internal node spans radial moderator dimension
Boundary Shocks

- Can see identical phenomena in simple Transient-Diffusion problem with quickly ramped boundary values
Moving Forward

- Move to simplified 2D-axisymmetric formulation, using same group constants and maintaining correct fuel-moderator volume fractions
- Structured mesh: allows much finer meshing in radial direction where gradients are much larger
- Return later to 3D problem, again using structured mesh
2D-axi geometry

- Repeating (narrow) fuel – (wide) moderator regions

Vacuum nts; T, Pres: OutflowBC

Vacuum nts; T = 922 (downcomer cooling)

Vacuum nts; T = 922; Pre = 0 (inlet)

Mesh

axis of symmetry

graphite

fuel
Results (neutrons)
Results (temperature)
Results (precursors)
New 3D Results

- Lessons learned from 2D-axisymmetric case: using *structured* 3D mesh, can evolve close to steady-state without observing boundary shocks
- Over heat conduction time scales, convergence slow
  - Unsure why
- Simulation run on 160 processors on Illinois’s Blue Waters supercomputer
Results (neutrons)

Fast

Thermal
Results (temperature)

- Not simulated out to long enough times for heat to finish conducting from fuel to moderator
- End time = 17 seconds
- Characteristic diffusion time through moderator = 24 seconds
Results (precursors)

Longest lived precursor

Shortest lived precursor
Moltres Summary/Future Work

- Good coupling with fine meshing in 2D
- Some good 3D results; slow convergence at long time scales

Next:
- Instead of assuming constant, couple in flow variables
- Start exploring transients, e.g. explore responses to reactivity insertion
- Implement cross section dependence on control rods
Moltres Summary/Future Work

- Presented preliminary results
- Still a bit to go to replicate state-of-the-art multi-physics simulation capabilities
- However, we hope use of modern development practices will accelerate growth and assist in review and transparency
GitHub with Continuous Integration

When learning Moltres and the underlying MOOSE framework, it may be best to start with a tutorial and examples.

The following pages give the math behind the Kernel and BC classes that are used to construct the weak forms of the PDEs required for the finite element method:

- Kernels
- BCs

The page below gives a detailed outline of a typical Moltres input file and should help the user prepare his/her own customized input:

- Example Input

Wiki with User Guides

Doxygen pages for helping new developers
Acknowledgements

- MOOSE team
- Andrei Rykhlevskii