Fuel performance simulation with MOOSE

Due date: May 7, 2021

Name: Ana Carolina Antunes Acosta Fernandes

Instructor: Dr. Benjamin Beeler

Contents

1 Problem description

This project aims at the modeling and simulation of nuclear fuel behavior in different conditions using the MOOSE framework. It consists in four exercises, divided in two sections: 1-D simulations and 2-D simulations. Figure 1 shows both configurations.

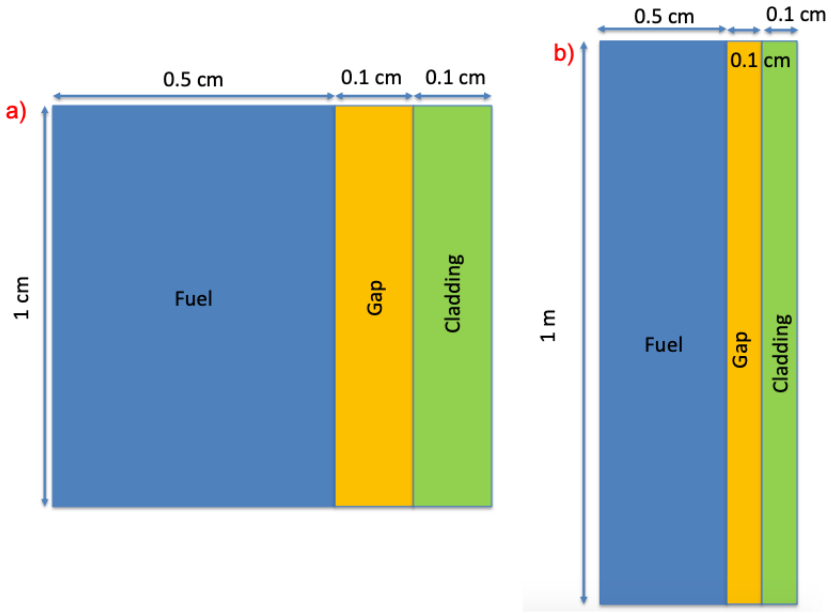

Figure 1: Geometry for a) 1-D simulations, and b) 2-D simulations (not to scale)

1.1 1-D case

The first section of this project was based on the geometry described in Figure 1 a). A boundary condition of 500 K temperature was set for the cladding outer temperature, with constant thermal conductivity values for all the materials. Two exercises were modeled:

1. Solve for temperature profile: steady state, with $q'' = 250 W/cm^2$, and
2. Solve for centerline temperature vs. time: transient (from $t = 0s$ to $t = 200s$) with

$$q'' = 150(1 - e^{-0.01t}) + 250. \tag{1}$$

1.2 2-D case

The second section was composed of two 2-D simulations with geometry as described in Figure 1 b). A boundary condition of axial cladding outer temperature varying with the height z was set. Two exercises were modeled:

1. Solve for temperature profile: steady state, with $q'' = 250W/cm^2$, $z = 0.25m$, $z = 0.50m$, $z = 1.00m$ and

2. Solve for centerline temperature vs. time: transient (from $t = 0s$ to $t = 200s$) with

$$q'' = 150(1 - e^{-0.01t}) + 250, \tag{2}$$

and $z = 0.25m$, $z = 0.5m$, and $z = 1m$.

It was also necessary to find the peak centerline temperature location at steady state and at $t = 200s$ in the transient simulations.

2 Methodology

2.1 Materials and properties

In order to properly simulate fuel behavior in the given conditions, it was necessary to define which materials would be used and assume reasonable values for the material properties. In this case, it was decided to use the most common materials: UO_2 as a fuel, a He gap, Zircaloy-2 cladding, and water coolant, due to simplicity and availability of data. Table 1 shows the properties related to these materials that were used in the simulations.

For the 2-D case, the outer cladding temperature was defined [1]:

$$T_{co}(z) = T_{\infty}(z) + \frac{LHR}{2\pi R_f h_c}, \tag{3}$$

Table 1: Constant values used in this problem

Property	Value	Units
h_c [1]	5.5E4	[W/m^2· K]
$\dot{m}$ [1]	0.22	[kg/s]
$c_{p,water}$ [1]	4200	[J/kg· K]
k_{UO_2} [2]	3.55	[W/m·K]
k_{clad}[3]	13.66	[W/m·K]
k_{gap}[1]	25	[W/m·K]
T_∞ [1]	500	[K]

with $T_\infty(z)$ our coolant temperature at point z, LHR our linear heating rate, R_f our fuel radius and h_c the coolant heat transfer coefficient. We can remember the definition of $T_\infty(z)$:

$$T_\infty(z) = T_\infty(0) + \frac{1}{1.2}\frac{Z_0 \times LHR^0}{\dot{m}c_{pw}}\left\{sin(1.2) + sin\left[1.2\left(\frac{z}{Z_0} - 1\right)\right]\right\}, \tag{4}$$

with Z_0 the midpoint (in this case, $Z_0 = 0.50m$), $\dot{m}$ the mass flow rate and c_{pw} the water specific heat, with values specified in Table 1. Our midpoint linear heating rate, or LHR^0, is related to our areal heating rate as follows:

$$LHR^0 = q''(t)\pi R_f^2, \tag{5}$$

For the steady state case, $q''(t)$ is a constant and we simply substitute for its value; for the transient case, $q''(t)$ is described by Equation 2.

2.2 MOOSE implementation

2.2.1 The Capybara application

Online resources [4] were consulted in order to facilitate the understanding of the MOOSE framework.

The first step is the creation of an application. MOOSE has a pre-made *makefile* for that and all you need to do is modify the file to include the modules you want your application to feature.

The Capybara application was created, with the heat conduction and tensor mechanics modules activated.

The heat conduction module is defined as allowing for "modeling conduction, radiation between gray, diffuse surfaces, and it contains provisions to couple temperature fields to fluid domains through boundary conditions" [4]. Or, it solves the heat conduction equation:

$$\rho(t,\vec{x})c(t,\vec{x})\frac{\partial T}{\partial t} = \nabla k(t,\vec{x})\nabla T + \dot{q}, \text{for } \vec{x}\,\epsilon\,\Omega, \tag{6}$$

with T being the temperature, t the time, $\vec{x}$ the vector of spacial coordinates, ρ is the density, k is the thermal conductivity of the material and $\dot{q}$ is a heat source in a Ω domain.

The equation can only be solved if the boundary conditions are specified. The two most common are the Dirichlet BC – where the BC is a constant value, ($u = g$) – and Neumann BC – where a gradient is set as the the BC, or $\frac{\partial u}{\partial n} = h$.

The tensor mechanics module solves for continuum mechanics problems; it can either be used alone, for pure mechanics simulations, or in combined physics simulations with the Heat Transfer module (as it is in this case), Phase Field, Contact, Porous Flow, and/or XFEM modules.

2.2.2 Input writing

With the Capybara application created, it was possible to proceed to the input writing. The 2-D transient input is available on Appendix A. MOOSE uses a format known as "hierarchical input text" (*hit*), with 6 basic parts or subdivisions, which are:

1. Mesh: describes the geometry of the problem. In this case, a general mesh was created and then subdivided into 3 different regions, called blocks (block_0 is fuel, block_1 is gap, and block_2 is cladding), that later will have different properties attributed to them.

2. Variables: define the variables that MOOSE should solve for – the unknowns of the problem. For this problem, our variable of interest is the temperature.

3. Kernels: equations to be solved. In this project, we have heat generation and heat conduction in the fuel and heat conduction in the gap and the cladding, so we have a kernel

for each phenomenon. For each kernel, it is necessary to specify its type, the variable it is modifying, and the material property linked to it. Note that since we only have heat generation in the fuel region, we restrict this kernel to block_0.

4. BCs: define the boundary conditions of the problem. The BCs can be set in a variety of ways; for the 2-D transient case, we set the previously defined *axial_tco* function as the right-side wall (outer cladding) temperature, which varies with fuel height.

5. Executioner: define how the problem will solve: as a steady-state or transient problem, and eventual extra parameters, such as time step, starting time, and ending time.

6. Outputs: define how the solution will be written. The exodus output produces a *.e file that can be read on Peacock or Paraview, for instance. The csv file was asked here for we had some points added in our post-processor.

Some optional parts were used for this problem:

7. Problem: this part was used to transform our cartesian coordinates into a cylindrical coordinate system.

8. Materials: describe the materials in our system. It is necessary to specify the type of material and its properties. It is also possible to relate each material to a geometry block_id (as we defined in the Mesh part), which is how we set different properties to all our regions.

9. Functions: define the functions that are necessary for the problem definition.

10. Post processors: this part runs after the problem has been solved, being used to obtain specific values in addition to the general output created. Here, we ask for the temperature values at $z = 0.25m$, $0.5m$, and $1m$, in addition to the maximum temperature for our last time step.

The input, then, gives the application all the necessary information to solve our problem.

3 Results and discussion

3.1 1-D simulations

Figure 2 shows the temperature profile for the 1-D steady state case, with the parameters as described in Section 2.1. It is possible to identify, in the Paraview generated plot (right-hand side), the materials boundaries according to the difference in their behavior: in the fuel we have heat generation (region between 0 and 0.5 cm); then, we have a gap with a linear decrease in temperature (from 0.5 cm to 0.6 cm) and a clad with a lower thermal conductivity (from 0.6 cm to 0.7 cm).

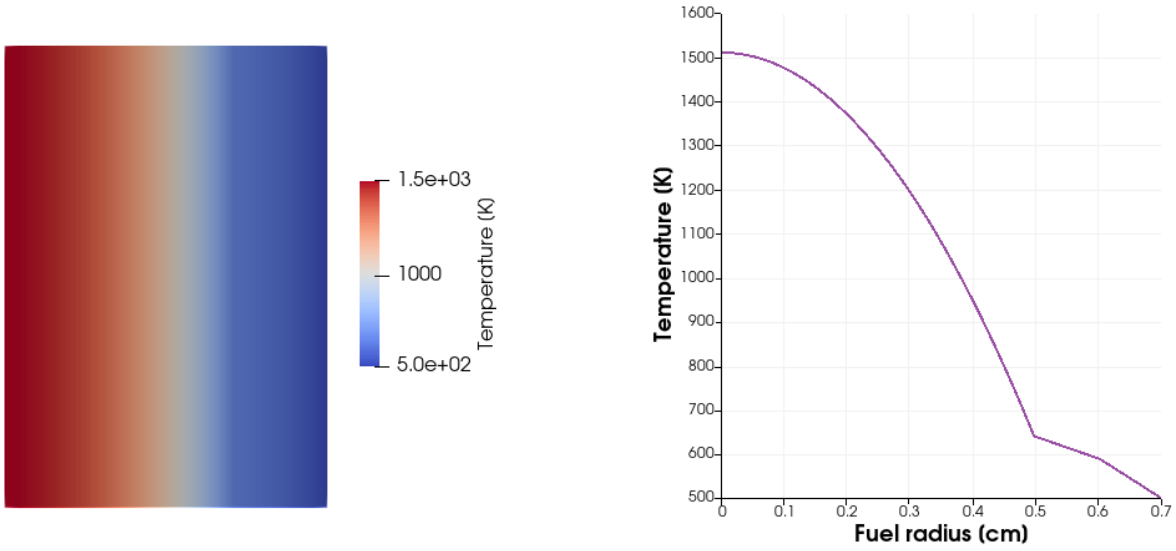

Figure 2: Temperature profile for 1-D steady state simulation a) Heat map; and b) Temperature vs. fuel radius

Figure 3 shows the centerline temperature over time for the 1-D transient. As expected, the temperature increases with time – which reflects the increase of heat generation with time.

For both steady state and transient simulations, the maximum temperatures in the fuel do not reach the fuel melting temperature (3120 K [2]) or the clad melting temperature (2122 K [3]).

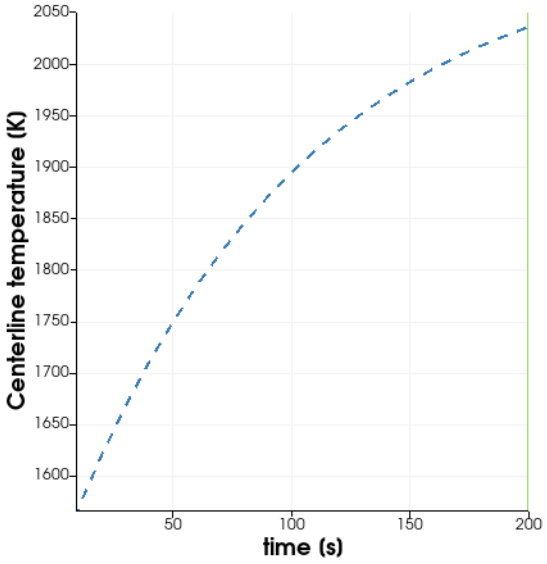

Figure 3: Centerline temperature over time for the 1-D transient

3.2 2-D simulations

Because the 2-D fuel height is so much larger than its radius, it is not interesting to see the fuel heat map for the 2-D case. Instead, Figure 4 shows the centerline temperature versus fuel height on the left and the temperature profiles for selected fuel heights on the right-hand side. It

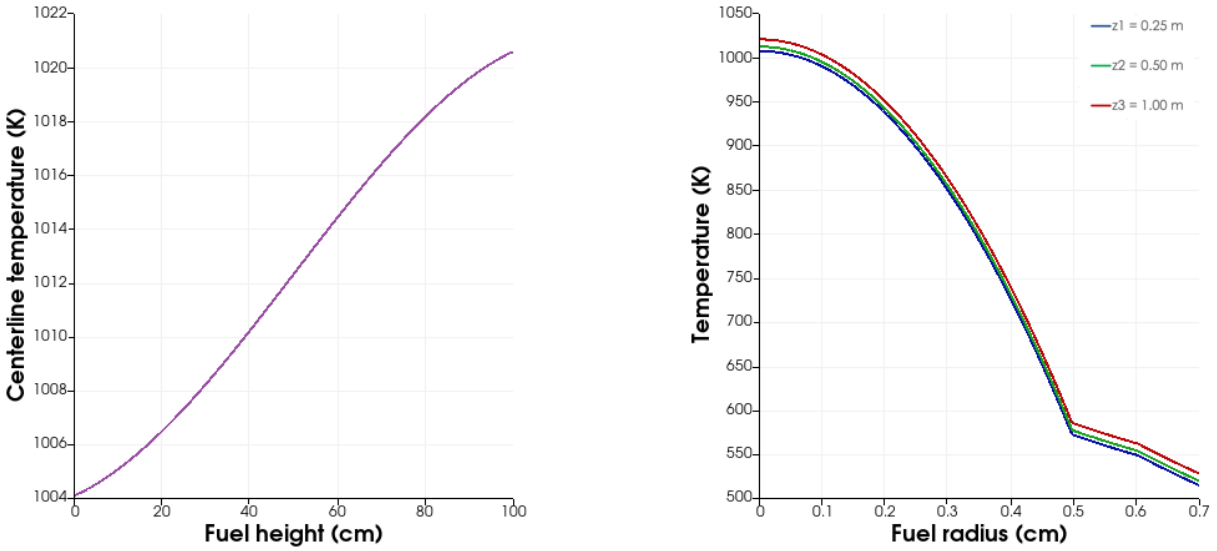

Figure 4: Temperatures for the 2-D steady state case a) Centerline temperature vs. height; and b) Temperature profiles for selected fuel heights

becomes clear that the highest temperature occurs at $z_3 = 1.00m$.

The temperatures at z_1, z_2 and z_3 were also observed at the 2-D transient. We can see their behavior over time for these three fuel heights in Figure 5.

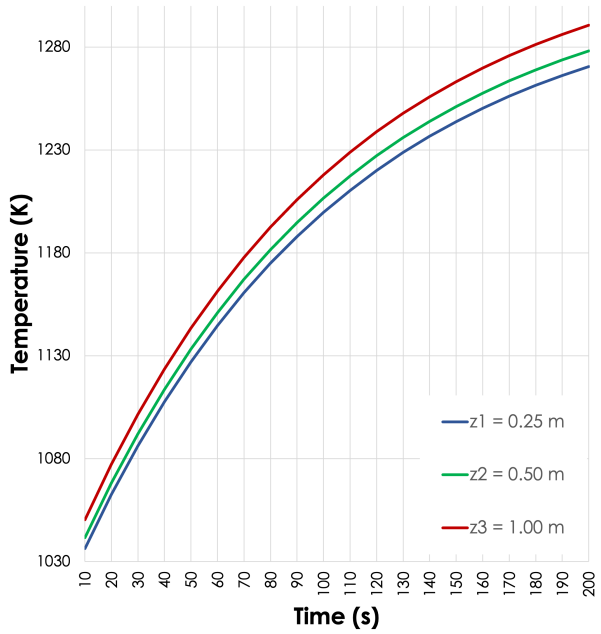

Figure 5: Centerline temperature over time for the 2-D transient in selected heights

Finally, Table 2 shows the temperatures for selected fuel heights for the steady state case and for the transient case at $t = 200s$ (the final time). Again, the maximum temperatures do not represent danger for the fuel or the clad integrity, for they are far from the melting temperatures.

Table 2: Centerline temperatures at selected heights

Height (m)	Temperature (K) steady state	Temperature (K) transient (at t=200s)
z_1 = 0.25	1007.34	1270.55
z_2 = 0.50	1012.34	1278.15
z_3 = 1.00	1020.58	1290.65

4 Conclusions

This work was challenging especially due to the lack of knowledge in MOOSE and Paraview. However, the online resources and instructor's aid were very helpful throughout the process.

With a successful implementation, it was possible to verify previously learned fuel thermal behavior in different circumstances. The 1-D steady state case, the simplest of all, shows a well known temperature profile shape. When implemented in a transient, in which q'' increased over time, it was possible to see the increasing centerline temperature over time. For the 2-D steady state case there was an increase in the centerline temperature with height that was also shown in the figure generated by Paraview. Lastly, in a 2-D transient simulation, the centerline temperature over time for different fuel heights was plotted, showing again that the highest temperature was at the top of the fuel for all time steps.

The maximum temperatures for all cases do not reach fuel or cladding melting temperatures, showing that fuel integrity is not jeopardized within the defined operation parameters.

Overall, the project was a success in its goals: to learn about a fuel performance code and implement some of its capabilities, with a possibility of doing more sophisticated work in the future.

References

[1] B. Beeler, “Lecture notes in Nuclear Fuel Performance,” April 2021.

[2] J. Carbajo, “Thermophysical properties of MOX and UO_2 fuels including the effects of irradiation,” Oak Ridge National Lab., Tech. Rep., 2001.

[3] C. Whitmarsh, *Review of Zircaloy-2 and Zircaloy-4 properties relevant to NS Savannah reactor design*. Oak Ridge National Laboratory for the US Atomic Energy Commission, 1962.

[4] Idaho National Laboratory. MOOSE - Multiphysics Object-Oriented Simulation Environment. [Online]. Available: http://https://mooseframework.inl.gov

A MOOSE 2-D transient input

```
[Mesh]
  [fuel]
    type = GeneratedMeshGenerator
    dim = 2
    nx = 100
    ny = 100
    xmax = 0.7
    ymax = 100
  []
  [gap]
    type = SubdomainBoundingBoxGenerator
    input = fuel
    bottom_left = '0.5 0 0'
    top_right = '0.7 100 0'
    block_id = 1
  []
  [clad]
    type = SubdomainBoundingBoxGenerator
    input = gap
    bottom_left = '0.6 0 0'
    top_right = '0.7 100 0'
    block_id = 2
  []
[]

[Problem]
  coord_type = RZ
[]

[Variables]
  [./temp]
  [../]
[]


[Materials]
  [./fuel]
    type = ADGenericConstantMaterial
    prop_names = 'thermal_conductivity'
    prop_values = '0.0355'
    block = 0
```

```
  [../]
  [./gap]
    type = ADGenericConstantMaterial
    prop_names = 'thermal_conductivity'
    prop_values = '0.25'
    block = 1
  [../]
  [./clad]
    type = ADGenericConstantMaterial
    prop_names = 'thermal_conductivity'
    prop_values = '0.1366'
    block = 2
  [../]
[]

[Functions]
  [./volumetric_heat]
    type = ParsedFunction
    value = (150*(1-exp(-0.01*t))+250)
  [../]
  [./axial_tco]
    type = ParsedFunction
    value = 'tinf + (1/1.2)*(z0*q*pi*rf^2/(m*cp))*(sin(1.2)+sin(1.2*(y/z0 -1)))
    + q*rf/(2*hc)'
    vars = 'tinf z0 q rf m cp hc'
    vals = '500 50 volumetric_heat 0.5 0.22 4200 5.5'
  [../]
[]

[Kernels]
  [./heat]
    type = ADHeatConduction
    variable = temp
    thermal_conductivity = thermal_conductivity
  [../]
  [./heatgen]
    type = HeatSource
    function = volumetric_heat
    variable = temp
    block = 0
  [../]
[]
```

```
[BCs]
  [./righttemp]
    type = FunctionDirichletBC
    boundary = right
    variable = temp
    function = axial_tco
  [../]
[]

[Executioner]
  type = Transient
  end_time = 200
  dt = 10
[]

[Postprocessors]
  [./z1]
    type = PointValue
    point = '0 25 0'
    variable = temp
  [../]
  [./z2]
    type = PointValue
    point = '0 50 0'
    variable = temp
  [../]
  [./z3]
    type = PointValue
    point = '0 100 0'
    variable = temp
  [../]
  [./peak]
    type = ElementExtremeValue
    variable = temp
    execute_on = timestep_end
    value_type = max
  [../]
[]

[Outputs]
  exodus = true
  csv = true
[]
```