Daniel Falkenbach

NE 533: Nuclear Fuel Performance

MOOSE Project Report

29 April 2022

MOOSE Project Report

Introduction

This paper investigates three separate simplified nuclear fuel, cladding, and gap configurations modeled in “Multiphysics Object Object-Oriented Simulation Environment”, or MOOSE. MOOSE was developed by INL and is a Multiphysics framework capable of coupling all physics together in Modules such as Tensor Mechanics, Chemical Reactions, Fluid Properties, Heat Conduction, Neutronics, etc. and is well suited to analyze nuclear fuel, cladding, and associated assemblies. MOOSE’s governing equations used in every analysis are Conservation of Mass, Conservation of Energy, and Darcy’s Law, and with these governing equations, Kernels were developed and can be selected as required. If the input file is done correctly, the user can be confident the Kernels are matching physics, and the results should closely replicate the physical real-world phenomena. Given the tool’s capabilities, confidence in inputs, and open-source platform, it is a widely used tool in engineering and especially in Nuclear Engineering. Each of the three configurations in this paper have two separate inputs and solutions required. In other words, Part 1 and Part 2 have one portion analyzing the proposed problem in SS, while the other portion analyzing in transient; Part 3 includes one part with a constant thermal conductivity (K) of UO_2, while the second part analyzes with K as a function of temperature (T). Given each part of these analyses have different required Executioner solver types, each required its own script, all of which can be found in Appendix A. The scripts were developed, drafted, and executed in MobaXterm via the “.i” file type, which is the required input file type for MOOSE. The scripts were loaded via WinSCP in order to be computationally solved via into the NCSU RDFMG cluster, resulting in outputs of the “.e” or exodus file type. All six exodus files were loaded into Paraview to plot and summarize the results visually and those results and discussion can be found in sections Part 1 – Part 3. All six scripts used for these simulations have a minimum of six individual blocks (namely Mesh, Variables, Kernels, BCs, Executioner, and Outputs) or definitions capturing aspects of the planned analysis; however, some were created with additional blocks based on the desired analytical result (such as Function, Materials, Modules, etc.). All six analysis parts shown in this paper used UO_2 as the fuel, air as the gap "material" for parts 1 and 2, and Zircaloy-4 as the cladding material. The following assumptions and constants were used for the Materials and applied to their respective blocks as noted in the Mesh descriptions in Part 1 – 3. The Gap and Clad materials below were not used in Part 3.1 nor 3.2, and constants $E, \nu, and\ \alpha$ defined below are unique to Part 3.1 and Part 3.2 and not applied elsewhere:

Material	Constant	Value	Units	Reference
Fuel ($\boldsymbol{UO_2}$)	Thermal Conductivity ($\boldsymbol{K}$)	10.5	$\frac{W}{cm-K}$	Lecture Notes
	Density ($\boldsymbol{\rho}$)	10.97	$\frac{g}{cm}$	Wikipedia
	Heat Capacity ($\boldsymbol{c}$)	1	$\frac{J}{K}$	Wikipedia
	Young's Modulus ($\boldsymbol{E}$)	200	MPa	Lecture Notes
	Poisson's Ratio ($\boldsymbol{\nu}$)	0.345	$None$	Lecture Notes
	CTE ($\boldsymbol{\alpha}$)	$1.2E-5$	K^{-1}	Lecture Notes
Gap ($\boldsymbol{air\ @\ STP}$)	Thermal Conductivity ($\boldsymbol{K}$)	$2.587E-4$	$\frac{W}{cm-K}$	Wikipedia
	Density ($\boldsymbol{\rho}$)	0.001204	$\frac{g}{cm}$	Wikipedia
	Heat Capacity ($\boldsymbol{c}$)	1	$\frac{J}{K}$	Wikipedia
Clad ($\boldsymbol{Zircaloy-4}$)	Thermal Conductivity ($\boldsymbol{K}$)	0.215	$\frac{W}{cm-K}$	Matweb
	Density ($\boldsymbol{\rho}$)	6.56	$\frac{g}{cm}$	Matweb
	Heat Capacity ($\boldsymbol{c}$)	1	$\frac{J}{K}$	Matweb

Table 1: Material Constants

Analysis Setup | Results | Discussion: Part 1

Part 1.1 analyzed the geometry as shown below in Figure 1, which consists of fuel with dimensions 1 x 1 cm, a gap of gas (air assumed here @ STP) at 0.002 x 1 cm, and cladding with 0.1 x1 cm.

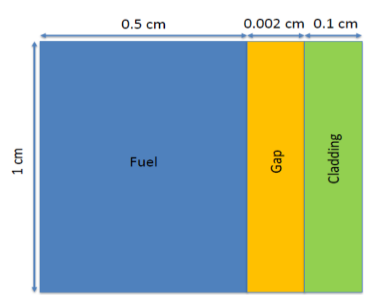

Figure 1: Part 1 Geometry

The problem is set up to determine temperature profile for a SS Linear Heat Rate (LHR) $= 150\ \frac{W}{cm}$. The problem was solved as 1-D; however, the geometry setup was in 2-D RZ-coordinates. A 2-D Mesh was set up with the GeneratedMeshGenerator type to create the overall Mesh space (1 x 0.602 cm) and provide enough elements to provide results that converged. This was determined to be a minimum of the smallest dimension (the gap) vs. overall x-dimension, and thus defined at 301 elements in x-direction by the following equation:

$$Required\ elements\ in\ x - dir = \frac{0.602}{0.002} = 301\ elements$$

In order to define the small area of the gap, mesh subdomains were required using the SubdomainBoundingBoxGenerator type, which was used to define 2 domains covering both the gap and cladding, and cladding itself, while an automatic block 3 was created for the fuel (excluded from subdomain). The Kernels utilized were HeatConduction, applied to all of the 3 blocks described in the Mesh, and HeatSource, to represent the scalar constant $LHR = 150\ \frac{W}{cm}$ in the fuel block. BCs were defined as $500\ K$, which was applied to the right cladding boundary only and the solver was set to steady in order to analyze the SS result. The results are shown below as simulated in Paraview:

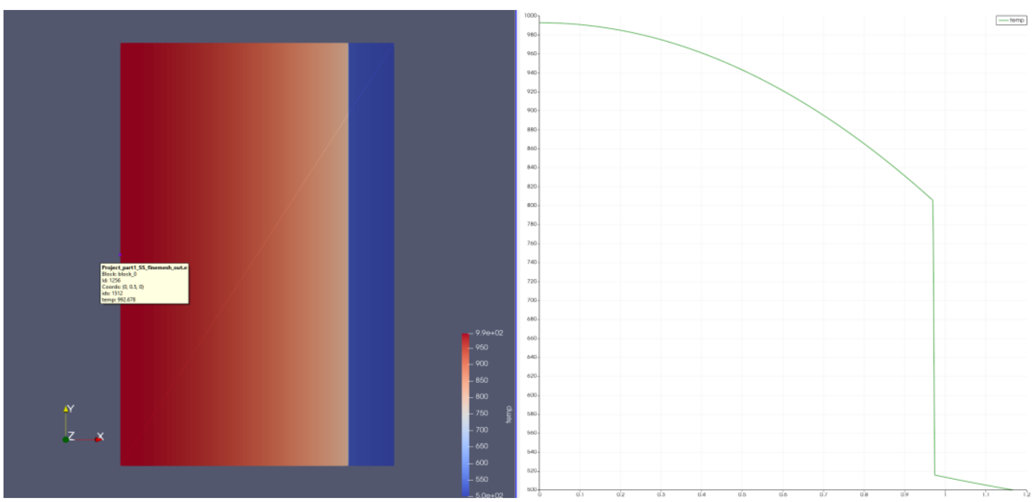

Figure 2: Part 1.1 SS Results

The left side of Figure 2 shows the temperature gradient, with the legend on the right showing hottest in red while dark blue is coldest. The node on the left-hand side is selected to show the max temperature, which was determined to be $993\ K$. The right hand-side plot of Figure 2 shows the temperature profile as a function of the diagonal distance (left image shows a faint line that is the reference for the plot) with start on bottom left corner, with end in top right corner. The sharp drop at ~1 cm diagonal distance occurs at the air gap, which is due to the very low thermal conductivity of the air. The dark blue section is the clad where there is $\sim 20\ K$ difference between the CO and CI with the CO set as a $BC = 500\ K$ and held constant. To correlate the results and provide confidence the resulting temperature profile is as expected, hand calcs were performed to determine the fuel centerline temperature (T_0), utilizing the same constants similar equations in spherical geometry:

$$T(r) - T_s = \frac{LHR}{6\,\pi k}\left(1 - \frac{r^2}{R_f^2}\right);\ where\ r = distance\ from\ T_o;\ R_f = fuel\ plate\ thickness$$

$$R_f = 0.5\ cm;\ LHR = 150\frac{W}{cm};\ T_{CO} = 500\ K$$

$$k_{fuel} = 0.05\frac{W}{cm - K};\ k_{gap} = 2.587E - 4\frac{W}{cm - K};\ k_{clad} = 0.215\frac{W}{cm - K}; R_f = 0.5\ cm$$

$$T_{CI} = \frac{LHR * t_{clad}}{2 * \pi * R_{fuel} * k_{clad}} + T_{CO} = \frac{150 * 0.1}{2 * \pi * 0.5 * 0.215} + 500 = 522.21\ K$$

$$T_{fuel} = \frac{LHR}{2 * \pi * R_{fuel} * h_{gap}} + T_{CI};\ where\ h_{gap} = \frac{k_{gap}}{t_{gap}} = \frac{2.587E - 4}{0.002} = 0.1294$$

$$T_{fuel} = T_s = \frac{150}{2 * \pi * 0.5 * 0.1294} + 522.21 = 891.19\ K$$

$$T_0 = \frac{LHR}{4 * \pi * k_{fuel}} + T_s = \frac{150}{4 * \pi * 0.05} + 891.19 = 1129.92\ K$$

As can be seen from the above result, the model provides a similar centerline temperature. There is some disparity (~100 K) between the two results, however that is likely due assumptions. These assumptions are required in order to easily solve the analytical solution and without them, the hand calc would take significantly more time to solve. The specific assumptions to note are axisymmetric, constant in z, and the analytical solutions using a linear profile only, as well as the assumed geometries.

The Part 1.2 model is set up to determine the temperature profile for a LHR as a function of time, i.e. transient formulas were required. The inputted LHR transient equation was defined as:

$$LHR_t = 150(1 - e^{-0.05t}) + 150$$

The same Mesh and subdomains were used as previously noted as the same geometry is used. The Kernels were adjusted as a scalar/constant LHR was not applicable, and instead the Function block was utilized to input the above LHR equation with the ParsedFunction type and call this function in the BCs and Kernel block for the fuel. A new Kernel, HeatCondunctionTimeDerivative was added here given the proposed problem is a function of time, while maintain the same HeatConduction Kernel setup. Additionally, the HeatSource Kernel type calls the above formula in the Function block. The temperature variable was applied as an IC = $500\ K$, as the CO temperature was to be solved for with the LHR equation in order to provide a more realistic equation, and to provide a comparison against the SS analysis result. BCs were defined to pull in the temp variable, as well as call the previously defined function with the ADFunctionDirichletBC type. The solver was set to transient in order to analyze the result as a function of time, and the timesteps were defined at 1 second for a total of 100 seconds:

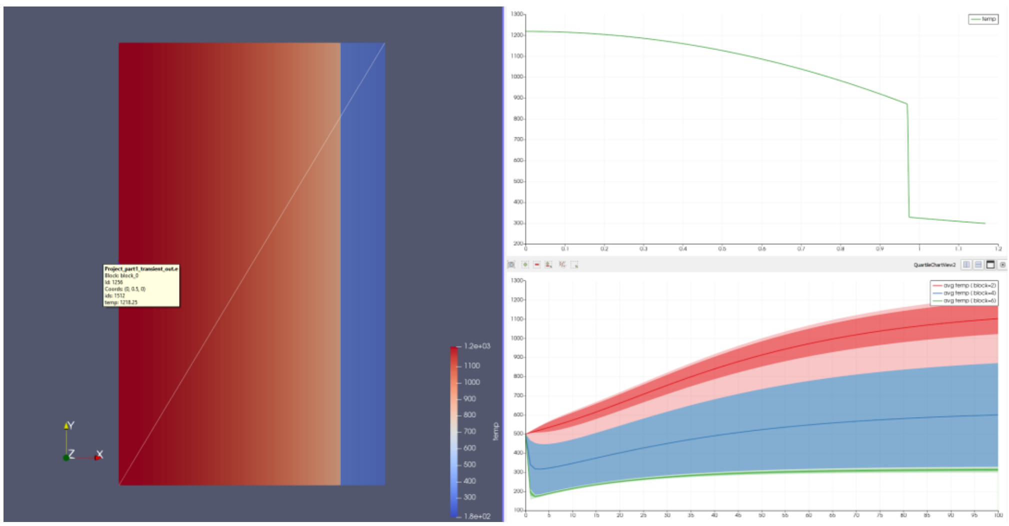

Figure 3: Part 1.2 Transient Results

The left side of Figure 3 shows the temperature gradient, with the legend on the right showing hottest in red while dark blue is coldest. The node on the left-hand side is selected to show the max temperature, which was determined to be $1218\ K$. The right hand-side of Figure 3 shows the temperature profile as a function of the diagonal distance (top right image), and the temperature profile as a function of time (bottom right image) for each block. The lowest temperature was calculated to be $\sim 300\ K$ on CO. Given the Executioner block was set to transient, the iterative calculation performed via the Implicit (or Backwards) Euler method. Given this method solves for a solution using both the current state and the later state, it provides a more accurate result than Forward Euler or simplified analytical methods at the cost of computing time. The resulting temperature difference between SS and transient at the extreme ends were found to be: $dT_{T_{CO}} = \sim 200\ K$ and $dT_{T_0} = 225\ K$. The main difference in temperatures here is due to SS models and analytical results taking the SS assumption, whereas in reality, the results would change as a function of time. These model result deltas are likely due to the assumptions required in the development of the SS model and BCs, as well as the size of the time step, as the larger the time step, the more deviation from the actual result. If a smaller time step was utilized and a finer mesh to provide better node to node transfer, the results would likely be closer; however, this was not done because this would come at a significant increase of computational time.

Analysis Setup | Results | Discussion: Part 2

The second Part 2 analysis performed was with the geometry shown in Figure 4 below:

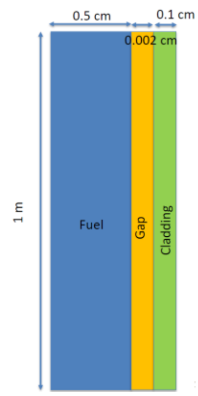

Figure 4: Part 2 Geometry

Many of the aspects of the Part 2 model, such as: Kernels, Mesh, Materials, and 2-D RZ setup were done in similar fashion than the previously described script setup in Part 1 for both SS and transient cases. As can be seen in Figure 4, the geometry is nearly identical to Part 1 as well, however with a stack length of 1 m vs. the previous 1 cm. In order to simplify and calculate the T_{cool} and enter into script, we assume perfect heat transfer b/w clad and coolant: $T_{clad} \cong T_{cool}$, and thus derive a resulting LHR equation for T_{cool}. This derived LHR function, which was captured in the Functions block and called in the BC block is defined as:

$$Assuming: \dot{m} = 0.25\frac{kg}{s}\ ; Z_0 = 50\ cm; C_{PW} = 4200\frac{J}{kg-K}; (constant\ across\ entire\ fuel\ rod)$$

$$Given: LHR^0 = 150\frac{W}{cm}$$

$$T_{cool} = \frac{1}{1.2}\frac{Z_0 * LHR^0}{\dot{m} * C_{pw}}\left[sin\ sin\ 1.2 + sin\ (1.2(\frac{z}{Z_0} - 1\ )\right] - T_{cool}^{in}$$

$$= \frac{1}{1.2}\frac{50 * 150}{0.25 * 4200}\left[sin\ sin\ 1.2 + sin\ (1.2(\frac{z}{50} - 1\ )\right] + 400\ K$$

$$= 5.95238\left[sin\ sin\ 1.2 + sin\ (1.2(\frac{z}{50} - 1\ )\right] + 400\ K\ \ll \boldsymbol{entered\ into\ model}$$

The above boxed formula was called in the BC block via the ADFunctionDirichletBC and applied to the right boundary of cladding. Additional BC was used to apply a constant 400 K temperature to the top boundary, resulting in:

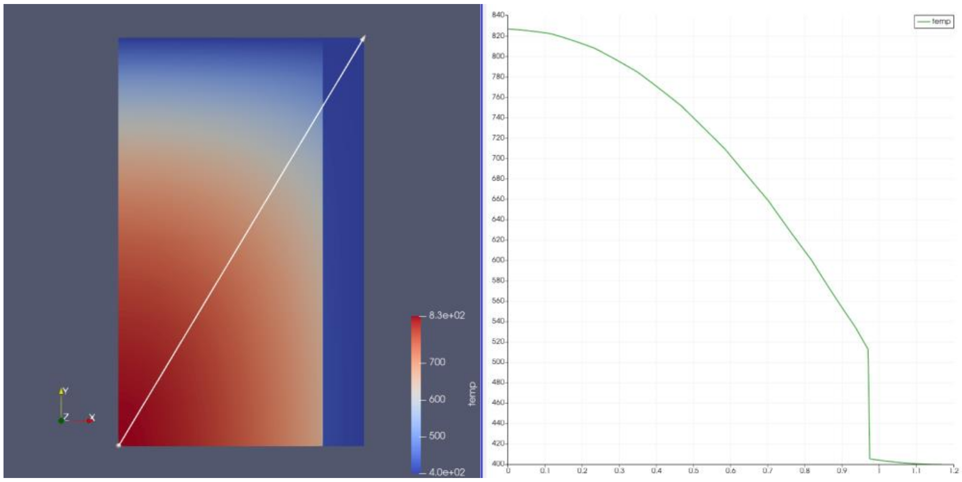

Figure 5: Part 2.1 SS Results

The above Figure 5 (in same format as Part 1) shows the max temperature seen at the bottom left of the fuel block was determined to be 826 K. The minimum temperature was found to be ~400K given the assumed BC previously noted. As can be seen on the right plot, a similar drop in temperature is seen at the gap due to its low thermal conductivity. The temperatures at the following locations on left fuel boundary are: $z_{0.25} = 809\ K;\ z_{0.5} = 755\ K;\ z_{1.0} = 400\ K$. The "centerline" temperature was found to be in the bottom left corner which is as expected given the temperature profile should be a function of both axial distance in the Y-axis, as well as distance from the fuel in the X-axis. A slight increase in clad temperature is seen on far-right side with the bottom right point of clad higher than top right due to the proximity to the fuel centerline, and the temperature increasing as a function of distance.

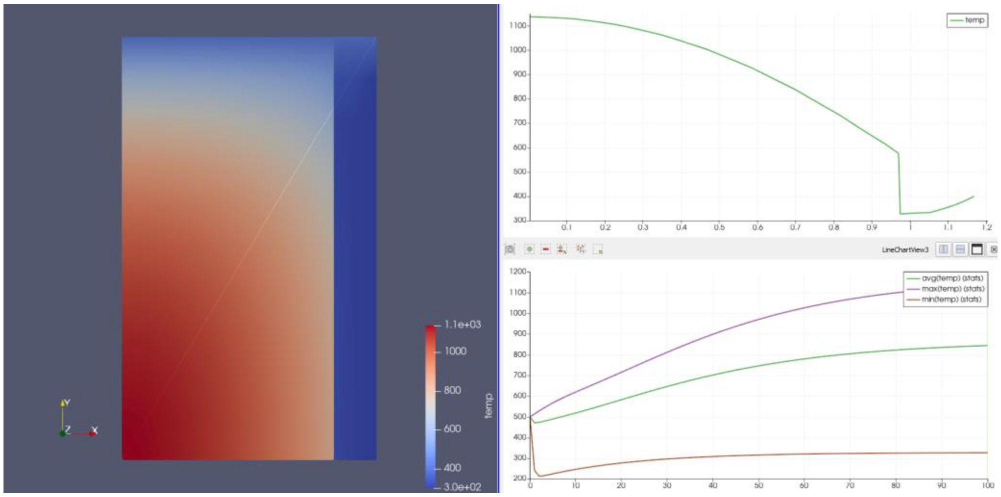

Figure 6: Part 2.2 Transient Results

The Part 2.2 transient result has a very similar trend as to the previously determined profile when comparing against Part 1 results. The max temperature was higher than the SS result by ~300 K (1136 K) and found in same location as SS case. The coolest temperature ~300 K was ~100 K lower than the SS result in same location. When comparing SS vs. transient, similar trends can be seen in Part 1's result. On the bottom right image, average, max, and minimum temperatures are plotted as a function of time with a similar profile to what was determined in Part 1. The temperatures at the following locations on left fuel boundary are: $z_{0.25} = 1110\,K$; $z_{0.5} = 1020\,K$; $z_{1.0} = 400\,K$. When comparing SS and transient results in Part 2, similar conclusions can be drawn due to the resulting deltas (caused by assumptions and time steps/mesh side etc.), and similar drop as a function of Z (or Y in model setup) when compared to the SS model, however slightly larger as shown in Part 1.

Analysis Setup | Results | Discussion: Part 3

The final Part 3 analysis had a different geometry as seen in the below Figure 7:

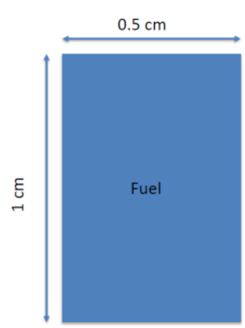

Figure 7: Part 3 Geometry

The model set up for Part 3 was significantly different than Parts 1 and 2 given the adjustments in geometry, and introduction of a new physics model Tensor Mechanics, coupled with Heat Conduction. The problem is set up to determine the resulting stresses due to thermal expansion in the fuel when assuming a uniform SS Linear Heat Rate (LHR) $= 175\ \frac{W}{cm}$. The problem was solved as 1-D; however, the geometry setup was in 2-D RZ-coordinates as done in previous Parts 1 and 2. The 2-D Mesh was set up with the GeneratedMeshGenerator type to create the overall Mesh space (1 x 0.5 cm), whoever with a decrease in total elements given no small gap to account for, and no subdomains were required given the fuel is isotropic and uniform. For Part 3.1, a constant thermal conductivity K was assumed per Table 1. Additional inputs were added to the Materials block with ComputeIsotropicEleasticityTensor type to define the $E\ and\ \nu$ as shown in Table 1, and ComputeLinearElasticStress. Additionally, the ComputeThermalExpansionEigenstrain was used to define the constants for CTE and stress-free temperature (defined as room temp). Given the new Tensor Mechanics Module was added, new Kernels were added to the previously noted HeatConduction and HeatSource Kernels, of type Diffusion (for x and y directions) in the fuel block. BCs were defined as $300\ K$ and was applied to the outer edges/parameter of the fuel to represent a constant fuel surface temperature. Another BC was applied to the bottom and left sides of the fuel in order to fix the FE, both done with the ADDirichletBC type. The solver was set to steady in order to analyze the SS result for both Part 3.1. and 3.2 however automatic scaling was required given the large OOM differences between stress and other variables in the model. Finally, the CTE was derived as a function of temperature using the following equation from IAEA TECHNICAL REPORTS SERIES No. 59 reference:

$$K_{Temp} = \frac{1}{5.33 + 0.0235\ T};\ where\ T\ is\ in\ K$$

This function for CTE in Part 3.2 was defined in the Function block and called in the Materials block via the GenericFunctionMaterial. The results are shown below as simulated in Paraview:

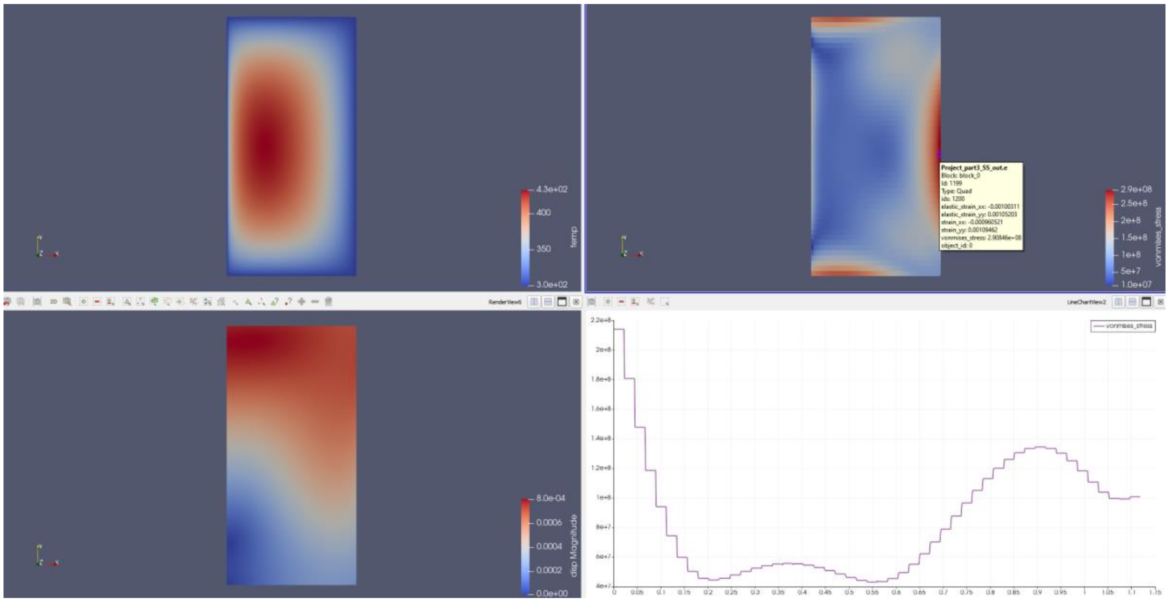

Figure 8: Part 3.1 SS Results

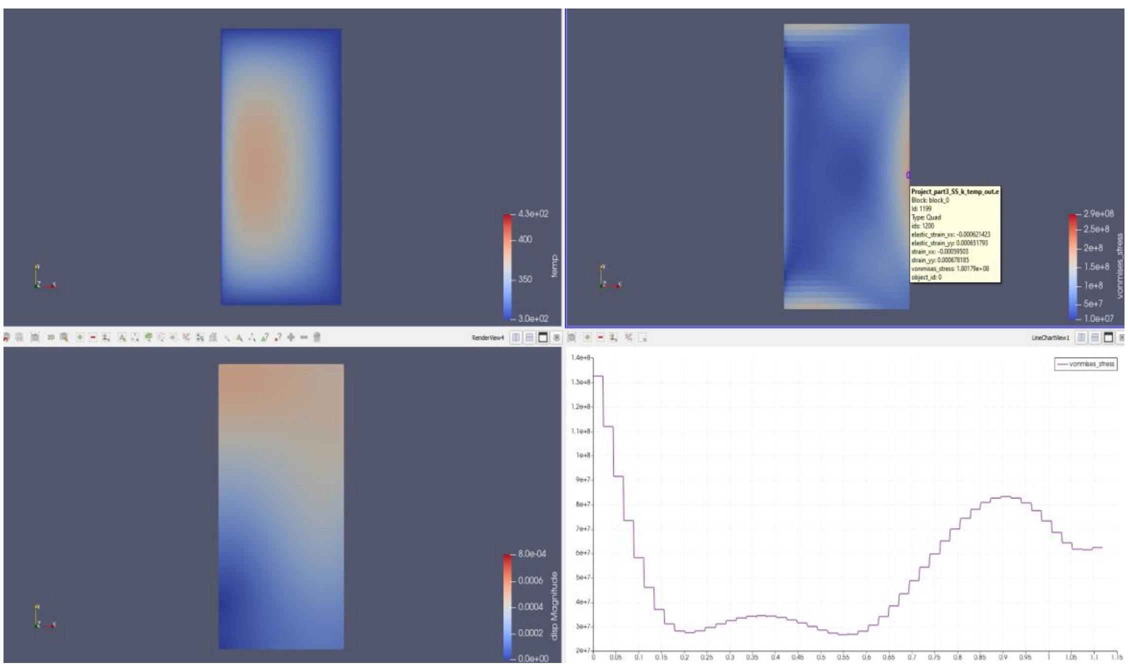

Figure 9: Part 3.2 SS K as a Function of Temp

Both Figure 8 and 9 were plotted on the same temperature scale which can be seen in top left of both images, same stress magnitude scales in the top right of the images, and same displacement scales in the bottom left of image. The result from Part 3.1 shows a higher max temperature of 430 K due to the constant CTE value and also a higher resulting Von Mises stress value of 291 MPa. This is compared to Part 3.2 max temperature of 381 K and a lower resulting Von Mises stress value of 180 MPa. The highest stresses in each case were in the expected ranges and were found to occur on the middle of the right edge, which is to be expected given the BCs of fixing the bottom and left edges.

Conclusion

Part 1 showed that if constant thermal conductivity is used in SS analysis, the analytical 1-D model matches well, however when K is a function of temperature, there will likely be a larger difference in results. All results consistently showed parabolic type temperature profile decreases in the fuel, large drops in temperature in the gas gap due to very low K, and slow gradual drop in the cladding temperatures. Additionally, the results showed that the assumed geometry has an effect on the results and that making assumptions, such as SS, axisymmetric, constant in Z, or constant CTE can impact results up to ~10-15 %. It also became clear that analytical hand calculations, while quick and efficient, should be taken as "ball-park" answers rather than utilized for design criteria or operational constraints in a reactor and when possible, numerical time (and functions of temperature) integration methods, e.g., Backwards Euler, should be used. However, there is still a good use case for analytical hand calcs as they provide a good sanity check against model results and provide a method of model correlation when test data does not exist. These results show that when able, modeling of physics should be performed as a function of time, temperature, and flow etc., and with a mesh size fine enough to not significantly impact results and capture all geometries. As a mesh becomes finer, and the more time steps utilized in transient analyses, the more accurate the result becomes; however equally increases computational time and model setup. This provides a justification for the existence and use of super computers/clusters for these analyses given the larger, 3D transient modeling of an entire reactor provides the most accurate results, especially when Multiphysics is used and coupling of physics is performed. Recommended forward work could include analyzing in 3-D, considering discrete fuel pellets with more realistic asymmetric geometries and deformation, allowing gap closures in the fuel-gap-clad stack, and coupling of more physics Modules to provide the interactions between heat transfer, mechanics, flow, and neutronics.

Appendix A: Scripts

Part 1.1 – "Project_part1_SS_finemesh.i"

```
[Kernels]
  [heatcond_fuel]
    type = HeatConduction
    block = 0
    variable = temp
  []
  [heatcond_cladding]
    type = HeatConduction
    block = 1
    variable = temp
  []
  [heatcond_gap]
    type = HeatConduction
    block = 2
    variable = temp
  []
  [fuel_LHR_SS]
    type = HeatSource
    variable = temp
    block = 0
    value = 150 # (W/cm)
  []
[]

[Mesh]
  [combined]
    type = GeneratedMeshGenerator
    dim = 2 # Dimension of mesh, 2-D
    nx = 301 # Number of elements in x-dir
    ny = 10 # Number of elements in y-dir
    xmax = 0.602 # (cm) width of the fuel, gap, and cladding
    ymax = 1 # (cm) length of the stack
  []
  [block1]
    type = SubdomainBoundingBoxGenerator
    input = combined
    bottom_left = '0.5 0 0'
    top_right = '0.602 1 0'
    block_id = 1
    block_name = 'non_fuel'
    location = INSIDE
  []
  [block2]
    type = SubdomainBoundingBoxGenerator
    input = block1
    bottom_left = '0.502 0 0'
    top_right = '0.602 1 0'
    block_id = 2
    block_name = 'clad'
    location = INSIDE
  []
[]
```

```
[Materials]
  [fuel_thermal_properties]
    type = GenericConstantMaterial
    prop_names = 'density thermal_conductivity heat_capacity'
    prop_values = '10.97         0.05                1'
    block = 0
  []
  [gap_thermal_properties]
    type = GenericConstantMaterial
    prop_names = 'density thermal_conductivity heat_capacity'
    prop_values = '0.001204      2.587e-4            1'
    block = 1
  []
  [clad_thermal_properties]
    type = GenericConstantMaterial
    prop_names = 'density thermal_conductivity heat_capacity'
    prop_values = '6.56          0.215                 1'
    block = 2
  []
[]

[Problem]
  coord_type = RZ # Axisymmetric RZ
[]

[Variables]
  [temp]
    #adding temp variable
  []
[]

[BCs]
  [cladding_temp]
    type = ADDirichletBC
    variable = temp # Variable to be set
    boundary = right
    value = 500 #  (K)
  []
[]

[Executioner]
  type = Steady # Steady state problem
  #solve_type = NEWTON # Perform a Newton solve
  #nl_rel_tol = 1e-8
  #nl_max_its = 20
  #l_max_its = 50
[]

[Outputs]
  exodus = true # Output Exodus format
[]
```

Part 1.2 – "Part1_Transient Script.i"

```
[Kernels]
  [heatcond_fuel]
    type = HeatConduction
    block = 0
    variable = temp
  []
  [heatcond_cladding]
    type = HeatConduction
    block = 1
    variable = temp
  []
  [heatcond_gap]
    type = HeatConduction
    block = 2
    variable = temp
  []
#  [fuel_LHR_SS]
#    type = HeatSource
#    variable = temp
#    block = 0
#    value = 150 # (W/cm)
#  []
  [heat_conduction_time_derivative_fuel]
    type = HeatConductionTimeDerivative
    variable = temp
    block = 0
#    density_name = density
    specific_heat = heat_capacity
  []
  [heat_conduction_time_derivative_cladding]
    type = HeatConductionTimeDerivative
    variable = temp
    block = 1
#    density_name = density
    specific_heat = heat_capacity
  []
  [heat_conduction_time_derivative_gap]
    type = HeatConductionTimeDerivative
    variable = temp
    block = 2
#    density_name = density
    specific_heat = heat_capacity
  []
  [fuel_LHR_transient]
    type = HeatSource
    function = LHR_fuel_transient_function
    variable = temp
    block = 0
  []
[]

[Mesh]
  [combined]
    type = GeneratedMeshGenerator
    dim = 2 # Dimension of mesh, 2-D
```

```
    nx = 301 # Number of elements in x-dir
    ny = 10 # Number of elements in y-dir
    xmax = 0.602 # (cm) width of the fuel, gap, and cladding
    ymax = 1 # (cm) length of the stack
  []
  [block1]
    type = SubdomainBoundingBoxGenerator
    input = combined
    bottom_left = '0.5 0 0'
    top_right = '0.602 1 0'
    block_id = 1
    block_name = 'non_fuel'
    location = INSIDE
  []
  [block2]
    type = SubdomainBoundingBoxGenerator
    input = block1
    bottom_left = '0.502 0 0'
    top_right = '0.602 1 0'
    block_id = 2
    block_name = 'clad'
    location = INSIDE
  []
[]

[Materials]
  [fuel_thermal_properties]
    type = GenericConstantMaterial
    prop_names = 'density thermal_conductivity heat_capacity'
    prop_values = '10.97         0.05                    1'
    block = 0
  []
  [gap_thermal_properties]
    type = GenericConstantMaterial
    prop_names = 'density thermal_conductivity heat_capacity'
    prop_values = '0.001204      2.587e-4              1'
    block = 1
  []
  [clad_thermal_properties]
    type = GenericConstantMaterial
    prop_names = 'density thermal_conductivity heat_capacity'
    prop_values = '6.56          0.215                    1'
    block = 2
  []
[]

[Problem]
  coord_type = RZ # Axisymmetric RZ
[]

[Variables]
  [temp]
    initial_condition = 500 #adding temp variable IC (K)
  []
[]

[Functions]
```

```
  [LHR_fuel_transient_function]
    type = ParsedFunction
    value = '150*(1-exp(-0.05*t))+150'
  []
[]

[BCs]
  [cladding_temp]
    type = DirichletBC
    variable = temp # Variable to be set
    boundary = right
    value = 500 #  (K)
  []
  [LHR_fuel_transient]
    type = ADFunctionDirichletBC
    variable = temp
    function = LHR_fuel_transient_function
    boundary = right
  []
[]

[Executioner]
  type = Transient # Transient problem
  dt = 1
  end_time = 100
  #automatic_scaling = true
  #solve_type = NEWTON # Perform a Newton solve
  #nl_rel_tol = 1e-8
  #nl_max_its = 20
  #l_max_its = 50
[]

[Outputs]
  exodus = true # Output Exodus format
  #csv = true #Output CSV
[]
```

Part 2.1 – "Part2__SS.i"

```
[Kernels]
  [heatcond_fuel]
    type = HeatConduction
    block = 0
    variable = temp
  []
  [heatcond_cladding]
    type = HeatConduction
    block = 1
    variable = temp
  []
  [heatcond_gap]
    type = HeatConduction
    block = 2
    variable = temp
  []
```

```
  [fuel_LHR_SS]
    type = HeatSource
    variable = temp
    block = 0
    value = 150 # (W/cm)
  []
[]

[Mesh]
  [combined]
    type = GeneratedMeshGenerator
    dim = 2 # Dimension of mesh, 2-D
    nx = 301 # Number of elements in x-dir
    ny = 10 # Number of elements in y-dir
    xmax = 0.602 # (cm) width of the fuel, gap, and cladding
    ymax = 1 # (cm) length of the stack
  []
  [block1]
    type = SubdomainBoundingBoxGenerator
    input = combined
    bottom_left = '0.5 0 0'
    top_right = '0.602 100 0'
    block_id = 1
    block_name = 'non_fuel'
    location = INSIDE
  []
  [block2]
    type = SubdomainBoundingBoxGenerator
    input = block1
    bottom_left = '0.502 0 0'
    top_right = '0.602 100 0'
    block_id = 2
    block_name = 'clad'
    location = INSIDE
  []
[]

[Materials]
  [fuel_thermal_properties]
    type = GenericConstantMaterial
    prop_names = 'density thermal_conductivity heat_capacity'
    prop_values = '10.97         0.05                    1'
    block = 0
  []
  [gap_thermal_properties]
    type = GenericConstantMaterial
    prop_names = 'density thermal_conductivity heat_capacity'
    prop_values = '0.001204      2.587e-4              1'
    block = 1
  []
  [clad_thermal_properties]
    type = GenericConstantMaterial
    prop_names = 'density thermal_conductivity heat_capacity'
    prop_values = '6.56          0.215                   1'
    block = 2
  []
[]
```

```
[Problem]
  coord_type = RZ # Axisymmetric RZ
#  rz_coord_axis = X # Which axis the symmetry is around
[]

[Variables]
  [temp]
    #adding temp variable
  []
#  [axial_distance]
#    #adding temp variable
#  []
[]

[Functions]
  [LHR_cool]
    type = ParsedFunction
    value = 5.95*(0.932+sin(1.2*(y/50-1.0)))+400
  []
#  [LHR_fuel_transient_function]
#    type = ParsedFunction
#    value = '150*(1-exp(-0.05*t))+150'
#  []
[]

[BCs]
  [cladding_temp]
    type = ADDirichletBC
    variable = temp # Variable to be set
    boundary = top
    value = 400 #  (K)
  []
  [LHR_clad]
    type = ADFunctionDirichletBC
    variable = temp
    function = LHR_cool
    boundary = right
  []
[]

[Executioner]
  type = Steady # Steady state problem
  #solve_type = NEWTON # Perform a Newton solve
  #nl_rel_tol = 1e-8
  #nl_max_its = 20
  #l_max_its = 50
[]

[Outputs]
  exodus = true # Output Exodus format
[]
```

Part 2.2 – "Part2_Transient.i"

```
[Kernels]
  [heatcond_fuel]
    type = HeatConduction
    block = 0
    variable = temp
  []
  [heatcond_cladding]
    type = HeatConduction
    block = 1
    variable = temp
  []
  [heatcond_gap]
    type = HeatConduction
    block = 2
    variable = temp
  []
#  [fuel_LHR_SS]
#    type = HeatSource
#    variable = temp
#    block = 0
#    value = 150 # (W/cm)
#  []
  [heat_conduction_time_derivative_fuel]
    type = HeatConductionTimeDerivative
    variable = temp
    block = 0
#    density_name = density
    specific_heat = heat_capacity
  []
  [heat_conduction_time_derivative_cladding]
    type = HeatConductionTimeDerivative
    variable = temp
    block = 1
#    density_name = density
    specific_heat = heat_capacity
  []
  [heat_conduction_time_derivative_gap]
    type = HeatConductionTimeDerivative
    variable = temp
    block = 2
#    density_name = density
    specific_heat = heat_capacity
  []
  [fuel_LHR_transient]
    type = HeatSource
    function = LHR_fuel_transient_function
    variable = temp
    block = 0
  []
[]

[Mesh]
  [combined]
    type = GeneratedMeshGenerator
```

```
    dim = 2 # Dimension of mesh, 2-D
    nx = 301 # Number of elements in x-dir
    ny = 10 # Number of elements in y-dir
    xmax = 0.602 # (cm) width of the fuel, gap, and cladding
    ymax = 1 # (cm) length of the stack
  []
  [block1]
    type = SubdomainBoundingBoxGenerator
    input = combined
    bottom_left = '0.5 0 0'
    top_right = '0.602 1 0'
    block_id = 1
    block_name = 'non_fuel'
    location = INSIDE
  []
  [block2]
    type = SubdomainBoundingBoxGenerator
    input = block1
    bottom_left = '0.502 0 0'
    top_right = '0.602 1 0'
    block_id = 2
    block_name = 'clad'
    location = INSIDE
  []
[]

[Materials]
  [fuel_thermal_properties]
    type = GenericConstantMaterial
    prop_names = 'density thermal_conductivity heat_capacity'
    prop_values = '10.97         0.05                1'
    block = 0
  []
  [gap_thermal_properties]
    type = GenericConstantMaterial
    prop_names = 'density thermal_conductivity heat_capacity'
    prop_values = '0.001204      2.587e-4           1'
    block = 1
  []
  [clad_thermal_properties]
    type = GenericConstantMaterial
    prop_names = 'density thermal_conductivity heat_capacity'
    prop_values = '6.56          0.215                1'
    block = 2
  []
[]

[Problem]
  coord_type = RZ # Axisymmetric RZ
[]

[Variables]
  [temp]
    initial_condition = 500 #adding temp variable IC (K)
  []
[]
```

```
[Functions]
  [LHR_fuel_transient_function]
    type = ParsedFunction
    value = '150*(1-exp(-0.05*t))+150'
  []
[]

[BCs]
  [cladding_temp]
    type = DirichletBC
    variable = temp # Variable to be set
    boundary = right
    value = 500 #  (K)
  []
  [LHR_fuel_transient]
    type = ADFunctionDirichletBC
    variable = temp
    function = LHR_fuel_transient_function
    boundary = right
  []
[]

[Executioner]
  type = Transient # Transient problem
  dt = 1
  end_time = 100
  #automatic_scaling = true
  #solve_type = NEWTON # Perform a Newton solve
  #nl_rel_tol = 1e-8
  #nl_max_its = 20
  #l_max_its = 50
[]

[Outputs]
  exodus = true # Output Exodus format
  #csv = true #Output CSV
[]
```

Part 3.1 - "Part3_SS.i"

```
[Kernels]
  [heatcond_fuel]
    type = HeatConduction
    variable = temp
  []
  [fuel_LHR_SS]
    type = HeatSource
    variable = temp
    value = 175 #  (W/cm)
  []
  [disp_x]
    type = Diffusion
    variable = disp_x
  []
  [disp_y]
    type = Diffusion
```

```
    variable = disp_y
  []
[]

[GlobalParams]
  displacements = 'disp_x disp_y'
[]

[Modules]
  [TensorMechanics/Master/All]
    add_variables = true
    strain = SMALL
    eigenstrain_names = thermal_expansion
    generate_output = 'vonmises_stress elastic_strain_xx elastic_strain_yy
strain_xx strain_yy'
  []
[]

[Mesh]
  [combined]
    type = GeneratedMeshGenerator
    dim = 2 # Dimension of mesh, 2-D
    nx = 50 # Number of elements in x-dir
    ny = 50 # Number of elements in y-dir
    xmax = 0.5 # (cm) width of the fuel, gap, and cladding
    ymax = 1 # (cm) length of the stack
  []
[]

[Materials]
  [fuel_thermal_properties]
    type = GenericConstantMaterial
    prop_names = 'density thermal_conductivity heat_capacity'
    prop_values = '10.97         0.05                 1'
  []
  [fuel_elasticity_tensor]
    type = ComputeIsotropicElasticityTensor
    youngs_modulus = 2.00e11 # (N/cm), 200 GPa, UO2
    poissons_ratio = 0.345
  []
  [fuel_stress]
    type = ComputeLinearElasticStress
  []
  [fuel_thermal_expansion]
    type = ComputeThermalExpansionEigenstrain
    thermal_expansion_coeff = 1.2e-5 #UO2 CTE (K^-1)
    stress_free_temperature = 300.0
    temperature = temp
    eigenstrain_name = thermal_expansion
  []
[]

[Problem]
  coord_type = RZ # Axisymmetric RZ
[]

[Variables]
```

```
  [temp]
    #initial_condition = 300
    #adding temp variable
  []
[]

[BCs]
  [left]
    type = ADDirichletBC
    boundary = 'left'
    variable = disp_x
    value = 0
  []
#   [top]
#     type = DirichletBC
#     boundary = 'top'
#     variable = disp_y
#     value = 0.0
#   []
#   [right]
#     type = DirichletBC
#     boundary = 'right'
#     variable = disp_x
#     value = 0.0
#   []
  [bottom]
    type = ADDirichletBC
    boundary = 'bottom'
    variable = disp_y
    value = 0
  []
  [fixed_surface_temp]
    type = ADDirichletBC
    boundary = 'left right top bottom'
    variable = temp
    value = 300
  []
[]

[Executioner]
  type = Steady # Steady state problem
  automatic_scaling = true
  #solve_type = linear # Perform a LINEAR solve
[]

[Outputs]
  exodus = true # Output Exodus format
[]
```

Part 3.2 – "Part3_SS_k_temp.i"

```
[Kernels]
  [heatcond_fuel]
    type = HeatConduction
    variable = temp
```

```
    #thermal_conductivity_temperature_function = K_function_of_temp #calls
function of K(Temp)
  []
  [fuel_LHR_SS]
    type = HeatSource
    variable = temp
    value = 175 #  (W/cm)
  []
  [disp_x]
    type = Diffusion
    variable = disp_x
  []
  [disp_y]
    type = Diffusion
    variable = disp_y
  []
[]

[GlobalParams]
  displacements = 'disp_x disp_y'
[]

[Modules]
  [TensorMechanics/Master/All]
    add_variables = true
    strain = SMALL
    eigenstrain_names = eigenstrain
    generate_output = 'vonmises_stress elastic_strain_xx elastic_strain_yy
strain_xx strain_yy'
  []
[]

[Mesh]
  [combined]
    type = GeneratedMeshGenerator
    dim = 2 # Dimension of mesh, 2-D
    nx = 50 # Number of elements in x-dir
    ny = 50 # Number of elements in y-dir
    xmax = 0.5 # (cm) width of the fuel, gap, and cladding
    ymax = 1 # (cm) length of the stack
  []
[]

[Materials]
  [fuel_density_HC_temp]
    type = GenericConstantMaterial
    prop_names = 'density heat_capacity'
    prop_values = '10.97       1'
  []
  [fuel_elasticity_tensor]
    type = ComputeIsotropicElasticityTensor
    youngs_modulus = 2.00e11 # (N/cm), 200 GPa, UO2
    poissons_ratio = 0.345
  []
  [fuel_stress]
    type = ComputeLinearElasticStress
  []
```

```
  [fuel_thermal_expansion]
    type = ComputeThermalExpansionEigenstrain
    thermal_expansion_coeff = 1.2e-5 #UO2 CTE (K^-1)
    stress_free_temperature = 300.0
    temperature = temp
    eigenstrain_name = eigenstrain
  []
  [fuel_k_function_temp]
    type = GenericFunctionMaterial
    prop_names = thermal_conductivity
    prop_values = K_function_of_temp
  []
[]

[Functions]
  [K_function_of_temp]
    type = ParsedFunction
    value = 1/(5.33+0.0235*temp)
    vars = temp
    vals = 300
  []
[]

[Problem]
  coord_type = RZ # Axisymmetric RZ
[]

[Variables]
  [temp]
    #initial_condition = 300
    #adding temp variable
  []
[]

[BCs]
  [left]
    type = ADDirichletBC
    boundary = 'left'
    variable = disp_x
    value = 0
  []
  [bottom]
    type = ADDirichletBC
    boundary = 'bottom'
    variable = disp_y
    value = 0
  []
  [fixed_surface_temp]
    type = ADDirichletBC
    boundary = 'left right top bottom'
    variable = temp
    value = 300
  []
[]

[Postprocessors]
  [K_min_man]
```

```
    type = FunctionValuePostprocessor
    function = K_function_of_temp
  []
[]

[Executioner]
  type = Steady # Steady state problem
  automatic_scaling = true
  #solve_type = linear # Perform a LINEAR solve
[]

[Outputs]
  exodus = true # Output Exodus format
[]
```