

AERO 430 HW 3

Judson Upchurch

2024-10-19

Table of Contents

1	Analytical Solution	4
1.1	Abstract	4
1.2	Derivation of General Solution	4
1.2.1	Conservation of Heat Flux	4
1.2.2	Newton's Law of Cooling	4
1.2.3	Fourier's Law of Conduction	4
1.2.4	General Solution	5
1.3	Interface Temperature Results	6
1.3.1	Interface Location 1: $\bar{x} = 0.5$	6
1.3.2	Interface Location 2: $\bar{x} = \frac{2}{\pi i}$	7
1.3.3	Discussion	8
1.4	Heat Convection Results	8
1.4.1	Interface Location 1: $\bar{x} = 0.5$	8
1.4.2	Interface Location 2: $\bar{x} = \frac{2}{\pi i}$	9
1.4.3	Discussion	10
2	FDM and FEM	10
2.1	Abstract	10
2.2	FDM Solution	10
2.3	FEM Solution	11
2.4	Interface Temperature Results	11
2.4.1	Interface Location 1: $\bar{x} = 0.5$	11
2.4.2	Interface Location 2: $\bar{x} = \frac{2}{\pi i}$	13
2.4.3	Discussion	15
2.5	Heat Convection Results	15
2.5.1	Interface Location 1: $\bar{x} = 0.5$	15
2.5.2	Interface Location 2: $\bar{x} = \frac{2}{\pi i}$	16
2.5.3	Discussion	17
3	FDM and FEM Convergence	17
3.1	Abstract	17
3.2	Interface Temperature Convergence	18
3.2.1	Interface Location 1: $\bar{x} = 0.5$	18
3.2.2	Interface Location 2: $\bar{x} = \frac{2}{\pi i}$	20
3.2.3	Discussion	21
3.3	Heat Convection Convergence	22
3.3.1	Interface Location 1: $\bar{x} = 0.5$	22
3.3.2	Interface Location 2: $\bar{x} = \frac{2}{\pi i}$	24
3.3.3	Discussion	25
4	Raw Code Output	26

5	Python Code	30
5.1	main.py	30
5.2	case1.py	46
5.3	Bar.py	51
5.4	common.py	51

1 Analytical Solution

1.1 Abstract

This section provides an analytical solution for the temperature distribution of a bimaterial beam connected at an interface point. The beam is constrained such that the environment is at 0 degrees, the right end of the beam is fixed at 100 degrees, and the left is fixed at 0 degrees.

In this report, the beam consists of two sections with the same area but different thermal conductivities. The affects of varying the ratio between the thermal conductivities will be analyzed as well as changing the interface location.

1.2 Derivation of General Solution

Let us first look at derivation for the general solution for temperature as a function along the rod. To begin, the governing equation will be conservation of heat flux.

1.2.1 Conservation of Heat Flux

In the case of a rod without internal heat generation, conservation of heat flux reduces to the total heat flux entering the system is equal and opposite to the heat flux leaving the system.

$$\dot{Q}_{out}(x) - \dot{Q}_{in}(x) = 0 \quad (1)$$

$\dot{Q}$ is the rate of heat transfer in energy / second units. We will use Joules/second, aka Watts.

1.2.2 Newton's Law of Cooling

Newton's Law of Cooling shows that the rate of heat transfer to the surroundings is proportional to the area of heat transfer and the temperature difference between the object and the surroundings.

$$\dot{Q}_{env}(x) = hA(T(x) - T_{amb}) \quad (2)$$

$\dot{Q}_{env}(x)$ is the rate of heat flux to the environment at a point along the rod in Joules/second. h is the heat transfer coefficient of the environment's fluid in $\frac{W}{m^2 \cdot ^\circ C}$. A is the area exposed to the surroundings in m^2 . $T(x)$ is the temperature of the rod at position x in $^\circ C$.

In both of our cases, the ambient temperature, T_{amb} is 0 $^\circ C$. This reduces the equation for Newton's Law of Cooling to:

$$\dot{Q}_{env}(x) = hAT(x) \quad (3)$$

1.2.3 Fourier's Law of Conduction

Fourier's Law of Conduction states that the rate of heat transfer through a material via conduction is proportional to the area of heat transfer and the negative temperature gradient.

$$\dot{Q}_c(x) = -kA \frac{dT(x)}{dx} \quad (4)$$

$\dot{Q}_c(x)$ is the rate of heat transfer via conduction through the material in Joules/second. k is the thermal conductivity of the rod in $\frac{W}{m^\circ C}$. A is the cross-sectional area of the rod in m^2 . $\frac{dT(x)}{dx}$ is the temperature gradient of the rod in $\frac{^\circ C}{m}$.

1.2.4 General Solution

If we look at an infinitesimal slice of the rod starting at position x and total length dx , we use conservation of heat flux (Equation 1) to get:

$$\dot{Q}(x+dx) - \dot{Q}(x) + \dot{Q}_{lat}(x;dx) = 0$$

From left to right, we have the heat flux leaving the right side of the rod slice, the heat flux entering the left side of the rod slice, and the heat flux leaving the rod slice from the outer circumference to the environment. Replacing $\dot{Q}_{lat}(x;dx)$ with $hPdxT(x)$ from Newton's Law of Cooling (Equation 3) gives:

$$\dot{Q}(x+dx) - \dot{Q}(x) + h(Pdx)T(x) = 0$$

Pdx is the area exposed to the surroundings for a given length dx where P is the perimeter. Dividing the equation by dx gives us:

$$\frac{\dot{Q}(x+dx) - \dot{Q}(x)}{dx} + hPT(x) = 0$$

Noting that $\frac{\dot{Q}(x+dx) - \dot{Q}(x)}{dx} = \dot{Q}'(x)$ as $dx \rightarrow 0$, the equation becomes:

$$\dot{Q}'(x) + hPT(x) = 0$$

Replacing $\dot{Q}(x)$ with $-kAT'(x)$ from Equation 4 gives:

$$-kAT''(x) + hPT(x) = 0$$

Rearranging and letting $\alpha = \sqrt{\frac{hP}{kA}}$ gives:

$$T''(x) - \alpha^2 T(x) = 0 \tag{5}$$

This will be the governing differential equation for both cases. Solving Equation 5 using the characteristic equation $r^2 - \alpha^2 = 0$ gives:

$$T(x) = Ae^{\alpha x} + Be^{-\alpha x}$$

Using the definitions $\sinh(\alpha x) = \frac{e^{\alpha x} - e^{-\alpha x}}{2}$ and $\cosh(\alpha x) = \frac{e^{\alpha x} + e^{-\alpha x}}{2}$, we get:

$$T(x) = C \sinh(\alpha x) + D \cosh(\alpha x) \tag{6}$$

Due to the interface in the bar, denoted at $\bar{x}$, the general solution is applied twice over each section with independent α_1 and α_2 .

$$\begin{aligned} T_1(x) &= C_1 \sinh(\alpha_1 x) + D_1 \cosh(\alpha_1 x) \quad \text{for } 0 \leq x \leq \bar{x} \\ T_2(x) &= C_2 \sinh(\alpha_2 x) + D_2 \cosh(\alpha_2 x) \quad \text{for } \bar{x} \leq x \leq L \end{aligned} \quad (7)$$

The constants C_1 , D_1 , C_2 , and D_2 must be solved simultaneously using the conditions that $T_1(0) = 0$, $T_1(L) = 100$, $T_1(\bar{x}) = T_2(\bar{x})$, and $-k_1 A_1 T_1'(\bar{x}) = k_2 A_2 T_2'(\bar{x})$.

1.3 Interface Temperature Results

1.3.1 Interface Location 1: $\bar{x} = 0.5$



Figure 1: Analytical Temperature Distribution for Varying $\frac{k_2}{k_1}$ with $\bar{x} = 0.5$

Table 1: Analytical Temperature Distribution for Varying $\frac{k_2}{k_1}$ with $\bar{x} = 0.5$

x	0.0625	0.125	0.25	0.5	$\frac{k_2}{k_1}$ 1.0	2.0	4.0	8.0	16.0
0.000	0	0	0	0	0	0	0	0	0
0.125	0.0117	0.0976	0.4648	1.471	3.440	6.352	9.653	12.57	14.65
0.250	0.0252	0.2107	1.003	3.175	7.425	13.71	20.83	27.13	31.63
0.375	0.0427	0.3572	1.700	5.381	12.59	23.24	35.31	45.98	53.60
0.500	0.0669	0.5602	2.667	8.439	19.74	36.44	55.38	72.11	84.07
0.625	0.8478	3.304	8.627	17.44	30.02	45.82	62.46	76.70	86.74
0.750	4.228	10.63	20.26	32.03	45.04	58.79	71.99	82.80	90.26
0.875	20.57	32.67	45.24	56.89	67.20	76.39	84.34	90.51	94.67
1.000	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0

1.3.2 Interface Location 2: $\bar{x} = \frac{2}{\pi}$

HW3/images/section1/x_bar_2/analytical_temperature_vs_position.png

Figure 2: Analytical Temperature Distribution for Varying $\frac{k_2}{k_1}$ with $\bar{x} = \frac{2}{\pi}$

Table 2: Analytical Temperature Distribution for Varying $\frac{k_2}{k_1}$ with $\bar{x} = \frac{2}{\pi}$

x	0.0625	0.125	0.25	0.5	$\frac{k_2}{k_1}$ 1.0	2.0	4.0	8.0	16.0
0.000	0	0	0	0	0	0	0	0	0
0.125	0.0433	0.2178	0.7238	1.775	3.440	5.484	7.435	8.922	9.877
0.250	0.0934	0.4700	1.562	3.831	7.425	11.84	16.05	19.26	21.32
0.375	0.158	0.7967	2.648	6.494	12.59	20.06	27.20	32.64	36.13
0.500	0.248	1.249	4.153	10.18	19.74	31.46	42.65	51.19	56.67
0.625	0.377	1.900	6.315	15.49	30.02	47.85	64.86	77.84	86.18
0.637	0.392	1.974	6.561	16.09	31.18	49.71	67.38	80.86	89.53
0.750	4.086	10.00	18.93	30.68	45.04	60.65	74.76	85.28	91.97
0.875	20.55	32.49	44.74	56.31	67.20	77.28	85.70	91.74	95.52
1.000	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0

1.3.3 Discussion

It can be seen from Figure 1 and Figure 2 that the section with lower thermal conductivity see a more gradual change in temperature compared to a higher k value. Additionally, comparing the two solutions with different interface locations shows a larger instantaneous temperature gradient at the interface.

1.4 Heat Convection Results

1.4.1 Interface Location 1: $\bar{x} = 0.5$

HW3/images/section1/x_bar_1/analytical_heat_convection_vs_k_ratio.png

Figure 3: Analytical Heat Convection for Varying $\frac{k_2}{k_1}$ with $\bar{x} = 0.5$

Table 3: Analytical Heat Convection for Varying $\frac{k_2}{k_1}$ with $\bar{x} = 0.5$

$\frac{k_2}{k_1}$	Heat Convection
0.0625	-1.242
0.125	-1.756
0.25	-2.487
0.5	-3.529
1.0	-4.985
2.0	-6.832
4.0	-8.797
8.0	-10.49
16.0	-11.68

1.4.2 Interface Location 2: $\bar{x} = \frac{2}{\pi}$

HW3/images/section1/x_bar_2/analytical_heat_convection_vs_k_ratio.png

Figure 4: Analytical Heat Convection for Varying $\frac{k_2}{k_1}$ with $\bar{x} = \frac{2}{\pi}$

Table 4: Analytical Heat Convection for Varying $\frac{k_2}{k_1}$ with $\bar{x} = \frac{2}{\pi}$

$\frac{k_2}{k_1}$	Heat Convection
0.0625	-1.242
0.125	-1.759
0.25	-2.501
0.5	-3.564
1.0	-4.985
2.0	-6.611
4.0	-8.119
8.0	-9.255
16.0	-9.981

1.4.3 Discussion

From Figure 3 and Figure 4, there is larger heat loss due to convection when the thermal conductivity of the right section of the beam becomes larger.

2 FDM and FEM

2.1 Abstract

If the analytical solution to the temperature distribution is not know, the finite difference method (FDM) and or the finite element method (FEM) can be applied to the bimaterial beam to find the nodal temperatures.

While this report follows two sections of constant area and differing thermal conductivities, the FDM and FEM are generalized such that varying areas can be accounted for.

2.2 FDM Solution

Using Equation 4 and Equation 1 applied at each node, the following matrix equation can be setup for a mesh of $N = 6$ sections which can be easily extrapolated for any system of N sections.

$$\begin{bmatrix} k_1 A_1 w_1 & -1 & 0 & 0 & 0 \\ -1 & k_1 A_1 w_1 & -1 & 0 & 0 \\ 0 & -\beta_1 & \rho & -\beta_2 & 0 \\ 0 & 0 & -1 & k_2 A_2 w_2 & -1 \\ 0 & 0 & 0 & -1 & k_2 A_2 w_2 \end{bmatrix} \begin{Bmatrix} T_2 \\ T_3 \\ T_4 \\ T_5 \\ T_6 \end{Bmatrix} = \begin{Bmatrix} T_1 = 0 \\ 0 \\ 0 \\ 0 \\ T_7 = 100 \end{Bmatrix} \quad (8)$$

Where:

$$w_1 = 2 + \alpha_1^2 \Delta x_1^2$$

$$w_2 = 2 + \alpha_2^2 \Delta x_2^2$$

$$\beta_1 = \frac{k_1 A_1}{\Delta x_1}$$

$$\beta_2 = \frac{k_2 A_2}{\Delta x_2}$$

$$\rho = \beta_1 + \beta_2 + \frac{k_1 A_1 \Delta x_1 \alpha_1^2}{2} + \frac{k_2 A_2 \Delta x_2 \alpha_2^2}{2}$$

2.3 FEM Solution

Using Equation 4 and Equation 1 applied at each node, the following matrix equation can be setup for a mesh of $N = 6$ sections which can be easily extrapolated for any system of N sections.

$$\begin{bmatrix} k_1 A_1 w_1 & -1 & 0 & 0 & 0 \\ -1 & k_1 A_1 w_1 & -1 & 0 & 0 \\ 0 & -\beta_1 & \rho & -\beta_2 & 0 \\ 0 & 0 & -1 & k_2 A_2 w_2 & -1 \\ 0 & 0 & 0 & -1 & k_2 A_2 w_2 \end{bmatrix} \begin{Bmatrix} T_2 \\ T_3 \\ T_4 \\ T_5 \\ T_6 \end{Bmatrix} = \begin{Bmatrix} T_1 = 0 \\ 0 \\ 0 \\ 0 \\ T_7 = 100 \end{Bmatrix} \quad (9)$$

Where:

$$w_1 = \frac{\frac{2}{\Delta x_1} + \frac{4\alpha_1^2 \Delta x_1}{6}}{\frac{1}{\Delta x_1} - \frac{\alpha_1^2 \Delta x_1}{6}}$$

$$w_2 = \frac{\frac{2}{\Delta x_2} + \frac{4\alpha_2^2 \Delta x_2}{6}}{\frac{1}{\Delta x_2} - \frac{\alpha_2^2 \Delta x_2}{6}}$$

$$\beta_1 = \frac{k_1 A_1}{\Delta x_1}$$

$$\beta_2 = \frac{k_2 A_2}{\Delta x_2}$$

$$\rho = \beta_1 + \beta_2 + \frac{k_1 A_1 \Delta x_1 \alpha_1^2}{2} + \frac{k_2 A_2 \Delta x_2 \alpha_2^2}{2}$$

2.4 Interface Temperature Results

2.4.1 Interface Location 1: $\bar{x} = 0.5$



Figure 5: FDM Temperature Distribution for Varying $\frac{k_2}{k_1}$ with $\bar{x} = 0.5$

Table 5: FDM Temperature Distribution for Varying $\frac{k_2}{k_1}$ with $\bar{x} = 0.5$

x	$\frac{k_2}{k_1}$								
	0.0625	0.125	0.25	0.5	1.0	2.0	4.0	8.0	16.0
0.000	0	0	0	0	0	0	0	0	0
0.125	0.0236	0.1312	0.5222	1.535	3.487	6.374	9.664	12.59	14.69
0.250	0.0508	0.2828	1.126	3.309	7.520	13.74	20.84	27.14	31.68
0.375	0.0860	0.4787	1.906	5.601	12.73	23.26	35.27	45.94	53.61
0.500	0.1346	0.7493	2.983	8.768	19.92	36.41	55.21	71.91	83.92
0.625	1.249	3.851	9.158	17.84	30.23	45.83	62.35	76.56	86.63
0.750	5.487	11.77	21.06	32.49	45.26	58.83	71.93	82.71	90.19
0.875	23.44	34.39	46.12	57.29	67.37	76.43	84.32	90.47	94.64
1.000	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0

HW3/images/section2/x_bar_1/fem_temperature_vs_position.png

Figure 6: FEM Temperature Distribution for Varying $\frac{k_2}{k_1}$ with $\bar{x} = 0.5$

Table 6: FEM Temperature Distribution for Varying $\frac{k_2}{k_1}$ with $\bar{x} = 0.5$

x	$\frac{k_2}{k_1}$								
	0.0625	0.125	0.25	0.5	1.0	2.0	4.0	8.0	16.0
0.000	0	0	0	0	0	0	0	0	0
0.125	0.00804	0.09242	0.4620	1.463	3.408	6.282	9.556	12.46	14.55
0.250	0.01737	0.1997	0.9980	3.161	7.363	13.57	20.65	26.93	31.44
0.375	0.02949	0.3389	1.694	5.366	12.50	23.04	35.05	45.71	53.37
0.500	0.04633	0.5326	2.662	8.432	19.64	36.20	55.07	71.82	83.87
0.625	0.4317	2.748	8.198	17.20	29.85	45.61	62.23	76.49	86.59
0.750	2.667	9.301	19.45	31.64	44.85	58.64	71.83	82.66	90.17
0.875	16.33	30.54	44.28	56.50	67.05	76.30	84.26	90.44	94.62
1.000	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0

2.4.2 Interface Location 2: $\bar{x} = \frac{2}{\pi}$

HW3/images/section2/x_bar_2/fdm_temperature_vs_position.png

Figure 7: FDM Temperature Distribution for Varying $\frac{k_2}{k_1}$ with $\bar{x} = \frac{2}{\pi}$

Table 7: FDM Temperature Distribution for Varying $\frac{k_2}{k_1}$ with $\bar{x} = \frac{2}{\pi}$

x	$\frac{k_2}{k_1}$								
	0.0625	0.125	0.25	0.5	1.0	2.0	4.0	8.0	16.0
0.000	0	0	0	0	0	0	0	0	0
0.159	0.0771	0.3197	0.9805	2.330	4.469	7.113	9.660	11.62	12.88
0.318	0.1736	0.7204	2.209	5.250	10.07	16.03	21.77	26.18	29.03
0.477	0.3142	1.304	3.998	9.501	18.22	29.00	39.39	47.37	52.53
0.637	0.5343	2.217	6.799	16.16	30.99	49.33	66.99	80.56	89.33
0.727	3.516	8.402	16.28	27.43	41.80	57.93	72.83	84.07	91.28
0.818	11.14	20.13	31.14	43.22	56.05	68.92	80.17	88.45	93.70
0.909	33.47	45.16	56.28	66.15	74.93	82.75	89.16	93.74	96.60
1.000	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0

HW3/images/section2/x_bar_2/fem_temperature_vs_position.png

Figure 8: FEM Temperature Distribution for Varying $\frac{k_2}{k_1}$ with $\bar{x} = \frac{2}{\pi}$

Table 8: FEM Temperature Distribution for Varying $\frac{k_2}{k_1}$ with $\bar{x} = \frac{2}{\pi}$

x	$\frac{k_2}{k_1}$								
	0.0625	0.125	0.25	0.5	1.0	2.0	4.0	8.0	16.0
0.000	0	0	0	0	0	0	0	0	0
0.159	0.0519	0.2759	0.9180	2.241	4.339	6.933	9.434	11.36	12.60
0.318	0.1175	0.6247	2.078	5.075	9.826	15.70	21.36	25.72	28.54
0.477	0.2141	1.139	3.789	9.251	17.91	28.62	38.94	46.89	52.03
0.637	0.3674	1.954	6.502	15.87	30.73	49.11	66.82	80.45	89.27
0.727	2.435	7.452	15.65	27.04	41.54	57.74	72.70	83.99	91.23
0.818	8.624	18.48	30.26	42.79	55.83	68.78	80.08	88.39	93.67
0.909	29.41	43.21	55.45	65.81	74.79	82.67	89.12	93.71	96.58
1.000	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0

2.4.3 Discussion

From the results in Figure 5, Figure 6, Figure 7, and Figure 8, both FDM and FEM showed similar results to the analytical temperature distribution in Figure 1 and Figure 2.

2.5 Heat Convection Results

2.5.1 Interface Location 1: $\bar{x} = 0.5$

HW3/images/section2/x_bar_1/heat_convection_vs_k_ratio.png

Figure 9: FDM and FEM Heat Convection for Varying $\frac{k_2}{k_1}$ with $\bar{x} = 0.5$

Table 9: FDM and FEM Heat Convection for Varying $\frac{k_2}{k_1}$ with $\bar{x} = 0.5$

$\frac{k_2}{k_1}$	FDM Q	FEM Q
0.0625	-1.58304	-1.36507
0.125	-2.01235	-1.84552
0.25	-2.67453	-2.54990
0.5	-3.66509	-3.57230
1.0	-5.08231	-5.01478
2.0	-6.90522	-6.86111
4.0	-8.86402	-8.84252
8.0	-10.5613	-10.5591
16.0	-11.7687	-11.7803

2.5.2 Interface Location 2: $\bar{x} = \frac{2}{\pi}$

HW3/images/section2/x_bar_2/heat_convection_vs_k_ratio.png

Figure 10: FDM and FEM Heat Convection for Varying $\frac{k_2}{k_1}$ with $\bar{x} = \frac{2}{\pi}$

Table 10: FDM and FEM Heat Convection for Varying $\frac{k_2}{k_1}$ with $\bar{x} = \frac{2}{\pi}$

$\frac{k_2}{k_1}$	FDM Q	FEM Q
0.0625	-1.43246	-1.30844
0.125	-1.89880	-1.80586
0.25	-2.60334	-2.53347
0.5	-3.63985	-3.58790
1.0	-5.04765	-5.01342
2.0	-6.67856	-6.66456
4.0	-8.20809	-8.21492
8.0	-9.37058	-9.39479
16.0	-10.1183	-10.1544

2.5.3 Discussion

Looking at the heat convection results in Figure 9 and Figure 10 compared to Figure 3 and Figure 4 show similar heat convection results at the end of the bar for both methods and interface locations.

3 FDM and FEM Convergence

3.1 Abstract

The convergence of the FDM and FEM are important quantities when verifying the method and predicting future performance for different setups. The rate of convergence of both methods will be calculated in reference to the exact quantity and the quantity found through Richardson Extrapolation.

The convergence of the interface temperature and the heat convection at the right end of the rod will be analyzed.

3.2 Interface Temperature Convergence

3.2.1 Interface Location 1: $\bar{x} = 0.5$

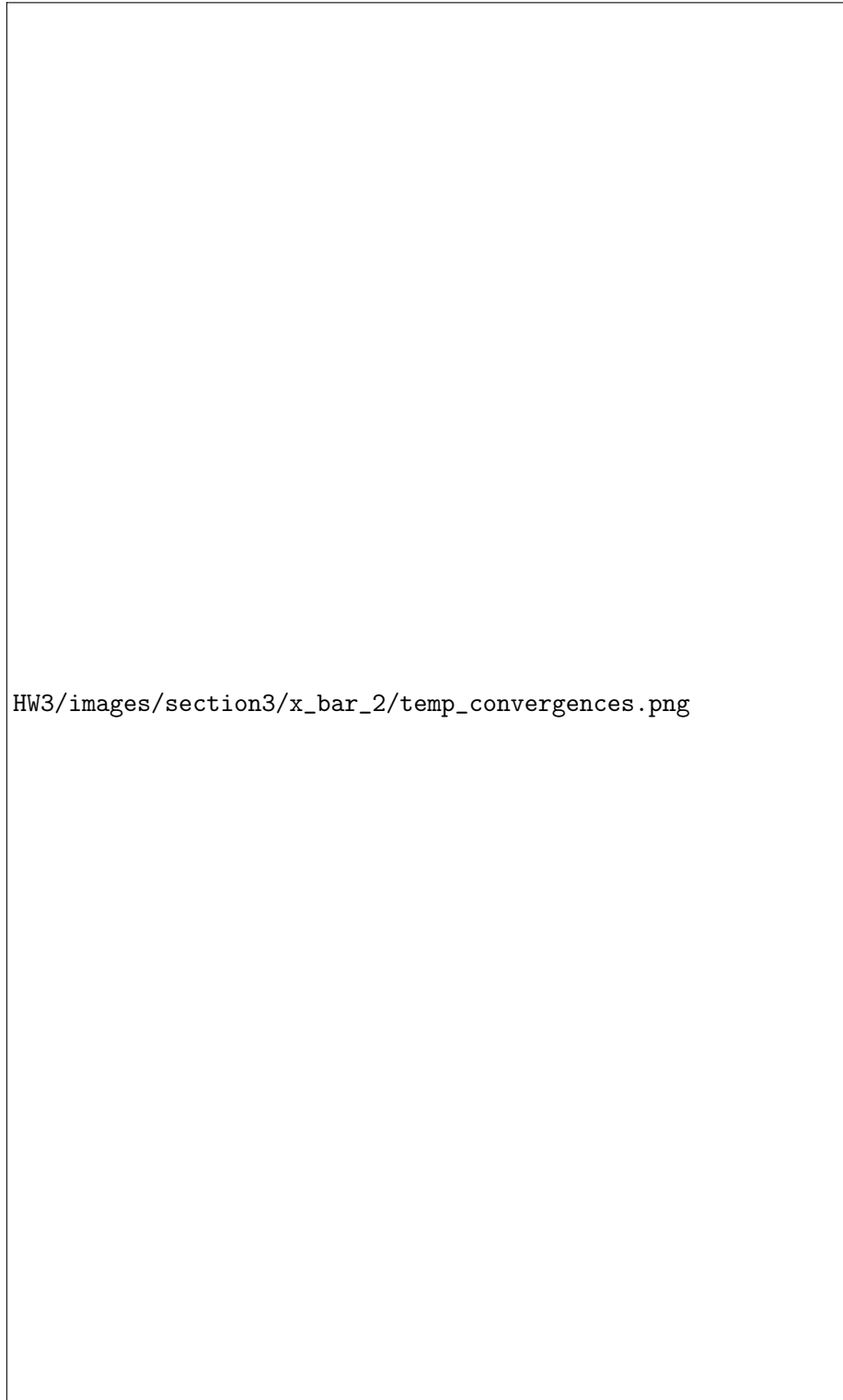


Figure 11: FDM and FEM Temperature Convergence for Varying $\frac{k_2}{k_1}$ with $\bar{x} = 0.5$

Table 11: FDM Temperature Convergence for $\frac{k_2}{k_1} = 4.0$ with $\bar{x} = 0.5$

N	True T	T	T_{EXTRAP}	%Error _{EX}	β_{EX}	%Error _{EXTRAP}	β_{EXTRAP}
4	55.3837	54.7385	N/A	0.01165	N/A	N/A	N/A
8	55.3837	55.2113	N/A	0.003114	1.90385	N/A	N/A
16	55.3837	55.3399	55.3879	0.000792	1.97491	0.000867	1.87879
32	55.3837	55.3727	55.3840	0.000199	1.99366	0.000204	1.96857
64	55.3837	55.3810	55.3838	0.000050	1.99841	0.000050	1.99207
128	55.3837	55.3831	55.3837	0.000012	1.99960	0.000012	1.99801

Table 12: FEM Temperature Convergence for $\frac{k_2}{k_1} = 4.0$ with $\bar{x} = 0.5$

N	True T	T	T_{EXTRAP}	%Error _{EX}	β_{EX}	%Error _{EXTRAP}	β_{EXTRAP}
4	55.3837	54.4102	N/A	0.01758	N/A	N/A	N/A
8	55.3837	55.0701	N/A	0.005663	1.63409	N/A	N/A
16	55.3837	55.2956	55.4126	0.001592	1.83095	0.002112	1.54918
32	55.3837	55.3604	55.3866	0.000421	1.91837	0.000472	1.79817
64	55.3837	55.3778	55.3841	0.000108	1.95986	0.000114	1.90373
128	55.3837	55.3822	55.3838	0.000027	1.98009	0.000028	1.95292

All other tables for their respective $\frac{k_2}{k_1}$ value are given in the Raw Code Output Appendix due to the sheer volume of data output.

3.2.2 Interface Location 2: $\bar{x} = \frac{2}{\pi}$

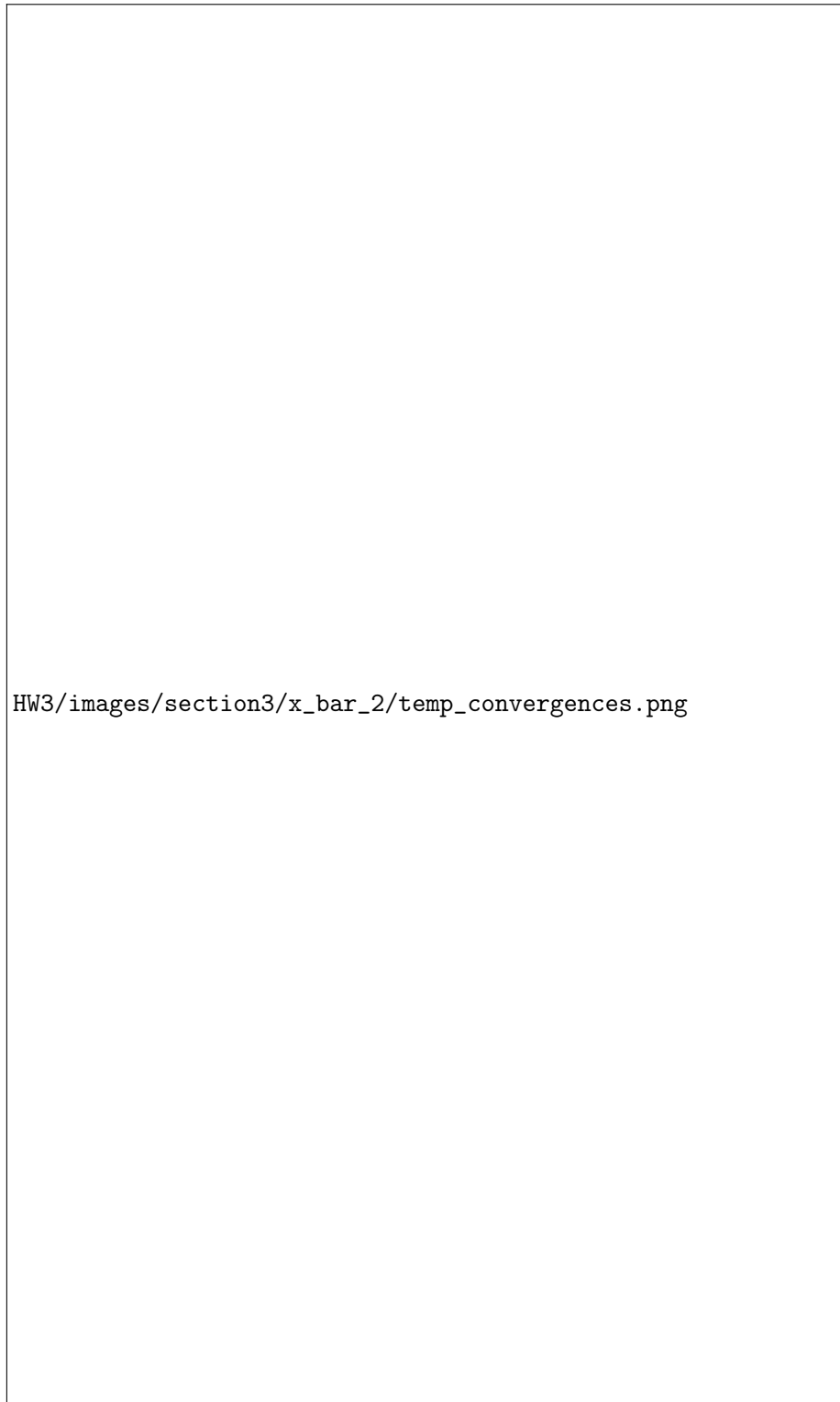


Figure 12: FDM and FEM Temperature Convergence for Varying $\frac{k_2}{k_1}$ with $\bar{x} = \frac{2}{\pi}$

Table 13: FDM Temperature Convergence for $\frac{k_2}{k_1} = 4.0$ with $\bar{x} = \frac{2}{\pi}$

N	True T	T	T_{EXTRAP}	%Error _{EX}	β_{EX}	%Error _{EXTRAP}	β_{EXTRAP}
4	67.3849	65.8951	N/A	0.022108	N/A	N/A	N/A
8	67.3849	66.9854	N/A	0.005928	1.89907	N/A	N/A
16	67.3849	67.2831	67.3949	0.001510	1.97291	0.001659	1.87294
32	67.3849	67.3593	67.3855	0.000379	1.99310	0.000389	1.96607
64	67.3849	67.3785	67.3849	0.000095	1.99827	0.000096	1.99137
128	67.3849	67.3833	67.3849	0.000024	1.99957	0.000024	1.99783

Table 14: FEM Temperature Convergence for $\frac{k_2}{k_1} = 4.0$ with $\bar{x} = \frac{2}{\pi}$

N	True T	T	T_{EXTRAP}	%Error _{EX}	β_{EX}	%Error _{EXTRAP}	β_{EXTRAP}
4	67.3849	65.5624	N/A	0.027046	N/A	N/A	N/A
8	67.3849	66.8226	N/A	0.008344	1.69667	N/A	N/A
16	67.3849	67.2284	67.4212	0.002321	1.84583	0.002859	1.63480
32	67.3849	67.3436	67.3892	0.000613	1.92198	0.000677	1.81752
64	67.3849	67.3743	67.3854	0.000157	1.96071	0.000165	1.90834
128	67.3849	67.3822	67.3849	0.000040	1.98028	0.000041	1.95401

All other tables for their respective $\frac{k_2}{k_1}$ value are given in the Raw Code Output Appendix due to the sheer volume of data output.

3.2.3 Discussion

For both methods, the rate of convergence for the interface temperature compared to the analytical and extrapolated solution was $\beta = 2$. Notably, for smaller values of $\frac{k_2}{k_1}$, the FEM convergence was non linear until the number of sections N surpassed 32. This occurred with both interface locations. As seen from looking at the temperature distributions in Figure 1 and Figure 2, the non uniform convergence values correspond to sections of the bar with much larger temperature gradients and therefore heat flux.

3.3 Heat Convection Convergence

3.3.1 Interface Location 1: $\bar{x} = 0.5$



Figure 13: FDM and FEM Heat Convection Convergence for Varying $\frac{k_2}{k_1}$ with $\bar{x} = 0.5$

Table 15: FDM Heat Convection Convergence for $\frac{k_2}{k_1} = 4.0$ with $\bar{x} = 0.5$

N	True Q	Q	Q_{EXTRAP}	%Error _{EX}	β_{EX}	%Error _{EXTRAP}	β_{EXTRAP}
4	-8.7970	-9.0603	N/A	0.029924	N/A	N/A	N/A
8	-8.7970	-8.8640	N/A	0.007612	1.97486	N/A	N/A
16	-8.7970	-8.8139	-8.7967	0.001912	1.99334	0.001956	1.96860
32	-8.7970	-8.8013	-8.7970	0.000479	1.99831	0.000481	1.99167
64	-8.7970	-8.7981	-8.7970	0.000120	1.99958	0.000120	1.99789
128	-8.7970	-8.7973	-8.7970	0.000030	1.99989	0.000030	1.99947

Table 16: FEM Heat Convection Convergence for $\frac{k_2}{k_1} = 4.0$ with $\bar{x} = 0.5$

N	True Q	Q	Q_{EXTRAP}	%Error _{EX}	β_{EX}	%Error _{EXTRAP}	β_{EXTRAP}
4	-8.7970	-8.9466	N/A	0.017004	N/A	N/A	N/A
8	-8.7970	-8.8425	N/A	0.005169	1.7180	N/A	N/A
16	-8.7970	-8.8095	-8.7942	0.001419	1.8653	0.001740	1.6581
32	-8.7970	-8.8003	-8.7967	0.000371	1.9340	0.000406	1.8401
64	-8.7970	-8.7979	-8.7970	0.000095	1.9673	0.000099	1.9224
128	-8.7970	-8.7973	-8.7970	0.000024	1.9838	0.000025	1.9617

All other tables for their respective $\frac{k_2}{k_1}$ value are given in the Raw Code Output Appendix due to the sheer volume of data output.

3.3.2 Interface Location 2: $\bar{x} = \frac{2}{\pi}$



Figure 14: FDM and FEM Heat Convection Convergence for Varying $\frac{k_2}{k_1}$ with $\bar{x} = \frac{2}{\pi}$

Table 17: FDM Heat Convection Convergence for $\frac{k_2}{k_1} = 4.0$ with $\bar{x} = \frac{2}{\pi}$

N	True Q	Q	Q_{EXTRAP}	%Error _{EX}	β_{EX}	%Error _{EXTRAP}	β_{EXTRAP}
4	-8.1187	-8.4608	N/A	0.042135	N/A	N/A	N/A
8	-8.1187	-8.2081	N/A	0.011009	1.93636	N/A	N/A
16	-8.1187	-8.1413	-8.1174	0.002786	1.98260	0.002953	1.92036
32	-8.1187	-8.1244	-8.1186	0.000699	1.99554	0.000710	1.97824
64	-8.1187	-8.1201	-8.1187	0.000175	1.99888	0.000175	1.99443
128	-8.1187	-8.1191	-8.1187	0.000044	1.99972	0.000044	1.99860

Table 18: FEM Heat Convection Convergence for $\frac{k_2}{k_1} = 4.0$ with $\bar{x} = \frac{2}{\pi}$

N	True Q	Q	Q_{EXTRAP}	%Error _{EX}	β_{EX}	%Error _{EXTRAP}	β_{EXTRAP}
4	-8.1187	-8.4335	N/A	0.038775	N/A	N/A	N/A
8	-8.1187	-8.2149	N/A	0.011850	1.71028	N/A	N/A
16	-8.1187	-8.1454	-8.1129	0.003283	1.85197	0.004001	1.65208
32	-8.1187	-8.1257	-8.1180	0.000864	1.92490	0.000951	1.82497
64	-8.1187	-8.1205	-8.1186	0.000222	1.96214	0.000233	1.91181
128	-8.1187	-8.1192	-8.1187	0.000056	1.98099	0.000058	1.95569

All other tables for their respective $\frac{k_2}{k_1}$ value are given in the Raw Code Output Appendix due to the sheer volume of data output.

3.3.3 Discussion

For both methods, the rate of convergence for the heat convection compared to the analytical and extrapolated solution was $\beta = 2$. Interestingly, the non linear behavior as seen in the temperature convergence is not reflected here.

4 Raw Code Output

```
Section 1: Analytical
Analytical Temperature Results x_bar = 0.5:
  x = 0.000  x = 0.125  x = 0.250  x = 0.375  x = 0.500  x = 0.500  x = 0.625  x = 0.750  x = 0.875  x = 1.000
k2/k1 = 0.0625  0.0  0.011667  0.025181  0.042681  0.066937  0.066937  0.847794  4.228179  20.573130  100.0
k2/k1 = 0.125  0.0  0.097641  0.210739  0.357195  0.560194  0.560194  3.303981  10.626282  32.674022  100.0
k2/k1 = 0.25  0.0  0.464798  1.003171  1.700341  2.666666  2.666666  8.626608  20.264910  45.242343  100.0
k2/k1 = 0.5  0.0  1.470842  3.174511  5.380689  8.438605  8.438605  17.437015  32.027881  56.890833  100.0
k2/k1 = 1.0  0.0  3.440413  7.425426  12.585848  19.738549  19.738549  30.015765  45.044332  67.203196  100.0
k2/k1 = 2.0  0.0  6.351689  13.708819  23.235988  36.441301  36.441301  45.815586  58.792577  76.392720  100.0
k2/k1 = 4.0  0.0  9.653341  20.834759  35.314219  55.383747  55.383747  62.464754  71.993744  84.344156  100.0
k2/k1 = 8.0  0.0  12.567937  27.125318  45.976504  72.105546  72.105546  76.701212  82.797389  90.513336  100.0
k2/k1 = 16.0  0.0  14.652607  31.625083  53.603455  84.066993  84.066993  86.741990  90.264767  94.669752  100.0

Analytical Temperature Results x_bar = 2/pi:
  x = 0.000  x = 0.125  x = 0.250  x = 0.375  x = 0.500  x = 0.625  x = 0.637  x = 0.750  x = 0.875  x = 1.000
k2/k1 = 0.0625  0.0  0.043265  0.093379  0.158275  0.248224  0.377466  0.392138  4.086217  20.545109  100.0
k2/k1 = 0.125  0.0  0.217770  0.470012  0.796654  1.249403  1.899926  1.973771  10.004385  32.490342  100.0
k2/k1 = 0.25  0.0  0.723845  1.562272  2.647999  4.152891  6.315166  6.560619  18.929061  44.739811  100.0
k2/k1 = 0.5  0.0  1.775153  3.831304  6.493931  10.184517  15.487261  16.089210  30.677503  56.308955  100.0
k2/k1 = 1.0  0.0  3.440413  7.425426  12.585848  19.738549  30.015765  31.182398  45.044332  67.203196  100.0
k2/k1 = 2.0  0.0  5.484175  11.836467  20.062415  31.464146  47.846498  49.706164  60.645947  77.284348  100.0
k2/k1 = 4.0  0.0  7.434698  16.046273  27.197894  42.654810  64.863775  67.384857  74.763593  85.702464  100.0
k2/k1 = 8.0  0.0  8.921798  19.255874  32.638061  51.186694  77.873932  80.863285  85.279652  91.742445  100.0
k2/k1 = 16.0  0.0  9.877475  21.318505  36.134152  56.669658  86.175696  89.525116  91.966805  95.516633  100.0

Analytical Heat Convection Results x_bar = 0.5:
  Heat Convection
k2/k1 = 0.0625  -1.241829
k2/k1 = 0.125  -1.756437
k2/k1 = 0.25  -2.486946
k2/k1 = 0.5  -3.529489
k2/k1 = 1.0  -4.985127
k2/k1 = 2.0  -6.832006
k2/k1 = 4.0  -8.797049
k2/k1 = 8.0  -10.486214
k2/k1 = 16.0  -11.680625

Analytical Heat Convection Results x_bar = 2/pi:
  Heat Convection
k2/k1 = 0.0625  -1.241978
k2/k1 = 0.125  -1.758799
k2/k1 = 0.25  -2.501202
k2/k1 = 0.5  -3.564213
k2/k1 = 1.0  -4.985127
k2/k1 = 2.0  -6.610807
k2/k1 = 4.0  -8.118713
k2/k1 = 8.0  -9.254592
k2/k1 = 16.0  -9.980639

Section 2: FDM and FEM with Varying k_ratio
FDM Temperature Results x_bar = 0.5:
  x = 0.000  x = 0.125  x = 0.250  x = 0.375  x = 0.500  x = 0.625  x = 0.750  x = 0.875  x = 1.000
k2/k1 = 0.0625  0.0  0.023566  0.050814  0.086002  0.134628  1.249213  5.486829  23.441518  100.0
k2/k1 = 0.125  0.0  0.131160  0.282813  0.478656  0.749289  3.850963  11.766340  34.389643  100.0
k2/k1 = 0.25  0.0  0.522237  1.126073  1.905858  2.983433  9.158380  21.057315  46.117073  100.0
k2/k1 = 0.5  0.0  1.534785  3.309379  5.601064  8.767916  17.841592  32.490766  57.293304  100.0
k2/k1 = 1.0  0.0  3.487383  7.519670  12.726906  19.922720  30.231460  45.263865  67.368749  100.0
k2/k1 = 2.0  0.0  6.373779  13.743460  23.260557  36.412116  45.832593  58.833740  76.431273  100.0
k2/k1 = 4.0  0.0  9.664493  20.839062  35.269736  55.211305  62.353389  71.931152  84.318726  100.0
k2/k1 = 8.0  0.0  12.587755  27.142347  45.937931  71.911316  76.562507  82.709060  90.471024  100.0
k2/k1 = 16.0  0.0  14.689868  31.675027  53.609410  83.920263  86.634225  90.194225  94.635027  100.0

FEM Temperature Results x_bar = 0.5:
  x = 0.000  x = 0.125  x = 0.250  x = 0.375  x = 0.500  x = 0.625  x = 0.750  x = 0.875  x = 1.000
k2/k1 = 0.0625  0.0  0.008040  0.017369  0.029485  0.046331  0.431694  2.667176  16.333414  100.0
k2/k1 = 0.125  0.0  0.092421  0.199668  0.338947  0.532604  2.747606  9.300941  30.539969  100.0
k2/k1 = 0.25  0.0  0.461952  0.988014  1.694185  2.662150  8.197855  19.452994  44.779989  100.0
k2/k1 = 0.5  0.0  1.463251  3.161248  5.366397  8.432466  17.199749  31.637279  56.504681  100.0
k2/k1 = 1.0  0.0  3.408346  7.363486  12.499933  19.641717  29.852520  44.852497  67.048062  100.0
k2/k1 = 2.0  0.0  6.282365  13.572595  23.040247  36.204196  45.614623  58.635706  76.298138  100.0
k2/k1 = 4.0  0.0  9.556088  20.645239  35.046460  55.070107  62.227195  71.830962  84.259013  100.0
k2/k1 = 8.0  0.0  12.463187  26.925816  45.708094  71.823222  76.490410  82.656431  90.442107  100.0
k2/k1 = 16.0  0.0  14.553527  31.441845  53.374310  83.869497  86.594484  90.166500  94.620483  100.0

FDM Temperature Results x_bar = 2/pi:
  x = 0.000  x = 0.159  x = 0.318  x = 0.477  x = 0.637  x = 0.727  x = 0.818  x = 0.909  x = 1.000
k2/k1 = 0.0625  0.0  0.077055  0.173629  0.314183  0.534322  3.515595  11.139042  33.471059  100.0
k2/k1 = 0.125  0.0  0.319719  0.720423  1.303613  2.217011  8.401972  20.134133  45.159372  100.0
k2/k1 = 0.25  0.0  0.980497  2.209358  3.997855  6.799021  16.282888  31.141948  56.281369  100.0
k2/k1 = 0.5  0.0  2.330072  5.250358  9.500576  16.157318  27.426136  43.221817  66.151538  100.0
k2/k1 = 1.0  0.0  4.469024  10.070065  18.221884  30.989359  41.796025  56.052044  74.933939  100.0
k2/k1 = 2.0  0.0  7.113289  16.028395  29.003540  49.325368  57.925784  68.916457  82.750907  100.0
k2/k1 = 4.0  0.0  9.660074  21.767073  39.387737  66.985431  72.825251  80.167607  89.163985  100.0
k2/k1 = 8.0  0.0  11.617645  26.178075  47.369498  80.559732  84.071349  88.450249  93.741604  100.0
k2/k1 = 16.0  0.0  12.883057  29.029430  52.529043  89.334419  91.281472  93.699357  96.600545  100.0

FEM Temperature Results x_bar = 2/pi:
  x = 0.000  x = 0.159  x = 0.318  x = 0.477  x = 0.637  x = 0.727  x = 0.818  x = 0.909  x = 1.000
k2/k1 = 0.0625  0.0  0.051867  0.117450  0.214096  0.367363  2.434575  8.623646  29.412987  100.0
k2/k1 = 0.125  0.0  0.275874  0.624708  1.138757  1.953971  7.451625  18.477330  43.210596  100.0
k2/k1 = 0.25  0.0  0.917965  2.078702  3.789190  6.501798  15.848464  30.261653  55.446231  100.0
k2/k1 = 0.5  0.0  2.241155  5.075024  9.251075  15.873740  27.039252  42.794017  65.812018  100.0
k2/k1 = 1.0  0.0  4.339198  9.825975  17.911409  30.733838  41.542855  55.828147  74.785094  100.0
k2/k1 = 2.0  0.0  6.933484  15.700652  28.620142  49.108745  57.744459  68.779447  82.672214  100.0
k2/k1 = 4.0  0.0  9.434441  21.363990  38.943632  66.822621  72.697960  80.078383  89.116689  100.0
k2/k1 = 8.0  0.0  11.388900  25.721867  46.887445  80.455325  83.990156  88.394995  93.713291  100.0
k2/k1 = 16.0  0.0  12.604001  28.541357  52.026991  89.272101  91.234400  93.667692  96.584539  100.0

FDM Heat Convection Results x_bar = 0.5:
  Heat Convection
k2/k1 = 0.0625  -1.583037
k2/k1 = 0.125  -2.012383
k2/k1 = 0.25  -2.674530
k2/k1 = 0.5  -3.665809
k2/k1 = 1.0  -5.082312
k2/k1 = 2.0  -6.905215
k2/k1 = 4.0  -8.864016
k2/k1 = 8.0  -10.561319
k2/k1 = 16.0  -11.768665

FEM Heat Convection Results x_bar = 0.5:
  Heat Convection
k2/k1 = 0.0625  -1.365085
k2/k1 = 0.125  -1.845516
k2/k1 = 0.25  -2.549900
k2/k1 = 0.5  -3.572301
k2/k1 = 1.0  -5.014775
k2/k1 = 2.0  -6.861111
k2/k1 = 4.0  -8.842519
k2/k1 = 8.0  -10.559111
k2/k1 = 16.0  -11.780305

FDM Heat Convection Results x_bar = 2/pi:
  Heat Convection
k2/k1 = 0.0625  -1.432483
k2/k1 = 0.125  -1.898803
k2/k1 = 0.25  -2.603336
k2/k1 = 0.5  -3.639854
k2/k1 = 1.0  -5.047653
k2/k1 = 2.0  -6.678556
k2/k1 = 4.0  -8.208090
k2/k1 = 8.0  -9.370579
k2/k1 = 16.0  -10.118258

FEM Heat Convection Results x_bar = 2/pi:
  Heat Convection
k2/k1 = 0.0625  -1.308439
k2/k1 = 0.125  -1.805860
```

k2/k1 = 0.25 -2.53474
k2/k1 = 0.5 -3.587897
k2/k1 = 1.0 -5.013420
k2/k1 = 2.0 -6.664558
k2/k1 = 4.0 -8.214918
k2/k1 = 8.0 -9.394792
k2/k1 = 16.0 -10.154415

Section 3: Convergence of FDM and FEM with Varying k Ratio

Convergence of FDM and FEM Temperature at x_bar = 0.500 and k_ratio = 0.0625

	True T	FDM T	FDM Extrap T	FDM Exact	% Error	FDM Exact B	FDM Extrap B	% Error	FDM Extrap B	FEM T	FEM Extrap T	FEM Exact	% Error	FEM Exact B	FEM Extrap B	% Error	FEM Extrap B
N = 4	0.066937	0.400229	Na	Na	4.979197	Na	Na	Na	Na	0.363773	Na	Na	6.434572	Na	Na	Na	Na
N = 8	0.066937	0.134623	Na	Na	0.111236	Na	Na	Na	Na	0.046331	Na	Na	0.307943	Na	Na	Na	Na
N = 16	0.066937	0.082528	0.069597	Na	0.231756	2.289743	0.183010	2.347809	0.062673	0.063352	0.063686	0.066675	0.010705	4.395578	0.010705	4.649734	0.011806
N = 32	0.066937	0.070718	0.067312	0.066482	2.036995	Na	Na	0.056888	0.056888	0.056888	0.056888	0.056888	0.015670	2.023150	0.015670	2.345888	0.015670
N = 64	0.066937	0.067876	0.066968	0.066968	0.014025	2.009682	0.013561	2.045896	0.066671	0.066671	0.066671	0.066671	0.003966	1.982240	0.003966	2.036753	0.003966
N = 128	0.066937	0.067171	0.066939	0.066939	0.003501	2.002449	0.003469	2.012080	0.066870	0.066870	0.066870	0.066870	0.001003	1.984061	0.001003	1.981623	0.001003

Convergence of FDM and FEM Heat Convection with x_bar = 0.500 and k_ratio = 0.0625

	True Q	FDM Q	FDM Extrap Q	FDM Exact	% Error	FDM Exact B	FDM Extrap B	% Error	FDM Extrap B	FEM Q	FEM Extrap Q	FEM Exact	% Error	FEM Exact B	FEM Extrap B	% Error	FEM Extrap B
N = 4	-1.241829	-2.323339	Na	Na	0.870901	[None]	Na	Na	-1.681631	Na	Na	Na	0.354157	[None]	Na	Na	Na
N = 8	-1.241829	-1.583037	Na	Na	0.274763	1.664323	Na	Na	-1.365065	Na	Na	Na	0.099238	1.835423	Na	Na	Na
N = 16	-1.241829	-1.353208	-1.210764	Na	0.075292	1.867617	0.102880	1.579472	-1.273756	-1.237562	-1.237562	-1.237562	0.025710	1.948582	0.025710	1.793664	0.025710
N = 32	-1.241829	-1.265851	-1.238768	Na	0.019344	1.960577	0.021863	1.834029	-1.249687	-1.241439	-1.241439	-1.241439	0.006489	1.98625	0.006489	1.936399	0.006489
N = 64	-1.241829	-1.247878	-1.241605	Na	0.004871	1.989623	0.005052	1.950668	-1.243848	-1.241803	-1.241803	-1.241803	0.001626	1.9965	0.001626	1.982806	0.001626
N = 128	-1.241829	-1.243344	-1.241614	Na	0.001220	1.997371	0.001232	1.987025	-1.242334	-1.241827	-1.241827	-1.241827	0.000407	1.999121	0.000407	1.995625	0.000407

Convergence of FDM and FEM Temperature at x_bar = 0.500 and k_ratio = 0.125

	True T	FDM T	FDM Extrap T	FDM Exact	% Error	FDM Exact B	FDM Extrap B	% Error	FDM Extrap B	FEM T	FEM Extrap T	FEM Exact	% Error	FEM Exact B	FEM Extrap B	% Error	FEM Extrap B
N = 4	0.560194	1.321586	Na	Na	1.359157	Na	Na	Na	Na	0.284007	Na	Na	0.493020	Na	Na	Na	Na
N = 8	0.560194	0.749289	Na	Na	0.337552	2.009531	Na	Na	0.532604	Na	Na	Na	0.049252	3.323401	Na	Na	Na
N = 16	0.560194	0.507101	0.560273	Na	0.083911	2.008170	0.083760	2.009981	0.553343	0.555230	0.555230	0.555230	0.012231	2.009667	0.012231	3.583391	0.012231
N = 32	0.560194	0.571925	0.560274	Na	0.020794	2.002601	0.020794	2.010016	0.558340	0.559926	0.559926	0.559926	0.003310	1.885500	0.003310	1.885500	0.003310
N = 64	0.560194	0.563125	0.560201	Na	0.005233	2.000690	0.005230	2.003327	0.559704	0.560217	0.560217	0.560217	0.000875	1.920215	0.000875	1.872828	0.000875
N = 128	0.560194	0.560927	0.560195	Na	0.001308	2.000175	0.001307	2.000862	0.560068	0.560200	0.560200	0.560200	0.000226	1.955459	0.000226	1.907766	0.000226

Convergence of FDM and FEM Heat Convection with x_bar = 0.500 and k_ratio = 0.125

	True Q	FDM Q	FDM Extrap Q	FDM Exact	% Error	FDM Exact B	FDM Extrap B	% Error	FDM Extrap B	FEM Q	FEM Extrap Q	FEM Exact	% Error	FEM Exact B	FEM Extrap B	% Error	FEM Extrap B
N = 4	-1.756437	-2.635211	Na	Na	0.500316	[None]	Na	Na	-2.090293	Na	Na	Na	0.190075	[None]	Na	Na	Na
N = 8	-1.756437	-2.012353	Na	Na	0.145701	1.779824	Na	Na	-1.845516	Na	Na	Na	0.050716	1.906073	Na	Na	Na
N = 16	-1.756437	-1.827802	-1.741900	Na	0.038341	1.926064	0.047006	1.723786	-1.779124	-1.754419	-1.754419	-1.754419	0.012916	1.972249	0.012916	1.982389	0.012916
N = 32	-1.756437	-1.773512	-1.755202	Na	0.009721	1.97963	0.010492	1.907407	-1.762136	-1.756295	-1.756295	-1.756295	0.003245	1.993049	0.003245	1.965455	0.003245
N = 64	-1.756437	-1.760722	-1.756356	Na	0.002439	1.99477	0.002486	1.974522	-1.757864	-1.756428	-1.756428	-1.756428	0.000812	1.998239	0.000812	1.991312	0.000812
N = 128	-1.756437	-1.757509	-1.756432	Na	0.000610	1.998684	0.000613	1.993462	-1.756794	-1.756437	-1.756437	-1.756437	0.000203	1.999555	0.000203	1.997799	0.000203

Convergence of FDM and FEM Temperature at x_bar = 0.500 and k_ratio = 0.25

	True T	FDM T	FDM Extrap T	FDM Exact	% Error	FDM Exact B	FDM Extrap B	% Error	FDM Extrap B	FEM T	FEM Extrap T	FEM Exact	% Error	FEM Exact B	FEM Extrap B	% Error	FEM Extrap B
N = 4	2.666666	3.862069	Na	Na	0.448276	Na	Na	Na	Na	2.715719	Na	Na	0.018395	Na	Na	Na	Na
N = 8	2.666666	2.983433	Na	Na	0.118788	1.916002	Na	Na	2.662150	Na	Na	Na	0.001694	3.441210	Na	Na	Na
N = 16	2.666666	2.747136	2.660209	Na	0.030178	1.976907	0.032877	1.894660	2.661948	2.661948	2.661948	2.661948	0.001807	-0.703302	6.420173e-07	7.420074	0.001807
N = 32	2.666666	2.686866	2.666231	Na	0.007575	1.994067	0.007739	1.971109	2.664898	2.662123	2.662123	2.662123	0.000663	1.446493	1.042592e-03	-3.337586	0.000663
N = 64	2.666666	2.671721	2.666638	Na	0.001896	1.998506	0.001906	1.992882	2.666147	2.667012	2.667012	2.667012	0.000195	1.787683	3.244563e-04	1.288577	0.000195
N = 128	2.666666	2.667794	2.666664	Na	0.000474	1.999626	0.000475	1.998132	2.666626	2.666692	2.666692	2.666692	0.000052	1.991986	6.204964e-05	1.718994	0.000052

Convergence of FDM and FEM Heat Convection with x_bar = 0.500 and k_ratio = 0.25

	True Q	FDM Q	FDM Extrap Q	FDM Exact	% Error	FDM Exact B	FDM Extrap B	% Error	FDM Extrap B	FEM Q	FEM Extrap Q	FEM Exact	% Error	FEM Exact B	FEM Extrap B	% Error	FEM Extrap B
N = 4	-2.486946	-3.171745	Na	Na	0.275357	[None]	Na	Na	-2.730060	Na	Na	Na	0.097756	[None]	Na	Na	Na
N = 8	-2.486946	-2.674530	Na	Na	0.075427	1.868146	Na	Na	-2.549900	Na	Na	Na	0.025314	1.949262	Na	Na	Na
N = 16	-2.486946	-2.535136	-2.480834	Na	0.019377	1.960724	0.021889	1.834707	-2.502844	-2.486207	-2.486207	-2.486207	0.006392	1.98551	0.006392	1.936807	0.006392
N = 32	-2.486946	-2.498080	-2.486500	Na	0.004879	1.989636	0.005050	1.950854	-2.490393	-2.486894	-2.486894	-2.486894	0.001602	1.996036	0.001602	1.981971	0.001602
N = 64	-2.486946	-2.488985	-2.486917	Na	0.001222	1.99738	0.001234	1.987073	-2.487943	-2.486943	-2.486943	-2.486943	0.000401	1.998875	0.000401	1.995088	0.000401
N = 128	-2.486946	-2.487706	-2.486944	Na	0.000306	1.999343	0.000306	1.996725	-2.487196	-2.486946	-2.486946	-2.486946	0.000100	1.999654	0.000100	1.998615	0.000100

Convergence of FDM and FEM Temperature at x_bar = 0.500 and k_ratio = 0.5

	True T	FDM T	FDM Extrap T	FDM Exact	% Error	FDM Exact B	FDM Extrap B	% Error	FDM Extrap B	FEM T	FEM Extrap T	FEM Exact	% Error	FEM Exact B	FEM Extrap B	% Error	FEM Extrap B
N = 4	8.438605	9.674633	Na	Na	0.146473	Na	Na	Na	Na	6.526238	Na	Na	0.025364	Na	Na	Na	Na
N = 8	8.438605	8.767916	Na	Na	0.039024	1.908190	Na	Na	8.432466	Na	Na	Na	0.000727	5.123736	Na	Na	Na
N = 16	8.438605	8.522370	8.431179	Na	0.009926	1.975035	0.010816	1.846459	8.429022	8.428967	8.428967	8.428967	0.001136	-0.642537	0.000006	5.998275	0.001136
N = 32	8.438605	8.455139	8.438114	Na	0.002433	1.983636	0.002551	1.968752	8.43145	8.431226	8.431226	8.431226	0.000410	1.469764	0.000465	-0.830074	0.000410
N = 64	8.438605	8.443869	8.438574	Na	0.000624	1.998394	0.000628	1.992015	8.437603	8.439252	8.439252	8.439252	0.000119	1.788181	0.000195	1.316727	0.000119
N = 128	8.438605	8.439921	8.438603	Na	0.000156	1.999598	0.000156	1.997993	8.438337	8.438650	8.438650	8.438650	0.000032	1.903464	0.000037	1.743720	0.000032

Convergence of FDM and FEM Heat Convection with x_bar = 0.500 and k_ratio = 0.5

	True Q	FDM Q	FDM Extrap Q	FDM Exact	% Error	FDM Exact B	FDM Extrap B	% Error	FDM Extrap B	FEM Q	FEM Extrap Q	FEM Exact	% Error	FEM Exact B	FEM Extrap B	% Error	FEM Extrap B
N = 4	-3.529489	-4.049425	Na	Na	0.146037	[None]	Na	Na	-3.695537	Na	Na	Na	0.047046	[None]	Na	Na	Na
N = 8	-3.529489	-3.665089	Na	Na	0.038419	1.926443	Na	Na	-3.572301	Na	Na	Na	0.012130	1.955519	Na	Na	Na
N = 16	-3.529489	-3.563689	-3.527097	Na	0.009741	1.979723	0.010426	1.907890	-3.540313	-3.529099	-3.529099	-3.529099	0.003067	1.983847	0.003178	1.945806	0.003067
N = 32	-3.529489	-3.531815	-3.524979	Na	0.000691	1.986303	0.000740	1.974839	-3.532007	-3.529486	-3.529486	-3.529486	0.000970	1.993661	0.000779	1.980542	0.000970
N = 64	-3.529489	-3.531648	-3.529479	Na	0.000512	1.996689	0.000514	1.993491	-3.530170	-3.529486	-3.529486	-3.529486	0.000193	1.997282	0.000194	1.992449	0.000193
N = 128	-3.529489	-3.530029	-3.529489	Na	0.000153	1.999672	0.000153	1.998362	-3.529660	-3.529489	-3.529489	-3.529489	0.000048	1.998756	0.000048	1.996790	0.000048

Convergence of FDM and FEM Temperature at x_bar = 0.500 and k_ratio = 1.0

convergence for FDM and FEM temperature at X_0bar = 0.500 and k_ratio = 1.0																	
	True T	FDM T	FDM Extrap T	FDM Exact	% Error	FDM Exact B	FDM Extrap B	% Error	FDM Extrap B	FEM T	FEM Extrap T	FEM Exact	% Error	FEM Exact B	FEM Extrap B	% Error	FEM Extrap B
N = 4	19.738549	20.447284	NaN	NaN	0.036906	NaN	NaN	NaN	NaN	19.679634	NaN	NaN	0.002985	NaN	NaN	NaN	NaN
N = 8	19.738549	19.922770	NaN	NaN	0.009331	1.944198	NaN	NaN	19.641717	NaN	NaN	NaN	0.004906	-0.716884	NaN	NaN	NaN
N = 16	19.738549	19.785060	19.736081	0.002356	1.985391	0.002482	1.930009	19.703350	19.665192	0.001783	0.002985	1.748651	0.000038	0.001783	0.001940	-0.700878	NaN
N = 32	19.738549	19.750206	19.738390	0.000591	1.996370	0.000599	1.981722	19.728328	19.745349	0.000518	0.000518	1.748651	0.000038	0.000518	0.000038	0.000038	1.303049
N = 64	19.738549	19.741485	19.738539	0.000148	1.999703	0.000148	1.990176	19.738539	19.738539	0.000136	0.000136	1.748651	0.000038	0.000136	0.000038	0.000038	1.700162
N = 128	19.738549	19.739278	19.738548	0.000033	1.999978	0.000033	1.998846	19.738542	19.738542	0.000036	0.000036	1.748651	0.000038	0.000036	0.000038	0.000038	1.834048

N = 32	-10.486214	-10.490944	-10.486176	0.000451	1.997766	0.000455	1.989096	-10.491393	-10.485787	0.000494	1.938138	0.000535	1.854587
N = 64	-10.486214	-10.487397	-10.486211	0.000113	1.999444	0.000113	1.997232	-10.487537	-10.488618	0.000126	1.969892	0.000131	1.927410
N = 128	-10.486214	-10.486510	-10.486214	0.000028	1.999861	0.000028	1.999305	-10.486548	-10.488207	0.000032	1.984445	0.000033	1.963687
Convergence of FDM and FEM Temperature at x_bar = 0.500 and k_ratio = 16.0													
N = 4	84.066993	83.499938	Na	0.006745	Na	Na	83.389648	Na	0.008057	Na	Na	Na	Na
N = 8	84.066993	83.920263	Na	0.001745	1.950326	Na	83.869497	Na	0.002349	1.778069	Na	Na	Na
N = 16	84.066993	84.029979	84.068733	0.000440	1.986999	0.000461	84.013509	84.075262	0.000636	1.884628	0.000735	1.736399	Na
N = 32	84.066993	84.057718	84.067105	0.000110	1.996770	0.000112	84.053063	84.068040	0.000166	1.940876	0.000178	1.864285	Na
N = 64	84.066993	84.064673	84.067000	0.000028	1.999175	0.000028	84.063437	84.067126	0.000042	1.970029	0.000044	1.930747	Na
N = 128	84.066993	84.066413	84.066994	0.000007	1.999794	0.000007	84.066095	84.067010	0.000011	1.984906	0.000011	1.964966	Na
Convergence of FDM and FEM Heat Convection with x_bar = 0.500 and k_ratio = 16.0													
N = 4	-11.680625	-12.024326	Na	0.029425	[None]	Na	-12.024123	Na	0.029407	[None]	Na	Na	Na
N = 8	-11.680625	-11.768665	Na	0.007537	1.664919	Na	-11.780305	Na	0.008534	1.784929	Na	Na	Na
N = 16	-11.680625	-11.702777	-11.679901	0.001896	1.990732	0.001959	-11.707559	-11.676625	0.002306	1.887865	0.002459	1.744873	Na
N = 32	-11.680625	-11.686172	-11.680475	0.000475	1.997649	0.000479	-11.687653	-11.680115	0.000600	1.942452	0.00644	1.868167	Na
N = 64	-11.680625	-11.682012	-11.680622	0.000119	1.999341	0.000119	-11.682413	-11.680560	0.000153	1.970807	0.000159	1.932611	Na
N = 128	-11.680625	-11.680972	-11.680625	0.000030	1.999852	0.000030	-11.681077	-11.680617	0.000039	1.985293	0.000039	1.965880	Na
Convergence of FDM and FEM Temperature at x_bar = 0.637 and k_ratio = 0.0625													
N = 4	0.392138	0.964920	Na	1.460667	Na	Na	0.149088	Na	0.619806	Na	Na	Na	Na
N = 8	0.392138	0.534322	Na	0.362587	2.010228	Na	0.387363	Na	0.063178	3.294328	Na	Na	Na
N = 16	0.392138	0.427445	0.392169	0.090051	2.009513	0.089965	0.210465	0.386709	0.015450	2.031788	0.004526	3.543820	Na
N = 32	0.392138	0.392947	0.392207	0.022465	2.003074	0.022294	0.211646	0.390525	0.004112	1.909892	0.03535	2.073563	Na
N = 64	0.392138	0.394339	0.392144	0.005613	2.000819	0.005597	0.203825	0.391716	0.001076	1.934512	0.011010	1.910667	Na
N = 128	0.392138	0.392688	0.392138	0.001403	2.000209	0.001402	0.201022	0.392029	0.000276	1.962964	0.000286	1.924564	Na
Convergence of FDM and FEM Heat Convection with x_bar = 0.637 and k_ratio = 0.0625													
N = 4	-1.241978	-1.892412	Na	0.523708	[None]	Na	-1.490265	Na	0.199912	[None]	Na	Na	Na
N = 8	-1.241978	-1.432463	Na	0.153372	1.771728	Na	-1.308439	Na	0.053512	1.901412	Na	Na	Na
N = 16	-1.241978	-1.292229	-1.230720	0.040461	1.922447	0.049978	-1.258921	-1.240389	0.013642	1.971793	0.014941	1.876530	Na
N = 32	-1.241978	-1.254718	-1.241914	0.010267	1.978528	0.010300	-1.246235	-1.241866	0.003428	1.992665	0.003818	1.964720	Na
N = 64	-1.241978	-1.245176	-1.241914	0.002677	1.994448	0.002628	-1.243044	-1.241971	0.000858	1.998145	0.000864	1.990831	Na
N = 128	-1.241978	-1.242779	-1.241974	0.000645	1.998612	0.000648	-1.242245	-1.241978	0.000215	1.999539	0.000215	1.997680	Na
Convergence of FDM and FEM Temperature at x_bar = 0.637 and k_ratio = 0.125													
N = 4	1.973771	2.884832	Na	0.461584	Na	Na	1.809387	Na	0.118822	Na	Na	Na	Na
N = 8	1.973771	2.217011	Na	0.128267	1.951566	Na	1.953971	Na	0.010032	2.236949	Na	Na	Na
N = 16	1.973771	2.035705	1.968139	0.031379	1.973570	0.034330	1.881035	1.965337	0.004172	1.265896	0.008546	-1.710453	Na
N = 32	1.973771	1.989328	1.973387	0.007882	1.993163	0.008078	1.968930	1.971213	0.001296	1.686822	0.002767	1.026937	Na
N = 64	1.973771	1.977665	1.973746	0.002973	1.998282	0.001985	1.991477	1.973958	0.000358	1.854433	0.000463	1.617231	Na
N = 128	1.973771	1.974744	1.973769	0.000493	1.999570	0.000494	1.997852	1.973585	0.000094	1.929744	0.000104	1.826653	Na
Convergence of FDM and FEM Heat Convection with x_bar = 0.637 and k_ratio = 0.125													
N = 4	-1.758799	-2.268096	Na	0.289571	[None]	Na	-1.940183	Na	0.103130	[None]	Na	Na	Na
N = 8	-1.758799	-1.898803	Na	0.079602	1.863034	Na	0.079602	Na	0.026757	1.946449	Na	Na	Na
N = 16	-1.758799	-1.794813	-1.754052	0.009254	1.973315	0.023238	1.828323	-1.770688	0.006760	1.984849	0.007096	1.933234	Na
N = 32	-1.758799	-1.767870	-1.758449	0.005158	1.989132	0.005358	1.848491	-1.761780	0.001695	1.995926	0.001718	1.981123	Na
N = 64	-1.758799	-1.761071	-1.758776	0.001292	1.997244	0.000586	1.964111	-1.758796	0.000424	1.998878	0.000443	1.994339	Na
N = 128	-1.758799	-1.759367	-1.758797	0.000323	1.999308	0.000324	1.996554	-1.758985	0.000106	1.999671	0.000106	1.998614	Na
Convergence of FDM and FEM Temperature at x_bar = 0.637 and k_ratio = 0.25													
N = 4	6.560619	7.452341	Na	0.135920	Na	Na	6.572238	Na	0.001771	Na	Na	Na	Na
N = 8	6.560619	6.799021	Na	0.036338	1.903201	Na	6.501798	Na	0.008966	-2.339890	Na	Na	Na
N = 16	6.560619	6.545951	6.545951	0.009254	1.973315	0.010127	6.537168	6.525344	0.003575	1.326578	0.001812	0.993864	Na
N = 32	6.560619	6.575870	6.560239	0.002325	1.993146	0.002383	6.566601	6.553589	0.001072	1.738076	0.002167	1.106969	Na
N = 64	6.560619	6.564437	6.560595	0.000582	1.998275	0.000586	6.558712	6.561034	0.000291	1.881813	0.000354	0.866666	Na
N = 128	6.560619	6.561574	6.560618	0.000146	1.999588	0.000146	6.560124	6.560619	0.000076	1.943682	0.000082	1.859435	Na
Convergence of FDM and FEM Heat Convection with x_bar = 0.637 and k_ratio = 0.25													
N = 4	-2.501202	-2.888923	Na	0.155014	[None]	Na	-2.626242	Na	0.049992	[None]	Na	Na	Na
N = 8	-2.501202	-2.603336	Na	0.040834	1.924555	Na	-2.533474	Na	0.012903	1.954046	Na	Na	Na
N = 16	-2.501202	-2.571038	-2.499354	0.010365	1.979071	0.011005	1.905549	-2.509362	0.003263	1.983522	0.003385	1.943932	Na
N = 32	-2.501202	-2.507703	-2.501075	0.002599	1.994616	0.002650	1.973826	-2.503251	0.000819	1.993664	0.000830	1.980114	Na
N = 64	-2.501202	-2.502828	-2.501194	0.000650	1.998644	0.000655	1.993205	-2.501715	0.000205	1.997315	0.000206	1.992400	Na
N = 128	-2.501202	-2.501608	-2.501201	0.000163	1.999666	0.000163	1.995305	-2.501330	0.000051	1.998785	0.000051	1.996825	Na
Convergence of FDM and FEM Temperature at x_bar = 0.637 and k_ratio = 0.5													
N = 4	16.08921	16.377482	Na	0.017917	Na	Na	15.648397	Na	0.027398	Na	Na	Na	Na
N = 8	16.08921	16.157318	Na	0.004233	2.081543	Na	15.873740	Na	0.013392	1.032880	Na	Na	Na
N = 16	16.08921	16.105900	16.090233	0.000974	2.028306	0.000974	16.020446	16.294141	0.000474	1.647775	0.016797	0.619196	Na
N = 32	16.08921	16.093365	16.089315	0.000258	2.007904	0.000251	16.070057	16.095406	0.001190	1.844072	0.001575	1.564197	Na
N = 64	16.08921	16.090246	16.089218	0.000064	2.002023	0.000064	16.084171	16.089784	0.000313	1.926462	0.000349	1.813482	Na
N = 128	16.08921	16.089469	16.089218	0.000016	2.000509	0.000016	16.087919	16.089273	0.000080	1.964289	0.000084	1.813195	Na
Convergence of FDM and FEM Heat Convection with x_bar = 0.637 and k_ratio = 0.5													
N = 4	-3.564213	-3.858649	Na	0.082609	[None]	Na	-3.651412	Na	0.024465	[None]	Na	Na	Na
N = 8	-3.564213	-3.639854	Na	0.021222	1.960716	Na	-3.587897	Na	0.006645	1.880339	Na	Na	Na
N = 16	-3.564213	-3.583260	-3.563514	0.005541	1.950868	0.005344	-3.563697	-3.564213	0.001728	1.942759	0.001874	1.857306	Na
N = 32	-3.564213	-3.568983	-3.564167	0.001338	1.997356	0.001351	-3.568980	-3.565783	0.000441	1.972821	0.000457	1.932522	Na
N = 64	-3.564213	-3.565406	-3.564210	0.000335	1.999336	0.000336	-3.566995	-3.564609	0.000128	1.963406	0.000113	1.967415	Na
N = 128	-3.564213	-3											

	True T	FDM T	FDM Extrapol T	FDM Exact	% Error	FDM Exact B	FDM Extrapol	% Error	FDM Extrapol B	FEM T	FEM Extrapol T	FEM Exact	% Error	FEM Exact B	FEM Extrapol	% Error	FEM Extrapol B
N = 4	80.863285	79.714831		NaN	0.014202	NaN		NaN	NaN	79.498914		NaN	0.016873	NaN		NaN	NaN
N = 8	80.863285	80.559732		NaN	0.003754	1.919673		NaN	NaN	80.453254		NaN	0.005071	1.734431		NaN	NaN
N = 16	80.863285	80.786251	80.869227		0.000953	1.978385		0.001026	1.899150	80.750296	80.884534		0.001397	1.859558		0.001660	1.683834
N = 32	80.863285	80.843953	80.863675		0.000239	1.994488		0.000244	1.972949	80.853578	80.866025		0.000367	1.927308		0.000401	1.834600
N = 64	80.863285	80.858447	80.863310		0.000060	1.998615		0.000060	1.993108	80.855665	80.863637		0.000094	1.962954		0.000099	1.914803
N = 128	80.863285	80.862075	80.863286		0.000015	1.999653		0.000015	1.998268	80.861355	80.863330		0.000024	1.981291		0.000024	1.956682

Convergence of FDM and FEM Heat Convection with x_bar = 0.637 and k_ratio = 8.0

	True Q	FDM Q	FDM Extrapol Q	FDM Exact	% Error	FDM Exact B	FDM Extrapol	% Error	FDM Extrapol B	FEM Q	FEM Extrapol Q	FEM Exact	% Error	FEM Exact B	FEM Extrapol	% Error	FEM Extrapol B
N = 4	-9.254592	-9.698320		NaN	0.047947	[None]		NaN	NaN	-9.722544		NaN	0.050564	[None]		NaN	NaN
N = 8	-9.254592	-9.370579		NaN	0.012533	1.935704		NaN	NaN	-9.394792		NaN	0.015149	1.738869		NaN	NaN
N = 16	-9.254592	-9.283942	-9.252810		0.003171	1.982521		0.003365	1.919493	-9.293169	-9.247500		0.004169	1.86165		0.004939	1.689379
N = 32	-9.254592	-9.261952	-9.254472		0.000795	1.995531		0.000898	1.978140	-9.264727	-9.253673		0.001095	1.928326		0.001195	1.837124
N = 64	-9.254592	-9.256435	-9.254584		0.000189	1.998876		0.000200	1.994413	-9.257151	-9.254473		0.000281	1.963456		0.000284	1.916010
N = 128	-9.254592	-9.255052	-9.254591		0.000050	1.999719		0.000050	1.998595	-9.255250	-9.254577		0.000071	1.98154		0.000073	1.957273

Convergence of FDM and FEM Temperature at x_bar = 0.637 and k_ratio = 16.0

	True T	FDM T	FDM Extrapol T	FDM Exact	% Error	FDM Exact B	FDM Extrapol	% Error	FDM Extrapol B	FEM T	FEM Extrapol T	FEM Exact	% Error	FEM Exact B	FEM Extrapol	% Error	FEM Extrapol B
N = 4	89.525116	88.797458		NaN	0.008128	NaN		NaN	NaN	88.671094		NaN	0.009539	NaN		NaN	NaN
N = 8	89.525116	89.334419		NaN	0.002130	1.931977		NaN	NaN	89.272101		NaN	0.002826	1.755051		NaN	NaN
N = 16	89.525116	89.476830	89.528233		0.000539	1.981624		0.000574	1.914748	89.465749	89.536659		0.000775	1.866912		0.000903	1.710436
N = 32	89.525116	89.513005	89.525323		0.000135	1.995309		0.000135	1.977014	89.506913	89.526672		0.000203	1.930093		0.000221	1.843749
N = 64	89.525116	89.520085	89.525129		0.000034	1.998821		0.000034	1.994135	89.520450	89.525321		0.000052	1.964098		0.000054	1.913185
N = 128	89.525116	89.524358	89.525116		0.000008	1.999705		0.000008	1.998526	89.523934	89.525142		0.000013	1.981797		0.000013	1.958049

Convergence of FDM and FEM Heat Convection with x_bar = 0.637 and k_ratio = 16.0

	True Q	FDM Q	FDM Extrapol Q	FDM Exact	% Error	FDM Exact B	FDM Extrapol	% Error	FDM Extrapol B	FEM Q	FEM Extrapol Q	FEM Exact	% Error	FEM Exact B	FEM Extrapol	% Error	FEM Extrapol B
N = 4	-9.980639	-10.508396		NaN	0.052878	[None]		NaN	NaN	-10.567911		NaN	0.058841	[None]		NaN	NaN
N = 8	-9.980639	-10.118258		NaN	0.013789	1.939192		NaN	NaN	-10.154415		NaN	0.017411	1.756795		NaN	NaN
N = 16	-9.980639	-10.015440	-9.978646		0.003487	1.983481		0.003687	1.923889	-10.028253	-9.972858		0.004771	1.867755		0.005555	1.712595
N = 32	-9.980639	-9.989364	-9.980505		0.000874	1.995777		0.000888	1.979342	-9.993130	-9.979579		0.001252	1.930509		0.001358	1.844764
N = 64	-9.980639	-9.982822	-9.980630		0.000219	1.998938		0.000220	1.994720	-9.983840	-9.980459		0.000321	1.964305		0.000335	1.918678
N = 128	-9.980639	-9.981185	-9.980638		0.000055	1.999734		0.000055	1.998673	-9.981449	-9.980621		0.000081	1.9819		0.000083	1.958292

5 Python Code

5.1 main.py

```
# main.py

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import math

import case1
import case2
import common
import Bar

bar1 = Bar.Bar(
    radius=0.1,
    k=0.5,
    h=0.25
)
bar2 = Bar.Bar(
    radius=0.1,
    k=2,
    h=0.25
)

data_dict = {}
data_dict["k_ratios"] = np.array([1/16, 1/8, 1/4, 1/2, 1, 2, 4, 8, 16])
data_dict["x_bar_values"] = np.array([1/2, 2/np.pi])
data_dict["length"] = 1

def section_1():
    '''Analytical solutions with varying k_ratio'''
    print("Section 1: Analytical")

    l = data_dict["length"]

    # Calculate the results for every k ratio for x_bar_1
    x_bar_1 = data_dict["x_bar_values"][0]

    data_dict["section1"] = {}
    data_dict["section1"]["x_bar_1"] = {}
    data_dict["section1"]["x_bar_1"]["x_values_fine"] = np.linspace(0, 1, 50)
    data_dict["section1"]["x_bar_1"]["x_values_fine"] =
        np.sort(np.append(data_dict["section1"]["x_bar_1"]["x_values_fine"], x_bar_1))
    data_dict["section1"]["x_bar_1"]["x_values_course"] = np.linspace(0, 1, 9)
    data_dict["section1"]["x_bar_1"]["x_values_course"] =
        np.sort(np.append(data_dict["section1"]["x_bar_1"]["x_values_course"], x_bar_1))
```

```

results = []
results_course = []
for k_ratio in data_dict["k_ratios"]:
    Bar.set_k_ratio(k_ratio, bar1, bar2) # Set the k value of bar2 = k*bar1
    result =
        case1.get_analytical_datapoints(data_dict["section1"]["x_bar_1"]["x_values_fine"],
                                         bar1=bar1,
                                         bar2=bar2,
                                         x_bar=x_bar_1,
                                         l=1)

    result_course =
        case1.get_analytical_datapoints(data_dict["section1"]["x_bar_1"]["x_values_course"],
                                         bar1=bar1,
                                         bar2=bar2,
                                         x_bar=x_bar_1,
                                         l=1)

    results.append(result)
    results_course.append(result_course)

data_dict["section1"]["x_bar_1"]["temp_results"] = np.array(results)

df_x_bar_1 = pd.DataFrame(results_course, index=[f'k2/k1 = {a}' for a in
    data_dict["k_ratios"]], columns=[f'x = {x:.3f}' for x in
    data_dict["section1"]["x_bar_1"]["x_values_course"]])

print("Analytical Temperature Results x_bar = 0.5:")
print(df_x_bar_1.to_string())
print("\n" * 2)

# Plotting x_bar_1
plt.figure(figsize=(10, 6))
for idx, k_ratio in enumerate(data_dict["k_ratios"]):
    plt.plot(data_dict["section1"]["x_bar_1"]["x_values_fine"],
             data_dict["section1"]["x_bar_1"]["temp_results"][idx], label=f'k2/k1 =
             {k_ratio}')
plt.axvline(x=x_bar_1, color='r', linestyle='--', label='x_bar')
plt.xlabel('Position (cm)')
plt.ylabel('Temperature (C)')
plt.legend()
plt.grid(True)
plt.savefig('images/section1/x_bar_1/analytical_temperature_vs_position.png', dpi=300)
plt.show()
plt.clf()

# Calculate the temperature results for every k ratio for x_bar_1
x_bar_2 = data_dict["x_bar_values"][1]

data_dict["section1"]["x_bar_2"] = {}
data_dict["section1"]["x_bar_2"]["x_values_fine"] = np.linspace(0, 1, 50)

```

```

data_dict["section1"]["x_bar_2"]["x_values_fine"] =
    np.sort(np.append(data_dict["section1"]["x_bar_2"]["x_values_fine"], x_bar_2))
data_dict["section1"]["x_bar_2"]["x_values_course"] = np.linspace(0, 1, 9)
data_dict["section1"]["x_bar_2"]["x_values_course"] =
    np.sort(np.append(data_dict["section1"]["x_bar_2"]["x_values_course"], x_bar_2))

results = []
results_course = []
for k_ratio in data_dict["k_ratios"]:
    Bar.set_k_ratio(k_ratio, bar1, bar2)
    result =
        case1.get_analytical_datapoints(data_dict["section1"]["x_bar_2"]["x_values_fine"],
                                         bar1=bar1,
                                         bar2=bar2,
                                         x_bar=x_bar_2,
                                         l=1)

    result_course =
        case1.get_analytical_datapoints(data_dict["section1"]["x_bar_2"]["x_values_course"],
                                         bar1=bar1,
                                         bar2=bar2,
                                         x_bar=x_bar_2,
                                         l=1)

    results.append(result)
    results_course.append(result_course)

data_dict["section1"]["x_bar_2"]["temp_results"] = np.array(results)

df_x_bar_2 = pd.DataFrame(results_course, index=[f'k2/k1 = {a}' for a in
    data_dict["k_ratios"]], columns=[f'x = {x:.3f}' for x in
    data_dict["section1"]["x_bar_2"]["x_values_course"]])

print("Analytical Temperature Results x_bar = 2/pi:")
print(df_x_bar_2.to_string())
print("\n" * 2)

# Plotting x_bar_2
plt.figure(figsize=(10, 6))
for idx, k_ratio in enumerate(data_dict["k_ratios"]):
    plt.plot(data_dict["section1"]["x_bar_2"]["x_values_fine"],
             data_dict["section1"]["x_bar_2"]["temp_results"][idx], label=f'k2/k1 =
             {k_ratio}')
plt.axvline(x=x_bar_2, color='r', linestyle='--', label='x_bar')
plt.xlabel('Position (cm)')
plt.ylabel('Temperature (C)')
plt.legend()
plt.grid(True)
plt.savefig('images/section1/x_bar_2/analytical_temperature_vs_position.png', dpi=300)
plt.show()
plt.clf()

```



```

# Calculate the analytical heat convection leaving the rod at x=l for varying k ratio

results_1 = []
results_2 = []
for k_ratio in data_dict["k_ratios"]:
    Bar.set_k_ratio(k_ratio, bar1, bar2)
    result_1 = case1.get_analytical_heat_convection(x=l,
                                                    bar1=bar1,
                                                    bar2=bar2,
                                                    x_bar=x_bar_1,
                                                    l=l)
    result_2 = case1.get_analytical_heat_convection(x=l,
                                                    bar1=bar1,
                                                    bar2=bar2,
                                                    x_bar=x_bar_2,
                                                    l=l)

    results_1.append([result_1])
    results_2.append([result_2])

data_dict["section1"]["x_bar_1"]["heat_results"] = np.array(results_1)
data_dict["section1"]["x_bar_2"]["heat_results"] = np.array(results_2)

df_x_bar_1 = pd.DataFrame(results_1, index=[f'k2/k1 = {a}' for a in
    data_dict["k_ratios"]], columns=["Heat Convection"])
df_x_bar_2 = pd.DataFrame(results_2, index=[f'k2/k1 = {a}' for a in
    data_dict["k_ratios"]], columns=["Heat Convection"])

print("Analytical Heat Convection Results x_bar = 0.5:")
print(df_x_bar_1.to_string())
print("\n" * 2)

print("Analytical Heat Convection Results x_bar = 2/pi:")
print(df_x_bar_2.to_string())
print("\n" * 2)

# Plotting x_bar_1 heat convection
plt.figure(figsize=(10, 6))
plt.plot(data_dict["k_ratios"], data_dict["section1"]["x_bar_1"]["heat_results"],
    label=f'Heat Convection')
plt.xlabel('k2/k1')
plt.ylabel('Q_Dot (W)')
plt.legend()
plt.grid(True)
plt.savefig('images/section1/x_bar_1/analytical_heat_convection_vs_k_ratio.png',
    dpi=300)
plt.show()
plt.clf()

```

```

# Plotting x_bar_2 heat convection
plt.figure(figsize=(10, 6))
plt.plot(data_dict["k_ratios"], data_dict["section1"]["x_bar_2"]["heat_results"],
         label=f'Heat Convection')
plt.xlabel('k2/k1')
plt.ylabel('Q_Dot (W)')
plt.legend()
plt.grid(True)
plt.savefig('images/section1/x_bar_2/analytical_heat_convection_vs_k_ratio.png',
          dpi=300)
plt.show()
plt.clf()

def section_2():
    '''FDM and FEM with varying k_ratio'''
    print("Section 2: FDM and FEM with Varying k_ratio")

    l = data_dict["length"]

    data_dict["section2"] = {}
    data_dict["section2"]["num_sections"] = 8

    # Calculate the results for every k ratio for x_bar_1
    x_bar_1 = data_dict["x_bar_values"][0]

    # For every k_ratio, calculate the FDM and FEM temperature results
    fdm_results = []
    fem_results = []
    x_fdm = []
    x_fem = []
    for k_ratio in data_dict["k_ratios"]:
        Bar.set_k_ratio(k_ratio, bar1, bar2) # Set the k value of bar2 = k*bar1
        fdm_temps, x_fdm = case1.solve(bar1=bar1,
                                       bar2=bar2,
                                       num_sections=data_dict["section2"]["num_sections"],
                                       x_bar=x_bar_1,
                                       l=l,
                                       method="FDM")
        fem_temps, x_fem = case1.solve(bar1=bar1,
                                       bar2=bar2,
                                       num_sections=data_dict["section2"]["num_sections"],
                                       x_bar=x_bar_1,
                                       l=l,
                                       method="FEM")

    fdm_results.append(fdm_temps)
    fem_results.append(fem_temps)

```

```

data_dict["section2"]["x_bar_1"] = {}
data_dict["section2"]["x_bar_1"]["x_values"] = x_fdm # fdm and fem use same x_values
data_dict["section2"]["x_bar_1"]["fdm_results"] = np.array(fdm_results)
data_dict["section2"]["x_bar_1"]["fem_results"] = np.array(fem_results)

df_fdm = pd.DataFrame(fdm_results, index=[f'k2/k1 = {a}' for a in
    data_dict["k_ratios"]], columns=[f'x = {x:.3f}' for x in x_fdm])
df_fem = pd.DataFrame(fem_results, index=[f'k2/k1 = {a}' for a in
    data_dict["k_ratios"]], columns=[f'x = {x:.3f}' for x in x_fem])

print("FDM Temperature Results x_bar = 0.5:")
print(df_fdm.to_string())
print("\n" * 2)
print("FEM Temperature Results x_bar = 0.5:")
print(df_fem.to_string())
print("\n" * 2)

# Now that the data is gathered for FDM and FEM, plot it
# Plotting x_bar_1, FDM
plt.figure(figsize=(10, 6))
for idx, k_ratio in enumerate(data_dict["k_ratios"]):
    plt.plot(data_dict["section2"]["x_bar_1"]["x_values"],
        data_dict["section2"]["x_bar_1"]["fdm_results"][idx], label=f'k2/k1 =
        {k_ratio}')
plt.axvline(x=x_bar_1, color='r', linestyle='--', label='x_bar')
plt.xlabel('Position (cm)')
plt.ylabel('Temperature (C)')
plt.legend()
plt.grid(True)
plt.savefig('images/section2/x_bar_1/fdm_temperature_vs_position.png', dpi=300)
#plt.show()
plt.clf()

# Plotting x_bar_1, FEM
plt.figure(figsize=(10, 6))
for idx, k_ratio in enumerate(data_dict["k_ratios"]):
    plt.plot(data_dict["section2"]["x_bar_1"]["x_values"],
        data_dict["section2"]["x_bar_1"]["fem_results"][idx], label=f'k2/k1 =
        {k_ratio}')
plt.axvline(x=x_bar_1, color='r', linestyle='--', label='x_bar')
plt.xlabel('Position (cm)')
plt.ylabel('Temperature (C)')
plt.legend()
plt.grid(True)
plt.savefig('images/section2/x_bar_1/fem_temperature_vs_position.png', dpi=300)
#plt.show()
plt.clf()

# Calculate the results for every k ratio for x_bar_2

```

```

x_bar_2 = data_dict["x_bar_values"][1]

# For every k_ratio, calculate the FDM and FEM temperature results
fdm_results = []
fem_results = []
x_fdm = []
x_fem = []
for k_ratio in data_dict["k_ratios"]:
    Bar.set_k_ratio(k_ratio, bar1, bar2) # Set the k value of bar2 = k*bar1
    fdm_temps, x_fdm = case1.solve(bar1=bar1,
                                   bar2=bar2,
                                   num_sections=data_dict["section2"]["num_sections"],
                                   x_bar=x_bar_2,
                                   l=1,
                                   method="FDM")
    fem_temps, x_fem = case1.solve(bar1=bar1,
                                   bar2=bar2,
                                   num_sections=data_dict["section2"]["num_sections"],
                                   x_bar=x_bar_2,
                                   l=1,
                                   method="FEM")

    fdm_results.append(fdm_temps)
    fem_results.append(fem_temps)

data_dict["section2"]["x_bar_2"] = {}
data_dict["section2"]["x_bar_2"]["x_values"] = x_fdm # fdm and fem use same x_values
data_dict["section2"]["x_bar_2"]["fdm_results"] = np.array(fdm_results)
data_dict["section2"]["x_bar_2"]["fem_results"] = np.array(fem_results)

df_fdm = pd.DataFrame(fdm_results, index=[f'k2/k1 = {a}' for a in
    data_dict["k_ratios"]], columns=[f'x = {x:.3f}' for x in x_fdm])
df_fem = pd.DataFrame(fem_results, index=[f'k2/k1 = {a}' for a in
    data_dict["k_ratios"]], columns=[f'x = {x:.3f}' for x in x_fem])

print("FDM Temperature Results x_bar = 2/pi:")
print(df_fdm.to_string())
print("\n" * 2)
print("FEM Temperature Results x_bar = 2/pi:")
print(df_fem.to_string())
print("\n" * 2)

# Plotting x_bar_2, FDM
plt.figure(figsize=(10, 6))
for idx, k_ratio in enumerate(data_dict["k_ratios"]):
    plt.plot(data_dict["section2"]["x_bar_2"]["x_values"],
             data_dict["section2"]["x_bar_2"]["fdm_results"][idx], label=f'k2/k1 = {k_ratio}')
plt.axvline(x=x_bar_2, color='r', linestyle='--', label='x_bar')
plt.xlabel('Position (cm)')

```

```

plt.ylabel('Temperature (C)')
plt.legend()
plt.grid(True)
plt.savefig('images/section2/x_bar_2/fdm_temperature_vs_position.png', dpi=300)
#plt.show()
plt.clf()

# Plotting x_bar_2, FEM
plt.figure(figsize=(10, 6))
for idx, k_ratio in enumerate(data_dict["k_ratios"]):
    plt.plot(data_dict["section2"]["x_bar_2"]["x_values"],
             data_dict["section2"]["x_bar_2"]["fem_results"][idx], label=f'k2/k1 = {k_ratio}')
plt.axvline(x=x_bar_2, color='r', linestyle='--', label='x_bar')
plt.xlabel('Position (cm)')
plt.ylabel('Temperature (C)')
plt.legend()
plt.grid(True)
plt.savefig('images/section2/x_bar_2/fem_temperature_vs_position.png', dpi=300)
#plt.show()
plt.clf()

# After calculating temperature values, extract the heat convection at x=1

# x_bar_1
# for every k ratio, calculate the heat convection from fdm and fem temp results
fdm_heat_results = []
fem_heat_results = []
for i, k_ratio in enumerate(data_dict["k_ratios"]):
    Bar.set_k_ratio(k_ratio, bar1, bar2) # Set the k value of bar2 = k*bar1
    dx = data_dict["section2"]["x_bar_1"]["x_values"][-1] -
          data_dict["section2"]["x_bar_1"]["x_values"][-2]
    (fdm_t0, fdm_t1) = data_dict["section2"]["x_bar_1"]["fdm_results"][i][-2:]
    fdm_heat_result = common.fdm_heat_extraction(fdm_t0, fdm_t1, dx, bar2, order=2)
    fdm_heat_results.append(fdm_heat_result)

    (fem_t0, fem_t1) = data_dict["section2"]["x_bar_1"]["fem_results"][i][-2:]
    fem_heat_result = common.fem_heat_extraction(fem_t0, fem_t1, dx, bar2)
    fem_heat_results.append(fem_heat_result)

data_dict["section2"]["x_bar_1"]["fdm_heat_results"] = np.array(fdm_heat_results)
data_dict["section2"]["x_bar_1"]["fem_heat_results"] = np.array(fem_heat_results)

df_fdm = pd.DataFrame(fdm_heat_results, index=[f'k2/k1 = {a}' for a in
        data_dict["k_ratios"]], columns=["Heat Convection"])
df_fem = pd.DataFrame(fem_heat_results, index=[f'k2/k1 = {a}' for a in
        data_dict["k_ratios"]], columns=["Heat Convection"])

```

```

print("FDM Heat Convection Results x_bar = 0.5:")
print(df_fdm.to_string())
print("\n" * 2)

print("FEM Heat Convection Results x_bar = 0.5:")
print(df_fem.to_string())
print("\n" * 2)

# x_bar_2
# for every k ratio, calculate the heat convection from fdm and fem temp results
fdm_heat_results = []
fem_heat_results = []
for i, k_ratio in enumerate(data_dict["k_ratios"]):
    Bar.set_k_ratio(k_ratio, bar1, bar2) # Set the k value of bar2 = k*bar1
    dx = data_dict["section2"]["x_bar_2"]["x_values"][-1] -
        data_dict["section2"]["x_bar_2"]["x_values"][-2]
    (fdm_t0, fdm_t1) = data_dict["section2"]["x_bar_2"]["fdm_results"][i][-2:]
    fdm_heat_result = common.fdm_heat_extraction(fdm_t0, fdm_t1, dx, bar2, order=2)
    fdm_heat_results.append(fdm_heat_result)

    (fem_t0, fem_t1) = data_dict["section2"]["x_bar_2"]["fem_results"][i][-2:]
    fem_heat_result = common.fem_heat_extraction(fem_t0, fem_t1, dx, bar2)
    fem_heat_results.append(fem_heat_result)

data_dict["section2"]["x_bar_2"]["fdm_heat_results"] = np.array(fdm_heat_results)
data_dict["section2"]["x_bar_2"]["fem_heat_results"] = np.array(fem_heat_results)

df_fdm = pd.DataFrame(fdm_heat_results, index=[f'k2/k1 = {a}' for a in
    data_dict["k_ratios"]], columns=["Heat Convection"])
df_fem = pd.DataFrame(fem_heat_results, index=[f'k2/k1 = {a}' for a in
    data_dict["k_ratios"]], columns=["Heat Convection"])

print("FDM Heat Convection Results x_bar = 2/pi:")
print(df_fdm.to_string())
print("\n" * 2)

print("FEM Heat Convection Results x_bar = 2/pi:")
print(df_fem.to_string())
print("\n" * 2)

# Plotting x_bar_1 heat convection
plt.figure(figsize=(10, 6))
plt.plot(data_dict["k_ratios"], data_dict["section2"]["x_bar_1"]["fdm_heat_results"],
    label=f'FDM Heat Convection')
plt.plot(data_dict["k_ratios"], data_dict["section2"]["x_bar_1"]["fem_heat_results"],
    label=f'FEM Heat Convection')
#plt.plot(data_dict["k_ratios"], data_dict["section1"]["x_bar_1"]["heat_results"],
    label=f'Analytical Heat Convection')

```

```

plt.xlabel('k2/k1')
plt.ylabel('Q_Dot (W)')
plt.legend()
plt.grid(True)
plt.savefig('images/section2/x_bar_1/heat_convection_vs_k_ratio.png', dpi=300)
#plt.show()
plt.clf()

# Plotting x_bar_2 heat convection
plt.figure(figsize=(10, 6))
plt.plot(data_dict["k_ratios"], data_dict["section2"]["x_bar_2"]["fdm_heat_results"],
         label=f'FDM Heat Convection')
plt.plot(data_dict["k_ratios"], data_dict["section2"]["x_bar_2"]["fem_heat_results"],
         label=f'FEM Heat Convection')
#plt.plot(data_dict["k_ratios"], data_dict["section1"]["x_bar_2"]["heat_results"],
         label=f'Analytical Heat Convection')
plt.xlabel('k2/k1')
plt.ylabel('Q_Dot (W)')
plt.legend()
plt.grid(True)
plt.savefig('images/section2/x_bar_2/heat_convection_vs_k_ratio.png', dpi=300)
#plt.show()
plt.clf()

def section_3():
    '''Convergence of FDM and FEM of interface temperature and heat convection'''
    print("Section 3: Convergence of FDM and FEM with Varying k Ratio")

    l = data_dict["length"]

    data_dict["section3"] = {}
    data_dict["section3"]["num_sections"] = [4, 8, 16, 32, 64, 128]

    for k, x_bar_1 in enumerate(data_dict["x_bar_values"]):
        # Calculate the number of rows needed for two columns of subplots
        num_k_ratios = len(data_dict["k_ratios"])
        num_rows = math.ceil(num_k_ratios / 2)

        # Define the figure for temperature convergence (two columns)
        fig_temp, axs_temp = plt.subplots(num_rows, 2, figsize=(15, 5 * num_rows))

        # Define the figure for heat convergence (two columns)
        fig_heat, axs_heat = plt.subplots(num_rows, 2, figsize=(15, 5 * num_rows))

        # Flatten the axs arrays for easier indexing
        axs_temp = axs_temp.flatten()
        axs_heat = axs_heat.flatten()

```

```

# For every k ratio, iterate through every number of points, calculate convergence
# (compared to analytical and extrapolated), print and add a subplot
for idx, k_ratio in enumerate(data_dict["k_ratios"]):
    Bar.set_k_ratio(k_ratio, bar1, bar2) # Set the k value of bar2 = k*bar1

    # Get the analytical data
    analy_interface_temp = case1.get_analytical_datapoints([x_bar_1], bar1, bar2,
        x_bar_1, 1)
    analy_heat = case1.get_analytical_heat_convection(1, bar1, bar2, x_bar_1, 1)

    # Setup data arrays
    fdm_interface_temps = []
    fem_interface_temps = []
    fdm_extrap_temps = [None, None]
    fem_extrap_temps = [None, None]
    fdm_heats = []
    fem_heats = []
    fdm_extrap_heats = [None, None]
    fem_extrap_heats = [None, None]
    fdm_interface_temp_errors = []
    fem_interface_temp_errors = []
    fdm_interface_temp_betas = [[None]]
    fem_interface_temp_betas = [[None]]
    fdm_heat_errors = []
    fem_heat_errors = []
    fdm_heat_betas = [[None]]
    fem_heat_betas = [[None]]
    fdm_interface_temp_extrap_errors = [None, None]
    fem_interface_temp_extrap_errors = [None, None]
    fdm_interface_temp_extrap_betas = [None, None]
    fem_interface_temp_extrap_betas = [None, None]
    fdm_heat_extrap_errors = [None, None]
    fem_heat_extrap_errors = [None, None]
    fdm_heat_extrap_betas = [None, None]
    fem_heat_extrap_betas = [None, None]

    # Get the fdm and fem data
    dx = 0
    for i, num_sections in enumerate(data_dict["section3"]["num_sections"]):
        fdm_temps, x_fdm =
            case1.solve(bar1=bar1, bar2=bar2, num_sections=num_sections, x_bar=x_bar_1, l=1, method="F")
        fem_temps, x_fem =
            case1.solve(bar1=bar1, bar2=bar2, num_sections=num_sections, x_bar=x_bar_1, l=1, method="F")

        # Get the interface temperature
        fdm_interface_temp = fdm_temps[num_sections // 2]
        fem_interface_temp = fem_temps[num_sections // 2]
        fdm_interface_temps.append(fdm_interface_temp)
        fem_interface_temps.append(fem_interface_temp)

    # Get the 2 last temperature to calculate heat convection

```



```

dx_prev = dx
dx = x_fdm[-1] - x_fdm[-2]
(fdm_t0, fdm_t1) = fdm_temps[-2:]
fdm_heat = common.fdm_heat_extraction(fdm_t0, fdm_t1, dx, bar2, order=2)
(fem_t0, fem_t1) = fem_temps[-2:]
fem_heat = common.fem_heat_extraction(fem_t0, fem_t1, dx, bar2)
fdm_heats.append(fdm_heat)
fem_heats.append(fem_heat)

# Calculated the Richardson extrpolution of interface temperature and heat
# convection
if i >= 2:
    fdm_extrap_temp = common.calc_extrapolated(fdm_interface_temps[-3],
        fdm_interface_temps[-2], fdm_interface_temps[-1])
    fem_extrap_temp = common.calc_extrapolated(fem_interface_temps[-3],
        fem_interface_temps[-2], fem_interface_temps[-1])
    fdm_extrap_heat = common.calc_extrapolated(fdm_heats[-3], fdm_heats[-2],
        fdm_heats[-1])
    fem_extrap_heat = common.calc_extrapolated(fem_heats[-3], fem_heats[-2],
        fem_heats[-1])
    fdm_extrap_temps.append(fdm_extrap_temp)
    fem_extrap_temps.append(fem_extrap_temp)
    fdm_extrap_heats.append(fdm_extrap_heat)
    fem_extrap_heats.append(fem_extrap_heat)

# Calculate the errors in reference to extrapolated values
fdm_interface_temp_extrap_error = common.calc_error(fdm_extrap_temp,
    fdm_interface_temp)
fem_interface_temp_extrap_error = common.calc_error(fem_extrap_temp,
    fem_interface_temp)
fdm_heat_extrap_error = common.calc_error(fdm_extrap_heat, fdm_heat)
fem_heat_extrap_error = common.calc_error(fem_extrap_heat, fem_heat)
fdm_interface_temp_extrap_errors.append(fdm_interface_temp_extrap_error)
fem_interface_temp_extrap_errors.append(fem_interface_temp_extrap_error)
fdm_heat_extrap_errors.append(fdm_heat_extrap_error)
fem_heat_extrap_errors.append(fem_heat_extrap_error)

# Calculate the betas in reference to extrapolated values
fdm_interface_temp_extrap_beta = common.calc_beta(fdm_extrap_temp,
    fdm_interface_temps[-1], fdm_interface_temps[-2], dx, dx_prev)
fem_interface_temp_extrap_beta = common.calc_beta(fem_extrap_temp,
    fem_interface_temps[-1], fem_interface_temps[-2], dx, dx_prev)
fdm_heat_extrap_beta = common.calc_beta(fdm_extrap_heat, fdm_heats[-1],
    fdm_heats[-2], dx, dx_prev)
fem_heat_extrap_beta = common.calc_beta(fem_extrap_heat, fem_heats[-1],
    fem_heats[-2], dx, dx_prev)
fdm_interface_temp_extrap_betas.append(fdm_interface_temp_extrap_beta)
fem_interface_temp_extrap_betas.append(fem_interface_temp_extrap_beta)
fdm_heat_extrap_betas.append(fdm_heat_extrap_beta)
fem_heat_extrap_betas.append(fem_heat_extrap_beta)

```

```

# Calculate the error and convergence rates for fdm temp, fem temp, fdm
heat, and fem heat
fdm_interface_temp_error = common.calc_error(analy_interface_temp,
      fdm_interface_temp)
fem_interface_temp_error = common.calc_error(analy_interface_temp,
      fem_interface_temp)
fdm_heat_error = common.calc_error(analy_heat, fdm_heat)
fem_heat_error = common.calc_error(analy_heat, fem_heat)
fdm_interface_temp_errors.append(fdm_interface_temp_error)
fem_interface_temp_errors.append(fem_interface_temp_error)
fdm_heat_errors.append(fdm_heat_error)
fem_heat_errors.append(fem_heat_error)

if i >= 1: # if i == 0 then we cannot calculate convergence
    fdm_interface_temp_beta = common.calc_beta(analy_interface_temp,
        fdm_interface_temps[-1], fdm_interface_temps[-2], dx, dx_prev)
    fem_interface_temp_beta = common.calc_beta(analy_interface_temp,
        fem_interface_temps[-1], fem_interface_temps[-2], dx, dx_prev)
    fdm_heat_beta = common.calc_beta(analy_heat, fdm_heats[-1],
        fdm_heats[-2], dx, dx_prev)
    fem_heat_beta = common.calc_beta(analy_heat, fem_heats[-1],
        fem_heats[-2], dx, dx_prev)
    fdm_interface_temp_betas.append(fdm_interface_temp_beta)
    fem_interface_temp_betas.append(fem_interface_temp_beta)
    fdm_heat_betas.append(fdm_heat_beta)
    fem_heat_betas.append(fem_heat_beta)

# Print the interface temps for this k ratio
table_data = []
for i in range(len(data_dict["section3"]["num_sections"])):
    table_data.append([
        analy_interface_temp[0], # True T

        fdm_interface_temps[i], # fdm result
        fdm_extrap_temps[i],
        fdm_interface_temp_errors[i][0], # fdm % error in reference to analytical
        fdm_interface_temp_betas[i][0], # fdm beta
        fdm_interface_temp_extrap_errors[i], # fdm % error in reference to
            extrapolated value
        fdm_interface_temp_extrap_betas[i], # fdm beta

        fem_interface_temps[i], # fem result
        fem_extrap_temps[i],
        fem_interface_temp_errors[i][0], # fem % error
        fem_interface_temp_betas[i][0], # fem beta
        fem_interface_temp_extrap_errors[i], # fem extrap % error
        fem_interface_temp_extrap_betas[i], # fem beta
    ])

```

```

    ])

columns = ['True T', 'FDM T', 'FDM Extrapol T', 'FDM Exact % Error', 'FDM Exact
          B', 'FDM Extrapol % Error', 'FDM Extrapol B', 'FEM T', 'FEM Extrapol T', 'FEM
          Exact % Error', 'FEM Exact B', 'FEM Extrapol % Error', 'FEM Extrapol B']
df = pd.DataFrame(table_data, index=[f'N = {i}' for i in
                                   data_dict["section3"]["num_sections"]], columns=columns)

print(f"Convergence of FDM and FEM Temperature at x_bar = {x_bar_1:.3f} and
      k_ratio = {k_ratio}")
print(df.to_string())
print("\n" * 2)

# Print the heat convection results for this k ratio
table_data = []
for i in range(len(data_dict["section3"]["num_sections"])):
    table_data.append([
        analy_heat, # True Q
        fdm_heats[i], # fdm result
        fdm_extrap_heats[i], # fdm extrapolated heat
        fdm_heat_errors[i], # fdm % error in reference to analytical
        fdm_heat_betas[i], # fdm beta in reference to analytical
        fdm_heat_extrap_errors[i], # fdm % error in reference to extrapolated
        value
        fdm_heat_extrap_betas[i], # fdm beta

        fem_heats[i], # fem result
        fem_extrap_heats[i], # fem extrapolated heat
        fem_heat_errors[i], # fem % error
        fem_heat_betas[i], # fem beta
        fem_heat_extrap_errors[i], # fem % error
        fem_heat_extrap_betas[i] # fem beta
    ])

columns = ['True Q', 'FDM Q', 'FDM Extrapol Q', 'FDM Exact % Error', 'FDM Exact
          B', 'FDM Extrapol % Error', 'FDM Extrapol B', 'FEM Q', 'FEM Extrapol Q', 'FEM
          Exact % Error', 'FEM Exact B', 'FEM Extrapol % Error', 'FEM Extrapol B']
df = pd.DataFrame(table_data, index=[f'N = {i}' for i in
                                   data_dict["section3"]["num_sections"]], columns=columns)

print(f"Convergence of FDM and FEM Heat Convection with x_bar = {x_bar_1:.3f}
      and k_ratio = {k_ratio}")
print(df.to_string())
print("\n" * 2)

# Add a subplot of the interface temp convergence for this k ratio
ax_temp = axs_temp[idx]

# Data to plot for temperature convergence
num_sections = data_dict["section3"]["num_sections"]

```

```

fdm_exact_errors = [e[0] for e in fdm_interface_temp_errors]
fem_exact_errors = [e[0] for e in fem_interface_temp_errors]
fdm_extrap_errors = fdm_interface_temp_extrap_errors
fem_extrap_errors = fem_interface_temp_extrap_errors

# Plotting temperature lines
ax_temp.plot(num_sections, fdm_exact_errors, label="FDM Exact")
ax_temp.plot(num_sections, fem_exact_errors, label="FEM Exact")
ax_temp.plot(num_sections, fdm_extrap_errors, label="FDM Extrap")
ax_temp.plot(num_sections, fem_extrap_errors, label="FEM Extrap")

ax_temp.set_xscale("log")
ax_temp.set_yscale("log")
ax_temp.set_xlabel("Number of Sections (N)")
ax_temp.set_ylabel("% Error")
ax_temp.set_title(f"k_ratio = {k_ratio}")
ax_temp.grid(True)
ax_temp.legend()

# Add a subplot of the heat convergence for this k ratio
ax_heat = axs_heat[idx]

# Data to plot for heat convergence
fdm_exact_heat_errors = fdm_heat_errors
fem_exact_heat_errors = fem_heat_errors
fdm_extrap_heat_errors = fdm_heat_extrap_errors
fem_extrap_heat_errors = fem_heat_extrap_errors

# Plotting heat lines
ax_heat.plot(num_sections, fdm_exact_heat_errors, label="FDM Exact")
ax_heat.plot(num_sections, fem_exact_heat_errors, label="FEM Exact")
ax_heat.plot(num_sections, fdm_extrap_heat_errors, label="FDM Extrap")
ax_heat.plot(num_sections, fem_extrap_heat_errors, label="FEM Extrap")

ax_heat.set_xscale("log")
ax_heat.set_yscale("log")
ax_heat.set_xlabel("Number of Sections (N)")
ax_heat.set_ylabel("% Error")
ax_heat.set_title(f"k_ratio = {k_ratio}")
ax_heat.grid(True)
ax_heat.legend()

# Hide any unused subplots (in case of an odd number of k_ratios)
for i in range(num_k_ratios, len(axs_temp)):
    fig_temp.delaxes(axs_temp[i])
    fig_heat.delaxes(axs_heat[i])

# Adjust layout for better spacing
fig_temp.tight_layout(rect=[0, 0.03, 1, 0.95])
fig_heat.tight_layout(rect=[0, 0.03, 1, 0.95])

```

```

    # Save the plots
    if k == 0:
        fig_temp.savefig("images/section3/x_bar_1/temp_convergences.png", dpi=300)
        fig_heat.savefig("images/section3/x_bar_1/heat_convergences.png", dpi=300)
    else:
        fig_temp.savefig("images/section3/x_bar_2/temp_convergences.png", dpi=300)
        fig_heat.savefig("images/section3/x_bar_2/heat_convergences.png", dpi=300)

def main():
    # Make directories for plots
    import os
    import shutil

    base_dir = "images"
    sub_dirs = ["section1", "section2", "section3"]
    nested_dirs = ["x_bar_1", "x_bar_2"]

    # Create the base directory if it doesn't exist
    if not os.path.exists(base_dir):
        os.mkdir(base_dir)

    # Create or clear subdirectories and their nested directories
    for sub_dir in sub_dirs:
        section_path = os.path.join(base_dir, sub_dir)
        if os.path.exists(section_path):
            # Remove all contents of the directory
            shutil.rmtree(section_path)
        # Create the section directory
        os.makedirs(section_path)

        # Create nested directories within each section
        for nested_dir in nested_dirs:
            nested_path = os.path.join(section_path, nested_dir)
            os.makedirs(nested_path)

    # import sys

    # # Redirect all print statements to a file
    # sys.stdout = open("output.txt", "w", encoding="utf-8")

    section_1() # Analytical solutions
    section_2() # FDM and FEM temperature and heat convection
    section_3() # Convergence of FDM and FEM temperature and heat convection

    # # Restore original stdout and close the file
    # sys.stdout.close()
    # sys.stdout = sys.__stdout__

```

```
if __name__ == "__main__":
    main()
```

5.2 case1.py

```
# case1.py

import numpy as np
import matplotlib.pyplot as plt

from Bar import Bar, set_k_ratio

def get_analytical_constants(bar1:'Bar', bar2:'Bar', x_bar=2/np.pi, l=1):
    '''Solve for C1, D1, C2, and D2
    Returns np.array([C1, D1, C2, D2])'''
    epsilon = (bar1.k*bar1.area*bar1.alpha) / (bar2.k*bar2.area*bar2.alpha)
    A = np.array([
        #C1                D1                C2                D2
        [0,                1,                0,                0],
        [0,                0,                np.sinh(bar2.alpha*l),
         np.cosh(bar2.alpha*l)],
        [np.sinh(bar1.alpha*x_bar), 0, -1*np.sinh(bar2.alpha*x_bar),
         -1*np.cosh(bar2.alpha*x_bar)],
        [epsilon*np.cosh(bar1.alpha*x_bar), 0, -1*np.cosh(bar2.alpha*x_bar),
         -1*np.sinh(bar2.alpha*x_bar)]
    ])

    B = np.array([
        0,
        100,
        0,
        0
    ])

    return np.linalg.solve(A, B)

def get_analytical_datapoints(x_points, bar1:'Bar', bar2:'Bar', x_bar=2/np.pi, l=1):
    '''Generates the temperature distribution given x_points.
    Returns np.array of tempature values'''
    constants = get_analytical_constants(bar1=bar1, bar2=bar2, x_bar=x_bar, l=1)
    C1, D1, C2, D2 = constants[0], constants[1], constants[2], constants[3]

    # Two temp functions for the left and right side of the interface
    left_temp = lambda x: C1*np.sinh(bar1.alpha*x) + D1*np.cosh(bar1.alpha*x)
    right_temp = lambda x: C2*np.sinh(bar2.alpha*x) + D2*np.cosh(bar2.alpha*x)
```

```

temp_values = np.array([])
for x in x_points:
    if x <= x_bar:
        temp = left_temp(x)
    else:
        temp = right_temp(x)
    temp_values = np.append(temp_values, temp)

return temp_values

def get_analytical_heat_convection(x, bar1:'Bar', bar2:'Bar', x_bar=2/np.pi, l=1):
    '''Gets the analytical heat convection at x from heat flux conservation'''
    constants = get_analytical_constants(bar1=bar1, bar2=bar2, x_bar=x_bar, l=1)
    C1, D1, C2, D2 = constants[0], constants[1], constants[2], constants[3]

    t_prime = C2*bar2.alpha*np.cosh(bar2.alpha*x) + D2*bar2.alpha*np.sinh(bar2.alpha*x)

    q_dot = -bar2.k*bar2.area*t_prime
    return q_dot

def fdm_array(bar1:'Bar', bar2:'Bar', num_sections: int=6, x_bar=2/np.pi, l=1):
    '''num_sections = total sections to cut the bar. Note that this is NOT evenly
        distributed over the entire
        bar. Half of the sections are left of x_bar, half are to the right.
        Returns the A matrix of the FDM equation Ax = B'''
    dx1 = x_bar / (num_sections / 2)
    dx2 = (1 - x_bar) / (num_sections / 2)
    k1 = 2 + bar1.alpha**2 * dx1**2
    k2 = 2 + bar2.alpha**2 * dx2**2

    beta1 = (bar1.k * bar1.area) / (dx1)
    beta2 = (bar2.k * bar2.area) / (dx2)

    #rho = beta1 - beta2 - (bar1.k * bar1.area * dx1 * bar1.alpha**2) / 2 + (bar2.k *
        bar2.area * dx2 * bar2.alpha**2) / 2
    rho = beta1 + beta2 + (bar1.k * bar1.area * dx1 * bar1.alpha**2) / 2 + (bar2.k *
        bar2.area * dx2 * bar2.alpha**2) / 2
    # Making the matrix
    # There are num_sections - 1 rows and columns
    # row 0, row n - 2, and row num_sections/2 - 1 are unique
    # the rest are -1, k, -1 where k changes from k1 to k2 at num_sections/2

    # Initialize matrix A with zeros
    A = np.zeros((num_sections - 1, num_sections - 1))

    # Fill the matrix A
    for row in range(num_sections - 1):
        if row == 0:
            # First row for boundary at x = 0 (for example Dirichlet or Neumann condition)
            A[row, row] = k1

```

```

        A[row, row+1] = -1
    elif row == num_sections // 2 - 1:
        # Special row at the interface between bar1 and bar2
        A[row, row-1] = -beta1
        A[row, row] = rho # This is the rho value you computed
        A[row, row+1] = -beta2
    elif row == num_sections - 2:
        # Last row for boundary at x = 1 (again assuming Dirichlet or Neumann condition)
        A[row, row-1] = -1
        A[row, row] = k2
    else:
        # Interior rows for bar1 and bar2 (uniform grid)
        if row < num_sections // 2 - 1:
            # In bar1 region
            A[row, row-1] = -1
            A[row, row] = k1
            A[row, row+1] = -1
        else:
            # In bar2 region
            A[row, row-1] = -1
            A[row, row] = k2
            A[row, row+1] = -1
    return A

def fem_array(bar1:'Bar', bar2:'Bar', num_sections: int=6, x_bar=2/np.pi, l=1):
    '''num_sections = total sections to cut the bar. Note that this is NOT evenly
        distributed over the entire
    bar. Half of the sections are left of x_bar, half are to the right.
    Returns the A matrix of the FEM equation Ax = B'''
    dx1 = x_bar / (num_sections / 2)
    dx2 = (1 - x_bar) / (num_sections / 2)
    k1 = (2/dx1 + 4*bar1.alpha**2*dx1/6) / (1/dx1 - bar1.alpha**2*dx1/6)
    k2 = (2/dx2 + 4*bar2.alpha**2*dx2/6) / (1/dx2 - bar2.alpha**2*dx2/6)

    beta1 = (bar1.k * bar1.area) / (dx1)
    beta2 = (bar2.k * bar2.area) / (dx2)

    rho = beta1 + beta2 + (bar1.k * bar1.area * dx1 * bar1.alpha**2) / 2 + (bar2.k *
        bar2.area * dx2 * bar2.alpha**2) / 2
    # Making the matrix
    # There are num_sections - 1 rows and columns
    # row 0, row n - 2, and row num_sections/2 - 1 are unique
    # the rest are -1, k, -1 where k changes from k1 to k2 at num_sections/2

    # Initialize matrix A with zeros
    A = np.zeros((num_sections - 1, num_sections - 1))

    # Fill the matrix A
    for row in range(num_sections - 1):
        if row == 0:

```



```

    # First row for boundary at x = 0 (for example Dirichlet or Neumann condition)
    A[row, row] = k1
    A[row, row+1] = -1
elif row == num_sections // 2 - 1:
    # Special row at the interface between bar1 and bar2
    A[row, row-1] = -beta1
    A[row, row] = rho # This is the rho value you computed
    A[row, row+1] = -beta2
elif row == num_sections - 2:
    # Last row for boundary at x = 1 (again assuming Dirichlet or Neumann condition)
    A[row, row-1] = -1
    A[row, row] = k2
else:
    # Interior rows for bar1 and bar2 (uniform grid)
    if row < num_sections // 2 - 1:
        # In bar1 region
        A[row, row-1] = -1
        A[row, row] = k1
        A[row, row+1] = -1
    else:
        # In bar2 region
        A[row, row-1] = -1
        A[row, row] = k2
        A[row, row+1] = -1
return A

def solve(bar1:'Bar', bar2:'Bar', num_sections: int=6, x_bar=2/np.pi, l=1, method="FDM"):
    '''Solves using FDM or FEM as specified. Returns (temps, x_values)'''
    if method == "FDM":
        A = fdm_array(bar1,
                      bar2,
                      num_sections=num_sections,
                      x_bar=x_bar,
                      l=1)
    elif method == "FEM":
        A = fem_array(bar1,
                      bar2,
                      num_sections=num_sections,
                      x_bar=x_bar,
                      l=1)

    B = np.zeros((num_sections - 1))
    B[-1] = 100

    # Add boundary conditions
    interior_temps = np.linalg.solve(A, B)
    all_temps = np.array([0])
    all_temps = np.append(all_temps, interior_temps)
    all_temps = np.append(all_temps, 100)

```

```

# Generate the x_values to return along with the temps
dx2 = (1 - x_bar) / (num_sections / 2)
x_values = np.linspace(0, x_bar, num_sections // 2 + 1) # [0, x_bar]
x_values = np.append(x_values, np.linspace(x_bar+dx2, 1, num_sections // 2))
return all_temps, x_values

if __name__ == "__main__":
    bar1 = Bar(
        radius=0.1,
        k=0.5,
        h=0.25
    )
    bar2 = Bar(
        radius=0.1,
        k=2,
        h=0.25
    )

    set_k_ratio(0.5, bar1, bar2)

    x_bar = 0.5

    x_values = np.linspace(0, 1, 1000)
    temp_values = get_analytical_datapoints(x_points=x_values,
                                            bar1=bar1,
                                            bar2=bar2,
                                            x_bar=x_bar)

# fdm tests
l = 1
num_sections = 4
dx1 = x_bar / (num_sections / 2)
dx2 = (1 - x_bar) / (num_sections / 2)
fdm_temps, x_fdm = solve(bar1=bar1,
                          bar2=bar2,
                          num_sections=num_sections,
                          x_bar=x_bar,
                          l=1,
                          method="FDM")
fem_temps, x_fem = solve(bar1=bar1,
                          bar2=bar2,
                          num_sections=num_sections,
                          x_bar=x_bar,
                          l=1,
                          method="FEM")

# Plot the data
plt.plot(x_values, temp_values, label='Temperature Distribution')

```

```

plt.axvline(x=x_bar, color='r', linestyle='--', label='x_bar')
plt.plot(x_fdm, fdm_temps, 'o-', label='Temperature Distribution (FDM)', color='g')
plt.plot(x_fem, fem_temps, 'o-', label='Temperature Distribution (FEM)', color='r')
plt.xlabel('x')
plt.ylabel('Temperature')
plt.title('Temperature Distribution Along the Bar')
plt.legend()
plt.grid(True)

# Show plot
plt.show()

```

5.3 Bar.py

```

# Bar.py

from numpy import pi

class Bar():
    def __init__(self, radius:float=0.1, k:float=0.5, h:float=0.0025):
        self.radius:float = radius
        self.k:float = k
        self.h:float = h

        self.update_properties()

    def update_properties(self):
        self.area:float = pi*self.radius**2
        self.p: float = 2*pi*self.radius
        self.alpha = ((self.h*self.p)/(self.k*self.area))**0.5

    def set_k(self, k:float):
        self.k = k
        self.update_properties()

def set_k_ratio(ratio: float, bar1:'Bar', bar2:'Bar'):
    '''Sets the value of bar2.k = ratio*bar1.k and updates internal properties'''
    bar2.set_k(bar1.k*ratio)

```

5.4 common.py

```

# common.py

import numpy as np
from Bar import Bar

def fdm_heat_extraction(t_0, t_1, dx, bar:'Bar', order=2):

```

```

'''Get the heat conduction at the point t_1 using Taylor series'''
if order == 1:
    return -1 * bar.k * bar.area * (t_1 - t_0) / dx
elif order == 2:
    return -1 * bar.k * bar.area * (((t_1 - t_0) / dx) + (bar.alpha**2 * dx * t_1 / 2))

def fem_heat_extraction(t_0, t_1, dx, bar:'Bar'):
    '''Get the heat conduction at the point t_1 using FEM equation'''
    term_1 = (-1/dx + bar.alpha**2*dx/6) * t_0
    term_2 = (1/dx + 2*bar.alpha**2*dx/6) * t_1
    return -1 * bar.k * bar.area * (term_1 + term_2)

def calc_error(exact, q_1):
    return np.abs((exact - q_1) / exact)

def calc_beta(exact, q_1, q_2, dx_1, dx_2):
    return np.log(np.abs((exact - q_1)/(exact - q_2))) / np.log(dx_1 / dx_2)

def calc_extrapolated(q1, q2, q3):
    '''Calcs the Richardson Extrapolation value based on 3 different meshes.
    Assumes that q3 is 2x finer than q2 is 2x finer than q1'''
    numerator = q1*q3 - q2**2
    denominator = q1 + q3 - 2*q2
    return numerator / denominator

```
