Development of a MATLAB GUI to Assist the Active Comprehension of Biomedical Transport Phenomena Using a Visual Aid

Pranav Suri
University of Arkansas, Fayetteville

Development of a MATLAB GUI to Assist the Active Comprehension of Biomedical Transport Phenomena Using a Visual Aid

Pranav Suri and Dr. Christopher Nelson

Department of Biomedical Engineering

E-mail: psuri@uark.edu

Abstract

Studies show that inductive teaching methods for Biomedical Transport Phenomena greatly benefit from an accompanying visual aid. The following project aimed to develop a MATLAB GUI application that illustrates steady-state heat transfer with a graph and heat map using user-defined boundary conditions and numerical parameters. The application was evaluated using a survey that first familiarized the user with the GUI by running through heat transfer exercises, then allowed the user to experiment with the application, and finally asked users about their experiences using a questionnaire. The responses indicate that the GUI was received positively overall, and that a MATLAB component to the class would have aided these students' understanding of the material.

Keywords: heat transfer, biomedical transport phenomena, MATLAB GUI

Introduction

Students of any discipline can recognize the broad scope of applications heat and mass transfer has in everyday life: from wearing a coat to insulate you during a cold winter day, to heating water for your morning cup of coffee, and even to the maintenance of your body's temperature. Heat and mass transfer's broad scope of applications lends to its intuitive nature: heat and mass both flow from high to low temperature/concentration. However, the intuitive nature of the concept somewhat breaks down when more complex mathematical models are applied to it. Models that incorporate multiple boundaries, heat generation, partition coefficients, and other nuances can be difficult to grasp without telling visual aids. Such deductive teaching methods often employed for heat and mass transfer have been shown to stifle students' natural learning processes for the subject [1].

Biomedical Transport Phenomena is arguably one of the most challenging courses within the Biomedical Engineering curriculum, drawing complex learning objectives from previous classes such as Thermodynamics, Biomaterials, and Fluid Mechanics to mathematically model heat and mass transport. Typical teaching methods for the course revolve around theoretical problem sets, but a 2000 study cites the overreliance on these problem sets as a deficiency of the deductive teaching method, and that it ultimately leads to inadequate critical thinking capabilities in students [2]. Teaching methods have since evolved, and have incorporated more innovative active-learning strategies, such as ones that utilize social and digital media, problem-based learning, and project-based learning [1].

1.1 Current teaching methods

Both deductive and inductive methods have been employed for teaching heat and mass transfer. Deductive methods begin with rules or general principles and draw predictions from those general rules. Most textbooks for the subject are written in a deductive format. Induction, according to a study by Felder *et al.* in 2000, is the “more natural learning style,” and is a style in which intructors present familiar phenomena, practical issues, or experimental observations before presenting or inferring a general principal [3].

The typical approach to teaching mass and heat transfer is beginning with an introductory chapter on the phenomenon concerned, and continuing with a derivation of the balances and other auxiliary equations. Students then typically work through problems in the classroom with the guidance of the professor, and are then given problem sets to work on independently or with peers as homework.

1.2 Objective of this Study

The objective of this paper and thesis is to describe and develop a novel MATLAB GUI application to supplement student learning and visualization of course material. The application is premised on steady-state heat transfer, which is taught at the beginning of the course, and is based on the heat equation:

$$\boldsymbol{\rho c_p \frac{\partial T}{dt} + \rho c_p \frac{\partial}{\partial x}(vT) = k\frac{\partial^2 T}{\partial x^2} + Q}$$

The GUI generates a graph and heat map from student-defined boundary conditions and parameters. The purpose of the application is to better illustrate the heat equation and its relation with the five boundary conditions. The objective of the illustration is for students to draw connections between the otherwise arcane heat equation derivations and train their own intuition. This application presents both a deductive and inductive learning method, as students can use the application to make predictions about other outcomes and derive general trends for different combinations of boundary conditions. Additionally, students have access to the MATLAB code itself. Through tracing the code, students can learn how the heat equation was simplified and how MATLAB code can be applied to the class. The application was evaluated by survey responses by students who had already taken the course. The study will be expanded as the GUI will be used for the next class, and students will have the opportunity to use and modify the application for their own studying.

Methods

2.1 Heat equation simplification

The application considers the heat profiles of steady-state heat transfer with or without heat generation ($\boldsymbol{Q}$). Thus, in the heat equation, the storage ($\boldsymbol{\rho c_p \frac{\partial T}{dt}}$) and bulk flow ($\boldsymbol{\rho c_p \frac{\partial}{\partial x}}$) terms can be ignored. For problems without heat generation, the heat equation simplifies to $\boldsymbol{k\frac{\partial^2 T}{\partial x^2} = 0}$ and for those with heat generation, the heat equation simplifies to $\boldsymbol{k\frac{\partial^2 T}{\partial x^2} = Q}$. After integration, the final heat profiles ($\boldsymbol{T(x)}$) become:

$$\boldsymbol{T(x) = c_1 x + c_2}$$

for problems with no heat generation and

$$\boldsymbol{T(x) = -\frac{Qx^2}{2k} + c_1 x + c_2}$$

for problems with heat generation.

Then, the constants of integration were solved for different combinations of boundary conditions. For the purposes of this transport phenomena class, five different boundary conditions were considered: type I (constant temperature), type II (constant flux), type IIa (insulative boundary), type IIb (symmetrical distribution) and type III (convective heat transfer). Given heat distribution in one dimension and a surface with length L, the assumptions for each first boundary condition (BC 1) were as follows:

Type I: at $x = 0$ the temperature is a constant temperature T_1, or $T(0) = T_1$.

Type II: at $x = 0$, the change in temperature is proportional to the flux (J), or $-k\frac{\partial T}{\partial x} = J$.

Type IIa: at $x = 0$, there is no change in temperature due to insulation, or $-k\frac{\partial T}{\partial x} = 0$.

Type IIb: at $x = \frac{L}{2}$, there is no change in temperature due to a symmetrical distribution (again, $-k\frac{\partial T}{\partial x} = 0$ but at $x = \frac{L}{2}$).

Type III: at $x = 0$, the change in temperature is proportional to the heat transfer coefficient (h) between the fluid and surface and the fluid's T_{inf}, or the temperature of the fluid:

$$-k\frac{\partial T}{\partial x} = h(T(0) - T_{inf})$$

The assumptions for each secondary boundary condition (BC 2) type were consistent with those outlined above, except exist at $x = L$ and that type I BC 2 temperature was notated as T_2 instead of T_1. Additionally, boundary condition combinations of any two type II boundaries were not possible to solve for, as a constant temperature at a given point is necessary to set a profile. The results when solving for constants of integration with no generation are shown in **Appendix Figure 1 (A) and (B)** and with heat generation in **Appendix Figure 1 (C) and (D)**. The final heat profiles for each combination ($T(x)$) are given in **Appendix Figure 1 (E) and (F)**.

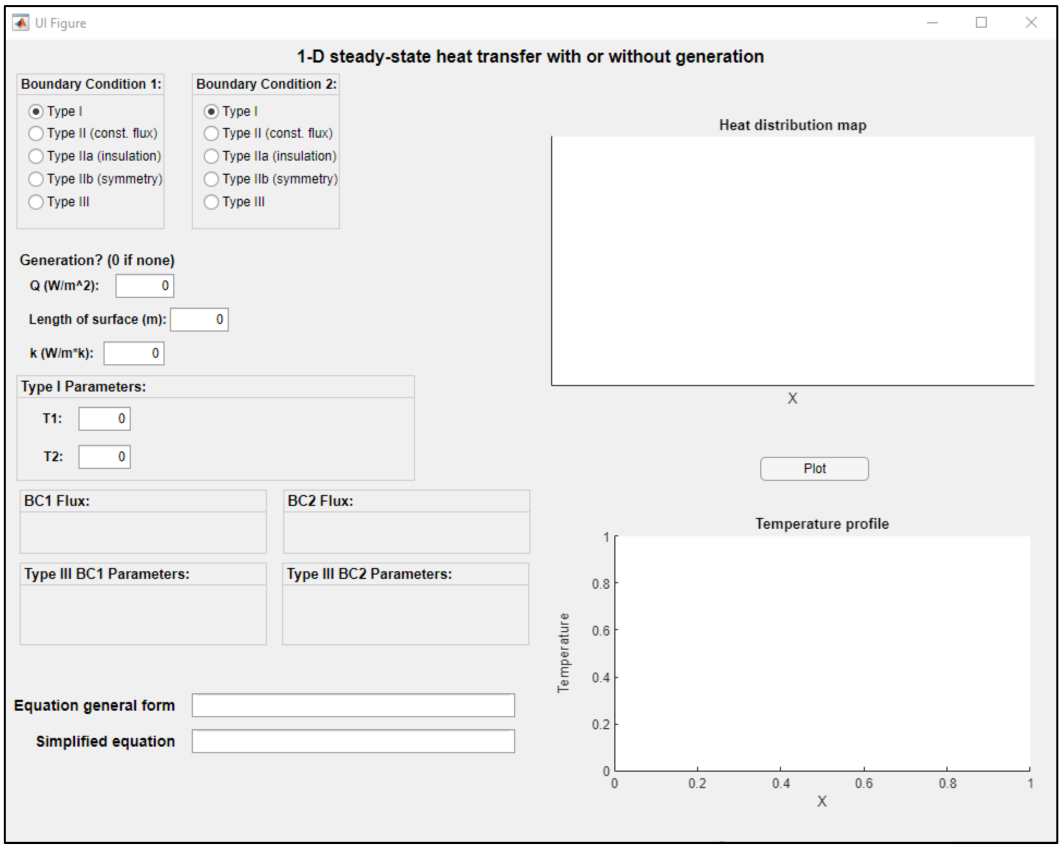
2.2 Constructing the application

The constants of integration and equations were then programmed into MATLAB (version R2018b) using the AppDesigner utility. The code prompts the user to select two boundary condition types, and the corresponding parameter edit fields appear. Parameters that are always active are $\boldsymbol{k}$ (thermal conductivity), $\boldsymbol{L}$ (length of the surface), and $\boldsymbol{Q}$ (the heat generation term). If $\boldsymbol{k}$ or $\boldsymbol{L}$ are not entered, the GUI prompts the user to fill the fields, and if $\boldsymbol{Q}$ is left as 0, the application assumes that there is no generation. Once users have finished filling in the edit fields, they may then select 'plot' to graph the function, generate the heat map, and display both the simplified and general forms of the heat equation for the profile. The user interface for the application and flow chart for the code are shown in **Figure 1** and **Figure 2**, respectively. Plots generated by the application, including those found in questions from the survey, are shown in **Figure 3**.

2.3 Surveying students

Surveys were issued via email to students who had taken the course during the fall semester of 2019. The survey response collection period lasted from February 20, 2020 to March 31, 2020 (41 days). The survey first familiarized students with the interface and options by modeling the heat distribution of surface tissue in a human at rest using two type I BCs (**Figure 3a)**. Students were directed to fill fields for heat generation, thermal conductivity, length, and two boundary temperatures intended to illustrate the temperature differential between tissue at the surface and viscera. Following the sample exercise, students were given two problems adapted from a heat transfer exam administered during the fall semester: one regarding the internal plate temperature of a steam iron (**Figure 3b**) and another regarding the convective cooling of skin during a cool day (**Figure 3c**).

After the exercises, students were asked to rate the GUI's help in understanding the material, utility as a visual aid, help in preparing in class, and overall performance. Additionally, they were asked how much they supported the use of a MATLAB component in the class and for any additional feedback. A full copy of the survey text is shown in **Appendix Figure 3** and **Appendix Figure 4**.



Boundary Condition 1: [✓ Type I] Type II (const. flux) Type IIa (insulation) Type IIb (symmetry) Type III

Boundary Condition 2: [✓ Type I] Type II (const. flux) Type IIa (insulation) Type IIb (symmetry) Type III

Generation? (0 if none)

Q (W/m^2): 0

Length of surface (m): 0

k (W/m*k): 0

Type I Parameters:

T1: 0

T2: 0

Figure 1 The user interface of the application.

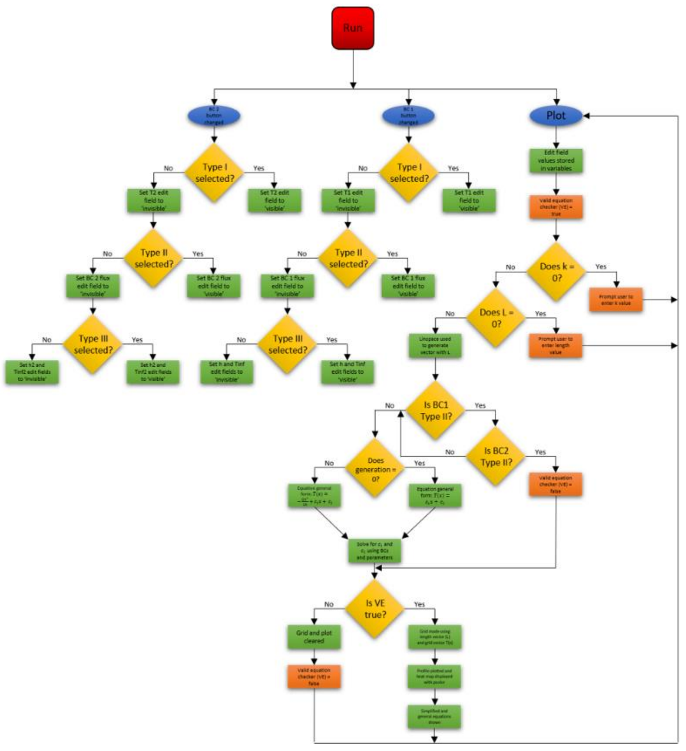
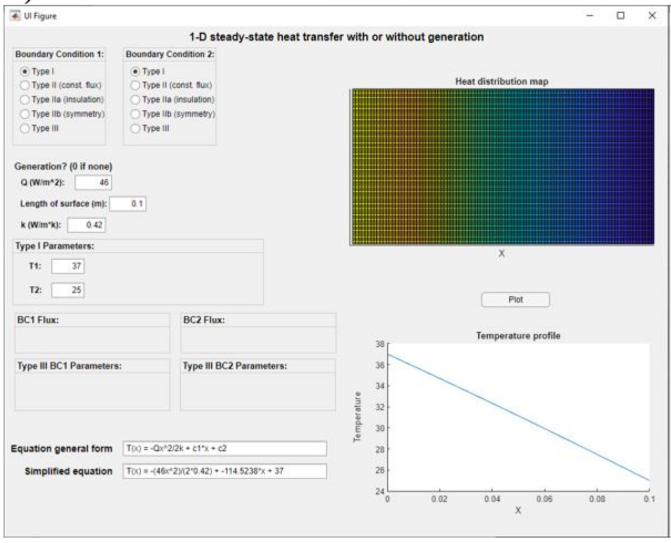

Run
BC 2 button changed
BC 1 button changed
Plot
Type I selected?
No
Yes
Set T2 edit field to 'invisible'
Set T2 edit field to 'visible'
Set T1 edit field to 'invisible'
Set T1 edit field to 'visible'
Type II selected?
Set BC 2 flux edit field to 'invisible'
Set BC 2 flux edit field to 'visible'
Set BC 1 flux edit field to 'invisible'
Set BC 1 flux edit field to 'visible'
Type III selected?
Set h2 and Tinf2 edit fields to 'invisible'
Set h2 and Tinf2 edit fields to 'visible'
Set h and Tinf edit fields to 'invisible'
Set h and Tinf edit fields to 'visible'
Edit field values stored in variables
Valid equation checker (VE) = true
Does k = 0?
Prompt user to enter k value
Does L = 0?
Linspace used to generate vector with L
Prompt user to enter length value
Is BC1 Type II?
Is BC2 Type II?
Valid equation checker (VE) = false
Does generation = 0?
Equation general form: $T(x) = -\frac{Qx^2}{2k} + c_1x + c_2$
Equation general form: $T(x) = c_1x + c_2$
Solve for c_1 and c_2 using BCs and parameters
Is VE true?
Grid and plot cleared
Valid equation checker (VE) = false
Grid made using length vector (L) and grid vector T(x)
Profile plotted and heat map displayed with pcolor
Simplified and general equations shown

Figure 2 Flow chart or pseudocode for the application.

A)

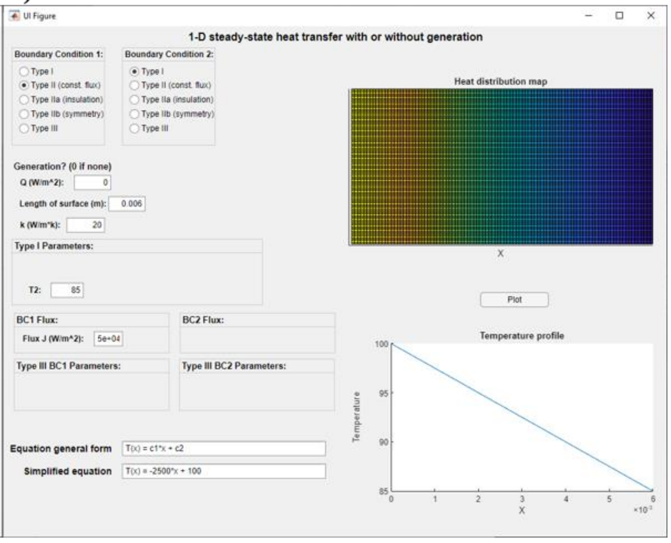


Boundary Condition 1: [✓ Type I] Type II (const. flux) Type IIa (insulation) Type IIb (symmetry) Type III

Boundary Condition 2: [✓ Type I] Type II (const. flux) Type IIa (insulation) Type IIb (symmetry) Type III

Generation? (0 if none)
Q (W/m^2): 46
Length of surface (m): 0.1
k (W/m*k): 0.42
Type I Parameters: T1: 37 T2: 25
Equation general form: T(x) = -Qx^2/2k + c1*x + c2
Simplified equation: T(x) = -(46x^2)/(2*0.42) + -114.5238*x + 37

B)

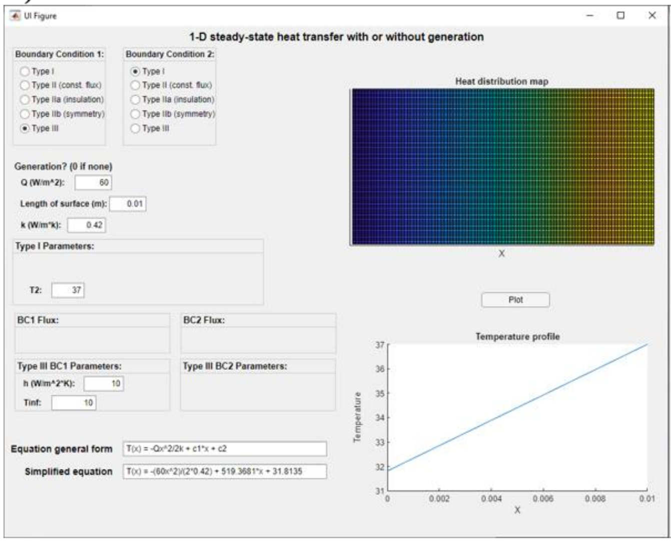


Boundary Condition 1: Type I [✓ Type II (const. flux)] Type IIa (insulation) Type IIb (symmetry) Type III

Boundary Condition 2: [✓ Type I] Type II (const. flux) Type IIa (insulation) Type IIb (symmetry) Type III

Generation? (0 if none)
Q (W/m^2): 0
Length of surface (m): 0.006
k (W/m*k): 20
Type I Parameters: T2: 85
BC1 Flux: Flux J (W/m^2): 5e+04
Equation general form: T(x) = c1*x + c2
Simplified equation: T(x) = -2500*x + 100

C)

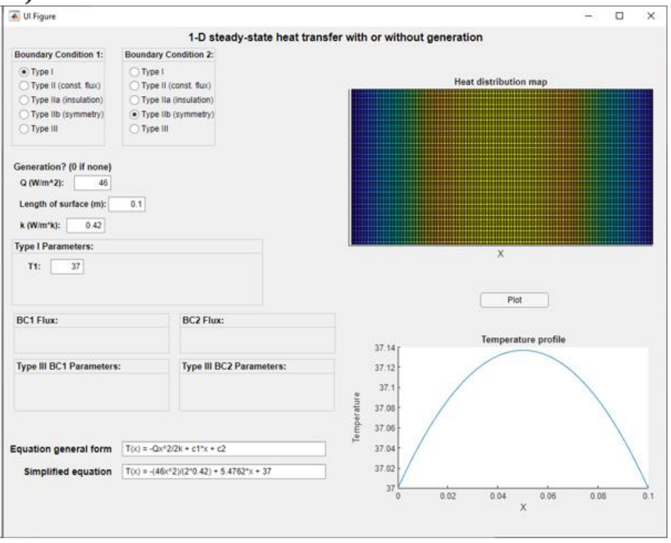


Boundary Condition 1: Type I Type II (const. flux) Type IIa (insulation) Type IIb (symmetry) [✓ Type III]

Boundary Condition 2: [✓ Type I] Type II (const. flux) Type IIa (insulation) Type IIb (symmetry) Type III

Generation? (0 if none)
Q (W/m^2): 60
Length of surface (m): 0.01
k (W/m*k): 0.42
Type I Parameters: T2: 37
Type III BC1 Parameters: h (W/m^2*K): 10 Tinf: 10
Equation general form: T(x) = -Qx^2/2k + c1*x + c2
Simplified equation: T(x) = -(60x^2)/(2*0.42) + 519.3681*x + 31.8135

D)

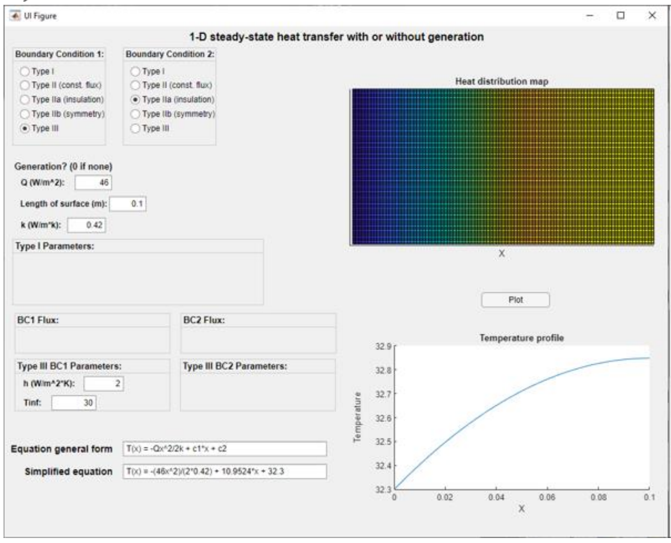


Boundary Condition 1: [✓ Type I] Type II (const. flux) Type IIa (insulation) Type IIb (symmetry) Type III

Boundary Condition 2: Type I Type II (const. flux) Type IIa (insulation) [✓ Type IIb (symmetry)] Type III

Generation? (0 if none)
Q (W/m^2): 46
Length of surface (m): 0.1
k (W/m*k): 0.42
Type I Parameters: T1: 37
Equation general form: T(x) = -Qx^2/2k + c1*x + c2
Simplified equation: T(x) = -(46x^2)/(2*0.42) + 5.4762*x + 37

E)

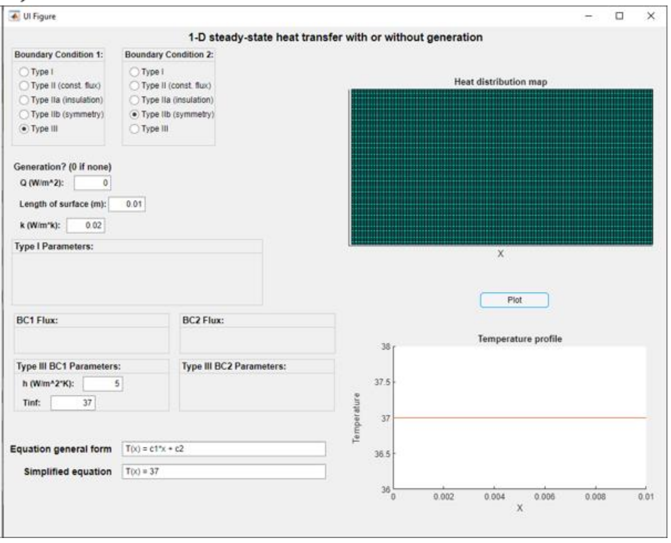


Boundary Condition 1: Type I Type II (const. flux) Type IIa (insulation) Type IIb (symmetry) [✓ Type III]

Boundary Condition 2: Type I Type II (const. flux) [✓ Type IIa (insulation)] Type IIb (symmetry) Type III

Generation? (0 if none)
Q (W/m^2): 46
Length of surface (m): 0.1
k (W/m*k): 0.42
Type III BC1 Parameters: h (W/m^2*K): 2 Tinf: 30
Equation general form: T(x) = -Qx^2/2k + c1*x + c2
Simplified equation: T(x) = -(46x^2)/(2*0.42) + 10.9524*x + 32.3

F)



Boundary Condition 1: Type I Type II (const. flux) Type IIa (insulation) Type IIb (symmetry) [✓ Type III]

Boundary Condition 2: Type I Type II (const. flux) Type IIa (insulation) [✓ Type IIb (symmetry)] Type III

Generation? (0 if none)
Q (W/m^2): 0
Length of surface (m): 0.01
k (W/m*k): 0.02
Type III BC1 Parameters: h (W/m^2*K): 5 Tinf: 37
Equation general form: T(x) = c1*x + c2
Simplified equation: T(x) = 37

Figure 3 GUI with various heat profiles plotted. Exercise 1 (A), exercise 2 (B), and exercise 3 (C) from the survey are shown,

illustrating combinations of two type I boundary conditions with heat generation (exercise 1), a type II and type I boundary condition without heat generation (exercise 2), and a type III and type I boundary condition with heat generation (exercise 3). Additionally, (D) shows type I and type IIb boundary conditions with heat generation, giving a symmetrical heat profile. (E) shows a type IIa (insulative) and type III boundary condition with heat generation, and (F) shows symmetry with a type III and type IIb boundary condition and no heat generation.

Results

Ratings collected from students are summarized in **Figure 4**, and track the ratings by question and by responder with a total of 9 responders. Answers to qualitative survey questions regarding support for a MATLAB component in class, feedback on the GUI, and additional comments can be found in **Appendix Figure 2**.

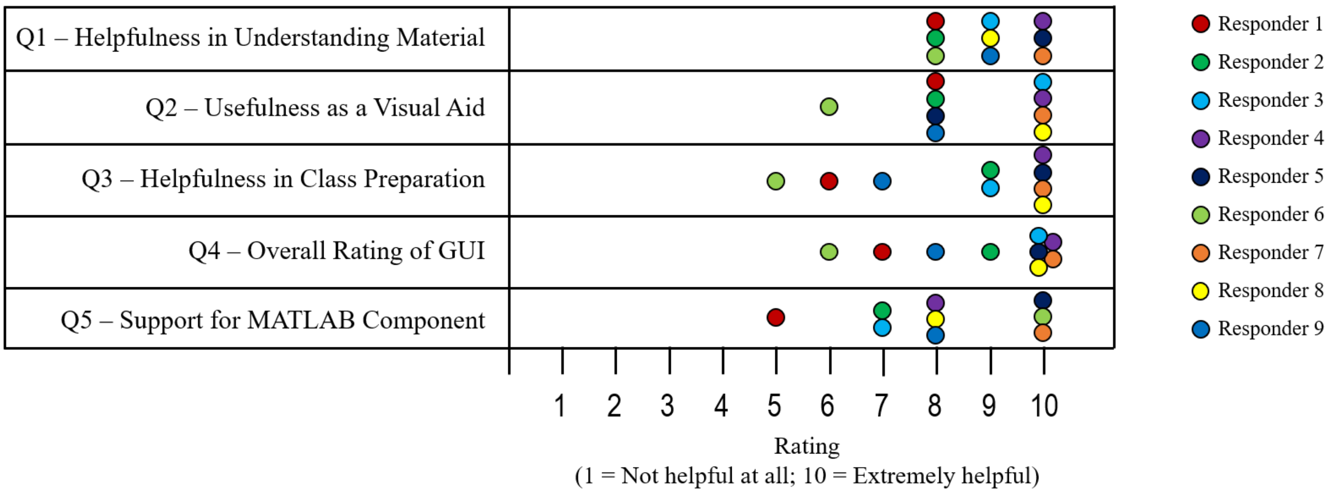


Figure 4 Rating results for each question collected by the survey over the 41 day period. Ratings for each responder are color-coded.

Discussion and Conclusion

Responses from the survey suggest that the rseponders believe the GUI was or would be fairly helpful in understanding the course material (avg. rating for question 1 = 9). Additionally, responders believe that the app functioned well as a visual aid (avg. response for question 2 = 8.56). Responders projected that the GUI would have helped with class preparation (avg. rating for question 3 = 8.44) and gave a final overall rating average of 8.89. Lower ratings on the first four questions could be explained by useability issues with the GUI (cited by two responders) or lack of overall experience with MATLAB (cited by one responder).

Responders appeared to support the addition of a MATLAB component to the course, as the average helpfulness rating for question 5 was 8.11. Students cited the practicality of using MATLAB for transport phenomena as providing greater context for how MATLAB could be used outside of a class setting (cited by three responders). However, some said it would be useful only if students have a stronger MATLAB background (cited by two responders). The visual aspect of the GUI was cited by three responders as being particularly useful, and two responders cited the GUI’s strong ease-of-use. Otherwise, two responders requested temperature labels for the temperature edit fields and one responder requested alternative label placement for heat generation.

While the survey responses are largely positive, the study is likely subject to sampling bias. Surveys were sent unanonymously, which may have made some students more likely to complete the survey than others. Additionally, it likely imparted bias on survey responses as students were aware that the survey was for an honors project conducted by a fellow student. The questionnaire issued to the students was not drafted either, and could confer bias based on syntax. Nonetheless, because this is a new component for the course and shows some promise among the respondents, this study should be viewed as a pilot worthy of follow-up.

Future studies for the GUI should aim for a greater sample size and thus more granular data. The data collected could benefit from a power analysis to determine ratings' significance for the given sample size using a target rating. The data collected so far for the application has been premised entirely on subjective feedback, and the GUI would be better evaluated by quantitative measures. Scores on an administered assessment for students who used the GUI compared with scores of those that did not would be more telling of the GUI's effectiveness. Students surveyed were not actively enrolled in the transport phenomena course either, and some of the material covered may have been forgotten since heat transfer was the first topic covered chronologically. If students were surveyed perhaps at the end of the transport course, more telling data and greater survey participation may be expected.

The GUI provides a platform upon which students may expand. The code itself shows students how MATLAB can be integrated within their coursework, and could be used pedagogically by simple revision of the code and its implementation. The application can be easily expanded to incorporate cylindrical and spherical geometries, which are also covered during the course. Mass transfer could be added as well by adding a partition coefficient parameter, as all 1-D steady-state mass transfer profiles in cartesian coordinates are analogous to the heat profiles programmed into the GUI. Another version of the application could be released covering transient transfer, though this is much more difficult to implement.

Should the GUI be provided to students during next fall's transport course, its uses as a platform and study/visual aid may become more clear. While students can use the application to memorize the heat profiles of different transport conditions, the derivation of the integration constants is not given explicitly. Ideally students will derive the constants themselves and refer to the GUI to check their work (though some hints are given in the code's annotations). Students can use the application to explore different combinations of boundary conditions not covered in class, and hopefully to bolster the development of their intuition in the context of heat and mass transfer.

References

[1] Wen, F and Khera, E (2016) Identify-Solve-Broadcast your own transport phenomenon: student-created YouTube videos to foster active learning in mass and heat transfer. *Chemical Engineering Education*, 50(3), 186-192.

[2] Felder, R.M. *et al*. (2000). The future of engineering education II. Teaching methods that work. *Chem. Eng. Ed.*, 34(1), 26.

[3] Farrell, S and Hesketh, R.P. (2000). An inductive approach to teaching heat and mass transfer. *2000 ASEE Annual Conference*.

Appendix

(A)

c_1 for No Generation

		BC 1				
		Type I	Type II	Type IIa	Type IIb	Type III
BC 2	Type I	$\frac{T_2 - T_1}{L}$	$-\frac{J}{k}$	0	0	$\frac{T_2 - T_\infty}{\frac{k}{h} + L}$
	Type II	$\frac{J}{k}$	-	-	-	$\frac{J}{k}$
	Type IIa	0	-	-	-	0
	Type IIb	0	-	-	-	0
	Type III	$\frac{h(T_\infty - T_1)}{k + hL}$	$-\frac{J}{k}$	0	0	$\frac{T_{\infty_2} - T_{\infty_1}}{L + k(\frac{1}{h_1} + \frac{1}{h_2})}$

(B)

c_2 for No Generation

		BC 1				
		Type I	Type II	Type IIa	Type IIb	Type III
BC 2	Type I	T_1	$\frac{JL}{k} + T_2$	T_2	T_2	$T_2 - \frac{T_2 - T_\infty}{\frac{k}{hL} + 1}$
	Type II	T_1	-	-	-	$\frac{J}{h} + T_\infty$
	Type IIa	T_1	-	-	-	T_∞
	Type IIb	T_1	-	-	-	T_∞
	Type III	T_1	$\frac{J}{h} + \frac{JL}{k} + T_\infty$	T_∞	T_∞	$\frac{T_{\infty_2} - T_{\infty_1}}{\frac{Lh_1}{k} + h_1(\frac{1}{h_1} + \frac{1}{h_2})} + T_{\infty_1}$

(C)

		c_1 for Generation				
		BC 1				
		Type I	Type II	Type IIa	Type IIb	Type III
BC 2	Type I	$\frac{T_2 + \frac{QL^2}{2k} - T_1}{L}$	$-\frac{J}{k}$	0	$\frac{QL}{2k}$	$\frac{T_2 - T_\infty + \frac{QL^2}{2k}}{\frac{k}{h} + L}$
	Type II	$\frac{1}{k}(QL + J)$	-	-	-	$\frac{J + QL}{k}$
	Type IIa	$\frac{QL}{k}$	-	-	-	$\frac{QL}{k}$
	Type IIb	$\frac{QL}{2k}$	-	-	-	$\frac{QL}{2k}$
	Type III	$\frac{h\left(\frac{QL^2}{2k} + T_\infty + T_1\right) + QL}{h + kL}$	$-\frac{J}{k}$	0	$\frac{QL}{2k}$	$\frac{T_{\infty_2} - T_{\infty_1} + \frac{QL}{h_2} + \frac{QL^2}{2k}}{L + k(\frac{1}{h_1} + \frac{1}{h_2})}$

(D)

		c_2 for Generation				
		BC 1				
		Type I	Type II	Type IIa	Type IIb	Type III
BC 2	Type I	T_1	$\frac{QL^2}{2k} + \frac{JL}{k} + T_2$	$\frac{QL^2}{2k} + T_2$	T_2	$T_2 + \frac{QL^2}{2k} - \frac{T_2 - T_\infty - \frac{QL^2}{2k}}{\frac{k}{hL} + 1}$
	Type II	T_1	-	-	-	$\frac{J}{h} + \frac{QL}{h} + T_\infty$
	Type IIa	T_1	-	-	-	$\frac{QL}{h} + T_\infty$
	Type IIb	T_1	-	-	-	$\frac{QL}{2h} + T_\infty$
	Type III	T_1	$\frac{1}{h}(QL + J) + \frac{1}{k}\left(\frac{QL^2}{2} + JL\right) + T_\infty$	$\frac{QL}{h} + T_\infty$	$\frac{QL}{2h} + T_\infty$	$\frac{T_{\infty_2} - T_{\infty_1} + \frac{QL}{h_2} + \frac{QL^2}{2k}}{\frac{Lh_1}{k} + h_1(\frac{1}{h_1} + \frac{1}{h_2})} + T_{\infty_1}$

(E)

		$T(x)$ (no generation)				
		BC 1				
		Type I	Type II	Type IIa	Type IIb	Type III
BC 2	Type I	$\frac{T_2 - T_1}{L}x + T_1$	$-\frac{J}{k}x + \frac{JL}{k} + T_2$	T_2	T_2	$\frac{T_2 - T_\infty}{\frac{k}{h} + L}x + T_2 - \frac{T_2 - T_\infty}{\frac{k}{h} + L}$
	Type II	$\frac{J}{k}x + T_1$	-	-	-	$\frac{J}{k}x + \frac{J}{h} + T_\infty$
	Type IIa	T_1	-	-	-	T_∞
	Type IIb	T_1	-	-	-	T_∞
	Type III	$\frac{h(T_\infty - T_1)}{k + hL}x + T_1$	$-\frac{J}{k}x + \frac{J}{h} + \frac{JL}{k} + T_\infty$	T_∞	T_∞	$\frac{T_{\infty_2} - T_{\infty_1}}{L + k\left(\frac{1}{h_1} + \frac{1}{h_2}\right)}x + \frac{T_{\infty_2} - T_{\infty_1}}{\frac{Lh_1}{k} + h_1(\frac{1}{h_1} + \frac{1}{h_2})} + T_{\infty_1}$

(F)

		$T(x)$ (with generation)				
		BC 1				
		Type I	Type II	Type IIa	Type IIb	Type III
BC 2	Type I	$-\frac{Qx^2}{2k}+\frac{T_2+\frac{QL^2}{2k}-T_1}{L}x+T_1$	$-\frac{Qx^2}{2k}-\frac{J}{k}x+\frac{QL^2}{2k}+\frac{JL}{k}+T_2$	$-\frac{Qx^2}{2k}+\frac{QL^2}{2k}+T_2$	$-\frac{Qx^2}{2k}+\frac{QL}{2k}x+T_2$	$-\frac{Qx^2}{2k}+\frac{T_2-T_\infty+\frac{QL^2}{2k}}{\frac{k}{h}+L}x+T_2+\frac{QL^2}{2k}-\frac{T_2-T_\infty-\frac{QL^2}{2k}}{\frac{k}{hL}+1}$
	Type II	$-\frac{Qx^2}{2k}+\frac{1}{k}(QL+J)x+T_1$	-	-	-	$-\frac{Qx^2}{2k}+\frac{J+QL}{k}x+\frac{J}{h}+\frac{QL}{h}+T_\infty$
	Type IIa	$-\frac{Qx^2}{2k}+\frac{QL}{k}x+T_1$	-	-	-	$-\frac{Qx^2}{2k}+\frac{QL}{k}x+\frac{QL}{h}+T_\infty$
	Type IIb	$-\frac{Qx^2}{2k}+\frac{QL}{2k}x+T_1$	-	-	-	$-\frac{Qx^2}{2k}+\frac{QL}{2k}x+\frac{QL}{2h}+T_\infty$
	Type III	$-\frac{Qx^2}{2k}+\frac{h\left(\frac{QL^2}{2k}+T_\infty+T_1\right)+QL}{h+kL}x+T_1$	$-\frac{Qx^2}{2k}-\frac{J}{k}x+\frac{1}{h}(QL+J)+\frac{1}{k}\left(\frac{QL^2}{2}+JL\right)+T_\infty$	$-\frac{Qx^2}{2k}+\frac{QL}{h}+T_\infty$	$-\frac{Qx^2}{2k}+\frac{QL}{2k}x+\frac{QL}{2h}+T_\infty$	$-\frac{Qx^2}{2k}+\frac{T_{\infty_2}-T_{\infty_1}+\frac{QL}{h_2}+\frac{QL^2}{2k}}{L+k\left(\frac{1}{h_1}+\frac{1}{h_2}\right)}x+\frac{T_{\infty_2}-T_{\infty_1}+\frac{QL}{h_2}+\frac{QL^2}{2k}}{\frac{Lh_1}{k}+h_1(\frac{1}{h_1}+\frac{1}{h_2})}+T_{\infty_1}$

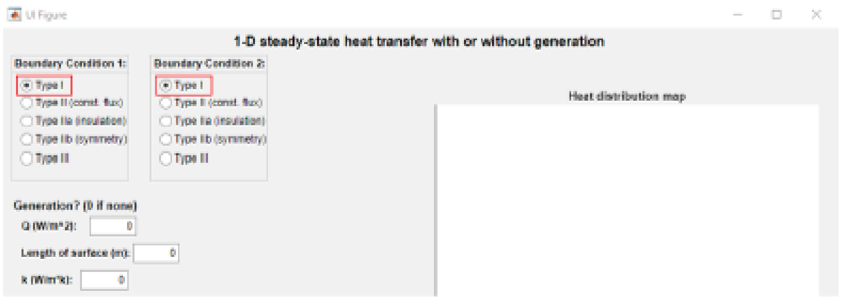
Appendix Figure 1 Solved constants of integration c_1 (A) and c_2 (B) for temperature profiles with no generation, and solved c_1 (C) and c_2 (D) for temperature profiles with generation. Final temperature equations for profiles with generation result in a quadratic equation instead of linear, so in most cases constants of integration are more complex for profiles with heat generation those of profiles with no generation. Constants of integration were solved for using the assumptions given for each boundary condition. Full temperature profiles for each boundary condition combination are shown without generation (E) and with generation (F).

Responder	Please elaborate on why you would support/not support a MATLAB component to the class?	Please share any feedback, if any, that you would have for the app in terms of additional features, fixes, etc.	Any additional comments?
1		Put units next to temperature	
2	I think a MATLAB component would be good because realistically a lot of transport problems will require you use a computer to solve	The line generation "Generation? (0 if none)" should be placed next to Q instead of above to make it more clear it's only referring to that input. Also temperatures should be labeled with Celsius or Kelvin so users would know which one to use.	
3	It is helpful for complex problems solving and it can provide good graphs for visual learners.	It is a great app to visualize boundary conditions. Great Job. Small note: for some reason, I couldn't move the app window. When I tried to select boundary condition I, it wasn't showing in my screen and I couldn't see it to select it (the whole app window was shifted up so the top section wasn't	
4	If we had a stronger matlab background, I would support this. However, when I took this class, I definitely did not understand matlab well enough to utilize it for understanding homework problems.		
5	Being able to click on the different types of boundary conditions allowed me to better understand which parameters needed to be used and why.	I never really tried to visualize a heat distribution map, so I'd need help on explaining that portion if I took the class again.	I really like this, and I wish I would have had it during the class. I'm definitely a visual learner and the boundary conditions were not taught all in the same day, and you did a great job of bringing together all of the information in a simple design.
6	More practical coding practice.	Had lots of trouble with the interface window. No way to scroll within it. Had to run it a couple times to be able for it to load where I could select a boundary condition.	Great interactive graphs!
7	A great way to visualize the materials.		
8	MATLAB component would incorporate your course knowledge and apply them to a technical scenario. If rightly done, it would always be a bonus to simplify the theory behind all the Transport problems.	I think this is a great idea and would save a lot of time while solving problems.	
9	It's an additional source that would help the students understand the material. Also, it's easy to use.		

Appendix Figure 2 Answers given by responders to open-ended questions.

Exercise 1

Let's start by modeling the heat distribution of surface tissue during rest in a human. First, make sure that Type I is selected for both boundary conditions:



UI Figure

1-D steady-state heat transfer with or without generation

Boundary Condition 1: [✓ Type I] Type II (const. flux) Type IIa (insulation) Type IIb (symmetry) Type III

Boundary Condition 2: [✓ Type I] Type II (const. flux) Type IIa (insulation) Type IIb (symmetry) Type III

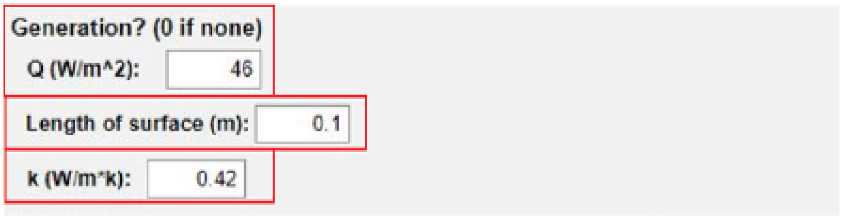
Generation? (0 if none)

Q (W/m^2): 0

Length of surface (m): 0

k (W/m*k): 0

We'll use 46 W/m^2 for heat generation (Q), which is the BMR for a sleeping human being. Input 0.42 W/m*K for thermal conductivity, k, and we'll consider a 10 cm (0.1m) piece of tissue.



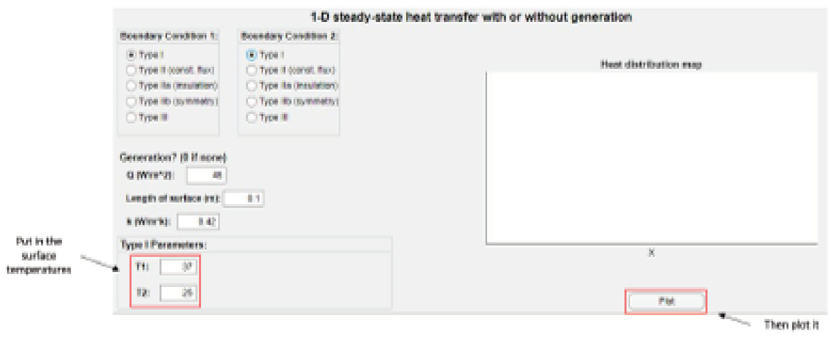
Generation? (0 if none)

Q (W/m^2): 46

Length of surface (m): 0.1

k (W/m*k): 0.42

Make sure Type I BCs are selected for both boundaries, and we'll assume one is at 37 degrees C (internal) and the other is at 25 degrees C (surface). Plot the graph.



1-D steady-state heat transfer with or without generation

Boundary Condition 1: [✓ Type I] Type II (const. flux) Type IIa (insulation) Type IIb (symmetry) Type III

Boundary Condition 2: [✓ Type I] Type II (const. flux) Type IIa (insulation) Type IIb (symmetry) Type III

Generation? (0 if none)

Q (W/m^2): 46

Length of surface (m): 0.1

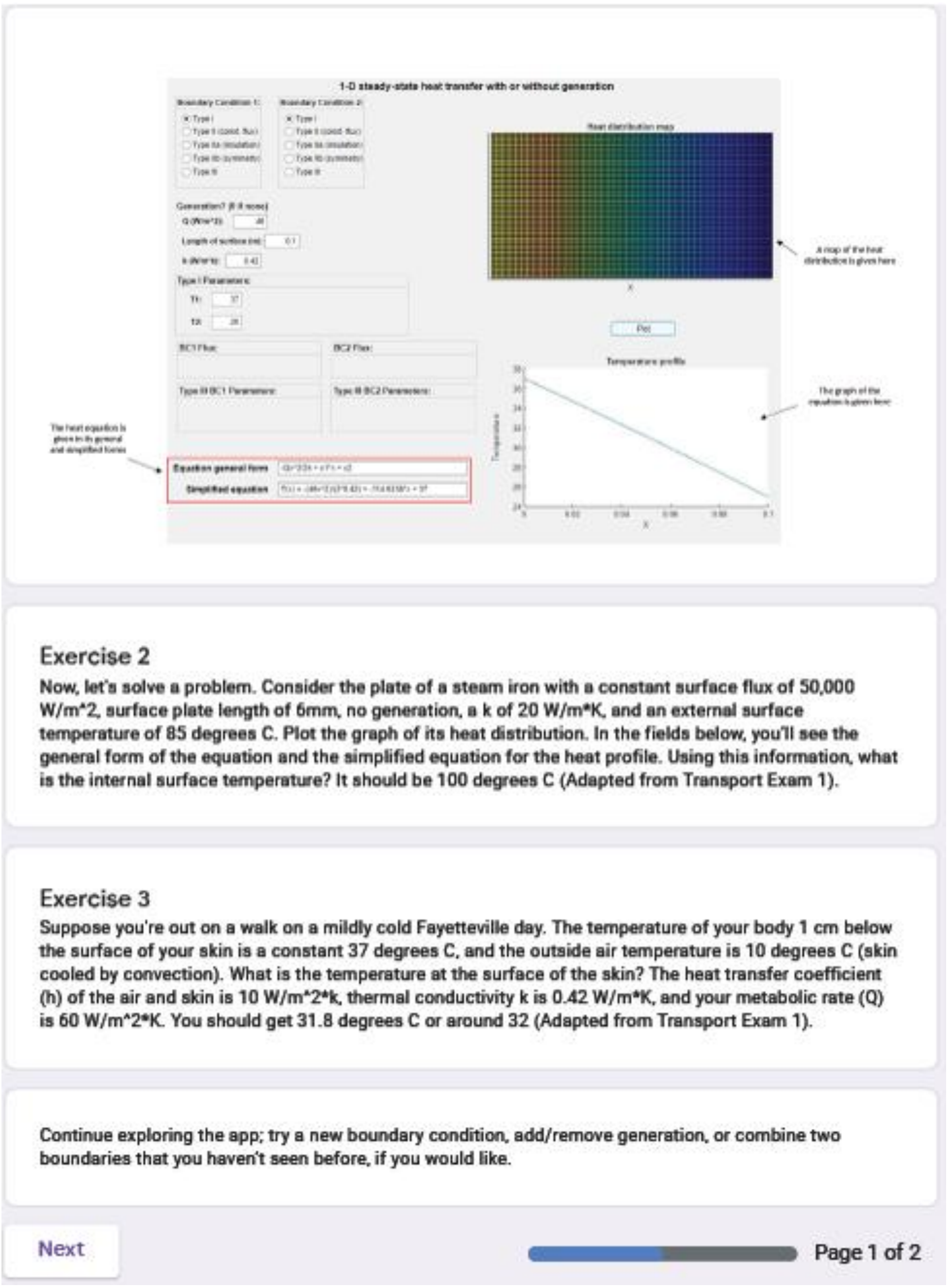
k (W/m*k): 0.42

Type I Parameters:

T1: 37

T2: 25

You should get a graph that looks like the bottom figure, and a heat distribution is shown in the top figure. The heat equation is given in its unsolved form (general form) and then solved form (simplified form).



Exercise 2

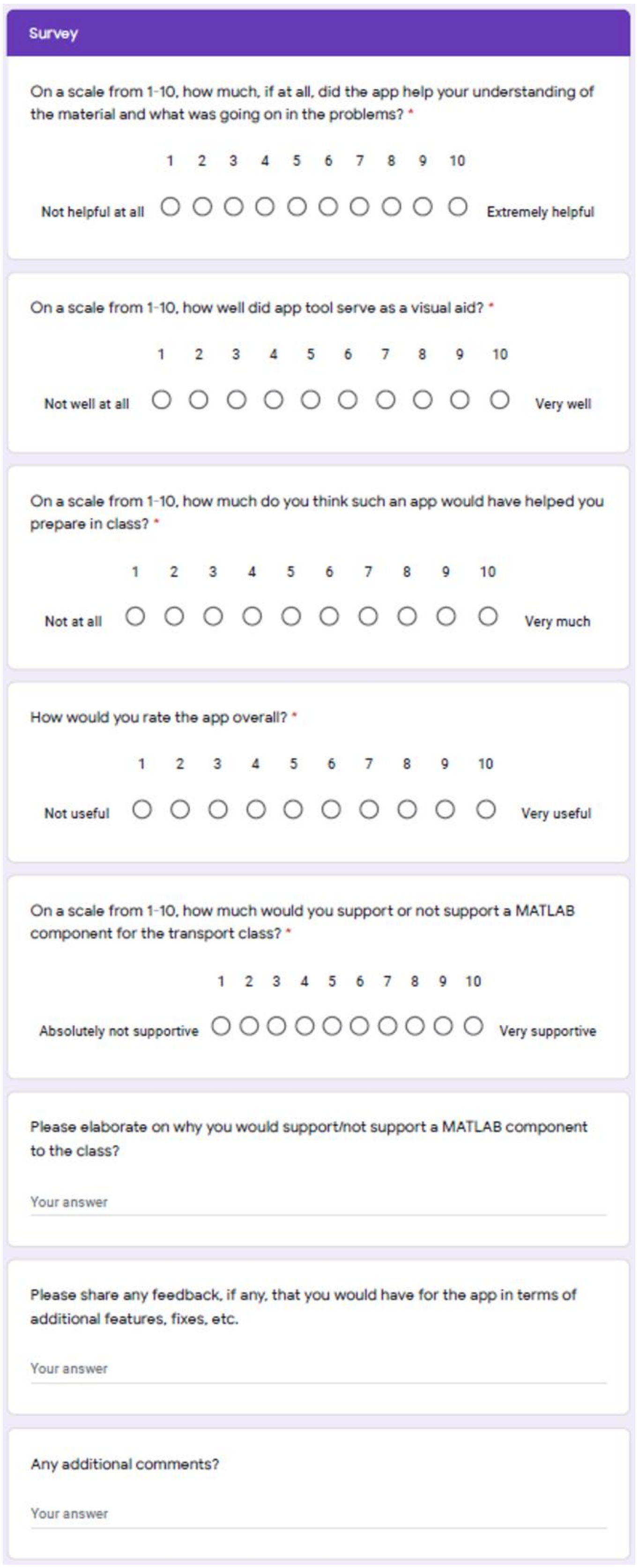
Now, let's solve a problem. Consider the plate of a steam iron with a constant surface flux of 50,000 W/m^2, surface plate length of 6mm, no generation, a k of 20 W/m*K, and an external surface temperature of 85 degrees C. Plot the graph of its heat distribution. In the fields below, you'll see the general form of the equation and the simplified equation for the heat profile. Using this information, what is the internal surface temperature? It should be 100 degrees C (Adapted from Transport Exam 1).

Exercise 3

Suppose you're out on a walk on a mildly cold Fayetteville day. The temperature of your body 1 cm below the surface of your skin is a constant 37 degrees C, and the outside air temperature is 10 degrees C (skin cooled by convection). What is the temperature at the surface of the skin? The heat transfer coefficient (h) of the air and skin is 10 W/m^2*k, thermal conductivity k is 0.42 W/m*K, and your metabolic rate (Q) is 60 W/m^2*K. You should get 31.8 degrees C or around 32 (Adapted from Transport Exam 1).

Continue exploring the app; try a new boundary condition, add/remove generation, or combine two boundaries that you haven't seen before, if you would like.

Appendix Figure 3 Section 1 of survey issued. Section responsible for familiarizing responders with GUI.



Survey

On a scale from 1-10, how much, if at all, did the app help your understanding of the material and what was going on in the problems? *

	1	2	3	4	5	6	7	8	9	10	
Not helpful at all	○	○	○	○	○	○	○	○	○	○	Extremely helpful

On a scale from 1-10, how well did app tool serve as a visual aid? *

	1	2	3	4	5	6	7	8	9	10	
Not well at all	○	○	○	○	○	○	○	○	○	○	Very well

On a scale from 1-10, how much do you think such an app would have helped you prepare in class? *

	1	2	3	4	5	6	7	8	9	10	
Not at all	○	○	○	○	○	○	○	○	○	○	Very much

How would you rate the app overall? *

	1	2	3	4	5	6	7	8	9	10	
Not useful	○	○	○	○	○	○	○	○	○	○	Very useful

On a scale from 1-10, how much would you support or not support a MATLAB component for the transport class? *

	1	2	3	4	5	6	7	8	9	10	
Absolutely not supportive	○	○	○	○	○	○	○	○	○	○	Very supportive

Please elaborate on why you would support/not support a MATLAB component to the class?

Your answer

Please share any feedback, if any, that you would have for the app in terms of additional features, fixes, etc.

Your answer

Any additional comments?

Your answer

Appendix Figure 4 Section 2 of survey issued. Questionnaire part of survey.

```
classdef HeatTransfer1D < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        UIFigure                       matlab.ui.Figure
        UIAxes                         matlab.ui.control.UIAxes
        Generation0ifnoneLabel         matlab.ui.control.Label
        QWm2Label                      matlab.ui.control.Label
        QEditField                     matlab.ui.control.NumericEditField
        DsteadystateheattransferwithorwithoutgenerationLabel  matlab.ui.control.Label
        BoundaryCondition1ButtonGroup  matlab.ui.container.ButtonGroup
        TypeIButton                    matlab.ui.control.RadioButton
        TypeIIButton                   matlab.ui.control.RadioButton
        TypeIIIButton                  matlab.ui.control.RadioButton
        TypeIIaButton                  matlab.ui.control.RadioButton
        TypeIIbButton                  matlab.ui.control.RadioButton
        PlotButton                     matlab.ui.control.Button
        SimplifiedequationEditFieldLabel  matlab.ui.control.Label
        SimplifiedequationEditField    matlab.ui.control.EditField
        EquationgeneralformEditFieldLabel  matlab.ui.control.Label
        EquationgeneralformEditField   matlab.ui.control.EditField
        BoundaryCondition2ButtonGroup  matlab.ui.container.ButtonGroup
        TypeIButton_2                  matlab.ui.control.RadioButton
        TypeIIButton_2                 matlab.ui.control.RadioButton
        TypeIIIButton_2                matlab.ui.control.RadioButton
        TypeIIaButton_2                matlab.ui.control.RadioButton
        TypeIIbButton_2                matlab.ui.control.RadioButton
        UIAxes2                        matlab.ui.control.UIAxes
        LengthofsurfacemLabel          matlab.ui.control.Label
        LengthEditField                matlab.ui.control.NumericEditField
        TypeIParametersPanel           matlab.ui.container.Panel
        T1EditFieldLabel               matlab.ui.control.Label
        T1EditField                    matlab.ui.control.NumericEditField
        T2Label                        matlab.ui.control.Label
        T2EditField                    matlab.ui.control.NumericEditField
        kWmkLabel                      matlab.ui.control.Label
        kWmkEditField                  matlab.ui.control.NumericEditField
        BC1FluxPanel                   matlab.ui.container.Panel
        FluxJWm2Label_2                matlab.ui.control.Label
        JEditField                     matlab.ui.control.NumericEditField
        TypeIIIBC1ParametersPanel      matlab.ui.container.Panel
        hWm2KLabel_2                   matlab.ui.control.Label
        hEditField                     matlab.ui.control.NumericEditField
        TinfEditFieldLabel_2           matlab.ui.control.Label
        TinfEditField                  matlab.ui.control.NumericEditField
        TypeIIIBC2ParametersPanel      matlab.ui.container.Panel
        h2Wm2KLabel                    matlab.ui.control.Label
        h2EditField                    matlab.ui.control.NumericEditField
        Tinf2EditFieldLabel            matlab.ui.control.Label
        Tinf2EditField                 matlab.ui.control.NumericEditField
        BC2FluxPanel                   matlab.ui.container.Panel
        FluxJWm2Label_3                matlab.ui.control.Label
        J2EditField                    matlab.ui.control.NumericEditField
    end

    methods (Access = private)

        % Code that executes after component creation
        function startupFcn(app)
            %Turning fields off besides T1 and T2 initially
            app.JEditField.Visible = 'off';
            app.FluxJWm2Label_2.Visible = 'off';
```

```
                app.hWm2KLabel_2.Visible = 'off';
                app.hEditField.Visible = 'off';
                app.TinfEditFieldLabel_2.Visible = 'off';
                app.TinfEditField.Visible = 'off';
                app.J2EditField.Visible = 'off';
                app.FluxJWm2Label_3.Visible = 'off';
                app.h2Wm2KLabel.Visible = 'off';
                app.h2EditField.Visible = 'off';
                app.Tinf2EditFieldLabel.Visible = 'off';
                app.Tinf2EditField.Visible = 'off';
            end

            % Button pushed function: PlotButton
            function PlotButtonPushed(app, event)
               %BC parameters:
                t1 = app.T1EditField.Value;
                t2 = app.T2EditField.Value;
                J = app.JEditField.Value;
                J2 = app.J2EditField.Value;
                h = app.hEditField.Value;
                h2 = app.h2EditField.Value;
                Tinf = app.TinfEditField.Value;
                Tinf2 = app.Tinf2EditField.Value;


                %General variables
                Eqn = "";
                GenEqn = "";
                k = app.kWmkEditField.Value;
                Q = app.QEditField.Value;
                L = app.LengthEditField.Value;

                %Equation validity check
                VE = 1;

                %Choosing boundary condition 1:
                bc1 = 1;

                if app.TypeIButton.Value
                    bc1 = 1;
                elseif app.TypeIIButton.Value
                    bc1 = 2;
                elseif app.TypeIIaButton.Value
                    bc1 = 21;
                elseif app.TypeIIbButton.Value
                    bc1 = 22;
                elseif app.TypeIIIButton.Value
                    bc1 = 3;
                end


                %Choosing boundary condition 2:
                bc2 = 1;

                if app.TypeIButton_2.Value
                    bc2 = 1;
                elseif app.TypeIIButton_2.Value
                    bc2 = 2;
                elseif app.TypeIIaButton_2.Value
                    bc2 = 21;
                elseif app.TypeIIbButton_2.Value
                    bc2 = 22;
                elseif app.TypeIIIButton_2.Value
                    bc2 = 3;
                end

                %Setting the surface length:
                X = linspace(0,L);

                %Checking for k-value input
                if k == 0

                    Eqn = 'Please enter a k value';
                    VE = 0;

                %Checking for a surface length input
                elseif L == 0

                    Eqn = 'Please enter a surface length';
                    VE = 0;

                elseif k ~= 0 && L ~= 0

                    %For a type I boundary condition 1:
                    if bc1 == 1

                        %Two Type I BCs
                        if bc2 == 1

                            %No heat gen
                            if Q == 0
                                c1 = (t2-t1)/L;
                                c2 = t1;
                                Z = (c1).*X + c2;
                                Eqn = "T(x) = " + num2str(c1) + "*x + " + num2str(c2);
                                GenEqn = "c1*x + c2";

                            %With heat gen
                            elseif Q ~= 0
                                c1 = (t2 + (Q*L^2)/(2*k) - t1)/L;
                                c2 = t1;
                                Z = -(Q.*X.^2)/(2*k) + c1.*X + c2;
                                Eqn = "T(x) = -(" + num2str(Q) + "x^2)/(2*" + num2str(k) + ") + "
    + num2str(c1) + "*x + " + num2str(c2);
                                GenEqn = "-Qx^2/2k + c1*x + c2";
                            end

                        %One type I and one type II BC
                        elseif bc2 == 2

                            %No heat gen
                            if Q == 0
                                c1 = -J2/(-k); %Negative J because flux is in negative x
    direction
                                c2 = t1;
                                Z = c1.*X + c2;
                                Eqn = "T(x) = " + num2str(c1) + "*x + " + num2str(c2);
                                GenEqn = "c1*x + c2";

                            %With heat gen
                            elseif Q ~= 0
                                c1 = -J2/(-k) + Q*L/k; %Setting dt/dx = -qL/k + c1 = J/k and
    solving (recall -k*dt/dx = -J)
                                c2 = t1;
                                Z = -(Q.*X.^2)/(2*k) + c1.*X + c2;
                                Eqn = "T(x) = -(" + num2str(Q) + "x^2)/(2*" + num2str(k) + ") + "
    + num2str(c1) + "*x + " + num2str(c2);
                                GenEqn = "-Qx^2/2k + c1*x + c2";
                            end

                        %One type I and one type IIa (insulative) BC
                        elseif bc2 == 21

                            %No heat gen
                            if Q == 0
                                c1 = 0;
                                c2 = t1;
                                Z = ones(length(X)).*c2;
                                Eqn = "T(x) = " + num2str(c2);
                                GenEqn = "c1*x + c2";

                            %With heat gen
                            elseif Q ~= 0
                                c1 = Q*L/k; %Setting dt/dx = -qL/k + c1 = 0 and solving since the
    insulative boundary is at x = L
                                c2 = t1;
                                Z = -(Q.*X.^2)/(2*k) + c1.*X + c2;
                                Eqn = "T(x) = -(" + num2str(Q) + "x^2)/(2*" + num2str(k) + ") + "
    + num2str(c1) + "*x + " + num2str(c2);
                                GenEqn = "-Qx^2/2k + c1*x + c2";
                            end

                        %One type I and one type IIb (symmetry) BC
                        elseif bc2 == 22

                            %No heat gen
                            if Q == 0
                                c1 = 0;
                                c2 = t1;
                                Z = ones(length(X)).*c2;
                                Eqn = "T(x) = " + num2str(c2);
                                GenEqn = "c1*x + c2";

                            %With heat gen
                            elseif Q ~= 0
                                c1 = (Q*L)/(2*k); %Setting dt/dx = -q(0.5L)/k + c1 = 0 and
    solving
                                c2 = t1;
                                Z = -(Q.*X.^2)/(2*k) + c1.*X + c2;
                                Eqn = "T(x) = -(" + num2str(Q) + "x^2)/(2*" + num2str(k) + ") + "
    + num2str(c1) + "*x + " + num2str(c2);
                                GenEqn = "-Qx^2/2k + c1*x + c2";
                            end


                        %One type I boundary and one type III BC
                        elseif bc2 == 3

                            %No heat gen
                            if Q == 0
                                c1 = (h2*(Tinf2 - t1))/(k + h2*L); %Solving -k*dt/dx = h*(T(L) -
    Tinf)
                                c2 = t1;
                                Z = c1.*X + c2;
                                Eqn = "T(x) = " + num2str(c1) + "*x + " + num2str(c2);
                                GenEqn = "c1*x + c2";

                            %With heat gen
                            elseif Q ~= 0
                                c1 = (h2*((Q*L^2)/(2*k) + Tinf2 - t1) + Q*L)/(k + h2*L); %Setting
    dt/dx = -qL/k + c1 = h(T(L) - Tinf)/k and solving for c1 (recall T(L) = -QL^2/2k + c1*L + c2)
                                c2 = t1;
                                Z = -(Q.*X.^2)/(2*k) + c1.*X + c2;
                                Eqn = "T(x) = -(" + num2str(Q) + "x^2)/(2*" + num2str(k) + ") + "
    + num2str(c1) + "*x + " + num2str(c2);
                                GenEqn = "-Qx^2/2k + c1*x + c2";
                            end
                        end

                    %For a type II (constant flux) boundary condition 1:
                    elseif bc1 == 2

                        %One type II and one type I BC
                        if bc2 == 1

                            %No heat gen
                            if Q == 0
                                c1 = J/(-k); %Negative because flux is in positive x direction (-
    k*dt/dx = J)
                                c2 = t2 - c1*L;
                                Z = c1.*X + c2;
                                Eqn = "T(x) = " + num2str(c1) + "*x + " + num2str(c2);
                                GenEqn = "c1*x + c2";

                            %With heat gen
                            elseif Q ~= 0
                                c1 = J/(-k); %setting dt/dx = -q(0)/k + c1 = -J/k
                                c2 = t2 + (Q*L^2)/(2*k) - c1*L;
                                Z = -(Q.*X.^2)/(2*k) + c1.*X + c2;
                                Eqn = "T(x) = -(" + num2str(Q) + "x^2)/(2*" + num2str(k) + ") + "
    + num2str(c1) + "*x + " + num2str(c2);
                                GenEqn = "-Qx^2/2k + c1*x + c2";
                            end

                        %One type II and one type III BC
                        elseif bc2 == 3

                            %No heat gen
                            if Q == 0
                                c1 = J/(-k);
                                c2 = J/h2 + (J*L)/k + Tinf2;
                                Z = c1.*X + c2;
                                Eqn = "T(x) = " + num2str(c1) + "*x + " + num2str(c2);
                                GenEqn = "c1*x + c2";

                            %With heat gen
                            elseif Q ~= 0
                                c1 = J/(-k);
                                c2 = (Q*L + J)/h2 + (0.5*Q*L^2 + J*L)/k + Tinf2;
                                Z = -(Q.*X.^2)/(2*k) + c1.*X + c2;
                                Eqn = "T(x) = -(" + num2str(Q) + "x^2)/(2*" + num2str(k) + ") + "
    + num2str(c1) + "*x + " + num2str(c2);
                                GenEqn = "-Qx^2/2k + c1*x + c2";
                            end

                        %For all other boundary conditions
                        elseif bc2 == 2 || bc2 == 21 || bc2 == 22

                            VE = 0;

                            %No heat gen
                            if Q == 0
                                Eqn = "Needs at least one type I or type III BC to graph";
                                GenEqn = "c1*x + c2";

                            %With heat gen
                            elseif Q ~= 0
                                Eqn = "Needs at least one type I or type III BC to graph";
                                GenEqn = "-Qx^2/2k + c1*x + c2";

                            end
                        end


                    %For a type IIa (insulative) boundary condition 1:
                    elseif bc1 == 21

                        %One type IIa and one type I BC
                        if bc2 == 1

                            %No heat gen
                            if Q == 0
                                c1 = 0;
                                c2 = t2;
                                Z = ones(length(X)).*c2; %Same as above, except with type I BC
    occurring at x = L
                                Eqn = "T(x) = " + num2str(t2);
                                GenEqn = "c1*x + c2";

                            %With heat gen
                            elseif Q ~= 0
                                c1 = 0; %Because dt/dx = 0 at x = 0 (-q(0)/k + c1 = 0)
                                c2 = t2 + (Q*L^2)/(2*k);
                                Z = -(Q.*X.^2)/(2*k) + c1.*X + c2;
                                Eqn = "T(x) = -(" + num2str(Q) + "x^2)/(2*" + num2str(k) + ") + "
    + num2str(c2);
                                GenEqn = "-Qx^2/2k + c1*x + c2";
                            end

                        %One type IIa and one type III BC
                        elseif bc2 == 3

                            %No heat gen
                            if Q == 0
                                c1 = 0; %At steady-state, the temperature will eventually reach
    Tinf
                                c2 = Tinf2;
                                Z = ones(length(X)).*c2;
                                Eqn = "T(x) = " + num2str(t2);
                                GenEqn = "c1*x + c2";

                            %With heat gen
                            elseif Q ~= 0
                                c1 = 0; %Because -k(dt/dx) = 0 at x = 0 (-k(-q(0)/k + c1) = 0)
                                c2 = Q*L + h2*Tinf2;
                                Z = -(Q.*X.^2)/(2*k) + c1.*X + c2;
                                Eqn = "T(x) = -(" + num2str(Q) + "x^2)/(2*" + num2str(k) + ") + "
    + num2str(c2);
                                GenEqn = "-Qx^2/2k + c1*x + c2";
                            end

                        %For all other boundary conditions
                        elseif bc2 == 2 || bc2 == 21 || bc2 == 22
```

```
                        VE = 0;

                        %No heat gen
                        if Q == 0
                            Eqn = "Needs at least one type I or type III BC to graph";
                            GenEqn = "c1*x + c2";

                        %With heat gen
                        elseif Q ~= 0
                            Eqn = "Needs at least one type I or type III BC to graph";
                            GenEqn = "-Qx^2/2k + c1*x + c2";

                        end
                    end

                %For a type IIb (symmetry) boundary condition 1:
                elseif bc1 == 22

                    %One type IIb and one type I BC
                    if bc2 == 1

                        %No heat gen
                        if Q == 0
                            c1 = 0;
                            c2 = t2;
                            Z = ones(length(X)).*c2;
                            Eqn = "T(x) = " + num2str(c2);
                            GenEqn = "c1*x + c2";

                        %With heat gen
                        elseif Q ~= 0
                            c1 = (Q*L)/(2*k); %Same as above, except with the type I BC
occurring at L
                            c2 = t2 + (Q*L^2)/(2*k) - c1*L; %These last two terms cancel,
leaving c2 = t2
                            Z = -(Q.*X.^2)/(2*k) + c1.*X + c2;
                            Eqn = "T(x) = -(" + num2str(Q) + "x^2)/(2*" + num2str(k) + ") + "
+ num2str(c1) + "*x + " + num2str(c2);
                            GenEqn = "-Qx^2/2k + c1*x + c2";
                        end

                    %One type IIb and one type III BC
                    elseif bc2 == 3

                        %No heat gen
                        if Q == 0
                            c1 = 0;
                            c2 = Tinf2;
                            Z = ones(length(X)).*c2;
                            Eqn = "T(x) = " + num2str(c2);
                            GenEqn = "c1*x + c2";

                        %With heat gen
                        elseif Q ~= 0
                            c1 = (Q*L)/(2*k);
                            c2 = (Q*L + 2*h2*Tinf2)/(2*h2);
                            Z = -(Q.*X.^2)/(2*k) + c1.*X + c2;
                            Eqn = "T(x) = -(" + num2str(Q) + "x^2)/(2*" + num2str(k) + ") + "
+ num2str(c1) + "*x + " + num2str(c2);
                            GenEqn = "-Qx^2/2k + c1*x + c2";
                        end

                    %For all other boundary conditions
                    elseif bc2 == 2 || bc2 == 21 || bc2 == 22

                        VE = 0;

                        %No heat gen
                        if Q == 0
                            Eqn = "Needs at least one type I or type III BC to graph";
                            GenEqn = "c1*x + c2";

                        elseif Q ~= 0
                            Eqn = "Needs at least one type I or type III BC to graph";
                            GenEqn = "-Qx^2/2k + c1*x + c2";

                        end
                    end

                %For a type III boundary condition 1:
                elseif bc1 == 3

                    %One type III and one type I BC
                    if bc2 == 1

                        %No heat gen
                        if Q == 0
                            c1 = (t2 - Tinf)/(k/h + L);
                            c2 = t2 - (t2 - Tinf)/(k/(h*L) + 1);
                            Z = c1.*X + c2;
                            Eqn = "T(x) = " + num2str(c1) + "*x + " + num2str(c2);
                            GenEqn = "c1*x + c2";

                        %With heat gen
                        elseif Q ~= 0
                            c1 = (t2 - Tinf + (Q*L^2)/(2*k))/(k/h + L);
                            c2 = t2 + (Q*L^2)/(2*k) - (t2 - Tinf + (Q*L^2)/(2*k))/(k/(h*L) +
1);

                            Z = -(Q.*X.^2)/(2*k) + c1.*X + c2;
                            Eqn = "T(x) = -(" + num2str(Q) + "x^2)/(2*" + num2str(k) + ") + "
+ num2str(c1) + "*x + " + num2str(c2);
                            GenEqn = "-Qx^2/2k + c1*x + c2";
                        end

                    %One type III and one type II BC
                    elseif bc2 == 2

                        %No heat gen
                        if Q == 0
                            c1 = -J2/(-k); %Flux going in negative x direction
                            c2 = J2/h + Tinf;
                            Z = c1.*X + c2;
                            Eqn = "T(x) = " + num2str(c1) + "*x + " + num2str(c2);;
                            GenEqn = "c1*x + c2";

                        %With heat gen
                        elseif Q ~= 0
                            c1 = (J2 + Q*L)/k;
                            c2 = (J2 + Q*L + h*Tinf)/h;
                            Z = -(Q.*X.^2)/(2*k) + c1.*X + c2;
                            Eqn = "T(x) = -(" + num2str(Q) + "x^2)/(2*" + num2str(k) + ") + "
+ num2str(c1) + "*x + " + num2str(c2);
                            GenEqn = "-Qx^2/2k + c1*x + c2";
                        end

                    %One type III and one type IIa (insulative) BC
                    elseif bc2 == 21

                        %No heat gen
                        if Q == 0
                            c1 = 0;
                            c2 = Tinf;
                            Z = ones(length(X)).*c2;
                            Eqn = "T(x) = " + num2str(c2);
                            GenEqn = "c1*x + c2";

                        %With heat gen
                        elseif Q ~= 0
                            c1 = (Q*L)/k;
                            c2 = (Q*L)/h + Tinf;
                            Z = -(Q.*X.^2)/(2*k) + c1.*X + c2;
                            Eqn = "T(x) = -(" + num2str(Q) + "x^2)/(2*" + num2str(k) + ") + "
+ num2str(c1) + "*x + " + num2str(c2);
                            GenEqn = "-Qx^2/2k + c1*x + c2";
                        end

                    %One type III and one type IIb (symmetry) BC
                    elseif bc2 == 22

                        %No heat gen
                        if Q == 0
                            c1 = 0;
                            c2 = Tinf;
                            Z = ones(length(X)).*c2; %Same as above
                            Eqn = "T(x) = " + num2str(c2);
                            GenEqn = "c1*x + c2";

                        %With heat gen
                        elseif Q ~= 0
                            c1 = (Q*L)/(2*k);
                            c2 = (Q*L)/(2*h) + Tinf;
                            Z = -(Q.*X.^2)/(2*k) + c1.*X + c2;
                            Eqn = "T(x) = -(" + num2str(Q) + "x^2)/(2*" + num2str(k) + ") + "
+ num2str(c1) + "*x + " + num2str(c2);
                            GenEqn = "-Qx^2/2k + c1*x + c2";
                        end

                    %Two type III BCs
                    elseif bc2 == 3

                        %No heat gen
                        if Q == 0
                            c1 = (Tinf2 - Tinf)/(L + k*(1/h + 1/h2));
                            c2 = (Tinf2 - Tinf)/((L*h)/k + h*(1/h + 1/h2)) + Tinf;
                            Z = c1.*X + c2;
                            Eqn = "T(x) = " + num2str(c1) + "*x + " + num2str(c2);
                            GenEqn = "c1*x + c2";

                        %With heat gen
                        elseif Q ~= 0
                            c1 = (Tinf2 - Tinf + Q*L + (Q*L^2)/(2*k))/(L + k*(1/h + 1/h2));
                            c2 = (Tinf2 - Tinf + Q*L + (Q*L^2)/(2*k))/((L*h)/k + h*(1/h +
1/h2)) + Tinf;

                            Z = -(Q.*X.^2)/(2*k) + c1.*X + c2;
                            Eqn = "T(x) = -(" + num2str(Q) + "x^2)/(2*" + num2str(k) + ") + "
+ num2str(c1) + "*x + " + num2str(c2);
                            GenEqn = "-Qx^2/2k + c1*x + c2";
                        end
                    end

                end
            end

            %Checking to see if equation is valid
            if VE == 1

                %Graphing temperature profile
                plot(app.UIAxes2,X,Z);

                %Generating matrix for heat map
                grid = zeros(length(X),length(Z));

                for i = 1:length(X)
                    grid(:,i) = Z(i);
                end

                pcolor(app.UIAxes,grid)


            elseif VE == 0
                cla(app.UIAxes);
                cla(app.UIAxes2);
            end

            %Displaying equations
            app.EquationgeneralformEditField.Value = "T(x) = " + GenEqn;
            app.SimplifiedequationEditField.Value = Eqn;




        end

        % Selection changed function: BoundaryCondition1ButtonGroup
        function BoundaryCondition1ButtonGroupSelectionChanged(app, event)
            selectedButton = app.BoundaryCondition1ButtonGroup.SelectedObject;

            %Making the T1 field appear when selected
            if app.TypeIButton.Value
                app.T1EditField.Visible = 'on';
                app.T1EditFieldLabel.Visible = 'on';
            elseif ~app.TypeIButton.Value
                app.T1EditField.Visible = 'off';
                app.T1EditFieldLabel.Visible = 'off';
            end

             %Making the BC1 flux field appear when selected
            if app.TypeIIButton.Value
                app.JEditField.Visible = 'on';
                app.FluxJWm2Label_2.Visible = 'on';
            elseif ~app.TypeIIButton.Value
                app.JEditField.Visible = 'off';
                app.FluxJWm2Label_2.Visible = 'off';
            end

            %Making the BC1 convective boundary fields appear when selected
            if app.TypeIIIButton.Value
                app.hWm2KLabel_2.Visible = 'on';
                app.hEditField.Visible = 'on';
                app.TinfEditFieldLabel_2.Visible = 'on';
                app.TinfEditField.Visible = 'on';
            elseif ~app.TypeIIIButton.Value
                app.hWm2KLabel_2.Visible = 'off';
                app.hEditField.Visible = 'off';
                app.TinfEditFieldLabel_2.Visible = 'off';
                app.TinfEditField.Visible = 'off';
            end
        end

        % Selection changed function: BoundaryCondition2ButtonGroup
        function BoundaryCondition2ButtonGroupSelectionChanged(app, event)
            selectedButton = app.BoundaryCondition2ButtonGroup.SelectedObject;

            %Making the T2 field appear when selected
            if app.TypeIButton_2.Value
                app.T2EditField.Visible = 'on';
                app.T2Label.Visible = 'on';
            elseif ~app.TypeIButton_2.Value
                app.T2EditField.Visible = 'off';
                app.T2Label.Visible = 'off';
            end

             %Making the BC2 flux field appear when selected
            if app.TypeIIButton_2.Value
                app.J2EditField.Visible = 'on';
                app.FluxJWm2Label_3.Visible = 'on';
            elseif ~app.TypeIIButton_2.Value
                app.J2EditField.Visible = 'off';
                app.FluxJWm2Label_3.Visible = 'off';
            end

            %Making the BC2 convective boundary fields appear when selected
            if app.TypeIIIButton_2.Value
                app.h2Wm2KLabel.Visible = 'on';
                app.h2EditField.Visible = 'on';
                app.Tinf2EditFieldLabel.Visible = 'on';
                app.Tinf2EditField.Visible = 'on';
            elseif ~app.TypeIIIButton_2.Value
                app.h2Wm2KLabel.Visible = 'off';
                app.h2EditField.Visible = 'off';
                app.Tinf2EditFieldLabel.Visible = 'off';
                app.Tinf2EditField.Visible = 'off';
            end
        end
    end

    % App initialization and construction
    methods (Access = private)

        % Create UIFigure and components
        function createComponents(app)

            % Create UIFigure
            app.UIFigure = uifigure;
            app.UIFigure.Position = [100 100 967 736];
            app.UIFigure.Name = 'UI Figure';

            % Create UIAxes
            app.UIAxes = uiaxes(app.UIFigure);
            title(app.UIAxes, 'Heat distribution map')
            xlabel(app.UIAxes, 'X')
            ylabel(app.UIAxes, '')
            app.UIAxes.PlotBoxAspectRatio = [1 0.513833992094862 0.513833992094862];
            app.UIAxes.XTick = [];
            app.UIAxes.YTick = [];
            app.UIAxes.Position = [499 372 454 324];

            % Create Generation0ifnoneLabel
            app.Generation0ifnoneLabel = uilabel(app.UIFigure);
            app.Generation0ifnoneLabel.FontSize = 13;
            app.Generation0ifnoneLabel.FontWeight = 'bold';
            app.Generation0ifnoneLabel.Position = [16 524 148 22];
            app.Generation0ifnoneLabel.Text = 'Generation? (0 if none)';

            % Create QWm2Label
            app.QWm2Label = uilabel(app.UIFigure);
            app.QWm2Label.HorizontalAlignment = 'right';
            app.QWm2Label.FontWeight = 'bold';
            app.QWm2Label.Position = [20 500 69 22];
            app.QWm2Label.Text = 'Q (W/m^2):';

            % Create QEditField
            app.QEditField = uieditfield(app.UIFigure, 'numeric');
            app.QEditField.Position = [104 500 54 22];

            % Create DsteadystateheattransferwithorwithoutgenerationLabel
            app.DsteadystateheattransferwithorwithoutgenerationLabel = uilabel(app.UIFigure);
            app.DsteadystateheattransferwithorwithoutgenerationLabel.HorizontalAlignment =
'center';
            app.DsteadystateheattransferwithorwithoutgenerationLabel.FontSize = 16;
            app.DsteadystateheattransferwithorwithoutgenerationLabel.FontWeight = 'bold';
            app.DsteadystateheattransferwithorwithoutgenerationLabel.Position = [165 705 640
32];
            app.DsteadystateheattransferwithorwithoutgenerationLabel.Text = '1-D steady-state
heat transfer with or without generation';

            % Create BoundaryCondition1ButtonGroup
            app.BoundaryCondition1ButtonGroup = uibuttongroup(app.UIFigure);
            app.BoundaryCondition1ButtonGroup.SelectionChangedFcn = createCallbackFcn(app,
@BoundaryCondition1ButtonGroupSelectionChanged, true);
            app.BoundaryCondition1ButtonGroup.Title = 'Boundary Condition 1:';
            app.BoundaryCondition1ButtonGroup.FontWeight = 'bold';
            app.BoundaryCondition1ButtonGroup.Position = [13 563 136 143];

            % Create TypeIButton
            app.TypeIButton = uiradiobutton(app.BoundaryCondition1ButtonGroup);
            app.TypeIButton.Text = 'Type I';
            app.TypeIButton.Position = [11 97 58 22];
            app.TypeIButton.Value = true;

            % Create TypeIIButton
            app.TypeIIButton = uiradiobutton(app.BoundaryCondition1ButtonGroup);
            app.TypeIIButton.Text = 'Type II (const. flux)';
            app.TypeIIButton.Position = [11 77 123 22];

            % Create TypeIIIButton
            app.TypeIIIButton = uiradiobutton(app.BoundaryCondition1ButtonGroup);
            app.TypeIIIButton.Text = 'Type III';
            app.TypeIIIButton.Position = [11 14 65 22];

            % Create TypeIIaButton
            app.TypeIIaButton = uiradiobutton(app.BoundaryCondition1ButtonGroup);
            app.TypeIIaButton.Text = 'Type IIa (insulation)';
            app.TypeIIaButton.Position = [11 56 127 22];

            % Create TypeIIbButton
            app.TypeIIbButton = uiradiobutton(app.BoundaryCondition1ButtonGroup);
            app.TypeIIbButton.Text = 'Type IIb (symmetry)';
            app.TypeIIbButton.Position = [11 35 128 22];

            % Create PlotButton
            app.PlotButton = uibutton(app.UIFigure, 'push');
            app.PlotButton.ButtonPushedFcn = createCallbackFcn(app, @PlotButtonPushed, true);
            app.PlotButton.Position = [696 332 100 22];
            app.PlotButton.Text = 'Plot';

            % Create SimplifiedequationEditFieldLabel
            app.SimplifiedequationEditFieldLabel = uilabel(app.UIFigure);
            app.SimplifiedequationEditFieldLabel.HorizontalAlignment = 'right';
            app.SimplifiedequationEditFieldLabel.FontSize = 14;
            app.SimplifiedequationEditFieldLabel.FontWeight = 'bold';
            app.SimplifiedequationEditFieldLabel.Position = [24 82 135 22];
            app.SimplifiedequationEditFieldLabel.Text = 'Simplified equation';

            % Create SimplifiedequationEditField
            app.SimplifiedequationEditField = uieditfield(app.UIFigure, 'text');
            app.SimplifiedequationEditField.Position = [174 82 297 22];

            % Create EquationgeneralformEditFieldLabel
            app.EquationgeneralformEditFieldLabel = uilabel(app.UIFigure);
            app.EquationgeneralformEditFieldLabel.HorizontalAlignment = 'right';
            app.EquationgeneralformEditFieldLabel.FontSize = 14;
            app.EquationgeneralformEditFieldLabel.FontWeight = 'bold';
            app.EquationgeneralformEditFieldLabel.Position = [5 115 154 22];
            app.EquationgeneralformEditFieldLabel.Text = 'Equation general form';

            % Create EquationgeneralformEditField
            app.EquationgeneralformEditField = uieditfield(app.UIFigure, 'text');
            app.EquationgeneralformEditField.Position = [174 115 297 22];

            % Create BoundaryCondition2ButtonGroup
            app.BoundaryCondition2ButtonGroup = uibuttongroup(app.UIFigure);
            app.BoundaryCondition2ButtonGroup.SelectionChangedFcn = createCallbackFcn(app,
@BoundaryCondition2ButtonGroupSelectionChanged, true);
            app.BoundaryCondition2ButtonGroup.Title = 'Boundary Condition 2:';
            app.BoundaryCondition2ButtonGroup.FontWeight = 'bold';
            app.BoundaryCondition2ButtonGroup.Position = [174 563 136 143];

            % Create TypeIButton_2
            app.TypeIButton_2 = uiradiobutton(app.BoundaryCondition2ButtonGroup);
            app.TypeIButton_2.Text = 'Type I';
            app.TypeIButton_2.Position = [11 97 58 22];
            app.TypeIButton_2.Value = true;

            % Create TypeIIButton_2
            app.TypeIIButton_2 = uiradiobutton(app.BoundaryCondition2ButtonGroup);
            app.TypeIIButton_2.Text = 'Type II (const. flux)';
            app.TypeIIButton_2.Position = [11 77 123 22];

            % Create TypeIIIButton_2
            app.TypeIIIButton_2 = uiradiobutton(app.BoundaryCondition2ButtonGroup);
            app.TypeIIIButton_2.Text = 'Type III';
            app.TypeIIIButton_2.Position = [11 14 65 22];

            % Create TypeIIaButton_2
            app.TypeIIaButton_2 = uiradiobutton(app.BoundaryCondition2ButtonGroup);
            app.TypeIIaButton_2.Text = 'Type IIa (insulation)';
            app.TypeIIaButton_2.Position = [11 56 127 22];

            % Create TypeIIbButton_2
            app.TypeIIbButton_2 = uiradiobutton(app.BoundaryCondition2ButtonGroup);
            app.TypeIIbButton_2.Text = 'Type IIb (symmetry)';
            app.TypeIIbButton_2.Position = [11 35 128 22];

            % Create UIAxes2
            app.UIAxes2 = uiaxes(app.UIFigure);
            title(app.UIAxes2, 'Temperature profile')
            xlabel(app.UIAxes2, 'X')
            ylabel(app.UIAxes2, {'Temperature'; ''})
            app.UIAxes2.PlotBoxAspectRatio = [1 0.56390977443609 0.56390977443609];
            app.UIAxes2.Position = [507 26 446 280];

            % Create LengthofsurfacemLabel
            app.LengthofsurfacemLabel = uilabel(app.UIFigure);
            app.LengthofsurfacemLabel.HorizontalAlignment = 'right';
            app.LengthofsurfacemLabel.FontWeight = 'bold';
            app.LengthofsurfacemLabel.Position = [13 469 138 22];
            app.LengthofsurfacemLabel.Text = 'Length of surface (m):';

            % Create LengthEditField
            app.LengthEditField = uieditfield(app.UIFigure, 'numeric');
            app.LengthEditField.Position = [154 469 54 22];

            % Create TypeIParametersPanel
            app.TypeIParametersPanel = uipanel(app.UIFigure);
            app.TypeIParametersPanel.Title = 'Type I Parameters:';
            app.TypeIParametersPanel.FontWeight = 'bold';
            app.TypeIParametersPanel.FontSize = 13;
            app.TypeIParametersPanel.Position = [13 332 366 97];

            % Create T1EditFieldLabel
            app.T1EditFieldLabel = uilabel(app.TypeIParametersPanel);
            app.T1EditFieldLabel.HorizontalAlignment = 'right';
            app.T1EditFieldLabel.FontWeight = 'bold';
            app.T1EditFieldLabel.Position = [17 46 25 22];
            app.T1EditFieldLabel.Text = 'T1:';

            % Create T1EditField
            app.T1EditField = uieditfield(app.TypeIParametersPanel, 'numeric');
            app.T1EditField.Position = [57 46 48 22];

            % Create T2Label
            app.T2Label = uilabel(app.TypeIParametersPanel);
            app.T2Label.HorizontalAlignment = 'right';
            app.T2Label.FontWeight = 'bold';
            app.T2Label.Position = [18 11 25 22];
            app.T2Label.Text = 'T2:';

            % Create T2EditField
            app.T2EditField = uieditfield(app.TypeIParametersPanel, 'numeric');
            app.T2EditField.Position = [57 11 48 22];

            % Create kWmkLabel
            app.kWmkLabel = uilabel(app.UIFigure);
            app.kWmkLabel.HorizontalAlignment = 'right';
            app.kWmkLabel.FontWeight = 'bold';
            app.kWmkLabel.Position = [20 438 64 22];
            app.kWmkLabel.Text = 'k (W/m*k):';

            % Create kWmkEditField
            app.kWmkEditField = uieditfield(app.UIFigure, 'numeric');
            app.kWmkEditField.Position = [93 438 56 22];
            % Create BC1FluxPanel
            app.BC1FluxPanel = uipanel(app.UIFigure);
            app.BC1FluxPanel.Title = 'BC1 Flux:';
            app.BC1FluxPanel.FontWeight = 'bold';
            app.BC1FluxPanel.FontSize = 13;
            app.BC1FluxPanel.Position = [16 265 227 59];

            % Create FluxJWm2Label_2
            app.FluxJWm2Label_2 = uilabel(app.BC1FluxPanel);
            app.FluxJWm2Label_2.HorizontalAlignment = 'right';
            app.FluxJWm2Label_2.FontWeight = 'bold';
            app.FluxJWm2Label_2.Position = [7 10 95 22];
            app.FluxJWm2Label_2.Text = 'Flux J (W/m^2):';

            % Create JEditField
            app.JEditField = uieditfield(app.BC1FluxPanel, 'numeric');
            app.JEditField.Position = [117 10 42 22];

            % Create TypeIIIBC1ParametersPanel
            app.TypeIIIBC1ParametersPanel = uipanel(app.UIFigure);
            app.TypeIIIBC1ParametersPanel.Title = 'Type III BC1 Parameters:';
            app.TypeIIIBC1ParametersPanel.FontWeight = 'bold';
            app.TypeIIIBC1ParametersPanel.FontSize = 13;
            app.TypeIIIBC1ParametersPanel.Position = [16 181 227 76];

            % Create hWm2KLabel_2
            app.hWm2KLabel_2 = uilabel(app.TypeIIIBC1ParametersPanel);
            app.hWm2KLabel_2.HorizontalAlignment = 'right';
            app.hWm2KLabel_2.FontWeight = 'bold';
            app.hWm2KLabel_2.Position = [8 29 80 22];
            app.hWm2KLabel_2.Text = 'h (W/m^2*K):';

            % Create hEditField
            app.hEditField = uieditfield(app.TypeIIIBC1ParametersPanel, 'numeric');
            app.hEditField.Position = [101 29 58 22];

            % Create TinfEditFieldLabel_2
            app.TinfEditFieldLabel_2 = uilabel(app.TypeIIIBC1ParametersPanel);
            app.TinfEditFieldLabel_2.HorizontalAlignment = 'right';
            app.TinfEditFieldLabel_2.FontWeight = 'bold';
            app.TinfEditFieldLabel_2.Position = [8 1 31 22];
            app.TinfEditFieldLabel_2.Text = 'Tinf:';

            % Create TinfEditField
            app.TinfEditField = uieditfield(app.TypeIIIBC1ParametersPanel, 'numeric');
            app.TinfEditField.Position = [54 1 65 22];

            % Create TypeIIIBC2ParametersPanel
            app.TypeIIIBC2ParametersPanel = uipanel(app.UIFigure);
            app.TypeIIIBC2ParametersPanel.Title = 'Type III BC2 Parameters:';
            app.TypeIIIBC2ParametersPanel.FontWeight = 'bold';
            app.TypeIIIBC2ParametersPanel.FontSize = 13;
            app.TypeIIIBC2ParametersPanel.Position = [257 181 227 76];

            % Create h2Wm2KLabel
            app.h2Wm2KLabel = uilabel(app.TypeIIIBC2ParametersPanel);
            app.h2Wm2KLabel.HorizontalAlignment = 'right';
            app.h2Wm2KLabel.FontWeight = 'bold';
            app.h2Wm2KLabel.Position = [1 29 87 22];
            app.h2Wm2KLabel.Text = 'h2 (W/m^2*K):';

            % Create h2EditField
            app.h2EditField = uieditfield(app.TypeIIIBC2ParametersPanel, 'numeric');
            app.h2EditField.Position = [101 29 58 22];

            % Create Tinf2EditFieldLabel
            app.Tinf2EditFieldLabel = uilabel(app.TypeIIIBC2ParametersPanel);
            app.Tinf2EditFieldLabel.HorizontalAlignment = 'right';
            app.Tinf2EditFieldLabel.FontWeight = 'bold';
            app.Tinf2EditFieldLabel.Position = [1 1 38 22];
            app.Tinf2EditFieldLabel.Text = 'Tinf2:';

            % Create Tinf2EditField
            app.Tinf2EditField = uieditfield(app.TypeIIIBC2ParametersPanel, 'numeric');
            app.Tinf2EditField.Position = [53 1 65 22];

            % Create BC2FluxPanel
            app.BC2FluxPanel = uipanel(app.UIFigure);
            app.BC2FluxPanel.Title = 'BC2 Flux:';
            app.BC2FluxPanel.FontWeight = 'bold';
            app.BC2FluxPanel.FontSize = 13;
            app.BC2FluxPanel.Position = [258 265 227 59];

            % Create FluxJWm2Label_3
            app.FluxJWm2Label_3 = uilabel(app.BC2FluxPanel);
            app.FluxJWm2Label_3.HorizontalAlignment = 'right';
            app.FluxJWm2Label_3.FontWeight = 'bold';
            app.FluxJWm2Label_3.Position = [7 10 95 22];
            app.FluxJWm2Label_3.Text = 'Flux J (W/m^2):';

            % Create J2EditField
            app.J2EditField = uieditfield(app.BC2FluxPanel, 'numeric');
            app.J2EditField.Position = [117 10 42 22];
        end
    end

    methods (Access = public)

        % Construct app
        function app = HeatTransfer1D

            % Create and configure components
            createComponents(app)

            % Register the app with App Designer
            registerApp(app, app.UIFigure)

            % Execute the startup function
            runStartupFcn(app, @startupFcn)

            if nargout == 0
                clear app
            end
        end

        % Code that executes before app deletion
        function delete(app)

            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```

Appendix Figure 5 Full code for the GUI. Created in MATLAB version R2018b.