A BRIEF TUTORIAL OF THE MATLAB PDE TOOLBOX

NOAH C. RUDERMAN, IAIN CROSBY, AND TAHA BAKHTIYAR

Abstract. For this project we explore some of applications of the PDE Toolbox in MATLAB 7.10.0. The work outlined here is ideal for someone who has a rudimentary knowledge of partial differential equations and has never used the PDE Toolbox before. First we briefly explain what a partial differential equation is and what partial differential equations the PDE Toolbox addresses. We then go on to explain applications of this toolbox and a brief descrption of the steps to solve a problem. To better aquaint the user with the toolbox, two examples are explained that nummerically solve a parabolic and elliptic differential equation.

What is a Partial Differential Equation?

A differential equation is a mathematical equation in which an unknown function of several variables is related its derivaties of various orders. A partial differential equation (PDE) is similar to a differential equation in all ways except that the function of several variables is related to its partial derivatives of various orders with respect to those independent variables. Partial differential equations have applications in physics, engineering, as well as the applied sciences and are often used to model physical systems.

What kind of PDEs can the PDE toolbox solve?

There basic PDE solved by the PDE tool box is the equation

$$-\nabla \cdot (c\nabla u) + au = f \tag{1}$$

We will express this equation as Ω. Above is the elliptic equation. Ω is a bounded domain in the plane. f, a, c, and the unknown u are scalar complex valued functions defined on Ω. The software can also handle the parabolic PDE

$$d\frac{\partial u}{\partial t} - \nabla \cdot (c\nabla u) + au = f \tag{2}$$

as well as the hyperbolic PDE

$$d\frac{\partial^2 u}{\partial t^2} - \nabla \cdot (c\nabla u) + au = f \tag{3}$$

and finally the eigenvalue problem

$$-\nabla \cdot (c\nabla u) + au = \lambda du \tag{4}$$

where d is a complex valued function on Ω, and λ is an eigenvalue which will be determined by the solution to the equation.

In what areas is the PDE tool applicable?

The PDEs listed above are used as mathematical models for a wide variety of problems and applications in engineering an the applied sciences.

Date: May 4, 2011.

Some of the problems that can be solved and modeled by the elliptic and parabolic equations include:

- Heat transfer in solids
- Diffusion
- Electrostatics of dielectric and conductive media

The hyperbolic equation can be used for:

- Wave propagation in sound and electromagnetics
- Transverse motion of membranes

The eigenvalue problems can model and solve:

- Finding the natural modes of vibrations in membranes and structural mechanics problems

How do I use the PDE tool box to solve these problems?

Using the PDE tool involves several steps:

- *Defining a PDE problem*
  To define a PDE problem we must know the general form of the PDE equation we will be using the solve the problem (parabolic, elliptic, hyperbolic, eigenvalue). We must also know the geometry of the problem that needs to be solved. For simple geometries, the built in functions of the PDE GUI will suffice. Simple geometries that can be drawn manually in the GUI include any intersection, union, or difference of shapes that include squares, rectangles, squares, ellipses, and irregularly shaped polygons. For more complex systems, an equation for the geometry of the shape can be entered through the command prompt in the form of an equation. Next, boundary conditions need to be specified for edges of the object and the edges of the subdomains. The boundary conditions come in the form of Dirichlet and Neumann boundary conditions. Dirichlet boundary conditions specify the value the solution must take on the boundary of the domain. A Neumann boundary condition specifies the value the derivative of a solution is to take on the boundary of the domain.
- *Solving the PDE*
  The PDE tool box uses the Finite Element Method (FEM) for solving each of the PDE problems outlined here. The FEM involves creating a triangular mesh across the geometry of the object that the PDE will be solved for and solving at each of the verticies of the triangles. The FEM is a useful tool because often exact solutions are unattainable and often sufficiently accurate estimations suffice. The triangular mesh must be initialized and refined carefully. If the mesh is initialized but not refined or improved, some areas of the geometry may have less accurate estimations of the solution than desired. If the mesh is made to be too fine, MATLAB will spend time computing unnecessarily and slow the program down. Once the triangular mesh is in place in the correct orientation, MATLAB will solve the PDE instantly.
- *Visualizing the results on a graph*
  The results of the PDE solution can be visualized in several ways. The default option is to see a color grid with a color bar axis. Other options include the use of contour lines, a 3D plot, and animation mode. Animation

mode plots the solution to the equation as a function over several different graphs depending on what time steps were specified in the GUI.

We will cover the following examples:

(1) The heat flow problem
(2) Conductive Media with an applied direct current

Note: For the benefit of the reader, images of GUI windows accessed from the main menu will be placed after their first reference so that the reader has a general idea of what using the GUI is like. To reduce redundancy, images of the windows referenced will not appear more than once.

Example 1 (The heat flow problem)**.** The basic form of the heat equation with a heat source is

$$\frac{\partial u}{\partial t} = \alpha\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}\right) + q \tag{5}$$

where u is a function of $t, x,$ and y. In MATLAB this equation is parabolic and has the form of equation (2). We can view this as

$$d(t,x,y)\frac{\partial u}{\partial t} - \nabla\cdot(c(t,x,y)\nabla u) + a(t,x,y)u = f(t,x,y) \tag{6}$$

On the main menu select **Options** $\rightarrow$ **Applications** $\rightarrow$ **Heat Transfer** to turn on the Heat Transfer Application mode. Our PDE retains the same form but becomes

$$\rho C u_t - \nabla\cdot(k\nabla u) = Q + h(u_{\text{ext}} - u) \tag{7}$$

where ρ is the density, C is the heat capacity, k is the coefficient of heat conduction, Q is the heat source, h is the convective heat transfer coefficient, and u_{ext} is the external temperature.

We are going to model heat transfer across a plane sheet with the PDE Toolbox GUI. To bring up the PDE Toolbox GUI type `pdetool` in the MATLAB Command Window and press the enter key.

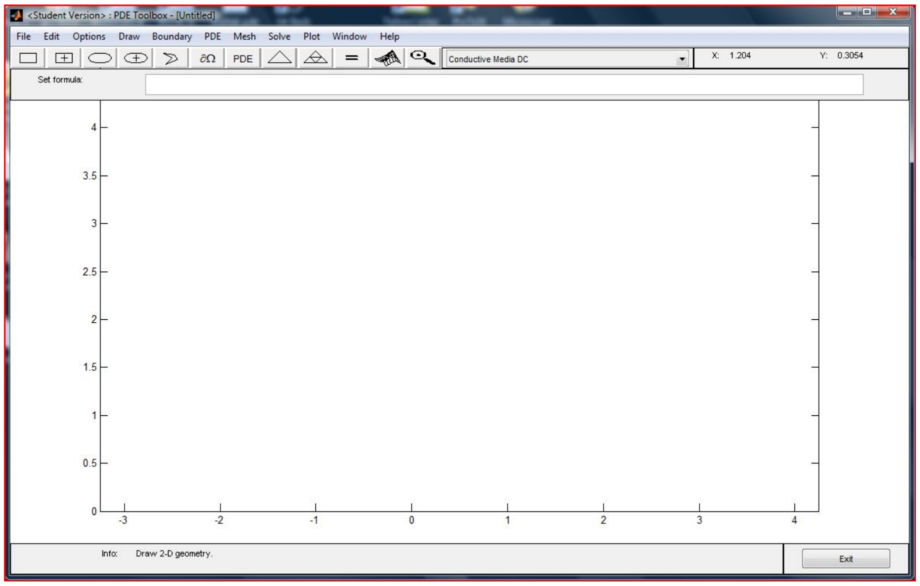
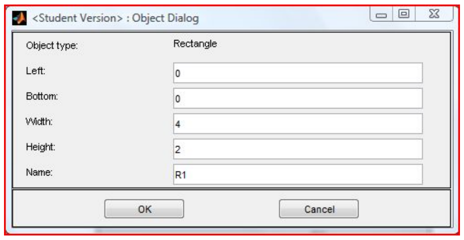


Figure 1. The pdetool GUI

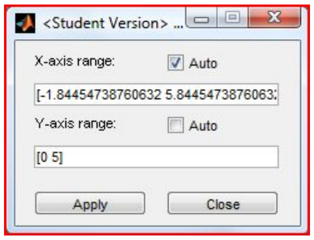
To create the geometry of our plane sheet, click the icon on the toolbar that looks like a rectangle. Once selected, click with the right mouse button on the grid to draw a square of any size. Double click on the square to pull up the object dialog box. For the left and bottom coordinates, enter 0 for both values. Enter 2 for the height and 4 for the width.


<Student Version> : Object Dialog

Object type:	Rectangle
Left:	0
Bottom:	0
Width:	4
Height:	2
Name:	R1

FIGURE 2. Object Dialog Box for a Rectangle

The axes may need to be changed to visualize the entire object. To do so, on the main menu click **Options** → **Axes Limits**, unclick the auto button for the y axis and adjust the y axis by changing it to [0 5]. Then on the main menu click **Options** → **Axes Equal**.


<Student Version> ...

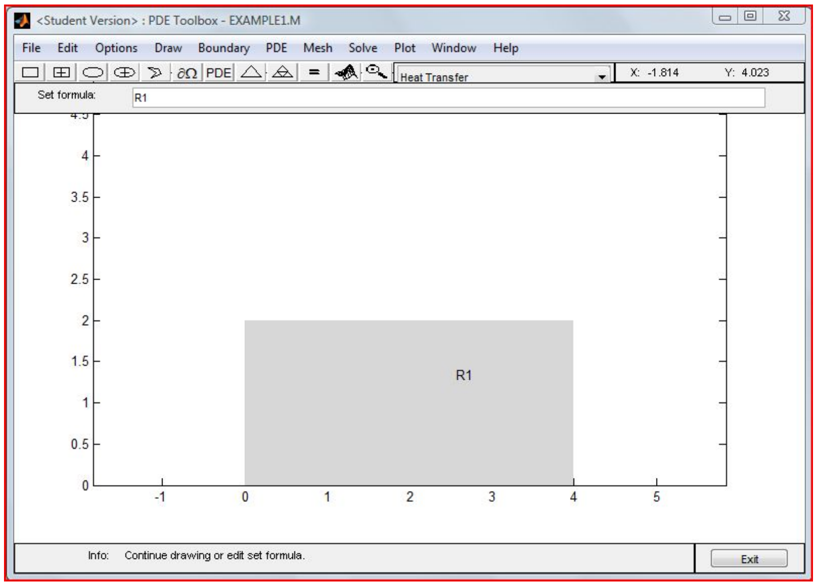
X-axis range: [✓ Auto]

[-1.84454738760632 5.8445473876063

Y-axis range: Auto

[0 5]

FIGURE 3. Axes Limits Window


<Student Version> : PDE Toolbox - EXAMPLE1.M

X: -1.814 Y: 4.023

Set formula: R1

Info: Continue drawing or edit set formula.

FIGURE 4. The Geometry of the Heat Transfer Problem

Next, click the $\partial\Omega$ icon on the toolbar to enable boundary mode. Our boundary conditions are as follows

$$\begin{aligned}
u_x(t,0,y) &= 1 && \text{left side}\\
u(t,4,y) &= 0 && \text{right side}\\
u_y(t,x,0) &= x && \text{bottom}\\
u(t,x,2) &= 0 && \text{top}\\
u(0,x,y) &= 0 && \text{initial temperature}
\end{aligned}$$

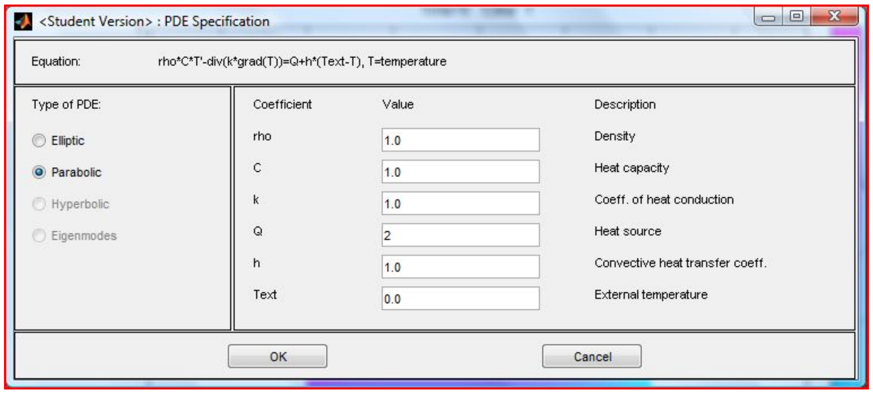
We see that the right and top sides are Dirichlet conditions, the left and bottom sides are Neumann conditions, and the last boundary condition is an initial condition.

In the Heat Transfer Application Mode, boundary conditions are of the form

$$\begin{aligned}
h\,u &= r && \text{Dirichlet}\\
n\,k\nabla(u) + q\,u &= g && \text{Neumann}
\end{aligned}$$

where h is the weight, r is the temperature, q is the heat transfer coefficient, and g is the heat flux. We may interpret Dirichlet conditions as a heat source or heat sink at a boundary in which the heat source or sink is very large compared to the slab so we have no change in this boundary condition over time. We may interpret Neumann conditions as the flow of heat across a boundary in the direction perpendicular to the surface. In this equation, n is a unit vector pointing in the direction perpendicular to the surface.

On the main menu click **PDE** $\rightarrow$ **PDE Specification.** On the left select the parabolic PDE option. Set $\rho = 1, C = 1, k = 1, Q = 2, h = 1,$ and $Text = 0$.



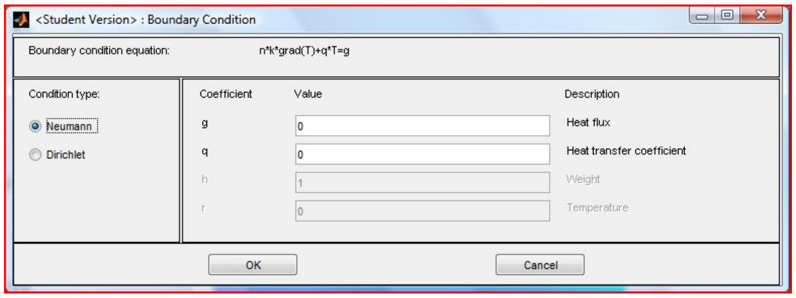
Equation: rho*C*T'-div(k*grad(T))=Q+h*(Text-T), T=temperature

Type of PDE: ( ) Elliptic [✓ Parabolic] ( ) Hyperbolic ( ) Eigenmodes

Coefficient	Value	Description
rho	1.0	Density
C	1.0	Heat capacity
k	1.0	Coeff. of heat conduction
Q	2	Heat source
h	1.0	Convective heat transfer coeff.
Text	0.0	External temperature

FIGURE 5. PDE Specification Window

To specify boundary conditions for the sides of the slab, double click on one side or click on one side and then from the main menu select **Boundary** $\rightarrow$ **Specify Boundary Conditions.** For the top and right side, select Dirichlet and change h to 1 and r to 0. For the left side, select Neumann conditions and set $g = 1$ and $q = 0$. For the bottom side, select Neumann conditions and set $g = x$ and $q = 0$.



<Student Version> : Boundary Condition

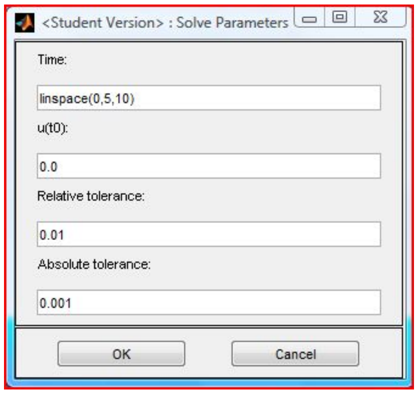
Boundary condition equation: n*k*grad(T)+q*T=g

Condition type: [✓ Neumann] Dirichlet

Coefficient	Value	Description
g	0	Heat flux
q	0	Heat transfer coefficient
h	1	Weight
r	0	Temperature

Figure 6. Boundary Condition Window

To specify the initial conditions, from the main menu select **Solve** → **Parameters.** Enter 0 for $u(t_0)$ for our initial temerature. This is also where we can specify the times that the PDE will be solved for. Input linspace(0,5,10) to specify the times solved for.



<Student Version> : Solve Parameters

Time: linspace(0,5,10)

u(t0): 0.0

Relative tolerance: 0.01

Absolute tolerance: 0.001

Figure 7. The Solve Parameters window

To initialize the mesh press the △ icon on the toolbar or use the main menu by selecting **Mesh** → **Initialize Mesh**. For greater accuracy, the mesh can be refined by clicking the icon with 4 triangles or clicking **Mesh** → **Refine Mesh.** This increases the number of points that the PDE is solved numerically reducing the error but increasing the calculation time. Generally mesh should only be refined only if more accuracy is needed. Whenever the mesh is refined, the mesh should be jiggled. You can jiggle the mesh by selecting **Mesh** → **Jiggle Mesh** from the main menu.

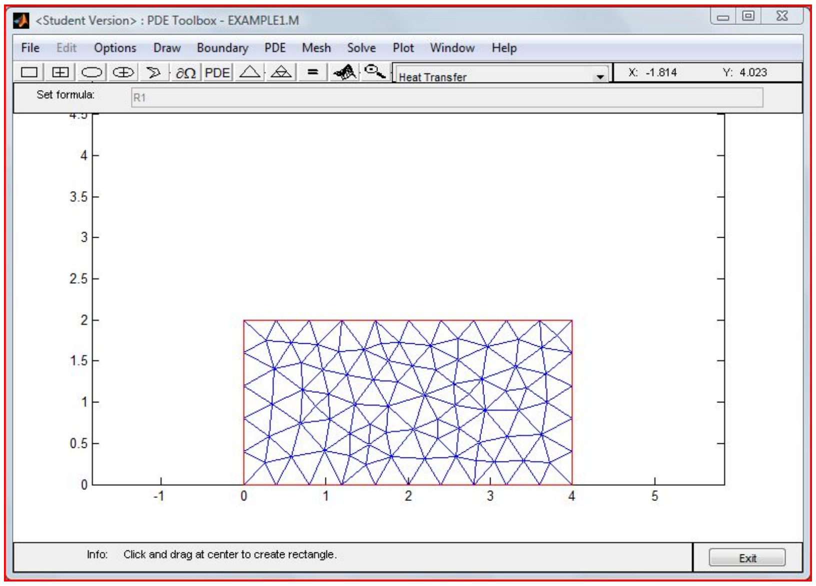

<Student Version> : PDE Toolbox - EXAMPLE1.M
File Edit Options Draw Boundary PDE Mesh Solve Plot Window Help
Heat Transfer
X: -1.814 Y: 4.023
Set formula: R1
Info: Click and drag at center to create rectangle.
Exit

FIGURE 8. A single triangle mesh

To solve the PDE simply click = icon on the toolbar. Alternatively use the main menu by clicking **Solve** $\rightarrow$ **Solve PDE.** The default solution is a colorbar of the system at the time of the last solution.

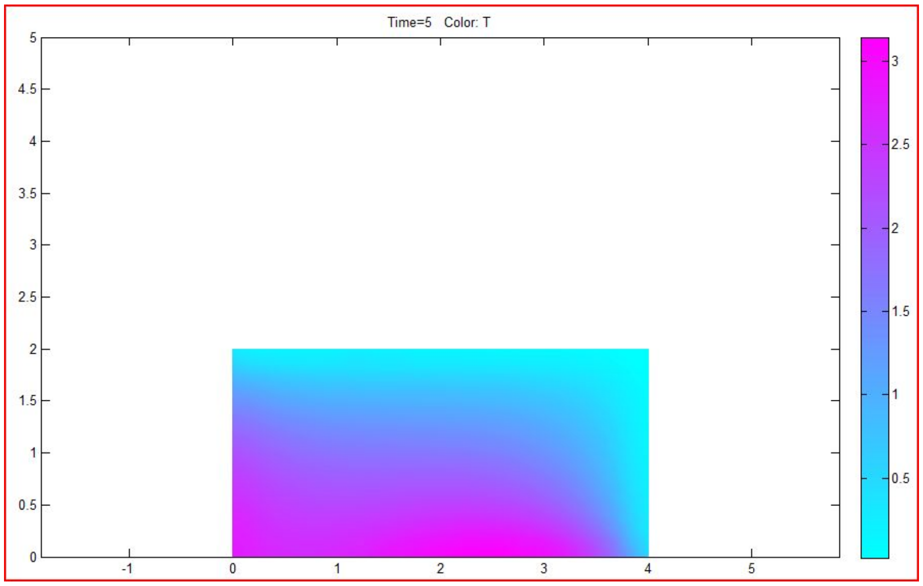
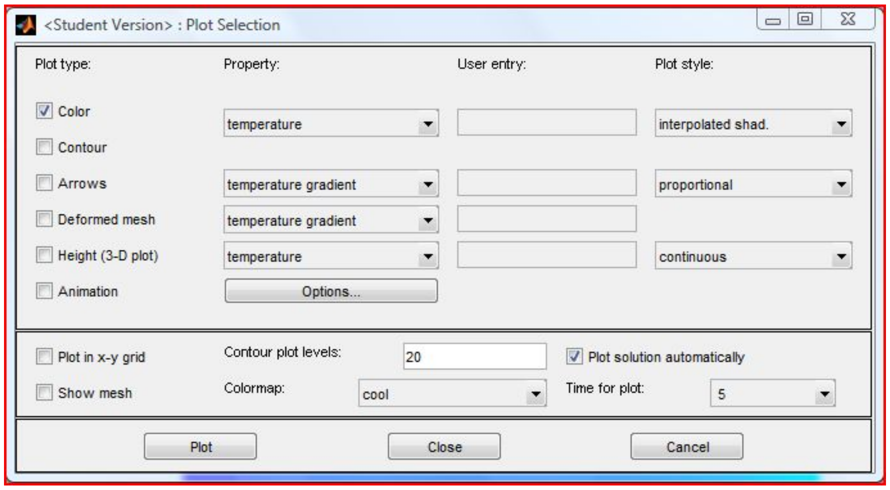


FIGURE 9. The PDE solved for the last time step

To see alternate visualizations, from the main menu select **Plot** → **Parameters.** On the left the type of plot can be specified. Select Height (3-D plot) option and the Animation option. Now plot the solution to see the animation.



<Student Version> : Plot Selection

Plot type:	Property:	User entry:	Plot style:
✓ Color	temperature		interpolated shad.
☐ Contour			
☐ Arrows	temperature gradient		proportional
☐ Deformed mesh	temperature gradient		
☐ Height (3-D plot)	temperature		continuous
☐ Animation	Options...		

☐ Plot in x-y grid	Contour plot levels: 20	✓ Plot solution automatically	
☐ Show mesh	Colormap: cool	Time for plot: 5	

Figure 10. The Plot Selection window

We can save our work by clicking on **File** → **Save** from the main menu. If we choose to open this file throught the PDE Toolbox GUI, our previous work appears back on the screen as it was at the time it was saved. Opening the file from the editor window in MATLAB opens the script file of our work.

Example 2 (Conductive Media DC)**.**

In this next example we will be modeling the potential and current flow on a metal sheet with several smaller circular metal sheets placed on top but each held at different voltages. We will make one of the circles have a negative potential, while the other two will be positive. To do this we will be using the Poisson equation. To derive this, we start with the knowledge that

$$J = \sigma E \tag{8}$$

$$\nabla \cdot J = q \tag{9}$$

$$E = -\nabla V \tag{10}$$

Note: σ is the conductivity, J is the volume current, E is the electric field, V is the electric potential, and q is our current source. We substitute the relation for J from equation (8) into equaiton (9) to get

$$\nabla \cdot \sigma E = q$$

Now we sub in E from equation (10) to get

$$\nabla \cdot \sigma(-\nabla V) = q$$

which is equal to

$$-\nabla \cdot \big(\sigma \nabla V\big) = q \tag{11}$$

which is the Poisson equation. We can see that this is in the form of an elliptic equation in which V is our solution, $c = \sigma, a = 0,$ and $f = q$.

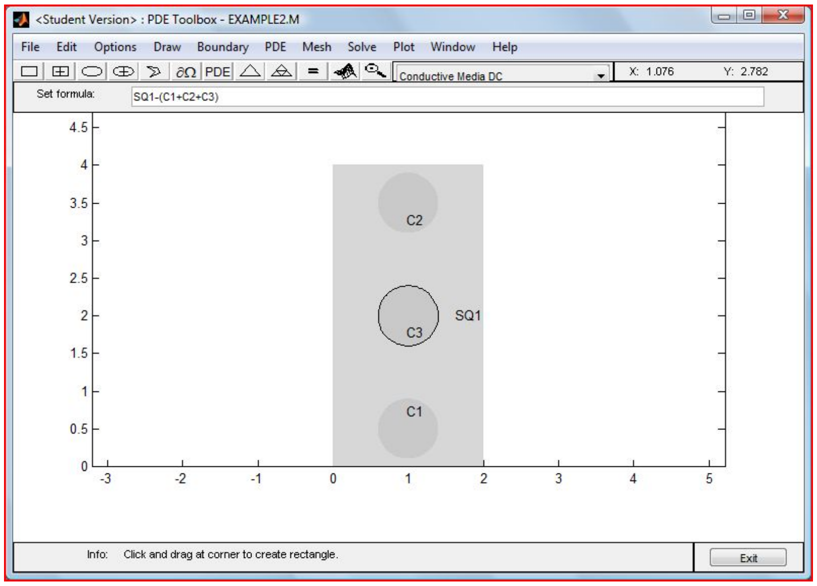
First we will create the geometry of our system. To specify our geometry we create four shapes. First create a rectangle by clicking on the rectangle icon on the toolbar and draw any sized rectangle. Double click on the rectangle to pull up the Object Dialog and enter the value 0 for the left and bottom coordinates. Make the width 2 and the height 4. On the main menu, go to **Options** $\rightarrow$ **Axes Limits.** Enter the vector [0 5] for the y axis limits. On the main menu select **Options** $\rightarrow$ **Axes Equal.**

Next click the circular icon on the toolbar and draw a circle by clicking the right mouse button and dragging. Repeat this three times. Now double click on each circle to open the object dialog box and change the parameters. Make the object designated C1 to have a center at (1, 3.5) with a radius of 0.4. Make the object designated C2 to have a center at (1, 2) with a raidus of 0.4. Make the object designated C3 to have a center at (1, 0.5) with a raidus of 0.4. Next, go to the set formula that is on the top of the GUI but below the toolbar to SQ1 - (C1+C2+C3).

Now click the $\partial\Omega$ symbol in the toolbar to enable boundary mode. The boundary conditions are as follows:

$V = 1$	on the top circular conductor
$V = 2$	on the middle circular conductor
$V = -1$	on the bottom circular conductor
$\dfrac{dV}{dn} = 0$	on the boundaries of the rectangular sheet

We can see that the first three boundary conditions are Dirichlet conditions and the last boundary condition is a Neumann condition. In the Conductive Media DC


<Student Version> : PDE Toolbox - EXAMPLE2.M
File Edit Options Draw Boundary PDE Mesh Solve Plot Window Help
Conductive Media DC
X: 1.076 Y: 2.782
C2
C3
SQ1
C1
Exit

Set formula: SQ1-(C1+C2+C3)

Info: Click and drag at corner to create rectangle.

Figure 11. The Geometry of the Conductive Media DC Problem

Application Mode, boundary conditions are of the form

$$hV = r \qquad \text{Dirichlet}$$

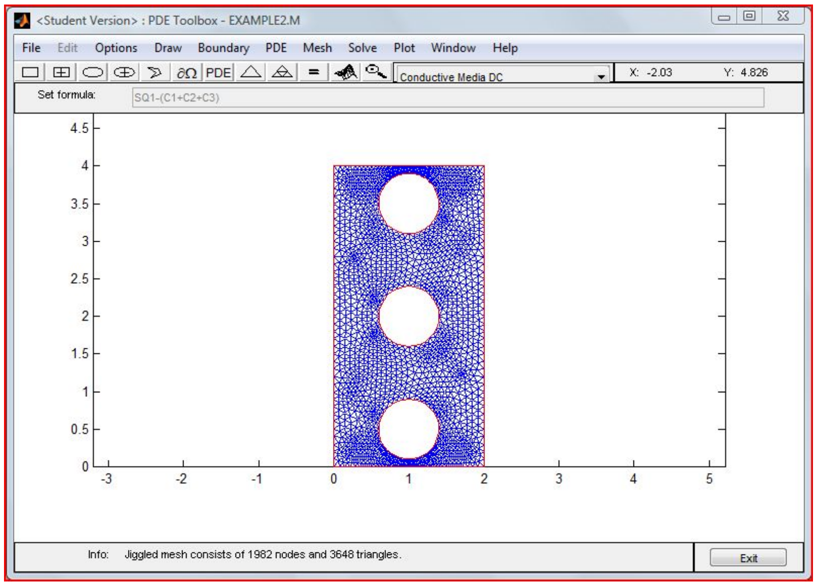
$$n\sigma\nabla(V) + qV = g \qquad \text{Neumann}$$

where h is the weight, r is the electric potential, n is a unit vector pointing perpendicular to the edge, g is the current source, and q is the film conductance.

To select boundary conditions, click on an edge and on the main menu click **Boundary → Specify Boundary Conditions.** On the edges of the the square (select multiple edges by holding shift and selecting each edge to include), select Neumann boundary conditions and set $g = q = 0$. For each circular conductor, select all edges and then select Dirichlet boundary conditions. Set $h = 1$ and set r to the appropriate potential.

Open the PDE specification box by selecting **PDE → PDE Specification** on the main menu. We the same PDE as modeled by equation (11). Set $\sigma = 1$ and $q = 0$.

Next initialize the mesh by selecting **Mesh → Initialize Mesh** from the main menu. Now select **Mesh → Refine Mesh** and jiggle the mesh once by selecting **Mesh → Jiggle Mesh.**


<Student Version> : PDE Toolbox - EXAMPLE2.M
File Edit Options Draw Boundary PDE Mesh Solve Plot Window Help
Conductive Media DC
X: -2.03 Y: 4.826
Set formula:
Exit

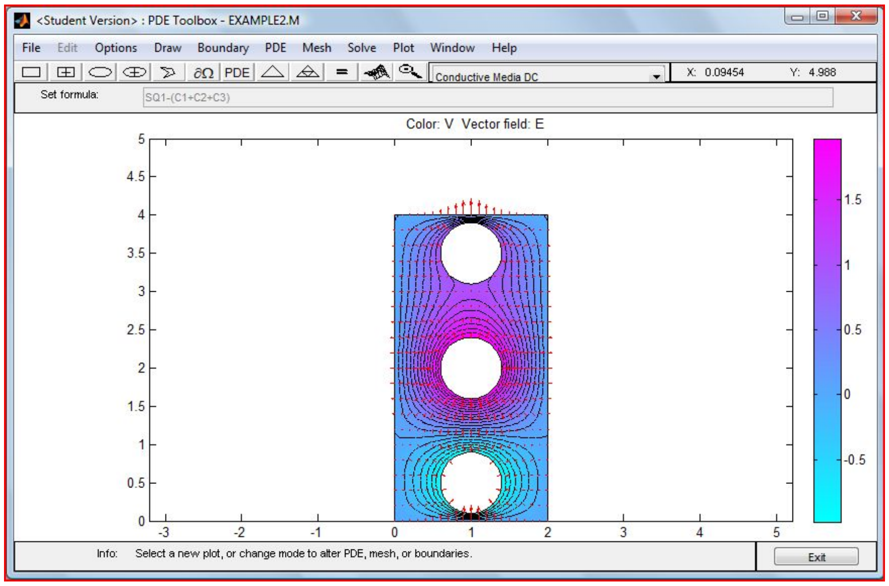
SQ1-(C1+C2+C3)

Info: Jiggled mesh consists of 1982 nodes and 3648 triangles.

FIGURE 12. Mesh mode after two mesh refinements and jiggling

Now to specify the graph we will plot, from the main menu select **Plot** $\rightarrow$ **Parameters.** On the left check the Countour and Arrow boxes under plot type. Close the box.

To solve the equation, click the $=$ button icon in the toolbar and the solution is plotted automatically.


<Student Version> : PDE Toolbox - EXAMPLE2.M
File Edit Options Draw Boundary PDE Mesh Solve Plot Window Help
Conductive Media DC
X: 0.09454 Y: 4.988
Set formula:
Color: V Vector field: E
Exit

SQ1-(C1+C2+C3)

Info: Select a new plot, or change mode to alter PDE, mesh, or boundaries.

FIGURE 13. The PDE solved for the last time step

References

[1] Peter Howard "Partial Differential Equations in MATLAB 7.0." Web. 4 May 2011
[2] MathWorks, ed. MATLAB PDE User Guide. Natick, MA: Mathworks, 2011. Print.
[3] Wikipedia contributors. "Partial differential equation." Wikipedia, The Free Encyclopedia. Wikipedia, The Free Encyclopedia, 20 Apr. 2011. Web. 5 May. 2011 http://en.wikipedia.org/w/index.php?title=Partial_differential_equation&oldid=425056169

E-mail address, Noah C. Ruderman: noah3@brandeis.edu
E-mail address, Taha A. Bakhtiyar: tahaalib@brandeis.edu
E-mail address, Iain F. Crosby: icrosby@brandeis.edu
URL: https://sites.google.com/a/brandeis.edu/cs177-final-project/