Joe Chen

Numerical Analysis

May 10, 2013

**Weber’s Disc and PDE Toolbox Documentation**

 Partial Differential Equations Toolbox (PDE Toolbox) is an add-on for Matrix Laboratory (MATLAB) that can solve partial differential equations. It lets the user specify the domain, set boundary conditions, enter the partial differential equation, create a mesh, and produce a solution in the form of data matrices. PDE Toolbox can be run either via script or via a graphical user interface (GUI). Although the script offers more flexibility, the GUI is recommended for new users. To open the GUI, type **pdetool** into the command window and hit enter.



**Figure 1. Opening the PDE Toolbox GUI.**

**Drawing the Boundaries**

 PDE Toolbox lets the user draw the specified boundaries in GUI itself with its **Drawing Mode**. Before drawing the boundaries, the user should consider changing a few settings for convenience. These can be adjusted via the “Options” dropdown menu accessible from the top of the GUI.



**Figure 2. Options menu**

Selecting **Grid** from this menu turns on dotted lines that show you where each division ends and begins, as shown in Figure 3.



**Figure 3. Grid option on.**

Also selecting the **Snap** option ensures that all your shapes align along either a dotted line or an outside edge. Even if the shape boundary isn’t perfectly on a dotted line on all sides, this option is recommended because usually at least one edge needs to be aligned along an axis, and small adjustments can be made later on.

 Next, the user should set the axes. These can be adjusted at any point in the process by selecting **Axes Limits...** from the Options menu. To start, you should set it so that you can see the entire boundary. Figure 4 shows how to set the boundary.



**Figure 4. Setting the axes limits.**

The range is specified by the 2-element vector. “[0 100]” specifies that the minimum is 0 and the maximum is 100. After both the x and y ranges are set, hit “Apply” to apply changes and “Close” to close the dialog.

 PDE Toolbox sets the defined boundaries by adding and subtracting basic shapes. These are created by selecting one of five buttons in the top left corner of the window.



**Figure 5. Selectable geometries (shown in red circle).**

The far left geometry lets the user create a rectangle whose corner starts from the initial click. The second button also creates a rectangle, but the first click instead sets the center of the rectangle. The third button creates an oval where the first click sets the top left corner of a rectangle that bounds the oval on four sides. The fourth button also creates an oval, but the first click sets the center of the oval. The last button lets the user create a polygon by drawing several line segments (the polygon must be closed and must not intersect itself to be valid).



**Figure 6. Example geometries creatable in PDE Toolbox.**

 If you mess up a shape and need to resize or move it, double click on the shape to bring up a popup dialog. The fields vary based on the type of shape you are editing, but these fields also let you alter shapes to more shapes and sizes than limited to by the gridlines.



Figure 7. Shape dialog for a rectangle.

To delete a shape, single click on the shape (does not always highlight) and hit the “Delete” key on your keyboard.

 To create more complex geometries, overlay shapes, as shown in Figure 6, and then alter the shape formula. Figure 8 shows the shape formula field in the main PDE Toolbox GUI.



**Figure 8. Shape formula field, which alters how shapes are combined.**

By default, whenever a new shape is created, it is added in by its shape name, meaning the area is added to the drawing (C is circle, E is ellipse, R is rectangle, P is polygon, and SQ [not depicted] is square). Each shape must have a unique name. Although each shape is given a new name by default whenever it is created, but it’s possible to create conflicting names by renaming via the shape dialog in Figure 7.

 Changing a shape to negative ‘-‘ subtracts the shape from a given geometry. By doing this, the user can create cutouts and thus more complex shapes. Figure 9 is an example of a rectangle that cuts out a piece of the polygon. Notice that R1 is negative in this window.



**Figure 9. Subtracting shapes to create complex geometries.**

The difference is not noticeable in **Drawing Mode** but will become apparent in other modes later on. Also, note that order of operations does matter, so shapes will be added and cut in order from left to right, meaning R1 must be after P1 to out a piece of P1.

**Defining the Geometry for Weber’s Disc**

A rectangle can accurately capture the results for Weber’s Disc since it assumes radially symmetric cylindrical coordinates. Thus, the ‘r’ variable can be mapped to ‘x’ in PDE Toolbox and the ‘z’ variable can be mapped to ‘y’ in PDE Toolbox to solve this problem. Below is the geometry used to solve Weber’s Disc.



**Figure 10. Geometry for Weber’s Disc.**

Notice that there’s three squares total: one that spans the entire range, one in the bottom left corner with a side length of 1, and another square in the bottom left corner with a side length of 2. All of these shapes are added together. The extra squares in the corner add extra mesh points to that corner (which improves accuracy because that’s the region with the most rapid change) and also allows for the mixed boundary conditions on the horizontal axis, assuming that the radius ‘a’ is 1.

**Defining Boundary Conditions**

 Next, the boundary conditions must be defined. By definition, boundary conditions are defined at the most extreme edges of your geometry. For simplicity, the geometry in Figure 11 will be used for the rest of the demonstration.



**Figure 11. Example geometry with a circle cut out of a square and a rectangle in the corner.**

 To enter **Boundary Mode**, use the **CTRL+B** shortcut or select the button circled in Figure 12.



**Figure 12. Boundary Mode button.**

The drawing itself will change to reflect boundaries. The shapes will no longer be filled in, and the edges of each shape will be shown in either red or grey. Red lines with arrows are boundaries on the exterior edges. This can be seen in Figure 13.



**Figure 13. Updated drawing in Boundary Mode.**

 Notice that the internal circle is selectable while the interior edges of the rectangle are not. This is because the circle is a cutout, meaning the edges of the circle are “outside” while the inside edges of the rectangle are “inside.” However, also notice that the outer edges of the rectangle are not only selectable but also selectable on that small range (0 < x < 20 and 0 < y < 10). This is how mixed boundary conditions are defined.

**Using the Nonlinear Solver for Weber’s Disc**

Weber’s Disc requires a special setting before you can define its boundary conditions. Select **Parameters** under the **Solve** dropdown menu.



**Figure 14. Parameters option under solve.**

The dialog shown in Figure 15 will then appear.



**Figure 15. Solve parameters dialog.**

In this window, check the “Use nonlinear solver” button in the top right corner. This lets you use x, y, ux (δu/δx), and uy (δu/δx) while defining boundary conditions. The default settings are fine for most purposes, including Weber’s Disc. Hit “OK” when ready to close the dialog.

 In **Boundary Mode**, all boundaries are set to Dirichlet prescribed conditions at u=0 by default. Double click on a boundary line to redefine it according to a problem. A dialog, such as the one in Figure 16, should appear.



**Figure 16. Dirichlet boundary condition dialog.**

 A Dirichlet boundary condition is one of the two types of boundaries that PDE Toolbox can handle. A Dirchlet boundary condition is when one edge is prescribed to a certain value. The simplest case is if it’s prescribed to a constant (e.g. 0). However, with the nonlinear solver selected, you can define the ‘r’ field as any combination of x, y, ux, uy, and constants. You must use element-by-element mathematical operations if you use these values though (e.g. ‘.\*’ ‘./’). In most cases, ‘h’ should be left as ‘1’ and all manipulation should be done to ‘r’ since the boundary condition equation is h\*u=r where u is the variable that PDE Toolbox is solving for.

 A Neumann boundary condition is one where the flux normal (perpendicular) to the surface is defined. To define a Neumann boundary condition, you also start by double clicking a red boundary.



**Figure 17. Neumann boundary conditions dialog window.**

In the dialog that pops up, select the “Neumann” button on the left side. You can now enter values for ‘g’ and ‘q’ only. There are more variables in this boundary condition equation (n\*c\*grad(u)+qu)=g, so the values are decomposed in Table 1.

**Table 1. Variables in the Neumann boundary equation.**

|  |  |
| --- | --- |
| **Variable** | **Meaning/Usage Notes** |
| n | Not a variable input but rather a unit vector that points in the direct normal to the selected boundary. |
| c | This value is set by the PDE, which will be defined in a later section. Usually, this value is 1 or -1.  |
| u | The variable that PDE Toolbox is solving for. |
| q | The coefficient of the ‘qu’ term. This value is usually 0. |
| g | The variable input, similar to ‘r’ with a Dirichlet boundary conditions. Usually 0 to indicate no flux in the normal direction. |
| grad(u) | The gradient operator on the variable PDE Toolbox is solving for. This turns u into the flux in the normal direction (e.g. δu/δx for a vertical boundary) |

**Boundary Conditions for Weber’s Disc**

Although most boundary conditions are given for Weber’s disc, some must be derived from the exact solution to optimize accuracy. Simply put, values are plugged into the exact solution to derive the concentration equation for that particular boundary. Figure 18 shows the normal domain and the associated boundary conditions for Weber’s disc assuming that the disc radius ‘a’ is 1 and the concentration at infinity ‘Co’ is 100.



**Figure 18. Weber’s Disc boundary conditions.**

Be careful to make sure that the left wall’s bottom boundary conditions get defined. By placing the boxes in the corner for the mixed boundary conditions for the horizontal axis, the boundary is also split on the vertical axis. For Weber’s disc, the boundary should be the same all the way up and down the left edge, so you can just define it as such.

**Defining the Partial Differential Equation**

 The PDE can be defined by pushing the PDE button circled in Figure 19. This pops up the dialog in Figure 20.



**Figure 19. Define PDE button in main GUI.**



**Figure 20. Define PDE popup window.**

Although the PDE can accept elliptic, parabolic, hyperbolic, and eigenmodes PDEs, this guide focuses on Elliptic PDEs. Elliptic PDEs are of the form –div(c\*grad(u))+a\*u=f. When c is a constant, it can be pulled out of the divergence and the equation becomes –c\*laplacian(u)+a\*u=f or –cΔu+au=f because div(grad(u)) = laplacian(u) = Δu. (Note: In 2D Cartesian coordinates, Δu decomposes to Eq. 1).

|  |  |  |
| --- | --- | --- |
|  | $$∆u=\frac{∂^{2}u}{∂x^{2}}+\frac{∂^{2}u}{∂y^{2}}$$ | (1) |

As an easy example, the PDE Δu = 0 is chosen, meaning ‘c’ is -1, ‘a’ is 0, and ‘f’ is 0. Hit “OK” after finishing your inputs to return to the main GUI.

**Partial Differential Equation for Weber’s Disc**

Weber’s Disc is actually defined as Eq. 2.

|  |  |  |
| --- | --- | --- |
|  | $$0=\frac{∂^{2}C}{∂r^{2}}+\frac{∂^{2}C}{∂z^{2}}+\frac{1}{r}\frac{∂C}{∂r}$$ | (2) |

However, the boundary setup assumes that ‘r’ is ‘x’, ‘z’ is ‘y’, and ‘C’ is ‘u’, so the first two terms can be combined together into the two dimensional Laplacian via Eq. 1. So your inputs for Figure 20 are ‘c’ is -1, ‘a’ is 0, and ‘f’ is ux./x. Note that the ‘f’ declaration only works with the nonlinear solver setting.

**Mesh Generation**

PDE Toolbox solves a PDE with its associated boundary conditions by building a mesh of grid points and numerically evaluating the PDE for each mesh point. Although there are many mesh settings that offer some flexibility, the default mesh creation settings suffice for most applications. To initialize the mesh, select “Initialize Mesh” under the “Mesh” dropdown at the top of the PDE Toolbox main GUI. Alternatively, hit CTRL+I.



**Figure 21. Initialized mesh.**

The circled info statement at the bottom tells you how many triangles and how many mesh points you have after initialization. However, with only the 190 mesh points shown above, the accuracy of the result is limited—notice how the circle in the middle looks more like an octagon than a circle. By selecting “Refine Mesh” under the “Mesh” dropdown or by hitting CTRL+M, you can increase the number of mesh points before calculation. Refining the mesh twice gives 2752 nodal points and the mesh shown in Figure 22, which has a much more circular center than the original mesh.



**Figure 22. Refined mesh.**

**Mesh for Weber’s Disc**

The solution for Weber’s Disc is generated by initializing the mesh and refining it five times, which results in 219,009 nodal points. Figure 23 is a zoom in of the mesh.



**Figure 23. Weber’s Disc refined mesh.**

**Solving the PDE**

 With all the parameters set, PDE Toolbox can solve the problem. Simply hit “Solve PDE” under the “Solve” dropdown at the top of the main GUI toolbar to solve the PDE (alternatively, hit **CTRL+E**).



**Figure 24. Solve PDE command.**

PDE Toolbox will solve the problem and automatically generate a color map of that solution.



**Figure 25. Example solution of PDE Toolbox.**

**Solution for Weber’s Disc**

PDE Toolbox produces the following plot as the solution to Weber’s Disc.



**Figure 26. Weber’s Disc solution.**

**Exporting the Solution**

 Although the graphical solution is helpful, having the numerical values is much more beneficial. Exporting this data to the MATLAB workspace could allow for extra plotting flexibility and opens up the opportunity for error analysis.

So that you only have to do this once, start by clearing your MATLAB Workspace. This can be done by typing **clear** into the MATLAB original command window. Then go back to PDE Toolbox to begin exporting data.

First, export the mesh by selecting “Export Mesh” from the “Mesh” dropdown at the top of the PDE Toolbox window. Another window will pop up asking for a name for points, edges, and triangles. The default variables are sufficient, but only points will be needed for most data analysis. Hit OK when ready to export the mesh to the Workspace.



**Figure 27. Export mesh window.**

 Similarly, export the actual solution by selecting “Export Solution…” from the “Solve” dropdown at the top of the PDE Toolbox window. The default ‘u’ variable is also sufficient, so hit OK when ready.



**Figure 28. Export solution window.**

The variable ‘p’ is a matrix with two rows. The first row represents the ‘x’ values while the second row represents the ‘y’ values. To pull off the ‘x’ values, use x= p(1,:). To pull off the ‘y’ values, use y=p(2,:). The variable ‘u’ is a single row variable with its values split across columns. These values correspond to values from the nodal points and can be manipulated as such.

After everything is exporting, it’s recommended that you save this workspace as is with the command **save nameOfSave**. The workspace can later be restored with the command **load nameOfSave**.

**Exporting Weber’s Disc**

The procedure for exporting the Weber’s Disc solution follows the same general procedure outlined above. As a reminder though, the ‘x’ from the solution corresponds to ‘r’ and the ‘y’ from the solution corresponds to ‘z’ in this particular problem.

**Round Drop Geometry & Other Boundary Conditions**

 Weber’s Disc can be adapted to other research problems. Dr. Kelly-Zion is currently studying the rate of evaporation of a drop to the air around in. In this problem, the drop can be modeled as a flat cylindrical disc, similar to Weber’s Disc. However, the boundary conditions change because the concentration on the disc itself is C­o while the concentration at infinity is zero. The exact solution from Weber’s Disc applies when transformed appropriately for the new boundary conditions. The old geometry and PDE works, but the new boundary conditions are defined in Figure 29.



**Figure 29. New boundary conditions.**

These conditions can produce a solution, and you still have an exact solution to check against.

 Next, to make the approximation more realistic, the drop is no longer considered flat. Although it would not be perfectly round either, a circular shape is chosen for demonstrative purposes here. Thus, the drop is assumed to have a radius of ‘1’ and be circular. A zoom-in of the geometry, essentially a circle subtracted from a square, is shown in Figure 30.



**Figure 30. Close up of round drop geometry.**

The partial differential equation from before still applies, as it simply describes diffusion. However, the exact solution no longer works because the problem is now in a different domain. Therefore, new boundary conditions are needed. The new boundary conditions are compiled in Figure 31.



**Figure 31. Boundary conditions for round drop geometry.**

 In the bottom left corner, the entire curve is prescribed as Co or 100. Since the curve is the surface of the drop, the air-drop has the highest concentration or Co. The bottom is still Neumann because there is a solid surface preventing the vapor from flowing below z=0.

The left edge of the boundary (r=0) is similarly set to Neumann conditions (zero flux normal to the surface) because of the radial symmetry assumption. By conservation of mass, the mass entering the radial center line must be equivalent to the mass exiting that line. However, according to radial symmetry, the net flux leaving that line must be mirrored in all directions radially. The only way that both of these conditions can be satisfied is if the flux normal to the radial center line is 0. Thus, this boundary conditions works.

The walls at z=100 and r=100 are prescribed as 0 concentration because there is no longer an exact solution that accurately describes the concentration at these edges and PDE Toolbox cannot define infinite boundary conditions. Thus, this approximates the actual solution by assuming that, at 100 units is far enough away so that the vapor concentration is close to 0 at that distance.