PdnMesh: A Tutorial Introduction

(Finite Element Program)

Sarod Yatawatta

sarod@users.sf.net

PdnMesh: A Tutorial Introduction: (Finite Element Program)
by Sarod Yatawatta

Copyright © 2001, 2004 Sarod Yatawatta

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with no Invariant Sections, with no Front-Cover Texts, and with no Back-Cover Texts.

Table of Contents

1. Installation and Running the Program 1
1.1. Introduction 1
1.2. The Program Internals 1
1.3. Program Input 1
1.4. Program Output 1
1.5. Installing the program 1
1.6. Running the program 2
1.7. PdnMesh Graphical User Interface 2
1.7.1. File Menu 2
1.7.2. Solve Menu 3
1.7.3. View Menu 3
1.7.4. Popup Menu 3
2. Input File Creation 5
2.1. Typical problems 5
2.2. Boundary Conditions 5
2.3. File Format 5
2.3.1. A Simple Example 6
2.3.2. Points 6
2.3.3. Edges 6
2.3.4. Boundaries 7
2.4. Fine Tuning 8
3. Converting DXF Files 10
3.1. A Worked Example 10
3.2. Editing Points 10
3.3. Editing Lines 11
3.4. Editing Boundaries 11
Further Information 13
Index 14

List of Tables

1-1. Command Line options 2

List of Figures

1-1. The contour plot 1
1-2. 3D contour plot 1
1-3. The gradient plot 1
1-4. The mesh 1
1-5. Solve Options Dialog 3
1-6. Triangle Info Dialog 4
2-1. A Solenoid 6
2-2. The problem geometry 6
2-3. Modified Solenoid 9
3-1. Magnetic Circuit 10
3-2. Magnetic Circuit solved 10
3-3. Editing Points 11
3-4. Editing Points Dialog 11
3-5. Editing Lines Dialog 11
3-6. Editing Boundaries 11
3-7. Editing Boundaries Dialog 12

List of Equations

2-1. Poisson Equation 5
2-2. Helmholtz Equation 5

Chapter 1. Installation and Running the Program

1.1. Introduction

pdnMesh is a 2 Dimensional finite element analysis program. It can be used to solve Electrostatic, Magneto-static, Heatflow and Fluid dynamic problems. It can also do post processing to visualise the solution. It is possible to import CAD DXF files (ASCII) into pdnMesh and convert them to a format for problem definition.

1.2. The Program Internals

If you are trying to hack the source, maybe this is not the correct article to read. The purpose of this tutorial is not to delve into the internals of the program. You are not required to know hardly anything about how it is written. Basically, pdnMesh performs a *Constrained Delaunay Triangulation* of the given region to generate a triangular mesh. It then assimilates a set of equations to be solved according to the mesh of the potentials of each node in the mesh. After solution of the equations, it performs post-processing to enable the user to easily visualise the solution.

1.3. Program Input

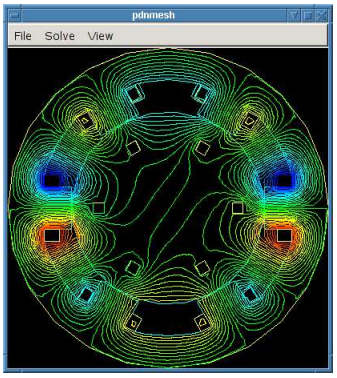
PdnMesh accepts an ASCII file as its input. The format of this is given in Chapter 2. It is also possible to use Drawing Interchange Format (DXF) files, generated by such programs as QCAD and AutoCAD to define the problem geometry.

1.4. Program Output

Perhaps the most important question to a user may be what the program can produce as output. It can produce following files types:

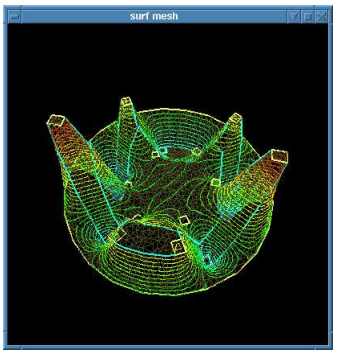
- The contour plot - encapsulated postscript file in colour and grayscale.
- The mesh - encapsulated postscript file.
- The gradient plot as an encapsulated postscript file.
- The mesh as a data file to be used by any other program.

Figure 1-1. The contour plot



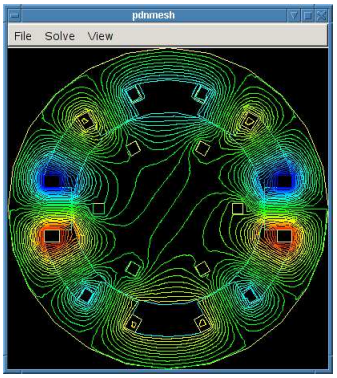
The above figure is a magnetic flux flow of a DC motor.

Figure 1-2. 3D contour plot



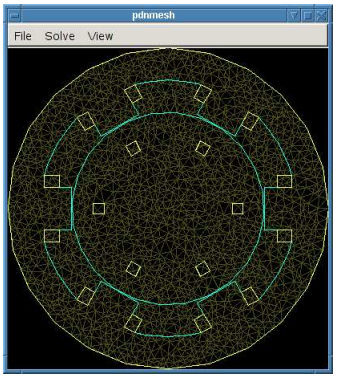
It is possible to visualise the contour plot in 3D using OpenGL.

Figure 1-3. The gradient plot



This is the magnitude and direction of the gradient of the magnetic potential.

Figure 1-4. The mesh



The above figure gives the mesh used to solve the problem. You can also see the different types of boundaries of the problem.

1.5. Installing the program

You can get the program in source code form as well as in binary form. I suggest you try the web page *http://pdnmesh.sourceforge.net/* first. Depending on your operating system, there might be binary versions available. However, they may be old and outdated. Source distribution will be in the form of a gzipped tar archive called `pdnmesh-*.*.*.tar.gz`, where the asterix will signify the available version such as *0.0.4*. For compilation, an ANSI C

compiler is required. GCC is ideal for this. The GUI is written in GTK. So you must have GTK and GTKGLExt installed (Sorry KDE Fans!). Apart from this, you need an implementation of OpenGL installed. In Linux and *BSD, you can use Mesa for this. In any UNIX like platform, the X11 header files are also required.

The basic commands for compilation in a UNIX like platform are:

```
$ gzip -dc pdnmesh-*.*.*.tar.gz | tar xvf -
$ cd pdnmesh-*.*.*/
$ ./configure
$ make
$ make install
```

The program will be installed in `/usr/local/` directory by default. In order to override this, give the option **`--prefix=/path/to/install`** to **`./configure`**. For more information on installation, please refer the documentation supplied with the program and documentation about autoconf and automake.

1.6. Running the program

Pdnmesh should be invoked at the command prompt. It can be run as a pure text mode program. However, if you need to exploit its full features, you need to run it in graphical (X window) mode. Pdnmesh also accepts command line options. Pdnmesh is typically invoked by:

```
$ pdnmesh -i input.file
```

The following flags can be used:

Table 1-1. Command Line options

option	description
`-h`	Print help
`-v`	be verbose
`-a value`	set minimum area of each triangle to ’value’
`-b value`	set maximum badness of each triangle to ’value’
`-c value`	set contour levels to ’value’
`-r value`	limit iterations to ’value’
`-nx`	work only in text mode, dump output to files
`-contours`	display contours by default
`-mesh`	display mesh by default
`-gradient`	display gradient by default
`-V`	print version

1.7. PdnMesh Graphical User Interface

When you run the program, you will see the main window as in Figure 1-1. The menubar has 3 menus.

1.7.1. File Menu

File⟶Open opens a new coordinate file.

File⟶Import DXF imports a CAD DXF file. For more information on this, please refer to Chapter 3.

File⟶Save saves contour plot, mesh, gradient etc.

File⟶Quit exits the program.

1.7.2. Solve Menu

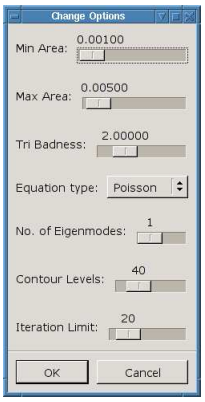
Solve⟶Mesh+Solve Once regenerates the mesh and solves the problem, using only one iteration.

Solve⟶Solve Once solves the problem without generating the mesh again.

Solve⟶Mesh+Solve Iteratively generates the mesh, and solves the problem iteratively, until iteration limit has reached. In each iteration, the mesh is refined.

Solve⟶Change Options... dialog window for changing solution options. See Figure 1-5 for more detail.

Figure 1-5. Solve Options Dialog



The options that can be changed using the above dialog are as follows:

- Min. Area: Minimum area of a triangle of the mesh.
- Max. Area: Maximum area of a triangle of the mesh.
- Tri. Badness: Maximum badness of a triangle in the mesh. Badness is the ratio of the longest edge to the shortest edge of a triangle.
- Equation Type: Poisson for Poisson and Laplace equations. Helmholtz for homogeneous Helmholtz equation (eigenvalue problems).
- No. of Eigenmodes: Only for Helmholtz equation, the number of eigenmodes to find, starting from the dominant mode.
- Contour Levels: No. of contour levels to plot
- Iteration Limit: Maximum no of iterations in the iterative form of solution.

Note that the units used in all of the above parameters are normalised.

1.7.3. View Menu

View⟶Mesh draws the mesh.

View⟶Contours draws the contours.

View⟶Grad draws the gradient.

View⟶Legend draws the legend.

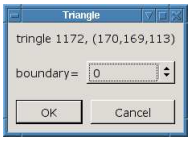
1.7.4. Popup Menu

This is the menu you get from right clicking the mouse. It has following items.

Popup⟶0_Mode switch to default mode. Basically does nothing. Useful if you want to stop zooming etc.

Popup⟶Triangle Info Mode Switch to retrieving and changing properties of triangles. In this mode, once you left-click on a triangle, you get a dialog like Figure 1-6 and you can change the boundary of the triangle.

Figure 1-6. Triangle Info Dialog



Popup⟶Triangle Split Mode In this mode, once you left-click on a triangle, it will be split. This is useful if you need to manually refine the generated mesh.

Popup⟶Zoom Window In this mode, you can select a rectangular area using the mouse left button and zoom in.

Popup⟶Zoom Back Once you select this option, the display will zoom back to previous view.

Popup⟶Zoom All Once you select this option, the display will zoom to global view.

Popup⟶3D Window Once you select this option, a new window will be created to display the contour plot in 3D like Figure 1-2. In the 3D window, you can zoom and rotate using mouse buttons. Move the mouse with left pressed for rotation. Move the mouse with middle pressed for zooming.

Popup⟶Next Eigenmode This option is useful if you solve the Helmholtz equation for more than one eigenmode. By selecting this option, you can switch between each eigenmode.

Popup⟶Quit Exits the program.

Chapter 2. Input File Creation

In this chapter, we will study in detail the input file format and defining a physical problem to be solved by pdnmesh. I must apologise that I use a description and notation heavily biased to electrical engineering problems. But if you know what you are doing, you can adapt any problem to be solvable by pdnMesh.

2.1. Typical problems

It should be noted that (currently) only 2 Dimensional plane problems can be solved. Typical physical problems are:

- Electrostatics
- Magnetostatics
- Fluid Flow: Laminar Flow Problems
- Heat Flow: Steady Heat Flow
- Eigenvalue Problems: Homogeneous Helmholtz Equation

The Poisson equation in rectangular coordinates can be written as:

Equation 2-1. Poisson Equation

$$\epsilon\{\frac{\partial^2\phi}{\partial x^2}+\frac{\partial^2\phi}{\partial y^2}\}=-\rho\,,\phi=\phi(x,y)\,,\rho=\rho(x,y)$$

x and y are the rectangular coordinates. phi is a scalar, generally called the potential. rho is also a scalar, dependent on x and y. epsilon is dependent on the medium properties and energy sources available. For instance, in Electrostatics, rho would actually mean the (negative) charge density and epsilon will be the permittivity of the medium. The first step should be to convert the PDE to be solved into above form.

The scalar, homogeneous Helmholtz equation, or Wave equation, can be written as:

Equation 2-2. Helmholtz Equation

$$\frac{\partial^2\phi}{\partial x^2}+\frac{\partial^2\phi}{\partial y^2}+k^2\phi=0,\phi=\phi(x,y)$$

This is an eigenvalue problem because phi appears in the derivative as well as in the stand alone term. We do not know the value of k in this equation as well. We can find possible values for k by solving an eigenvalue problem. The eigenvalues or k gives the cutoff wave number. The eigenvectors will give us the values for phi. The smallest k will give us the dominant mode.

Note that the Poisson's equation can be considered a special case of the Helmholtz equation. In the next sections, we will see how to prepare a problem suitable for input to pdnMesh.

2.2. Boundary Conditions

The most important information in any Finite Element Formulation is the Boundaries and the Boundary Conditions. There are two kinds of boundaries:

- Dirichlet Boundaries: On these boundaries, the potential is fixed and known beforehand.
- Neumann Boundaries: On these boundaries, the potential is variable and unknown.

A conductor will generally have a fixed potential and hence be a Dirichlet boundary. On the other hand, a dielectric will generally have a Neumann boundary. However, in real life there will be no pure conductors or dielectrics. So it is really dependent on the problem.

2.3. File Format

There are two essential items that we need to know define a problem. The first one is the geometry or physical shapes. The geometry should be specified using rectangular coordinates. The other important aspect is the material properties, such as permeability, permittivity and charge density. The potentials of known boundaries can also be taken as a material property.

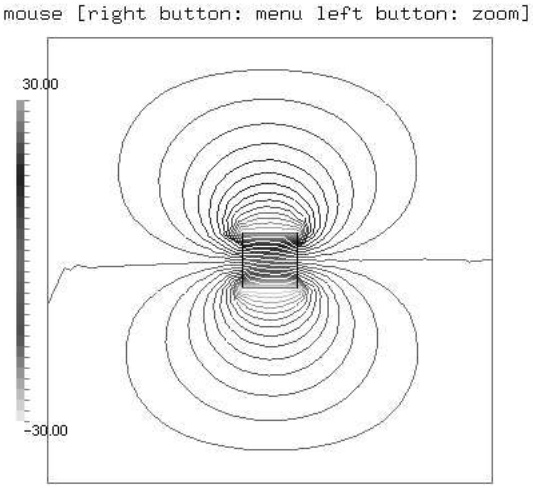
In any problem, the geometry is specified using a set of points and a set of lines or edges. So if there are any circular boundaries, they need to be represented as a set of straight lines. In order to specify material properties, you need to have a closed boundary. A boundary can be specified using a set of edges. Although this description seems confusing, an example will hopefully make things simple and clear.

2.3.1. A Simple Example

We will now consider solving a simple problem using pdnmesh. We will consider a small solenoid and plot its field pattern. The solution will be as given in the following figure.

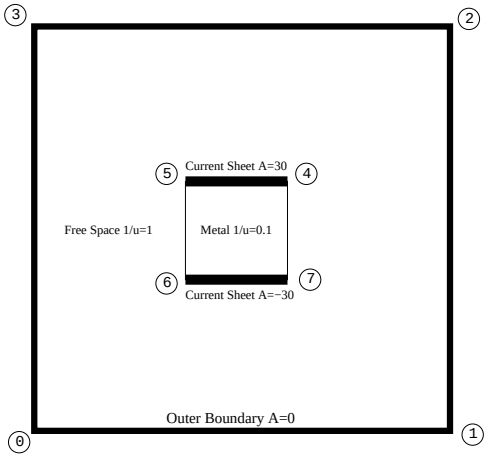
A solenoid is a coil, wound around a metal like iron. When a current is passed through the coil, the metal core acts as a magnet. Since the coil is very thin, we can assume it to be a thin sheet in which a current is flowing. Moreover, we can assume the potential of this sheet to be a constant. Note that the coil is sectioned along the axial plane in the figure. So we see two currents, one coming to us and one going down to the paper. So the potentials of the top and bottom sheets will have equal magnitudes and opposite signs.

Figure 2-1. A Solenoid



The problem can be defined as seen in

Figure 2-2. The problem geometry



2.3.2. Points

There are eight points numbered 0 to 7. Correspondingly, there are eight edges. Generally points should be numbered such that the ascending order is anti clockwise.

In the data file, the first lines should give the detail of individual points. Hence the first portion of the file may look like as given in figure. However, comments starting with a '\#' and empty lines will be ignored.

```
#this is a comment
# no. of points
8
# point data
0 -8 -8 0 1
1  8 -8 0 1
2 8 8  0 1
3 -8 8 0 1
4 1 1 30 1
5 -1 1 30 1
6 -1 -1 -30 1
7 1 -1 -30 1
```

The first line should give the number of points. Then data about each point follows line by line. In each line, the format is:

```
(point number) (x coordinate) (y coordinate) (the potential) (fixed or variable)
```

Point and line numbering should start from 0. Each point may have a known, fixed potential or an unknown, variable potential. If the potential is known, 1 should be given as the last entry of each line. In this example all points have fixed potentials. Note that points 4 and 5 have a potential of 30 and points 6 and 7 have a potential of -30. All other points have zero potential.

2.3.3. Edges

Next, the input should specify the edge data. First line should be the number of edges. Do not insert any blank line between point data and edge data. The detail is given in figure

```
# no. of edges
8
# edge data
0 0 1 1
1 1 2 1
2 2 3 1
3 3 0 1
4 4 5 1
```

```
5 5 6 0
6 6 7 1
7 7 4 0
```

For each edge, the following should be specified.

```
(edge number) (point 1) (point 2) (Dirichlet or Neumann)
```

Edges can either be Dirichlet or Neumann depending on the boundary condition. For Dirichlet edges, the potential is fixed and for Neumann edges the gradient of the potential is fixed. Hence for a Dirichlet edge, any point upon which will have fixed potential. We give 1 for a Dirichlet edge and the potential of any point is linearly interpolated from the edge points. For a Neumann edge we give 0.

For material edges, that need not satisfy any boundary condition we give 0 as well. Moreover, it should be remembered to have all edges of the outer boundary with known potential for a unique solution. Otherwise the program will warn you about singular matrices.

In figure edges 5 and 7 are not Dirichlet but that does not mean that they are Neumann. Moreover, edge 4 will have a potential of 30 because its end points both have the same potential. The same is true for edge 5. Both these edges represent the surface currents running on the periphery of the solenoid.

2.3.4. Boundaries

Finally, in the input file, we should specify the boundary detail. Boundaries need not be closed except the outer one, but almost always they are closed. Boundaries are required to define the shape of the problem as well as the material properties of the problem.

For instance, in this problem, There are two boundaries as given in figure. The point set 0, 1, 2, and 3 specify the outer boundary and the point set 4, 5, 6 and 7 represent the solenoid core.

Always, the outer boundary will be need to be specified and this should be done before other boundaries.

For any material, two parameters must be specified. For instance, in an magnetostatic problem, they can be the permeability μ and the current density J and for an electrostatic problem, they can be the permittivity ϵ and the charge density ρ. In fact, for an electro-magnetic problem, the inverse of the permeability should be given. For any other problem, the values need to be given can be determined depending on the differential equation as in .

It should be remembered that for no material, will the values of permeability or permittivity be zero. In the worst case, they would take values of free space. It is much easier to work with relative permeabilities or permittivities. Once the answer is obtained, it can be scaled back to the true value.

Sometimes the detail within a boundary may not be important or interesting. For example, within a small conductor, the potential variation can be considered negligible. Hence, the triangles within this boundary can be removed and this reduces the size of the problem. This can be done by giving 0 as a parameter in the boundary detail specification.

```
# no. of boundaries
2
# outer boundary
4 1 1 0
0
1
2
3
# solenoid core
4 1 0.1 0
4
5
6
```

```
7
```

The boundary data for this problem is given in figure. First the number of boundaries is specified. Next the outer boundary detail is given. The outer boundary has four edges, it is closed, the triangles will not be deleted, its 1/permeability is 1.0 (relative value) and its current density is 0. Next, the numbers of the edges in that boundary is given line by line. The general format for boundary information is

```
(number of edges) (remove triangles?) (1/permeability) (current density)
```

Hence, we can see that the inner boundary has 0.1 as its 1/permeability value and 0 as its current density.

The complete data file is given in figure

```
# complete data file
8
0 -8 -8 0 1
1  8 -8 0 1
2 8 8  0 1
3 -8 8 0 1
4 1 1 30 1
5 -1 1 30 1
6 -1 -1 -30 1
7 1 -1 -30 1
8
0 0 1 1
1 1 2 1
2 2 3 1
3 3 0 1
4 4 5 1
5 5 6 0
6 6 7 1
7 7 4 0
2
4 1 1 0
0
1
2
3
4 1 0.1 0
4
5
6
7
```

2.4. Fine Tuning

In the preceding example, the material properties were constant within one boundary. Now we will modify the value of charge density rho to be a variable of x and y in the space between the outer boundary and the solenoid. We can give an expression for rho, instead of a constant.

Moreover, let us remove the mesh inside the solenoid core because we are not interested in the filed pattern inside the core (besides, it is too small for us to see any pattern).

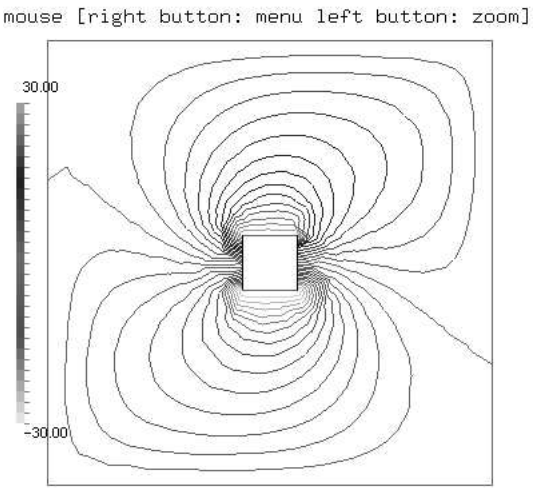
Here is the modified segment of the old input file.

```
# modified boundaries
2
# outer boundary
# with variable rho
4 1 1 0.1*(x+y)
0
1
2
```

```
3
# solenoid core
# with no triangles
4 0 0.1 0
4
5
6
7
```

Here is the new contour plot.

Figure 2-3. Modified Solenoid



You will see the skewed contours due to internal charge distribution. Moreover, the mesh will not contain any triangles inside the solenoid core.

Any expression of x and y can be given for rho. Please refer the manual page pdnmesh_input(5) for further information.

By reading this chapter you might have realized that creating an input file is a tedious task. In the next chapter we will see a way to speed thing up.

Chapter 3. Converting DXF Files

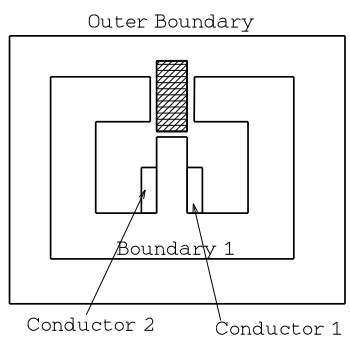
The hardest part of using pdnMesh is generating the input file. PdnMesh is NOT a CAD program. So do not expect that functionality. However, there are plenty of free CAD programs available. In order to define the geometry of the problem, you can use any one of them. Then after some simpler manipulations, you can convert the drawing you produced using the CAD program to a one suitable for pdnMesh. All you have to do is to produce a drawing of the problem, using only straight line segments and save it as an ASCII DXF file (not a binary DXF file). Any CAD program can be used so long as it produces a DXF file.

We will go through this whole process in this chapter. We have used QCAD, a free and excellent CAD program for this. You can get it from *http://www.ribbonsoft.com/qcad.html*.

3.1. A Worked Example

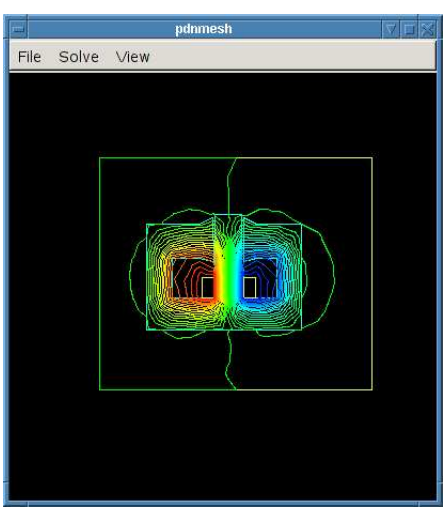
We consider a simple magnetic circuit with a moving armature and a core with two electromagnets. Please refer to Figure 3-1 for more detail. We first draw all the boundaries of this problem using straight lines. Please note that no two lines can lie on top of one another. For instance, The conductors and the iron core (Boundary 1) share some line segments. There cannot be two lines for the same boundary. Also note that the hatching of the armature is not required. It is used here just for illustration. Once again, remember you only need line segments of the boundaries in the drawing.

Figure 3-1. Magnetic Circuit



The solved problem is given in Figure 3-2.

Figure 3-2. Magnetic Circuit solved



We assume that you managed to create a drawing according to the geometry of the problem and saved it as a DXF file. Now, we run pdnMesh and choose FileImport DXF option. A dialog will appear and ask you for the file name. Just select it from the location where you saved it. After a while, if everything goes well, a new window will appear with the drawing.

If you right click on that window, you will see a menu. Let us consider this popup menu in detail.

Popup⟶Edit Points In this mode, once you left-click on a point, you can edit it. See Section 3.2 for more detail.

Popup⟶Edit Edges In this mode, you can select any edge by left-clicking and edit its properties. See Section 3.3 for more detail.

Popup⟶Edit Boundaries In this mode, you can add new boundaries and edit their properties. You can also add/remove edges of each boundary. See Section 3.4 for more detail.

Popup⟶Zoom Window In this mode, you can select a rectangular area using the mouse left button and zoom in.

Popup⟶Zoom Back Once you select this option, the display will zoom back to previous view.

Popup⟶Zoom All Once you select this option, the display will zoom to global view.

Popup⟶Save Once you select this option, you will be asked for a file name to save your coordinate file to be used as input to pdnMesh.

Popup⟶Close Closes the window. However, your work will not be saved.

3.2. Editing Points

First, you have to define the properties of all the points of the drawing. Remember, in a CAD drawing, each point will have only (x,y) coordinates. However, for our problem, we have to define the type of the point, whether they are fixed points (with a fixed potential) of variable points.

Once you select Popup Edit Points you will see each point highlighted as in Figure 3-3. You can select each point by moving the mouse cursor. Once you left click on a point, you will see a dialog window like Figure 3-4.

Figure 3-3. Editing Points

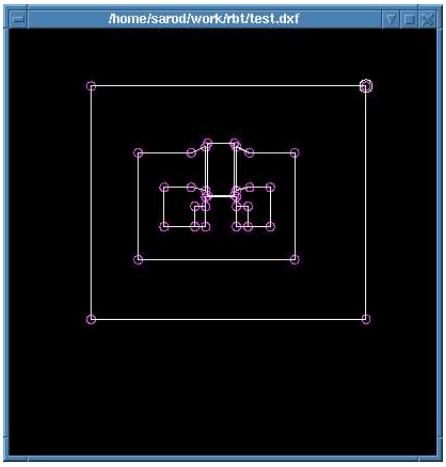
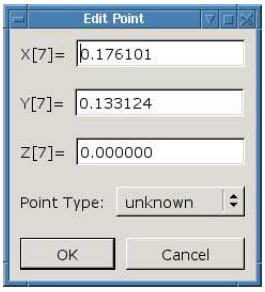


Figure 3-4. Editing Points Dialog



Using this dialog window, you can change the potential of each point. Moreover, you can select whether a point is fixed or variable. If a point is not fixed (i.e. unknown) it is assumed to be variable. We should also remember that if a point is variable, we do not need to specify its potential value, because it will be found after the solution.

3.3. Editing Lines

From Chapter 2 you know that there are two types of edges, i.e. Dirichlet and Neumann edges. So it is essential that you specify this for each and every edge in the drawing. In this mode, you will be able to left-click on any edge and edit its type. Once you left click, you will see a dialog window like Figure 3-5. Then you just have to select its type. We should note that all edges are assumed to be of type Neumann, unless specified otherwise.

Figure 3-5. Editing Lines Dialog

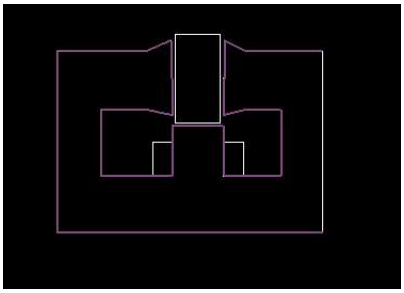
3.4. Editing Boundaries

Editing boundaries is the most important step in your DXF file conversion. In this step, you have to think about the material properties of the various shapes you have drawn in the DXF file. Moreover, you have to connect edges together to form a closed boundary. Please note that all boundaries should be closed. In other words, if you start from one point on the boundary, you should be able to return to the same point by traversing the edges of the boundary. Furthermore, all boundaries should be simply connected. That means you cannot have shapes like doughnuts for your boundaries.

Let us consider the example Figure 3-1 in more detail here. Here you will have four boundaries. Can you guess what they are? First you will have the outer boundary with the four outer edges. Next, you have the iron core, which will have 20 edges (see Figure 3-6). The hatched area forms another boundary with four edges. Finally, the two conductors form another two boundaries with four edges each.

You will also notice that some of these boundaries will include other boundaries. For example, the outer boundary will include all other boundaries. When we define boundaries, we always start with outer boundaries and finish with small, inner boundaries that do not include any other boundary within themselves. There is a very good reason for doing that. When we generate the mesh, we have to find the boundary for each triangle in the mesh. Since some boundaries are included in other boundaries, some triangles will belong to more than one boundary. However, you will see that the correct boundary for each triangle is the smallest boundary that includes that triangle. Hence, it is important to think ahead when you edit boundaries. As a rule, we always start defining the outer boundary. If you do not do that, your input will not work.

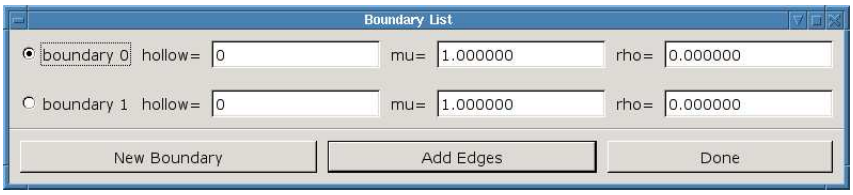
Figure 3-6. Editing Boundaries



The dialog window for editing boundaries will be like Figure 3-7. By clicking the left hand side radio buttons, you can select any boundary for editing. The fields hollow, mu, and rho are described in the input file format Section 2.3.4.

The first button on the last row is for adding a new boundary. The next button is for adding edges to the the selected boundary (using radio button). The final button is for closing the dialog.

Figure 3-7. Editing Boundaries Dialog



We should elaborate on adding edges to a boundary. Once you click on the Add Edges button, the edges belonging to that boundary will be highlighted as seen in Figure 3-6. The edges that are coloured are the edges on the boundary being edited. Just by clicking on any edge, you can add it to this boundary, or you can remove it from the boundary if it is already included. Note also that the edges need not be in any order. The ordering is done by the program. However, the program will warn you if a boundary does not close.

Once you have edited the boundaries, and if everything is fine, the final step will be just saving it as a coordinate file. Just use the popup menu PopupSave and you are good to go.

Further Information

If you need help in using pdnMesh, please send mail to

`<pdnmesh-users@lists.sourceforge.net>`

If you find any bugs or if you need to send any patches, please use

`<pdnmesh-bugs@lists.sourceforge.net>`

Here is a list of references that we found useful in developing pdnMesh.

Books

[Hoole89] S. R. H. Hoole, 1989, 0-444-01327-X, Elsevier, New York, *Computer-aided analysis and design of electromagnetic devices*.

[deBerg98] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf, 1997, 2000, 3-540-65620-0, Springer Verlag, Berlin Heidelberg, *Computational geometry, algorithms and applications, second edition*.

Other

[NiemannXX] *Sorting and searching algorithms, a cookbook*, Thomas Niemann, XXXX.

Index