LAB #4

In previous two computer labs, you have seen how the analytical techniques for solving electrostatic problems can be approximately solved using numerical methods. For example, you have approximated the solution to Coulomb's law problems which would be impossible to solve analytically, such as the off axis electric field for a charged ring. The main purpose of LAB 4 is to discover how MATLAB can be used to approximate the solution to electrostatic boundary value problems. To do this you will use the finite difference technique, which is one of many methods that are used to solve BVPs numerically. This is an extremely important lab because it provides you with an introduction to how electromagnetic problems are solved in current engineering practice.

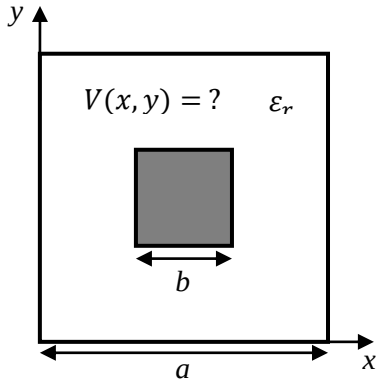
INTRODUCTION
Finite Difference Method

In class you have seen how electrostatic boundary value problems (BVP) can be solved relatively easily if the electric potential depends on only one variable. When the potential depends on, or varies with respect to two or three variables solving the BVP may still be possible using analytical techniques, but for some cases the analytical approach may be much too difficult.

In practice, many of these types of problems are solved using *numerical* methods using software such as MATLAB, or other custom-designed programs[1]. Numerical methods rely on breaking up the problem into smaller pieces so that the software can deal with the problem in a "digital" form. This is often referred to as meshing or discretization. This means that the "analog" equations, such as Laplace's or Poisson's equations, that are at the foundation of the BVP must be approximated in some manner. There are a number of ways to approach this approximation and the resulting solution technique[2], but in this lab we will focus on the *Finite Difference Method*[3].

To introduce you to the finite difference method let us consider following electrostatic boundary value problem:

A coaxial cable consists of two square conductors as shown to the right. A dielectric with relative permittivity ε_r fills the space between the conductors. If a DC source is connected to this cable such that the inner conductor has a potential of V_0, and the outer conductor has a potential of 0 V, determine the potential everywhere between the conductors.



To solve this using the finite difference method we begin with the appropriate form of Laplace's equation:

$$\nabla^2 V = 0$$

Since V will depend on both x and y, this becomes:

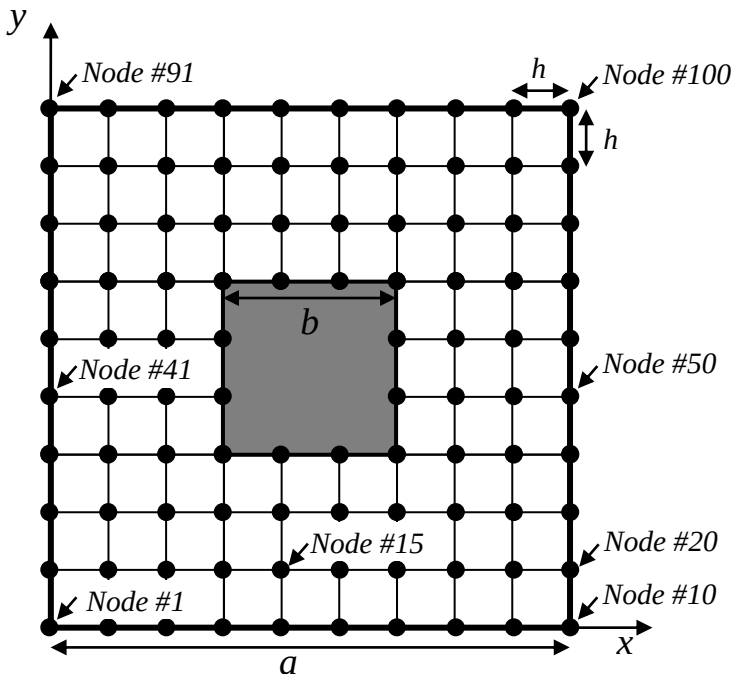
$$\frac{\partial^2 V}{\partial x^2} + \frac{\partial^2 V}{\partial y^2} = 0$$

In order to solve this problem using MATLAB, this "continuous" differential equation must be converted into a discrete "difference" equation. This is the same idea as taking the "continuous" Coulomb's integral problem and converting it into a discrete "summation" as you have done in the first two labs.

[1] One example of this type of software is Quickfield (http://www.quickfield.com), which offers a free student's version
[2] Other possible techniques include the Finite Element Method, and the Moment Method.
[3] Another simple example of the finite difference method can be found at http://www.dcjenn.com/EO2652/finite_diffs.pdf.

The process of converting this into a discrete problem which is known as the finite difference method involves the following steps:

1) *Discretize the problem geometry*: In this step the region in which you would like to determine the potential is broken up into small pieces. For this square coaxial cable problem, this results in the formation of a grid as shown below.



Some important quantities are:

N_x = the number of nodes in the *x*-direction
N_y = the number of nodes in the *y*-direction
h = the spacing between nodes

For the discretization shown to the right, we have

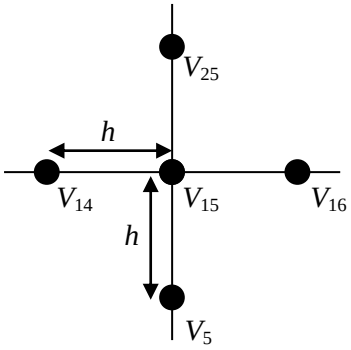
$$N_x = N_y = N_r = 10$$
$$h = \frac{a}{N_r - 1}$$

This gives a total of $N = 100$ nodes. We could count these nodes starting with 1 at the bottom left and counting upwards along each row (i.e., along the *x*-axis.) For some of the nodes are called *fixed nodes* because the potential is already known from the given the boundary values. In this case the fixed nodes are,

a) All the nodes on the outer square would have a known potential value of 0 V.
b) All the nodes on the inner square would have a known potential value of V_0 V. These are nodes #34 – 37, #44 – 47, #54 – 57, and #64 – 67.

The remaining nodes are called the *free nodes* since the potential value at these points are not known. Indeed, the purpose of this method is to figure out exactly how the potential changes from the value of 0 V on the outer square to V_0 V on the inner square. This means that we use MATLAB to calculate the potential at all the free nodes.

2) *Approximate Laplace's Equation*: This step involves turning Laplace's equation, $\frac{\partial^2 V}{\partial x^2} + \frac{\partial^2 V}{\partial y^2} = 0$, into a discrete form, so that MATLAB will be able to be used to solve for *V* at each free node. To do this let us consider how we can turn these second-order partial derivatives into difference equations. To begin we can focus on one particular node, say for example node #15, and the neighbouring nodes. We have labeled each node with the potential value (known or unknown) of that node:



At this node we can *approximate* the derivatives of the function $V(x, y)$ with respect to *x* and *y* using the *central difference method*:

$$\left.\frac{\partial V}{\partial x}\right|_{Node\ \#15} \approx \frac{V(x_{15} + h, y_{15}) - V(x_{15} - h, y_{15})}{2h} \approx \frac{V_{16} - V_{14}}{2h}$$
$$\left.\frac{\partial V}{\partial y}\right|_{Node\ \#15} \approx \frac{V(x_{15}, y_{15} + h) - V(x_{15}, y_{15} - h)}{2h} \approx \frac{V_{25} - V_5}{2h}$$

So, at this node #15 the second-order partial derivatives can then be approximated as:

$$\left.\frac{\partial^2 V}{\partial x^2}\right|_{Node\ \#15} = \frac{\partial}{\partial x}\left(\frac{\partial V}{\partial x}\right) \approx \frac{\left.\frac{\partial V}{\partial x}\right|_{x_{15}+h/2} - \left.\frac{\partial V}{\partial x}\right|_{x_{15}-h/2}}{h} \approx \frac{\frac{V_{16}-V_{15}}{h} - \frac{V_{15}-V_{14}}{h}}{h} \approx \frac{V_{16}-2V_{15}+V_{14}}{h^2}$$

$$\left.\frac{\partial^2 V}{\partial y^2}\right|_{Node\ \#15} = \frac{\partial}{\partial y}\left(\frac{\partial V}{\partial y}\right) \approx \frac{\left.\frac{\partial V}{\partial y}\right|_{y_{15}+h/2} - \left.\frac{\partial V}{\partial y}\right|_{y_{15}-h/2}}{h} \approx \frac{\frac{V_{25}-V_{15}}{h} - \frac{V_{15}-V_{5}}{h}}{h} \approx \frac{V_{25}-2V_{15}+V_{5}}{h^2}$$

These are *finite difference* approximations to the second-order partial derivatives[4]. In order to easily implement this in MATLAB, we can use two-dimensional indexing to represent the potentials at each node. We can use *i* to represent the location of node along the *x*-axis, and *j* to represent the location of the node along the *y*-axis, i.e., $i = 1 \ldots N_x$, and $j = 1 \ldots N_y$. Therefore, at a general node specified by (i, j), these expressions could be written as:

$$\left.\frac{\partial^2 V}{\partial x^2}\right|_{Node\ (i,j)} \approx \frac{V_{(i+1,j)} - 2V_{(i,j)} + V_{(i-1,j)}}{h^2}$$

$$\left.\frac{\partial^2 V}{\partial y^2}\right|_{Node\ (i,j)} \approx \frac{V_{(i,j+1)} - 2V_{(i,j)} + V_{(i,j-1)}}{h^2}$$

This means that Laplace's equation can be approximated at each node, so for node *i*, this would be:

$$\nabla^2 V = \frac{\partial^2 V}{\partial x^2} + \frac{\partial^2 V}{\partial y^2} \approx \frac{1}{h^2}\left[V_{(i+1,j)} + V_{(i-1,j)} + V_{(i,j+1)} + V_{(i,j-1)} - 4V_{(i,j)}\right] = 0$$

From this the unknown potential at node *i* can be solved for in terms of the potential values at its neighbouring nodes, and is thus:

$$V_{(i,j)} = \frac{1}{4}\left[V_{(i+1,j)} + V_{(i-1,j)} + V_{(i,j+1)} + V_{(i,j-1)}\right] \quad (1)$$

3) *Solve for the Unknown Potentials at the Free Nodes*: In this last step, the unknown potential values at all the free nodes can be found by using the approximate form of Laplace's equation stated above. There are a number of ways in which this can be done, but we will focus on using the *iteration method* in this lab[5].

Iteration Method

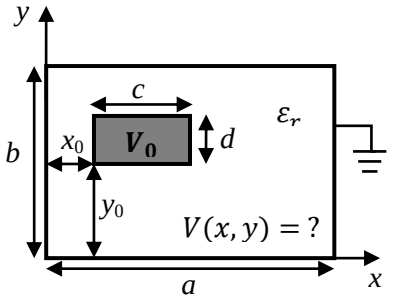
This method begins by assuming a value for the unknown potentials at the free nodes. Usually this is done by setting all the free nodes potentials to a value of 0 V. Using the equation (1) from above, one can calculate a new set of potentials at the free nodes based on the known fixed node values and the assumed free node values (i.e., 0V in the first iteration). This new set can then be used again to generate another set of values for the potentials at the free nodes. Note that the fixed node values (i.e., the boundary values) are kept the same for each iteration.

[4] Note that for the second-order derivatives, the spacing is $h/2$ not h. This is how the *central difference method* for approximating the derivatives works. There are other methods for approximating these derivatives, but we will use this method in this lab.

[5] Another popular technique is the *Matrix Method*. In this method the problem is considered as a set of N equations with N unknowns ($N = 100$ in this case). By writing out these N equations in matrix form the solution for the unknown potential values at the free nodes can be easily found through matrix inversion.

This iterative process can continue until it appears that the new set of free node potentials is not significantly different than the old set of free node potentials. Once this stage is reached, then the new set is likely a good approximation to the actual solution for $V(x, y)$ within the problem geometry.

Now consider a general rectangular coaxial cable design as shown to the right. In this case, both the outer conductor and the inner conductor are rectangles, and the inner conductor is not located at the center of the outer conductor. This lab will be focused around analyzing this type of structure using the finite difference method. To implement this method in MATLAB one could use the following code structure:



1) *Define the function* with:
   a. The output variables should be: *V*, which represents the potential values at all the nodes (free nodes and fixed nodes), and *Ex*, *Ey*, which represents the electric field components in the *x* and *y* directions.
   b. Input variables that should include: geometry (*a*, *b*, *c*, *d*, x_0, and y_0), and the boundary value (V_0).
2) *Define the number of nodes in each direction*. A larger number of nodes will lead to a more accurate solution, but the code will take longer to run. A good starting point which provides a balance between accuracy and speed would be $N_x = N_y = 101$.
3) *Initialize the potential value matrix, V, with zeros*. This matrix will have a size of N_x x N_y, and it contains the values of the potential at all the free and fixed nodes.
4) *Set the known values of the potential matrix, V*. Since the "boundary values are known (i.e., it is known that the potential value is 0 V on the outer conductor and V_0 on the inner conductor), you can "fix" the values of the corresponding elements within the matrix *V*. This will involve determining the indexes of the nodes within the N_x x N_y matrix *V* that represent the inner and outer conductors.
5) *Iterate to find the solution for the potential values at the free nodes*. For each iteration, you can use the equation:

$$V_{(i,j)} = \frac{1}{4}\left[V_{(i+1,j)} + V_{(i-1,j)} + V_{(i,j+1)} + V_{(i,j-1)}\right]$$

   to cycle through all the *free nodes* (using two nested for loops) and assign a "new" value for the potential at the free node (i, j) based on the "old" potential values at the surrounding nodes. You need to pay particular attention to make sure you do not assign any "new" values to the fixed nodes (i.e., the nodes that represent the outer and inner conductors).

   Once the new values are determined, you need to check to see if the new values are significantly different from the old values. If they are significantly different than another iteration would be needed. One way to do this would be to use the following commands:

```
Vnew=V(2:Nx-1,2:Ny-1);
maxdev=max(max(abs(100*(Vnew-Vold)./(Vnew))));
Vold=Vnew;
```

The first command creates a matrix which omits the points on the outer conductor (where the potential value is zero). The second command determines the maximum absolute percent deviation between new potential values and the old potential values within the matrix *V*. If `maxdev` is below a certain threshold then the new potential values can be consider to be the "solution" to the electrostatic boundary value problem and thus another iteration is not needed. A reasonable threshold for this lab could be 1×10^{-4}. One way to implement this iteration process is to use a `while end` loop structure.

6) *Evaluate the electric field components*: Once the potential distribution is known, then the electric field intensity can be found by taking the negative gradient of the potential. In MATLAB this can be approximated by using the command:

```
[Ey,Ex]=gradient(-V,hx,hy);
```

Note, that since a row of the matrix *V* represents a fixed *x* value and its columns represent fixed *y* values, the `gradient` function returns `Ey` as the first output variable (i.e., this is the variation of the matrix *V* from column to column) (see `help gradient`).

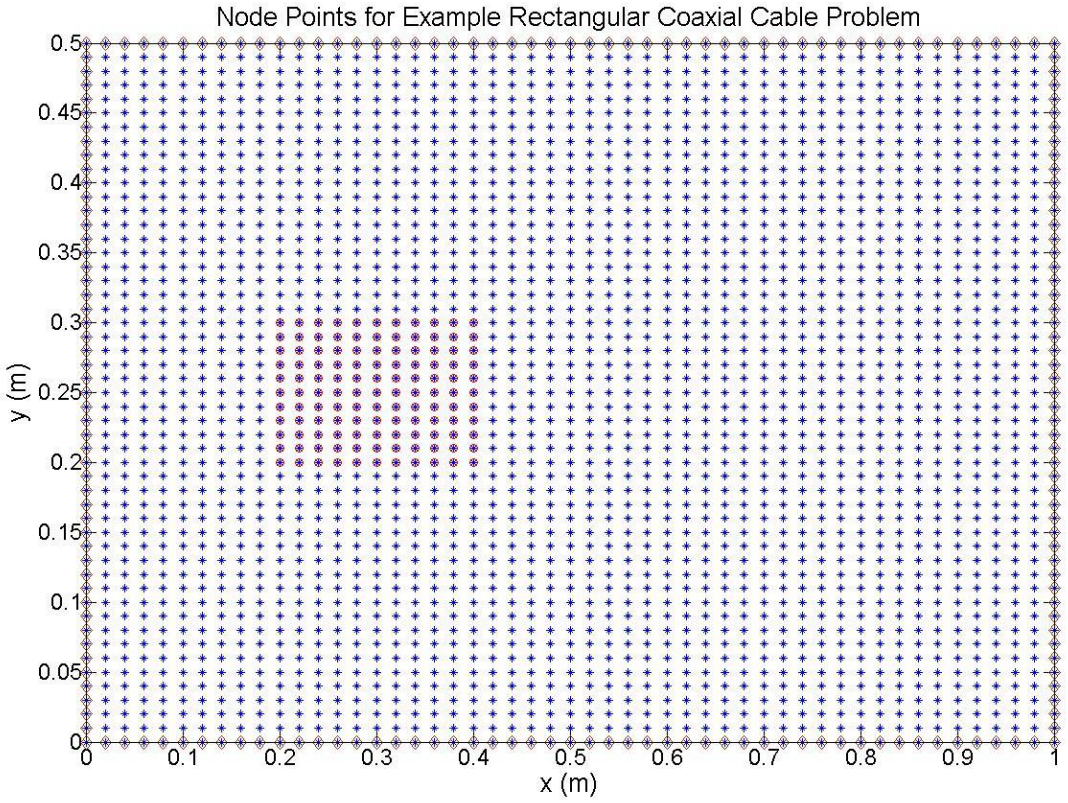
Example:

Consider the rectangular coaxial cable design in which a = 1 m, b = 0.5 m, c = 0.2 m, d = 0.1 m, x_0 = 0.2 m, and y_0 = 0.2 m. The first step in the process of solving this using the finite difference method is to discretize the problem geometry, or create the grid of nodes. For this example we will use $N_x = N_y = 51$. In solving these types of problems in MATLAB, it is often helpful to visualize the grid of nodes to make sure that you have defined it correctly. The function given below could be used to create a plot of the free and fixed nodes:

```
function
[gridpointsx,gridpointsy,innerx,innery,outerx,outery]=fdrectcoaxplotnodes(a,b,c,d,xo,yo,Nx,Ny)
%
% [gridpointsx,gridpointsy,innerx,innery,outerx,outery]=fdrectcoaxplotnodes(a,b,c,d,xo,yo,Nx,Ny)
% This function creates a plot of the grid of nodes for the finite difference
% method solution for the potential within a rectangular coaxial cable.
%

% Determine the node spacing in the x and y directions
hx=a/(Nx-1);
hy=b/(Ny-1);

% Determine indexes that represent the inner conductor
% These are rounded to ensure that they are integers
innerstartx=round(xo/hx+1);
innerendx=round(innerstartx+c/hx);
innerstarty=round(yo/hy+1);
innerendy=round(innerstarty+d/hy);

% Plot the grid points and known voltage nodes

% gridpointsx is a matrix (size Nx x Ny) that contains the x-values of the
% locations of the nodes within the grid.
% gridpointsy is a matrix (size Nx x Ny) that contains the y-values of the
% locations of the nodes within the grid.
[gridpointsx,gridpointsy]=meshgrid(0:hx:a,0:hy:b);
```

```
% innerx and innery are matrices that contains the x- and y-values of the
% locations of the nodes that relate to the inner conductor.
[innerx,innery]=meshgrid((innerstartx-1)*hx:hx:(innerendx-1)*hx,(innerstarty-1)*hy:hy:(innerendy-
1)*hy);

% outerx and outery are matrices that contains the x- and y-values of the
% locations of the nodes that relate to the outer conductor.
outerx=[0:hx:a,zeros(1,Ny-2),a:-hx:0,zeros(1,Ny-2)];
outerx((Nx+1):(Nx+Ny-2))=a;
outery=[zeros(1,Nx),hy:hy:(b-hy),zeros(1,Nx),(b-hy):-hy:hy];
outery((Nx+Ny-1):(2*Nx+Ny-2))=b;

figure
plot(gridpointsx,gridpointsy,'b*');hold;
plot(outerx,outery,'kd');
plot(innerx,innery,'ro');
```

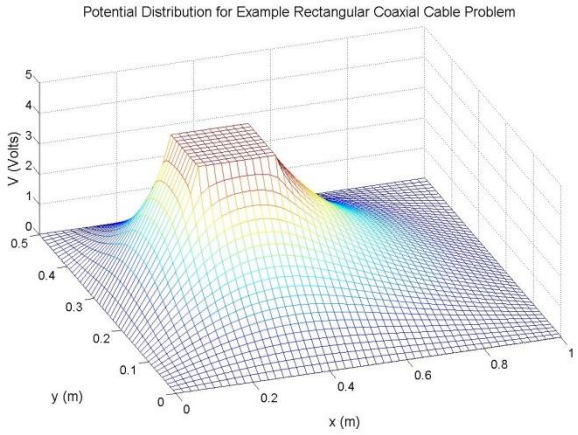
For the rectangular coaxial cable example described above, this function creates the plot shown below:

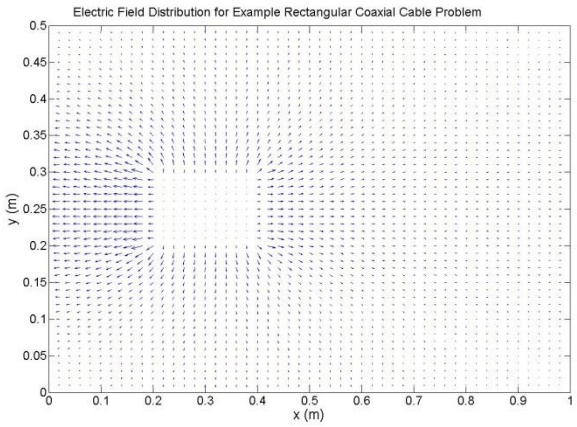


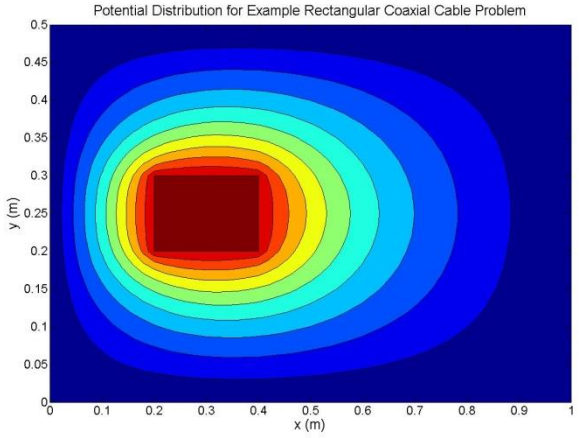
You can see that this plots all 51 x 51 = 2601 nodes and represents the fixed nodes on the inner conductor with red circles (`'ro'`) and the fixed nodes on the outer conductor with black diamonds (`'kb'`). In addition, this function returns the matrices, `gridpointsx,gridpointsy`, which are helpful for plotting the potential and electric field variation (see below).

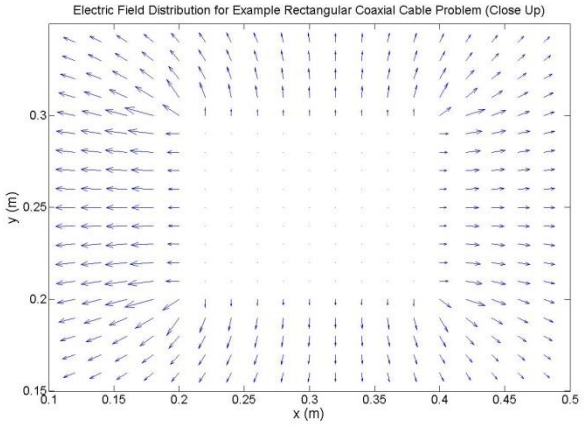
This function has been posted on the course website so you can use it if you would like. The posted function has additional functionality which is described in the comments of that code.

Solving this problem with the finite difference method in MATLAB using $N_x = N_y = 51$ and `maxdev` < 1 x 10^{-4}, the following plots can be created:









The plots of the potential distribution were created using the commands:

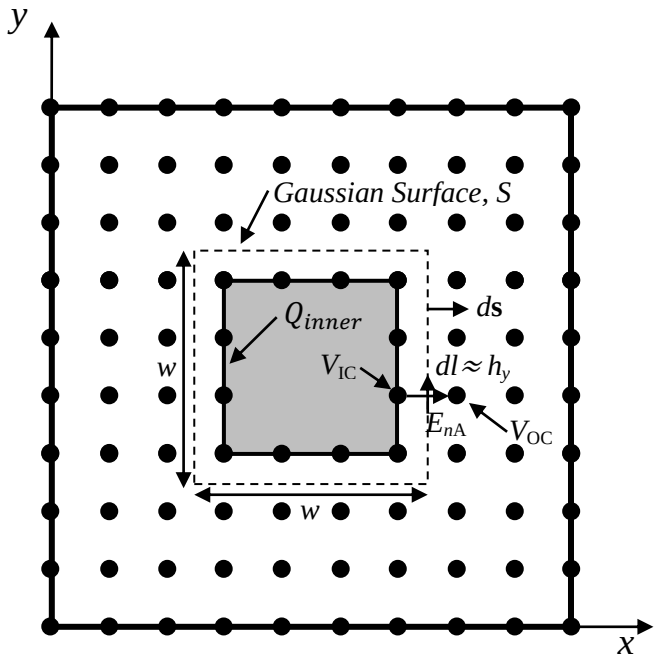
```
meshc(gridpointsx,gridpointsy,V');
contourf(gridpointsx,gridpointsy,V');
```

and the electric field plot was created using the command:

```
quiver(gridpointsx,gridpointsy,Ex',Ey');
```

Note that the `V`, `Ex`, and `Ey`, matrices had to be transposed since the size of these matrices are given by $N_x \times N_y$, while the `gridpointsx` and `gridpointsy` matrices have a size of $N_y \times N_x$.

From these plots we can observe that the potential solution satisfies the boundary values and that the electric field behaves as it would be expected. Meaning that it points from areas of high potential to areas of low potential and it is normal to the inner (and outer) conductor.



In the second part of the lab you will be asked to determine the capacitance per unit length of a certain rectangular coaxial cable design. This concept of capacitance *per unit length* is a very useful idea when working with cables and transmission lines. In calculating this quantity, we ignore the effects of any fringing fields, which mean that we assume that the electric potential and electric field distributions would be the same at every point along the cable. These distributions are the ones that we find using the two-dimensional finite difference method (for example, like the ones shown in the example plots above).

To find the capacitance per unit length of this cable we must first determine the charge per unit length on the inner conductor, i.e.,

$$\frac{C}{L} = \frac{1}{V_0}\left(\frac{Q_{inner}}{L}\right)$$

This quantity can be found from Gauss's law:

$$Q_{inner} = \oiint_S \mathbf{D} \cdot d\mathbf{s} = \oiint_S \varepsilon_r \varepsilon_0 \mathbf{E} \cdot d\mathbf{s}$$

In this case, the Gaussian surface, S, is a cylinder of length L in the z-direction with a square cross-section of size w x w. If we assume that the fields do not change as we move to different values of z (i.e., we ignore the effects of the fringing fields at the ends of the cable), then this simplifies to:

$$Q_{inner} = \oint_\alpha \varepsilon_r \varepsilon_0 E_n dl \int_0^L dz = L \oint_\alpha \varepsilon_r \varepsilon_0 E_n dl$$

where L represents the length of the Gaussian surface, α is the closed contour which encloses the inner conductor (in the figure shown this is the w x w square), and E_n is the component of the electric field which is normal to the contour at that point. Therefore,

$$\frac{Q_{inner}}{L} = \oint_\alpha \varepsilon_r \varepsilon_0 E_n dl$$

In order to evaluate this in MATLAB, we must be able to approximate this integration using a summation. One way to do this is to realize that between each set of nodes which lie on either side of this contour, the normal component of the electric field can be approximately determined by the values of the electric potential at these nodes (this comes from $\mathbf{E} = -\nabla V$). For example,

$$E_{nA} = \frac{V_{node\ inside\ contour} - V_{node\ outside\ contour}}{h_x} = \frac{V_{IC} - V_{OC}}{h_x}$$

This expression applies at each point on this right-hand segment of the contour α, so the total contribution to the integral $\oint_\alpha \varepsilon_r \varepsilon_0 E_n dl$ for this segment could be approximated by:

$$\sum_{i=1}^{N_{C1}} \varepsilon_r \varepsilon_0 E_{ni} h_y$$

where N_{C1} represents the number of node pairs that straddle that segment of the contour α (which is 4 node pairs in the above example). By repeating this summation for all four segments of the contour you can reach an approximate value for $\frac{Q_{inner}}{L}$ and thus, $\frac{C}{L}$.

PREPARATION - Individual

1) Read through these Lab #4 notes carefully. This prior review will save you a lot of time during the lab session, which will enable you to complete the lab in the time available.

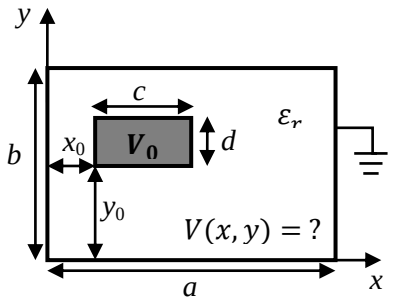
2) Using the initial code given below, complete this MATLAB function that can be used to solve for the electric potential and electric field within an arbitrary rectangular coaxial cable. As you can see the input variables include the cable's geometry (a, b, c, d, x_0, and y_0), and the boundary value (V_0). At a minimum the output variables should include V, Ex, and Ey. Make sure that your code works by verifying the plots for the example rectangular coaxial cable as given above.

```
function [V,Ex,Ey,C,We,We2,gridpointsx,gridpointsy,innerx,innery,outerx,outery]=
bvprectangularcoax(a,b,c,d,xo,yo,er,Vo)
%
% This function used the finite difference method to solve the
% two-dimensional electrostatic boundary value problem related to a square
% coaxial cable.
%   a = width of outer conductor
%   b = height of outer conductor
%   c = width of inner conductor
%   d = height of inner conductor
%   xo = the x-coordinate of the location of the bottom left corner of the inner conductor
%   yo = the y-coordinate of the location of the bottom left corner of the inner conductor
%   er = the relative permittivity of the dielectric which fills the space
%   between the inner and outer conductor
%   Vo = electric potential of the inner conductor (outer is grounded)

% Define the fundamental constant eo
eo=8.854e-12;

% Set number of nodes and node spacings
Nx=201;
hx=a/(Nx-1)
hy=hx;
Ny=round(b/hy+1)

% Set the initial values of V to zero
V = zeros(Nx,Ny);

% Set the known potential values (or boundary values)
V(1,1:Ny)=0; % Grounded left side
V(1:Nx,1)=0; % Grounded bottom side
V(Nx,1:Ny)=0; % Grounded right side
V(1:Nx,Ny)=0; % Grounded top side

innerstartx=round(xo/hx+1);
innerendx=round(innerstartx+c/hx);
innerstarty=round(yo/hy+1);
innerendy=round(innerstarty+d/hy);
V(innerstartx:innerendx,innerstarty:innerendy)=Vo; % Set potentials of inner conductor

% Determine the final voltage distributions (your code goes here…)
```

3) Using the approach discussed in the introduction above, write a few more lines of code which will enable you to estimate the capacitance per unit length of a general rectangular coaxial cable. You can assume that you have already determined the electric potential values at each node within the grid using your code from above.

IN-LAB WORK - Group

Analysis of a Rectangular Coaxial Cable



In this lab you will be asked to analyze a rectangular coaxial cable using the code that you wrote in your preparation. Each group will be given a unique cable design by their TAs at the beginning of the lab.

Your cable design is:

a = _____, b = _____, c = _____, d = _____,

x_0 = _____, y_0 = _____, ε_r = _____, V_0 = _____

1. Potential and Field Distribution

   Using your finite difference code with $N_x = 101$, and `maxdev` < 1 x 10^{-4}. Ensure that your h_x and h_y are the same by calculating your N_y value from: `Ny=round(b/hx+1)` .

   Prepare a two-dimensional plot (`contourf`) and three-dimensional plot (`meshc`) of the electric potential variation within your rectangular coaxial cable. As well, create a two-dimensional plot (`quiver`) of the electric field within this cable. Zoom into the inner conductor and into the edges of the outer conductor to verify that the field is correct.

Make sure you discuss this section of the lab with your TA before you leave.

2. Capacitance Per Unit Length

   a) *Code Verification*: Using the code that you have written in the preparation that enables you to calculate the capacitance per unit length, verify that your code works by considering the following three test cases[6]:

Rectangular Coaxial Cable Design	Capacitance Per Unit Length Actual Value	Capacitance Per Unit Length Value From Your Code
a = 2.2 mm, b = 2 mm, c = 1 mm, d = 0.8 mm, x_0 = 0.6 mm, y_0 = 0.6 mm, $\varepsilon_r = 1$	71.5 pF/m	
a = 4 mm, b = 4 mm, c = 2 mm, d = 2 mm, x_0 = 1 mm, y_0 = 1 mm, $\varepsilon_r = 1$	90.7 pF/m	
a = 1 m, b = 0.5 m, c = 0.2 m, d = 0.1 m, x_0 = 0.2 m, y_0 = 0.2 m, $\varepsilon_r = 2$	88.2 pF/m	

[6] The first two test cases are taken from the paper S. M. Musa and M. N. O. Sadiku, "Analysis of Rectangular Coaxial Lines," in *Proc. 2007 IEEE Region 5 Technical Conference*, April 20 – 21, 2007, pgs. 322 – 325, which has been posted on the course website.

2. Capacitance Per Unit Length (cont'd)

b) In this part you are asked to determine the capacitance per unit length for your rectangular coaxial cable design. To make sure that your calculations are correct, evaluate the capacitance for three different contours (i.e., three α's) around the inner conductor.

Contour #1: $\frac{C}{L}$ = _____

Contour #2: $\frac{C}{L}$ = _____

Contour #3: $\frac{C}{L}$ = _____

Use the posted function, `fdrectcoaxplotnodes.m`, to plot your node grid and the contours that you have used.

c) Now, determine the energy stored per unit length within a capacitor made from this rectangular coaxial cable that is charged to a potential difference of V_0 = 5 V. Calculate this energy stored per unit length from both the calculated capacitance per unit length, and the calculated electric fields:

Stored Energy Per Unit Length #1: $\frac{W_E}{L} = \frac{1}{2}\left(\frac{C}{L}\right)V_0^2 \approx$ _____

Stored Energy Per Unit Length #2: $\frac{W_E}{L} = \frac{1}{2}\iint_S \varepsilon_r\varepsilon_0|\mathbf{E}|^2 ds \approx$ _____

Make sure you discuss this section of the lab with your TA before you leave.