Elec 4700

Assignment 2 – Finite Difference Method

Submitted By: Jarikre Efe Jeffery

Date Submitted: 23/02/2020

Carleton University

Ottawa, Ontario Canada

Introduction:

This experiment involved studying the finite difference method. The finite difference method was used to solve Laplace’s electrostatic problem by modelling the problem as a mesh of voltages and resistances. The first part of the experiment involved modelling the voltage in a simple rectangular device using both numerical techniques and analytical techniques. This would enable one to be able to see clearly the advantages and disadvantages both techniques pose. The second part of the experiment involved adding rectangular bottlenecks which were of high resistance and modelling the voltage in order to observe what happens to the current flow in the device when they came in contact with the rectangular bottlenecks.

Part1:

a.) Using the finite difference method with the boundaries applied:

By solving the simple case where $V = V_0$ at x=0 and V=0 at x=L. the following plots were able to be gotten:

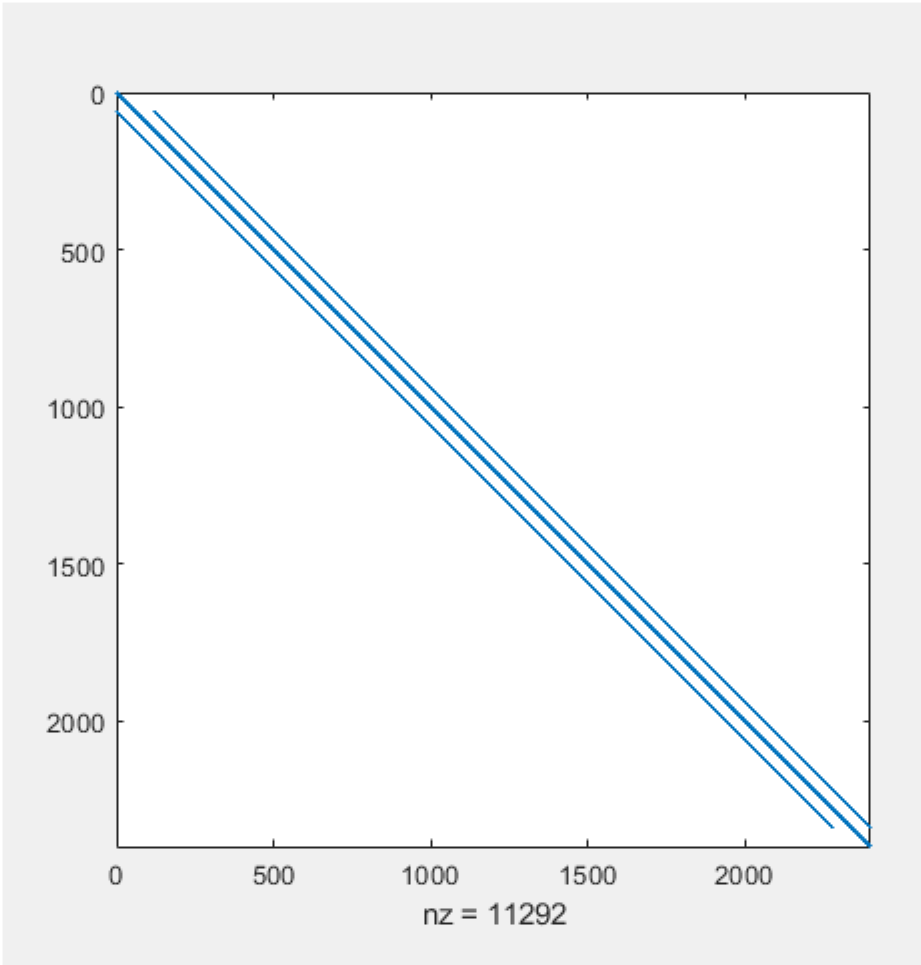

Figure 1: MATLAB plot gotten after taking spy(G Matrix)

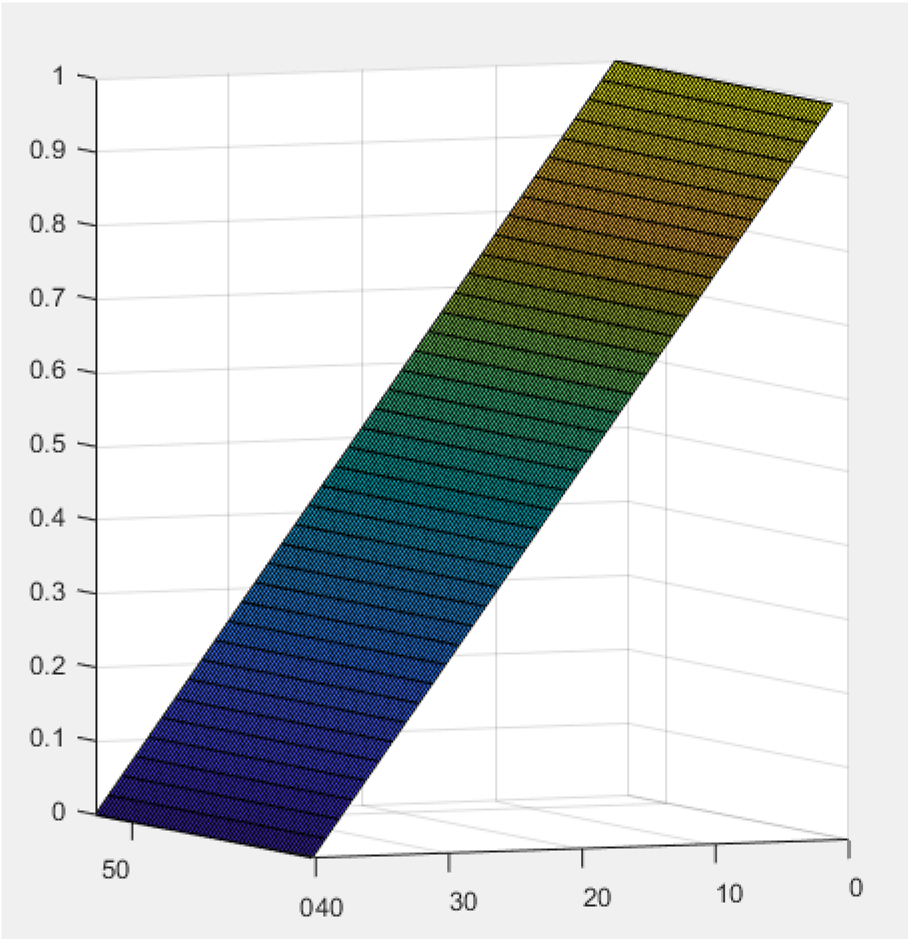

Figure 2: Voltage model for the fixed BCs on two sides

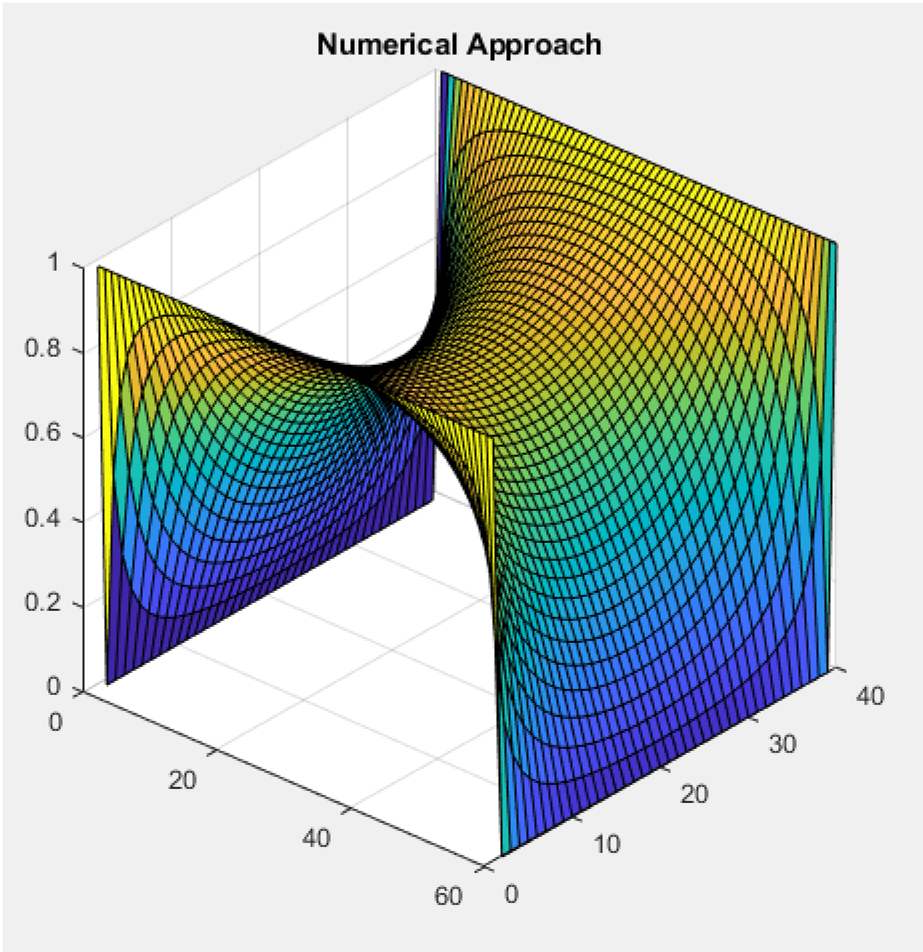

Figure 3: result gotten when using the meshing / numerical approach to solve the problem.

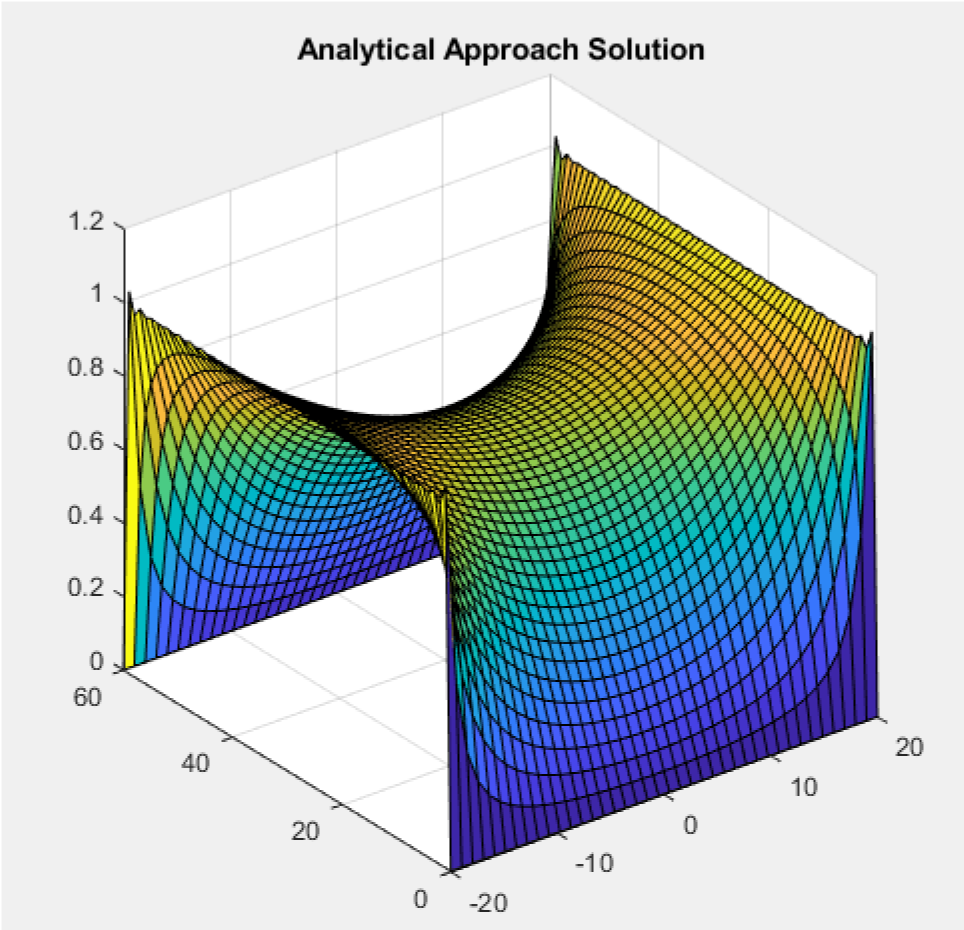

Figure 4: Result gotten when using the analytical approach to solve the problem.

One can see that the two solutions seemed to produce very similar results meaning the solution approaches the analytical approach however the two methods both have their advantages and disadvantages. The advantage of using the numerical approach is that by using a smaller space step the results becomes more accurate and also unlike the analytical approach in the numerical approach we don't have to worry about how many sums we need to take to get a more accurate result. The disadvantage of the numerical approach however is that with this approach we are working with a lot of approximations and this can contribute to how accurate of a model we get. The advantage of using the Analytical approach is that with the analytical approach since we are working with an actual equation in order to get our model and so we are not approximating as much as the numerical method. The disadvantage however with this method is that since we are working with a series of summations, it becomes difficult to be able to predict when to stop the summation. For this experiment I simply tried a bunch of different summations until I got to a point where the plot or result gotten seemed more defined as it is not possible to sum to infinity. Final conclusions about the meshing method: The meshing method is a good way to get a model when dealing with approximations however the analytical solution would be able to get a more accurate model if one could sum the series up to infinity.

Part 2.):

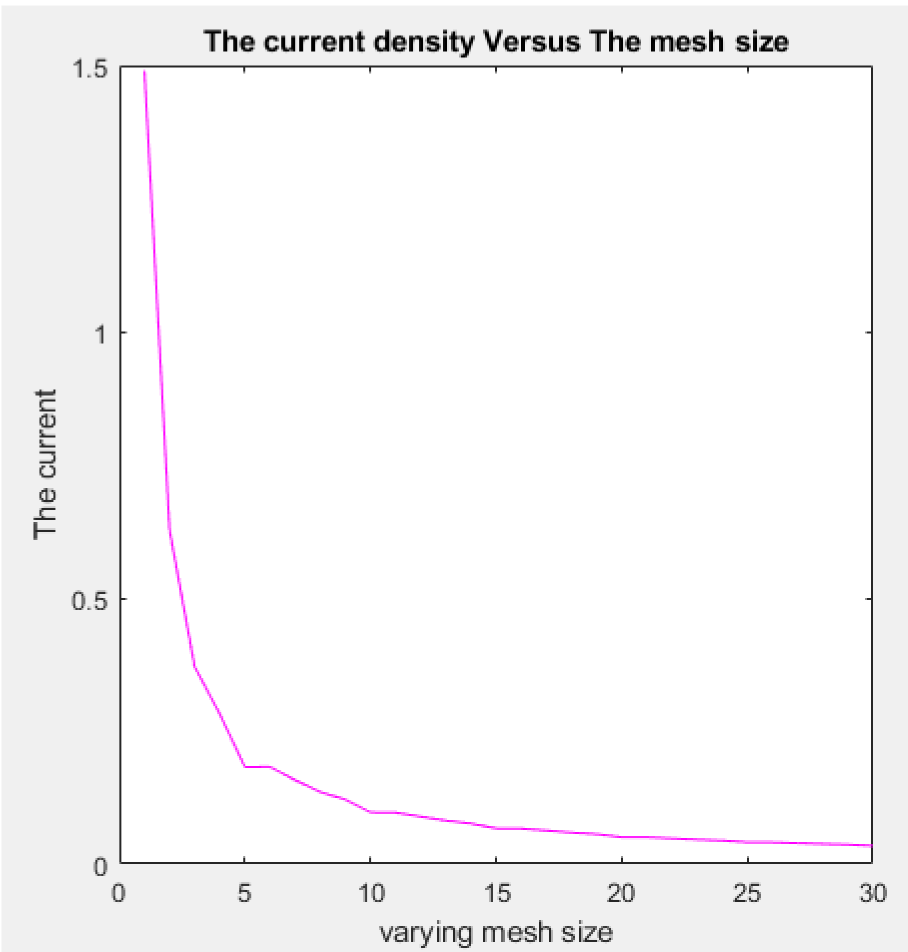

Figure 5: Current vs varying mesh number

By observing figure 5, one can see that as the mesh size used increases, the current tends to decrease.

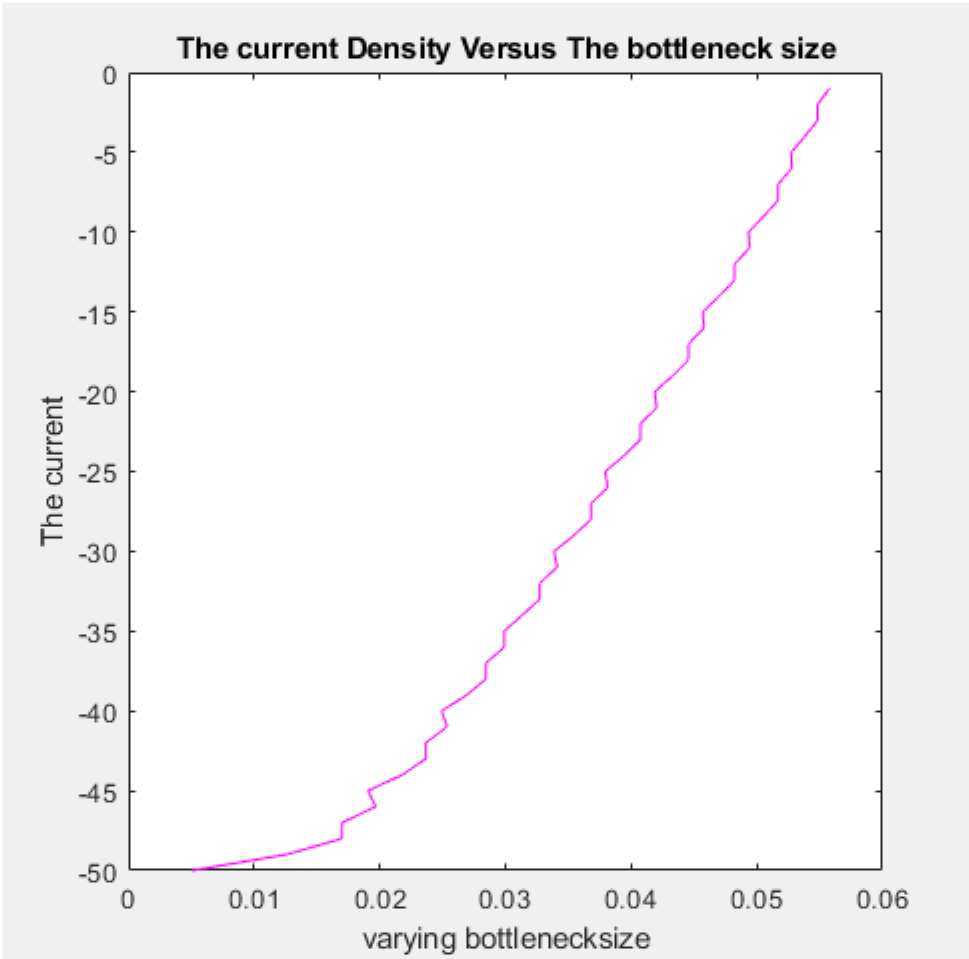

Figure 6: current vs varying bottleneck width

Looking at figure 6, we see that by increasing the bottleneck width, the current increases as well.

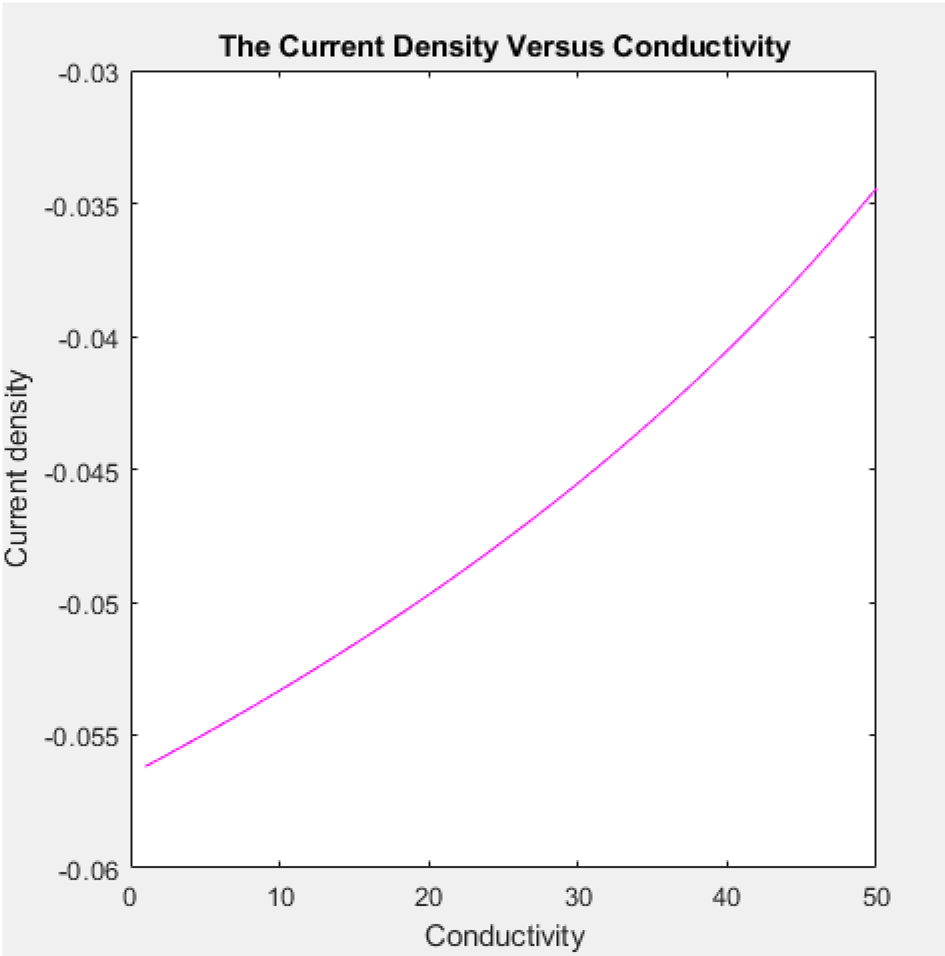

Figure 7: current vs varying conductance

According the equation J = conductivity*Electric field we know that the relationship between current and conductivity should be linear however due to a few minor errors performed during the coding process the graph gotten isn't perfectly linear as can be seen in Figure 7.

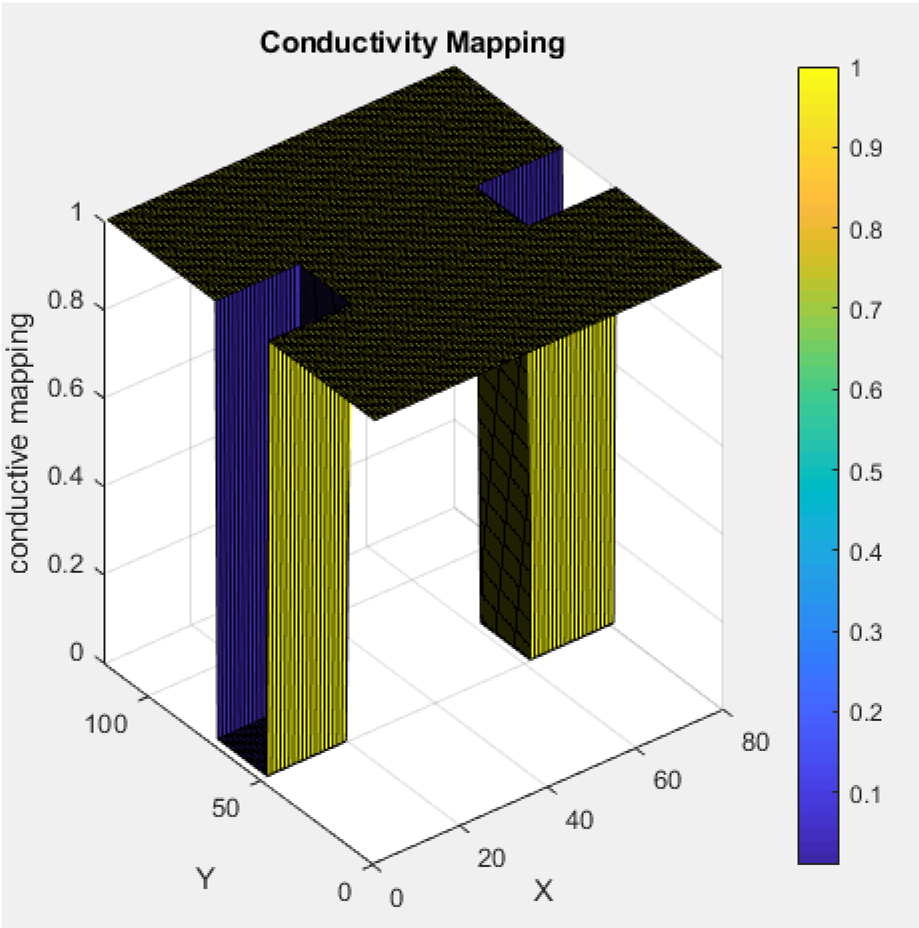

Figure 8: Conductivity mapping plot

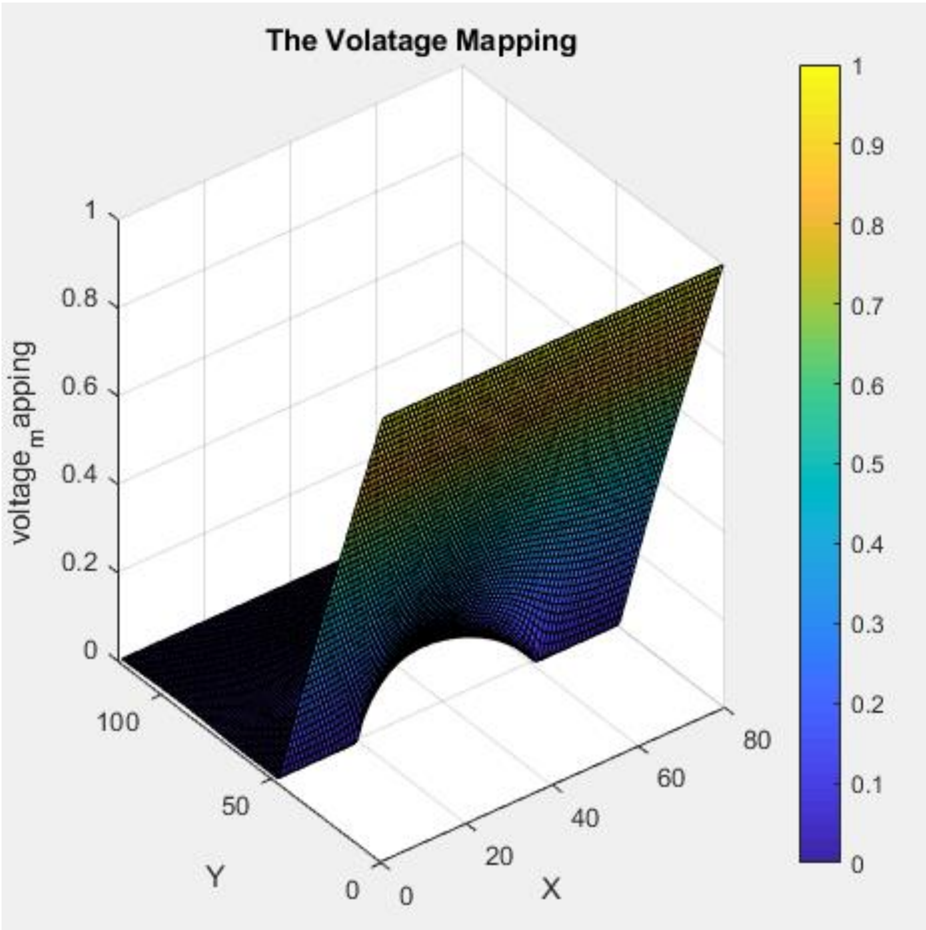

Figure 9: Voltage mapping plot

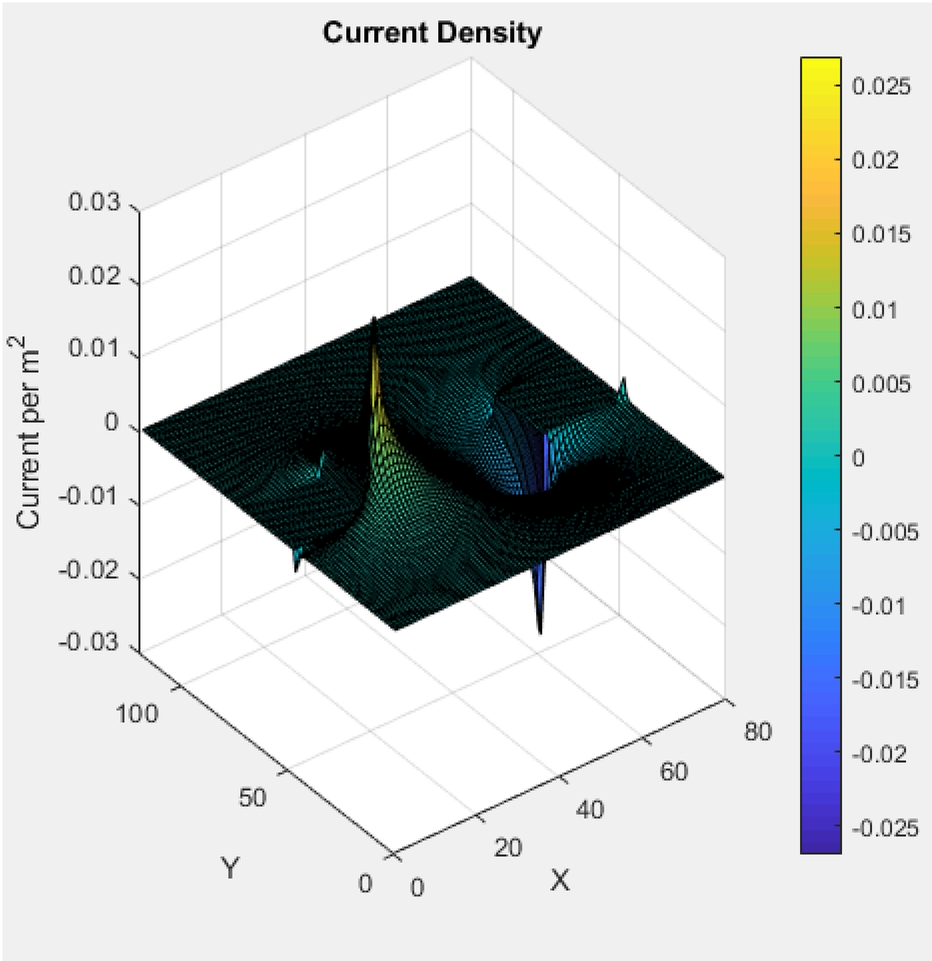

Figure 10: Current Density Plot

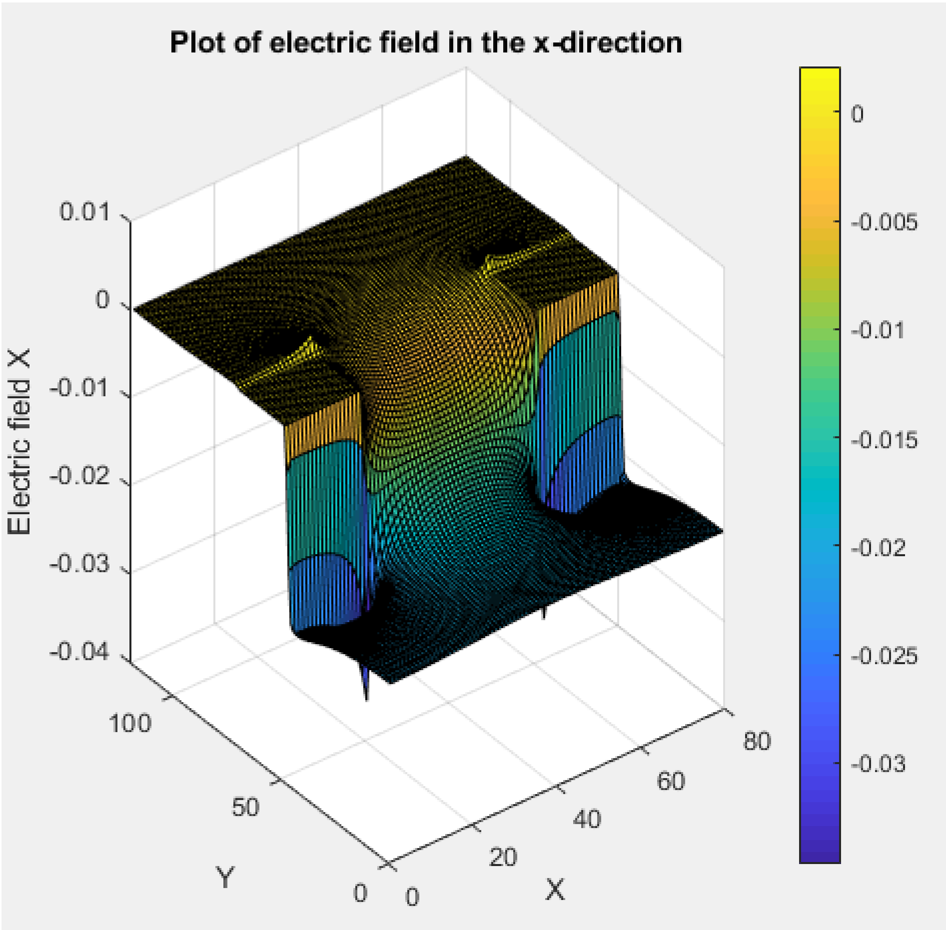

Figure 11: Plot of the electric field in the x-Direction

Looking at figure 11 we can see that there is a region at a higher voltage and also a region of the plot at a lower voltage there by creating a change in potential which creates the electric field. An increase in the change in voltage potential would cause the electric field created to increase likewise a decrease in the change in voltage potential should cause the electric field created to be smaller.

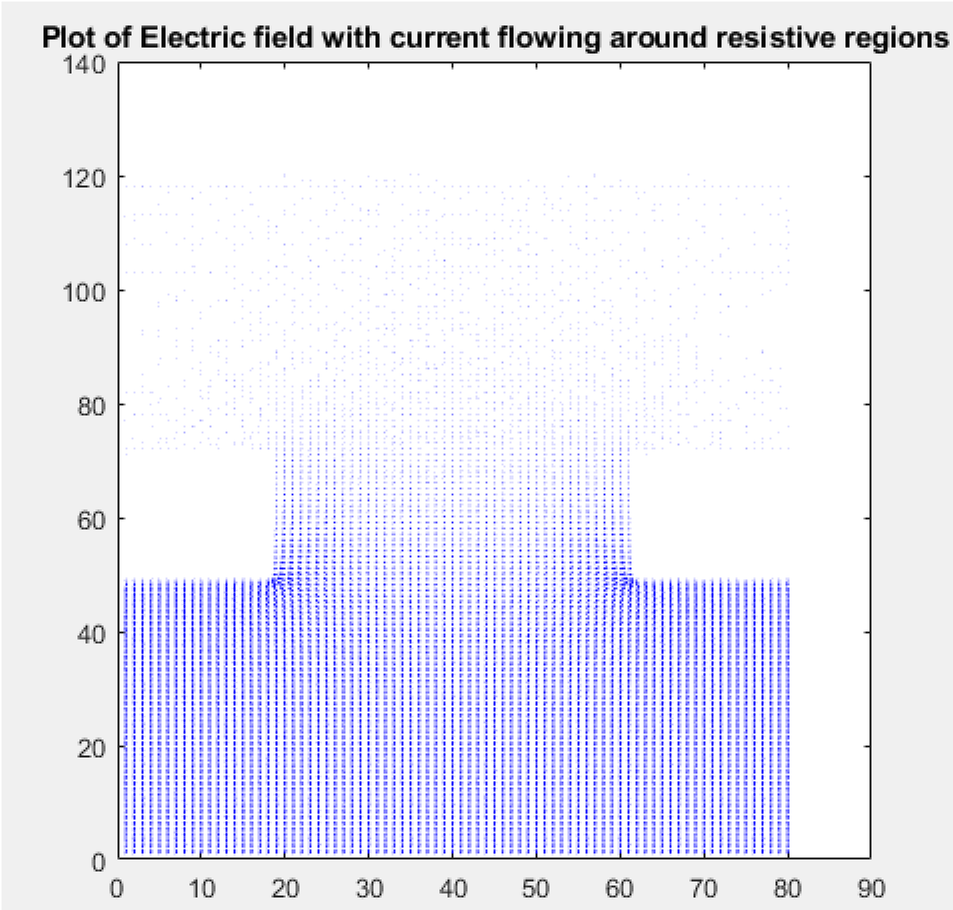

Figure 12: Quiver plot of current density

Looking at the figure above one we can easily observe what occurs when the resistive bottleneck has been added. One can see that the current flows around the highly resistive regions as they should. As we know no current should be able to flow through highly resistive materials and as such the behaviour of the current flow makes sense.

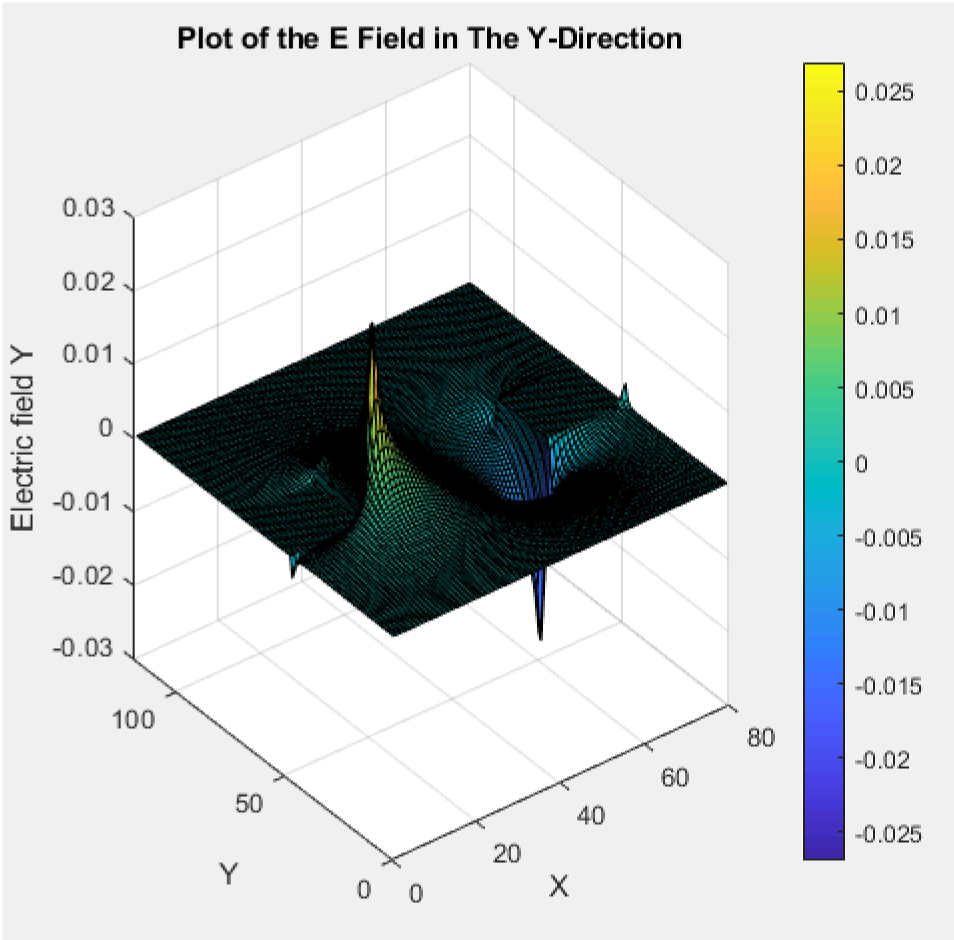

Figure 13: Electric field in the Y-direction

Conclusion:

This assignment was performed successfully. As we were able to successfully use the finite difference method to solve a bunch of problems one of them being Laplace's electrostatic potential.

Appendix :

```
% Jarikre Efe Jeffery - 101008461
% ELEC 4700 - Assignment 2 - Question 1
% using the 3/2 ratio for the length and the width

clc
clear
set(0,'DefaultFigureWindowStyle','docked')

width_x = 40;
length_y = 60;
```

```
G_matrix = sparse((width_x * length_y), (width_x * length_y));
V_matrix = zeros(1, (width_x * length_y));
v0 = 1;

for i = 1:width_x

    for j = 1:length_y
        n = j + (i - 1) * length_y;
        nxm = j + ((i-1) - 1) * length_y;
        nxp = j + ((i+1) - 1) * length_y;
        nym = (j-1) + (i - 1) * length_y;
        nyp = (j+1) + (i - 1) * length_y;

        if (i == 1)
            G_matrix(n, :) = 0;
            G_matrix(n, n) = 1;
            V_matrix(1, n) = 1;
        elseif (i == width_x)
            G_matrix(n, :) = 0;
            G_matrix(n, n) = 1;

        elseif (j == 1 && i > 1 && i < width_x)
            G_matrix(n, n) = -1;
            G_matrix(n, nyp) = 1;

        elseif (j == length_y && i > 1 && i < width_x)

            G_matrix(n, n) = -1;
            G_matrix(n, nym) = 1;

        else
            G_matrix(n, n) = -4;
            G_matrix(n, nxm) = 1;
            G_matrix(n, nxp) = 1;
            G_matrix(n, nym) = 1;
            G_matrix(n, nyp) = 1;

        end
    end


end

figure (1);
spy(G_matrix);

matrix_solution = G_matrix\V_matrix';
figure (2);

surfacet = zeros(width_x, length_y);

for i = 1:width_x

    for j = 1:length_y
```

```matlab
        n = j + (i - 1) * length_y;
        nxm = j + ((i-1) - 1) * length_y;
        nxp = j + ((i+1) - 1) * length_y;
        nym = (j-1) + (i - 1) * length_y;
        nyp = (j+1) + (i - 1) * length_y;
        surfacet(i, j) = matrix_solution(n);
    end
end

surf(surfacet);

% Second Part of Question 1.)
G_matrix2 = sparse((width_x * length_y), (width_x * length_y));
V_matrix2 = zeros(1, (width_x * length_y));
vo = 1;

for i = 1:width_x

    for j = 1:length_y
        n = j + (i - 1) * length_y;
        nxm = j + ((i-1) - 1) * length_y;
        nxp = j + ((i+1) - 1) * length_y;
        nym = (j-1) + (i - 1) * length_y;
        nyp = (j+1) + (i - 1) * length_y;

        if i == 1
            G_matrix2(n, :) = 0;
            G_matrix2(n, n) = 1;
            V_matrix2(1, n) = vo;
        elseif i == width_x
            G_matrix2(n, :) = 0;
            G_matrix2(n, n) = 1;
            V_matrix2(1, n) = vo;
        elseif j == 1
            G_matrix2(n, :) = 0;
            G_matrix2(n, n) = 1;
        elseif j == length_y
            G_matrix2(n, :) = 0;
            G_matrix2(n, n) = 1;
        else
            G_matrix2(n, :) = 0;
            G_matrix2(n, n) = -4;
            G_matrix2(n, nxm) = 1;
            G_matrix2(n, nxp) = 1;
            G_matrix2(n, nym) = 1;
            G_matrix2(n, nyp) = 1;
        end
    end


end

solution2 = G_matrix2\V_matrix2';
figure (3);
```

```
surface2 = zeros(width_x, length_y);


for i = 1:width_x

    for j = 1:length_y
        n = j + (i - 1) * length_y;
        nxm = j + ((i-1) - 1) * length_y;
        nxp = j + ((i+1) - 1) * length_y;
        nym = (j-1) + (i - 1) * length_y;
        nyp = (j+1) + (i - 1) * length_y;
        surface2(i, j) = solution2(n);
    end
end

surf(surface2);
title("Numerical Approach");

% Question 1.) Part B - The Analytical Solution Approach

zone = zeros(60, 40);
a = 60;
b = 20;

x = linspace(-20,20,40);
y = linspace(0,60,60);

[x_mesh, y_mesh] = meshgrid(x, y);

for n = 1:2:300

    zone =  (zone + (4 * v0/pi).*(cosh((n * pi * x_mesh)/a) .* sin((n * pi *
y_mesh)/a)) ./ (n * cosh((n * pi * b)/a)));

    figure(4);
    surf(x, y, zone);
    title("Analytical Approach Solution");
    pause(0.01);

end

% Assignment 2 elec 4700 Question 2
% Jarikre Efe Jeffery
% 101008461
clc
clear
set(0,'DefaultFigureWindowStyle','docked')

% Width and length are following the 3/2 ratio
width_x = 120;
length_y = 80;
num_x = 80;
num_y = 100;
```

```
% matrix
G_matrix = sparse((width_x * length_y), (width_x * length_y));
V_matrix = zeros(1, (width_x * length_y));
vo = 1;

%Conductivity outside the boxes
conductivity_outside = 1;

%Conductivity inside the boxes
conductivity_inside = 1e-2;

% Bottleneck for 1 and 2
bottleneck1 = [(width_x * 0.4), (width_x * 0.6),  length_y, (length_y *
0.75)];
bottleneck2 = [(width_x * 0.4), (width_x * 0.6), 0, (length_y * 0.25)];

% Creating the Conductivity Mapping
conductivity_mapping = ones(width_x, length_y);

% Including the bottlenecks
for i = 1:width_x

    for j = 1:length_y
        if(i > bottleneck1(1) && i < bottleneck1(2) && ((j < bottleneck2(4))
|| (j > bottleneck1(4))))
            conductivity_mapping(i,j) = 1e-2;
        end
    end

end

% Conductivity Mapping Plot
figure(5);
surf(conductivity_mapping);
colorbar
title('Conductivity Mapping');
xlabel('X')
ylabel('Y')
zlabel('conductive mapping')

% G-Matrix and Boundary Conditions
for i = 1:width_x

    for j = 1:length_y

        % defining location on boundary
        n = j + (i - 1) * length_y;
        nxm = j + ((i-1) - 1) * length_y;
        nxp = j + ((i+1) - 1) * length_y;
        nym = (j-1) + (i - 1) * length_y;
        nyp = (j+1) + (i - 1) * length_y;

        % Indexes for conditions that need to be fulfilled
        index1 = (i == 1);
```

```
        index2 = (i == width_x);
        index3 = (j == 1 && i > 1 && i < width_x);
        index4 = (i == bottleneck1(1));
        index5 = (i == bottleneck1(2));
        index6 = (i > bottleneck1(1) && i < bottleneck1(2));
        index7 = (j == length_y && i > 1 && i < width_x);
        index8 = (i == bottleneck1(2));
        index9 = (i > bottleneck1(1) && i < bottleneck1(2));
        index10 = (i == bottleneck1(1) && ((j < bottleneck2(4)) || (j >
bottleneck1(4))));
        index11 = (i == bottleneck1(2) && ((j < bottleneck2(4)) || (j >
bottleneck1(4))));
        index12 = (i > bottleneck1(1) && i < bottleneck1(2) && ((j <
bottleneck2(4)) || (j > bottleneck1(4))));

        if (index1)
            G_matrix(n, :) = 0;
            G_matrix(n, n) = 1;
            V_matrix(1, n) = 1;
        elseif (index2)
            G_matrix(n, :) = 0;
            G_matrix(n, n) = 1;

        elseif (index3)

            if (index4)
                G_matrix(n, n) = -3;
                G_matrix(n, nyp) = conductivity_inside;
                G_matrix(n, nxp) = conductivity_inside;
                G_matrix(n, nxm) = conductivity_outside;

            elseif (index5)
                G_matrix(n, n) = -3;
                G_matrix(n, nyp) = conductivity_inside;
                G_matrix(n, nxp) = conductivity_outside;
                G_matrix(n, nxm) = conductivity_inside;

            elseif (index6)
                G_matrix(n, n) = -3;
                G_matrix(n, nyp) = conductivity_inside;
                G_matrix(n, nxp) = conductivity_inside;
                G_matrix(n, nxm) = conductivity_inside;
            else
                G_matrix(n, n) = -3;
                G_matrix(n, nyp) = conductivity_outside;
                G_matrix(n, nxp) = conductivity_outside;
                G_matrix(n, nxm) = conductivity_outside;
            end

        elseif (index7)

            if (index4)
                G_matrix(n, n) = -3;
                G_matrix(n, nym) = conductivity_inside;
                G_matrix(n, nxp) = conductivity_inside;
```

```
                G_matrix(n, nxm) = conductivity_outside;

            elseif (index8)
                G_matrix(n, n) = -3;
                G_matrix(n, nym) = conductivity_inside;
                G_matrix(n, nxp) = conductivity_outside;
                G_matrix(n, nxm) = conductivity_inside;

            elseif (index9)
                G_matrix(n, n) = -3;
                G_matrix(n, nym) = conductivity_inside;
                G_matrix(n, nxp) = conductivity_inside;
                G_matrix(n, nxm) = conductivity_inside;
            else
                G_matrix(n, n) = -3;
                G_matrix(n, nym) = conductivity_outside;
                G_matrix(n, nxp) = conductivity_outside;
                G_matrix(n, nxm) = conductivity_outside;
            end

        else

            if (index10)
                G_matrix(n, n) = -4;
                G_matrix(n, nyp) = conductivity_inside;
                G_matrix(n, nym) = conductivity_inside;
                G_matrix(n, nxp) = conductivity_inside;
                G_matrix(n, nxm) = conductivity_outside;

            elseif (index11)
                G_matrix(n, n) = -4;
                G_matrix(n, nyp) = conductivity_inside;
                G_matrix(n, nym) = conductivity_inside;
                G_matrix(n, nxp) = conductivity_outside;
                G_matrix(n, nxm) = conductivity_inside;

            elseif (index12)
                G_matrix(n, n) = -4;
                G_matrix(n, nyp) = conductivity_inside;
                G_matrix(n, nym) = conductivity_inside;
                G_matrix(n, nxp) = conductivity_inside;
                G_matrix(n, nxm) = conductivity_inside;
            else
                G_matrix(n, n) = -4;
                G_matrix(n, nyp) = conductivity_outside;
                G_matrix(n, nym) = conductivity_outside;
                G_matrix(n, nxp) = conductivity_outside;
                G_matrix(n, nxm) = conductivity_outside;
            end

        end
    end
end

solution1 = G_matrix\V_matrix';
```

```
surface = zeros(width_x, length_y);

% Mapping the solution vector to a matrix
for i = 1:width_x

    for j = 1:length_y
        n = j + (i - 1) * length_y;
        nxm = j + ((i-1) - 1) * length_y;
        nxp = j + ((i+1) - 1) * length_y;
        nym = (j-1) + (i - 1) * length_y;
        nyp = (j+1) + (i - 1) * length_y;
        surface(i, j) = solution1(n);
    end
end

figure (6);
surf(surface);
colorbar
title('The Volatage Mapping');
xlabel('X')
ylabel('Y')
zlabel('voltage_mapping')

[Efield_y1, Efield_x1] = gradient(surface);
J = conductivity_mapping.*gradient(surface);
Jx = conductivity_mapping.*(-Efield_y1);
Jy = conductivity_mapping.*(-Efield_x1);
% quiver plot of current density
figure(7)
quiver (Jx,Jy,'m');
title('plot of resistive bottlenecks and current flow')

% Current Density Plot
figure (8)
surf(J)
colorbar
title('Current Density');
xlabel('X')
ylabel('Y')
zlabel('Current per m^2')

% Plot of electric Field in the X -direction
figure (9)
surf (Efield_y1)
colorbar
title('Plot of the E Field in The Y-Direction');
xlabel('X')
ylabel('Y')
zlabel('Electric field Y')

% Plot of electric field in the X-direction
figure (10)
surf(Efield_x1)
colorbar
```

```
title('Plot of electric field in the x-direction')
xlabel('X')
ylabel('Y')
zlabel('Electric field X')

% Plot of the E-field(x,y)
E_field = sqrt(Efield_y1.^2 + Efield_x1.^2);
figure (11)
surf(E_field)

% quiver plot of electric field
figure (12)
quiver (-Efield_y1, -Efield_x1, 'b');
title('Plot of Electric field with current flowing around resistive regions')

%Jarikre Efe Jeffery
%101008461
% Calculate the Current density versus mesh size
clc
set(0,'DefaultFigureWindowStyle','docked')

clear
num = 30;
% Using width and length ratio as provided in the assignment instructions
width_x = 2;
length_y = 3;

current_density = [];

for num = 1:num
    width_x = 3*num;
    length_y = 2*num;
    V0 = 5;

    G_matrix = sparse(length_y*width_x,length_y*width_x);
    solution1 = zeros(length_y*width_x,1);

    % sigma's as provided in the assignment instructions
    conductivity_outside = 1;
    conductivity_inside = 1e-2;
    conduction = conductivity_outside.*ones(length_y,width_x);

    for i = 1:width_x
        for j = 1:length_y
            if((i <= 0.8*width_x && i >= (0.3*width_x) && j <=
(0.3*length_y)) || (i <= (0.8*width_x) && i >= (0.3*width_x) && j >=
(0.8*length_y)))
                conduction(j,i) = conductivity_inside;
            end
        end
    end

    for i = 1:width_x
        for j = 1:length_y
```

```
            n = j + (i - 1) * length_y;
            nxm = j + ((i-1) - 1) * length_y;
            nxp = j + ((i+1) - 1) * length_y;
            nym = (j-1) + (i - 1) * length_y;
            nyp = (j+1) + (i - 1) * length_y;

            if(i == 1)
                solution1(n,1) = V0;
                G_matrix(n,n) = 1;
            elseif(i == width_x)
                solution1(n,1) = 0;
                G_matrix(n,n) = 1;
            elseif(j == 1)

                G_matrix(n,n) = -(((conduction(j,i) + conduction(j,i-
1))/2)+((conduction(j,i) + conduction(j,i+1))/2)+((conduction(j,i) +
conduction(j+1,i))/2));
                G_matrix(n,nxm) = (conduction(j,i) + conduction(j,i-1))/2;
                G_matrix(n,nxp) = (conduction(j,i) + conduction(j,i+1))/2;
                G_matrix(n,nyp) = (conduction(j,i) + conduction(j+1,i))/2;

                solution1(n,1) = 0;
            elseif(j == length_y)
                G_matrix(n,n) = -(((conduction(j,i) + conduction(j,i-
1))/2)+((conduction(j,i) + conduction(j,i+1))/2)+((conduction(j,i) +
conduction(j-1,i))/2));
                G_matrix(n,nxm) = (conduction(j,i) + conduction(j,i-1))/2;
                G_matrix(n,nxp) = (conduction(j,i) + conduction(j,i+1))/2;
                G_matrix(n,nym) = (conduction(j,i) + conduction(j-1,i))/2;

                solution1(n,1) = 0;
            else
                G_matrix(n,n) = -(((conduction(j,i) + conduction(j,i-
1))/2)+((conduction(j,i) + conduction(j,i+1))/2)+((conduction(j,i) +
conduction(j-1,i))/2)+((conduction(j,i) + conduction(j+1,i))/2));
                G_matrix(n,nxm) = (conduction(j,i) + conduction(j,i-1))/2;
                G_matrix(n,nxp) = (conduction(j,i) + conduction(j,i+1))/2;
                G_matrix(n,nym) = (conduction(j,i) + conduction(j-1,i))/2;
                G_matrix(n,nyp) = (conduction(j,i) + conduction(j+1,i))/2;
                solution1(n,1) = 0;
            end
        end
    end

    V_matrix = G_matrix\solution1;

    for i = 1:width_x
        for j = 1:length_y
            n = (i-1)*length_y+j;
            surface(j,i) = V_matrix(n,1);
        end
    end

    [Efield_x,Efield_y] = gradient(surface);
    J_xdir = conduction.*(-Efield_x);
```

```
    J_ydir = conduction.*(-Efield_y);
    current_density(num) = mean(mean((((J_xdir.^2)+(J_ydir.^2)).^0.5)));
end

% Plot the current density vs the mesh size
figure(13)
plot(1:num,current_density,'m')
title('The current density vs The mesh size')

clear
num = 50;
current_density = [];
for num = 1:num
    width_x = 90;
    length_y = 60;
    V0 = 5;
    G_matrix = sparse(length_y*width_x,length_y*width_x);
    solution1 = zeros(length_y*width_x,1);
    conductivity_outside = 1;
    conductivity_inside = 0.01;
    conduction = conductivity_outside.*ones(length_y,width_x);

    for i = 1:width_x
        for j = 1:length_y
            if((i <= 0.8*width_x && i >= 0.3*width_x && j <=
0.01*num*length_y) || (i <= (1-num*0.01)*length_y && i >= 0.25*width_x && j
>= (1-num*0.01)*length_y))
                conduction(j,i) = conductivity_inside;
            end
        end
    end

    for i = 1:width_x
        for j = 1:length_y
            n = j + (i - 1) * length_y;
            nxm = j + ((i-1) - 1) * length_y;
            nxp = j + ((i+1) - 1) * length_y;
            nym = (j-1) + (i - 1) * length_y;
            nyp = (j+1) + (i - 1) * length_y;
            if(i == 1)
                solution1(n,1) = V0;
                G_matrix(n,n) = 1;
            elseif(i == width_x)
                solution1(n,1) = 0;
                G_matrix(n,n) = 1;
            elseif(j == 1)
                G_matrix(n,n) = -(((conduction(j,i) + conduction(j,i-
1))/2)+((conduction(j,i) + conduction(j,i+1))/2)+((conduction(j,i) +
conduction(j+1,i))/2));
                G_matrix(n,nxm) = (conduction(j,i) + conduction(j,i-1))/2;
                G_matrix(n,nxp) = (conduction(j,i) + conduction(j,i+1))/2;
                G_matrix(n,nyp) = (conduction(j,i) + conduction(j+1,i))/2;

                solution1(n,1) = 0;
            elseif(j == length_y)
```

```
                G_matrix(n,n) = -(((conduction(j,i) + conduction(j,i-
1))/2)+((conduction(j,i) + conduction(j,i+1))/2)+((conduction(j,i) +
conduction(j-1,i))/2));
                G_matrix(n,nxm) = (conduction(j,i) + conduction(j,i-1))/2;
                G_matrix(n,nxp) = (conduction(j,i) + conduction(j,i+1))/2;
                G_matrix(n,nym) = (conduction(j,i) + conduction(j-1,i))/2;

                solution1(n,1) = 0;
            else
                G_matrix(n,n) = -(((conduction(j,i) + conduction(j,i-
1))/2)+((conduction(j,i) + conduction(j,i+1))/2)+((conduction(j,i) +
conduction(j-1,i))/2)+((conduction(j,i) + conduction(j+1,i))/2));
                G_matrix(n,nxm) = ((conduction(j,i) + conduction(j,i-1))/2);
                G_matrix(n,nxp) = (conduction(j,i) + conduction(j,i+1))/2;
                G_matrix(n,nym) = (conduction(j,i) + conduction(j-1,i))/2;
                G_matrix(n,nyp) = (conduction(j,i) + conduction(j+1,i))/2;
                solution1(n,1) = 0;
            end
        end
    end

    % solving for the V matrix
    V_matrix = G_matrix\solution1;

    % Loop to map answer vector to a matrix to be plotted
    for i = 1:width_x
        for j = 1:length_y
            n = j + (i - 1) * length_y;
            nxm = j + ((i-1) - 1) * length_y;
            nxp = j + ((i+1) - 1) * length_y;
            nym = (j-1) + (i - 1) * length_y;
            nyp = (j+1) + (i - 1) * length_y;
            surface(j,i) = V_matrix(n,1);
        end
    end

    [Efield_x,Efield_y] = gradient(surface);
    J_xdir = conduction.*(-Efield_x);
    J_ydir = conduction.*(-Efield_y);
    current_density(num) = mean(mean((((J_xdir.^2)+(J_ydir.^2)).^0.5)));
end

% Plot of the current density vs the bottleneck size
figure(14)
plot(current_density,(-1)*(1:num), 'm')
title('The current Vs The bottleneck size')

clear
num = 50;
current_density = [];

for num = 1:num

    width_x = 90;
    length_y = 60;
```

```
    V0 = 5;
    G_matrix = sparse(length_y*width_x,length_y*width_x);
    solution1 = zeros(length_y*width_x,1);
    conductivity_outside = 1;
    conductivity_inside = 1.02-num*0.02;
    conduction = conductivity_outside.*ones(length_y,width_x);

    for i = 1:width_x
        for j = 1:length_y
            if((i <= 0.8*width_x && i >= 0.3*width_x && j <= 0.3*length_y) ||
(i <= 0.8*width_x && i >= 0.3*width_x && j >= 0.8*length_y))
                conduction(j,i) = conductivity_inside;
            end
        end
    end

    for i = 1:width_x
        for j = 1:length_y
            n = j + (i - 1) * length_y;
            nxm = j + ((i-1) - 1) * length_y;
            nxp = j + ((i+1) - 1) * length_y;
            nym = (j-1) + (i - 1) * length_y;
            nyp = (j+1) + (i - 1) * length_y;

            if(i == 1)
                solution1(n,1) = V0;
                G_matrix(n,n) = 1;
            elseif(i == width_x)
                solution1(n,1) = 0;
                G_matrix(n,n) = 1;
            elseif(j == 1)

                G_matrix(n,n) = -(((conduction(j,i) + conduction(j,i-
1))/2)+((conduction(j,i) + conduction(j,i+1))/2)+((conduction(j,i) +
conduction(j+1,i))/2));
                G_matrix(n,nxm) = (conduction(j,i) + conduction(j,i-1))/2;
                G_matrix(n,nxp) = (conduction(j,i) + conduction(j,i+1))/2;
                G_matrix(n,nyp) = (conduction(j,i) + conduction(j+1,i))/2;

                solution1(n,1) = 0;
            elseif(j == length_y)

                G_matrix(n,n) = -(((conduction(j,i) + conduction(j,i-
1))/2)+((conduction(j,i) + conduction(j,i+1))/2)+((conduction(j,i) +
conduction(j-1,i))/2));
                G_matrix(n,nxm) = (conduction(j,i) + conduction(j,i-1))/2;
                G_matrix(n,nxp) = (conduction(j,i) + conduction(j,i+1))/2;
                G_matrix(n,nym) = (conduction(j,i) + conduction(j-1,i))/2;

                solution1(n,1) = 0;
            else

                G_matrix(n,n) = -(((conduction(j,i) + conduction(j,i-
1))/2)+((conduction(j,i) + conduction(j,i+1))/2)+((conduction(j,i) +
conduction(j-1,i))/2)+((conduction(j,i) + conduction(j+1,i))/2));
```

```
                G_matrix(n,nxm) = (conduction(j,i) + conduction(j,i-1))/2;
                G_matrix(n,nxp) = (conduction(j,i) + conduction(j,i+1))/2;
                G_matrix(n,nym) = (conduction(j,i) + conduction(j-1,i))/2;
                G_matrix(n,nyp) = (conduction(j,i) + conduction(j+1,i))/2;
                solution1(n,1) = 0;

            end
        end
    end

    V_matrix = G_matrix\solution1;

    for i = 1:width_x
        for j = 1:length_y
            n = j + (i - 1) * length_y;
            nxm = j + ((i-1) - 1) * length_y;
            nxp = j + ((i+1) - 1) * length_y;
            nym = (j-1) + (i - 1) * length_y;
            nyp = (j+1) + (i - 1) * length_y;
            surface(j,i) = V_matrix(n,1);
        end
    end

    [Efield_x,Efield_y] = gradient(surface);
    J_xdir = conduction.*(-Efield_x);
    J_ydir = conduction.*(-Efield_y);
    current_density(num) = mean(mean((((J_xdir.^2)+(J_ydir.^2)).^0.5)));
end

% Plotting the current density vs the conductivity
figure(15)
plot(1:num,(-1)*current_density,'m')
title('The Current vs the conductivity')
```