Carleton University

Course: ELEC 4700 Modelling of Integrated Device

Assignment No: 2

Finite Difference Method

Kwabena Gyasi Bawuah

101048814

Due: 02/28/2021

Table of Contents

Introduction:

This report is for ELEC 4700 in response to the call for an assignment report. The assignment was to model carriers as a population of electrons in an N-type Si semiconductor crystal (Monte-Carlo model). These particles are then to be giving velocities using the Maxwell-Boltzmann distribution. Lastly, an enhancement is to put the system to test by adding a bottle neck boundary. This report will detail the results from the built simulation, observation of results, discussion of results, answers to specific questions asked in the assignment and conclusions derived. Samples of the code used to perfume these models will also be produced within the sections.

Finite Difference Method Question 1:

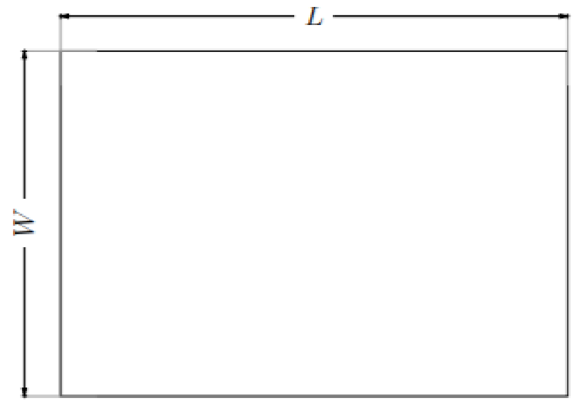


Figure 1: Rectangular region with isolated conducting sides

Use the Finite Difference Method to solve for the electrostatic potential in the rectangular region L × W shown in Figure 1 using $\nabla^2 V = 0$.

a)

Solve the simple case where V = V0 at x = 0 and V = 0 at x = L. The related 2D plot is shown below:

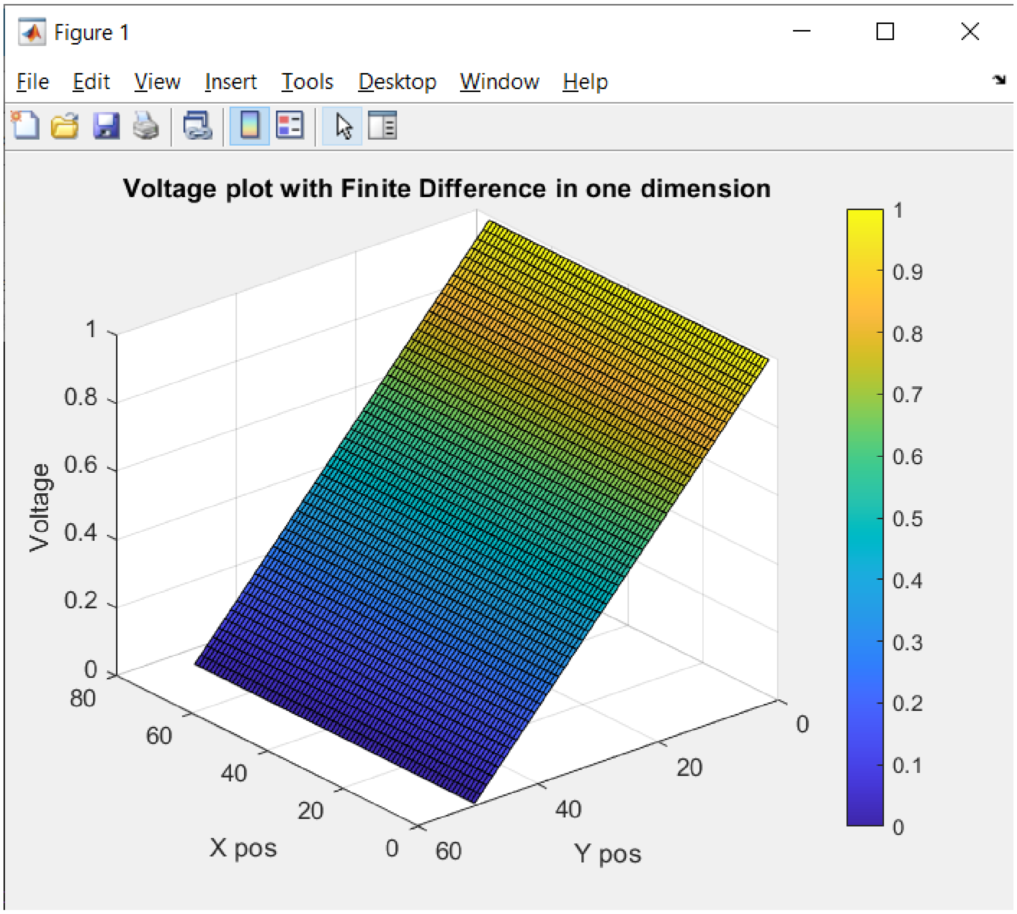


Figure 2: Voltage plot with finite difference in 1D

Code Used for Q1.a:

Two matrices are used for this part. Not just a G matrix, the matrix can be used for the operations of the G matrix. the solution will be in the form Ax = b, and to get x this will be b\A, in this case it will be G\Op.

```
%Assign 2
%Kwabena Gyasi Bawuah
%101048814

%initiailizing the dimensions of our matrices, ensuring L
is 3/2 times W
W = 50;
L = (3/2)*W;

G = sparse(L*W , L*W);
Op = zeros(L*W , 1);

%fill the g matrix
```

```
for i = 1:W
    for j = 1:L

        n = j + (i-1)*L;

        if i == 1  %left edge

            G(n,:) = 0;
            G(n,n) = 1;
            Op(n) = 1;

        elseif i == W %right edge

            G(n,:) = 0;
            G(n,n) = 1;
            Op(n) = 0;

        elseif j == 1 %bottom edge

            G(n, :) = 0;
            G(n, n) = -3;
            G(n, n+1) = 1;
            G(n, n+L) = 1;
            G(n, n-L) = 1;

        elseif j == L %top edge

            G(n, n) = -3;
            G(n, n-1) = 1;
            G(n, n+L) = 1;
            G(n, n-L) = 1;

        else   %inside parts

            G(n, n) = -4;
            G(n, n+1) = 1;
            G(n, n-1) = 1;
            G(n, n+L) = 1;
            G(n, n-L) = 1;

        end
    end
end

Voltage = G\Op;
```

```
%surfed matrix (x,y,voltage)
sol = zeros(W,L);

for i = 1:W
    for j = 1:L
        n = j + (i-1)*L;
        sol(i,j) = Voltage(n);
    end
end

figure(1)
surf(sol)
colorbar
title("Voltage plot with Finite Difference in one
dimension")
xlabel("X pos")
ylabel("Y pos")
zlabel("Voltage")
view(-130,30)
%the end
```

b) Solve the case where V = V0 at x = 0, x = L and V = 0 at y = 0, y = W. The related 2D plot is shown below:

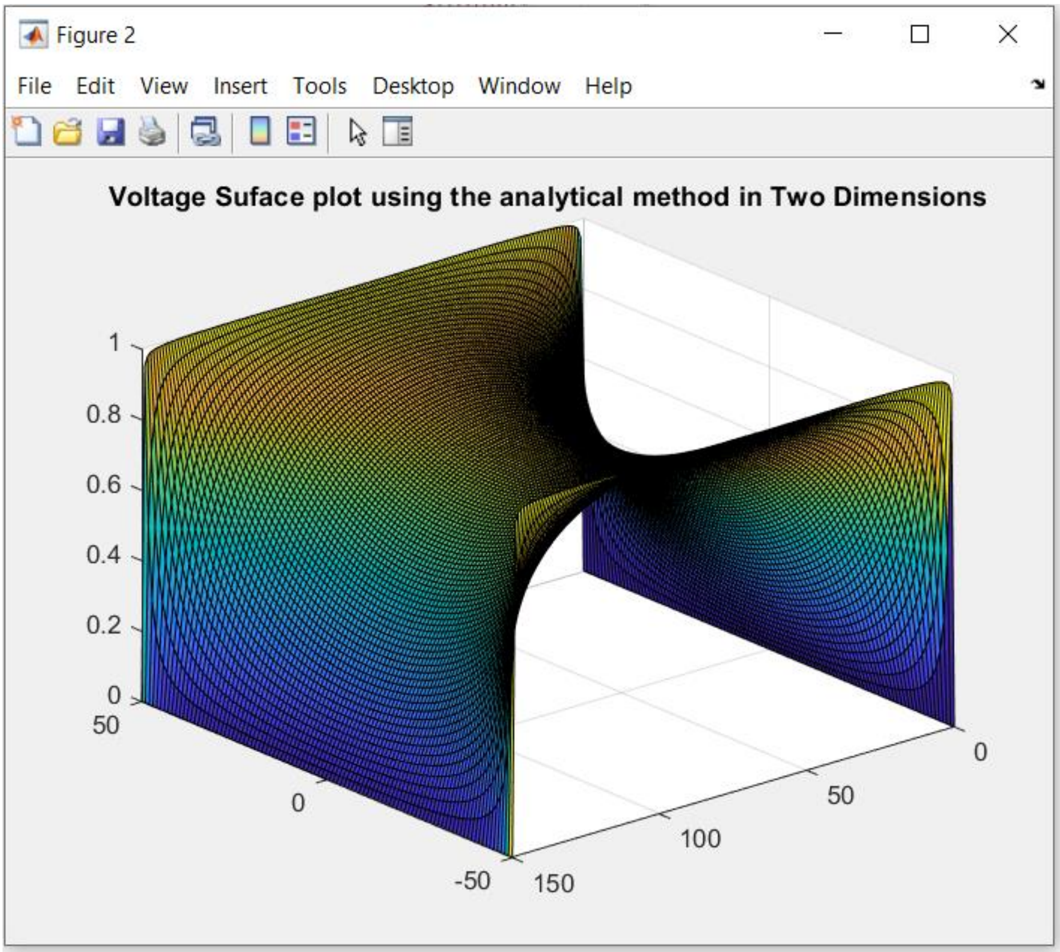


Figure 3: Voltage surface plot using the analytical method in 2D

Again, we are using finite difference method, but this time in 2-D. We are then finding a solution using the analytical method, which works by iterating to complete the summation of an infinite series. It will not, however, be infinite in this case.

Observation from results:
The solution from a series does approach the solution that was created using the FD method. Due to cosh and sinh the iterations are limited to 600. When I iterate above 600 the plot no longer looks like the true solution. This is because the cosh and sinh values approach infinity around this value, which increases the error in the solution, so we should stop at 600 iterations for best results.

Through judging the results obtained using both methods,
It seems that numerical solutions would be an applicable means of finding a solution, given that the information you are feeding it is not too complicated. It is a method that will work given you have the right computing power to handle the equations you throw into it. For very complex equations, the hardware one uses may not be able to handle it.

The analytical method, on the other hand, is better (quicker) at competing simpler equations, and is the method of choice when dealing with relatively small data sets (simpler equations). The limitations, however, can be surmised by observing this part of the assignment. Certain iteration values may cause a breakdown in the equation which limits its reliable accuracy. One must understand the limits of the equation to avoid these possible pitfalls.

Code Used for Q1.b:

```
%Assign 2
%Kwabena Gyasi Bawuah
%101048814

%initiailizing the dimensions of our matrices, ensuring L
is 3/2 times W
W = 100;
L = (3/2)*W;

G = sparse(W*L,W*L);
Op = sparse(W*L,1);

%filling in the G matrix's
for x = 1:W
    for y = 1:L
        n = y + (x-1)*L;

        if x == 1
            G(n, :) = 0;
            G(n, n) = 1;
            Op(n) = 1;
        elseif x == W
            G(n, :) = 0;
            G(n, n) = 1;
            Op(n) = 1;
        elseif y == 1
            G(n, :) = 0;
            G(n, n) = 1;
        elseif y == L
            G(n, :) = 0;
            G(n, n) = 1;
        else
            G(n, n) = -4;
            G(n, n+1) = 1;
            G(n, n-1) = 1;
            G(n, n+L) = 1;
            G(n, n-L) = 1;
```

```
        end
    end
end

Voltage = G\Op;
sol = zeros(W,L,1);

%surfed matrix (x,y,voltage)
for x = 1:W
    for y = 1:L
        n = y + (x-1)*L;
        sol(x,y) = Voltage(n);
    end
end

%variables to be used in our anayltical solution
a = L;
b = W/2;

x2 = linspace(-W/2,W/2, W);
y2 = linspace(0,L,L);

[i,j] = meshgrid(x2,y2);

sol2 = sparse(L,W);

figure(2)
surf(sol)
pause(0.01)
xlabel('X position');
ylabel('Y position');
zlabel('Voltage(x,y)');
title('Voltage Suface plot using the analytical method
2D)');
%iterating to create a summation of the infinite series
(finite in this
%case)

for n = 1:2:600
    if rem(n,2)==1
    sol2 = (sol2 +
(cosh(n*pi*i/a).*sin(n*pi*j/a))./(n*cosh(n*pi*b/a)));
    surf(x2,y2,(4/pi)*sol2)
```

```
    title("Voltage Suface plot using the analytical method
in Two Dimensions")
    axis tight
    view(-130,30);
    pause(0.001)
    end
end
```

Finite Difference Method Question 2:

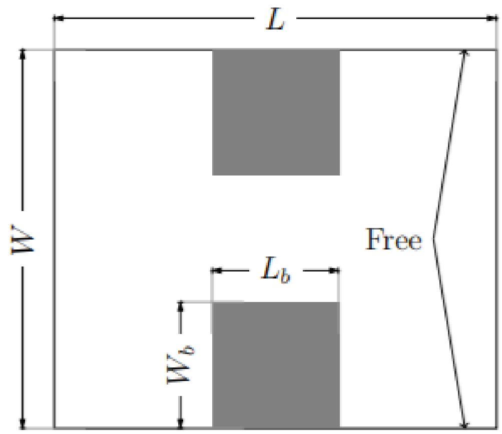


Figure 4: Rectangular region with isolated conducting sides and “bottle-neck”.

Use the Finite Difference Method to solve for the current flow in the rectangular region L×W shown in Figure 3 using ∇ (σx,y ∇V) = 0.

a)

Calculate the current flow at the two contacts. Generate plots of σ(x, y), V (x, y), E~ x, E~ y, J~(x, y)

In this part of the assignment, we are setting up 5 surface plots: sigma, voltage, the x and y components of the electric the field and finally the current density plot.

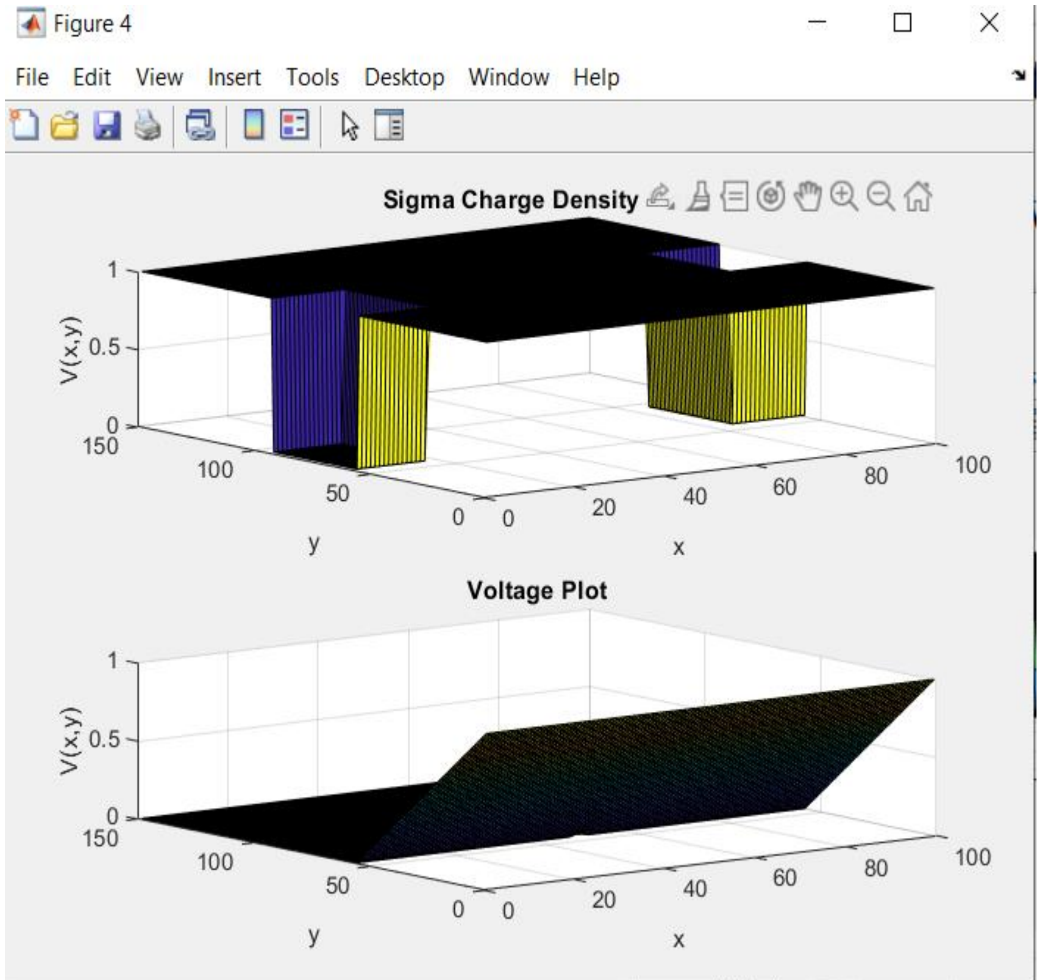


Figure 5: sigma and voltage plot

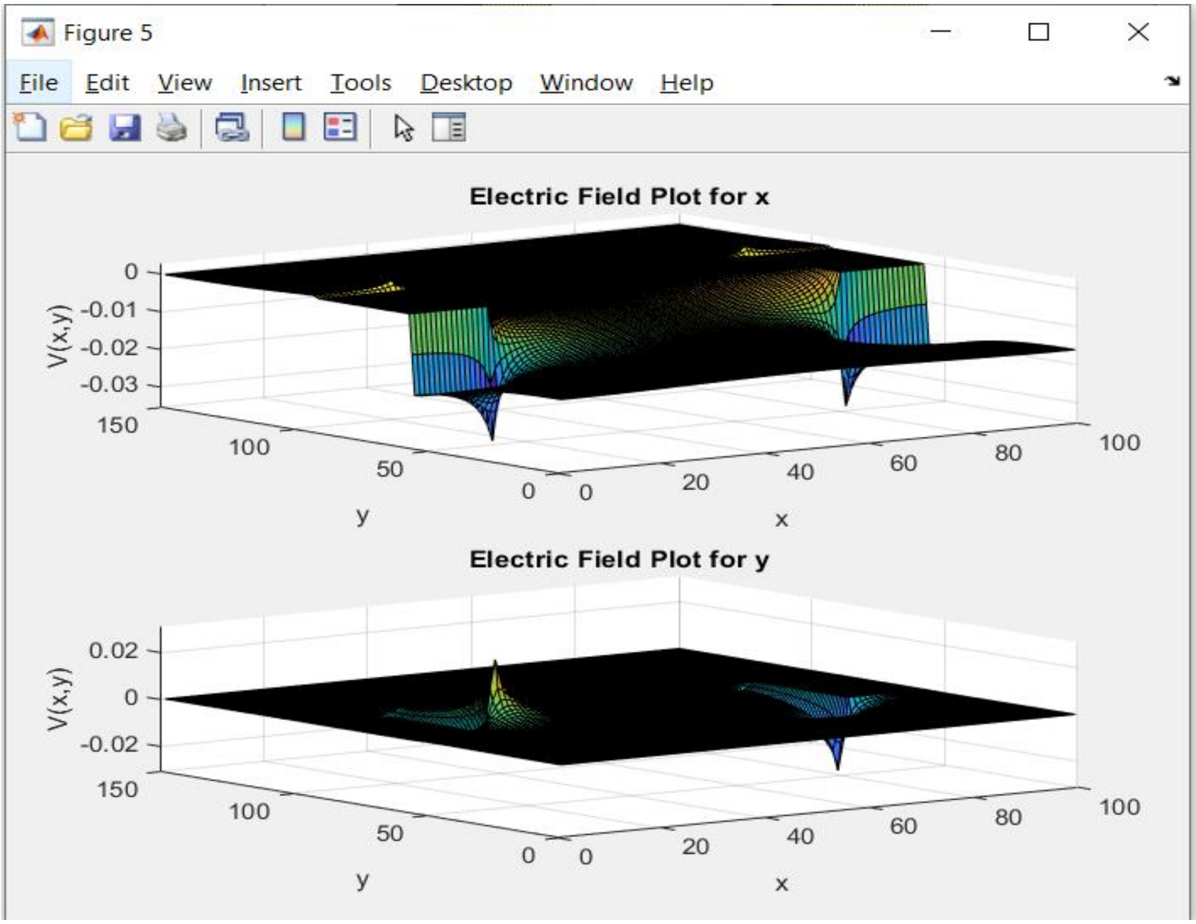


Figure 6: Electrical field plot for x and y

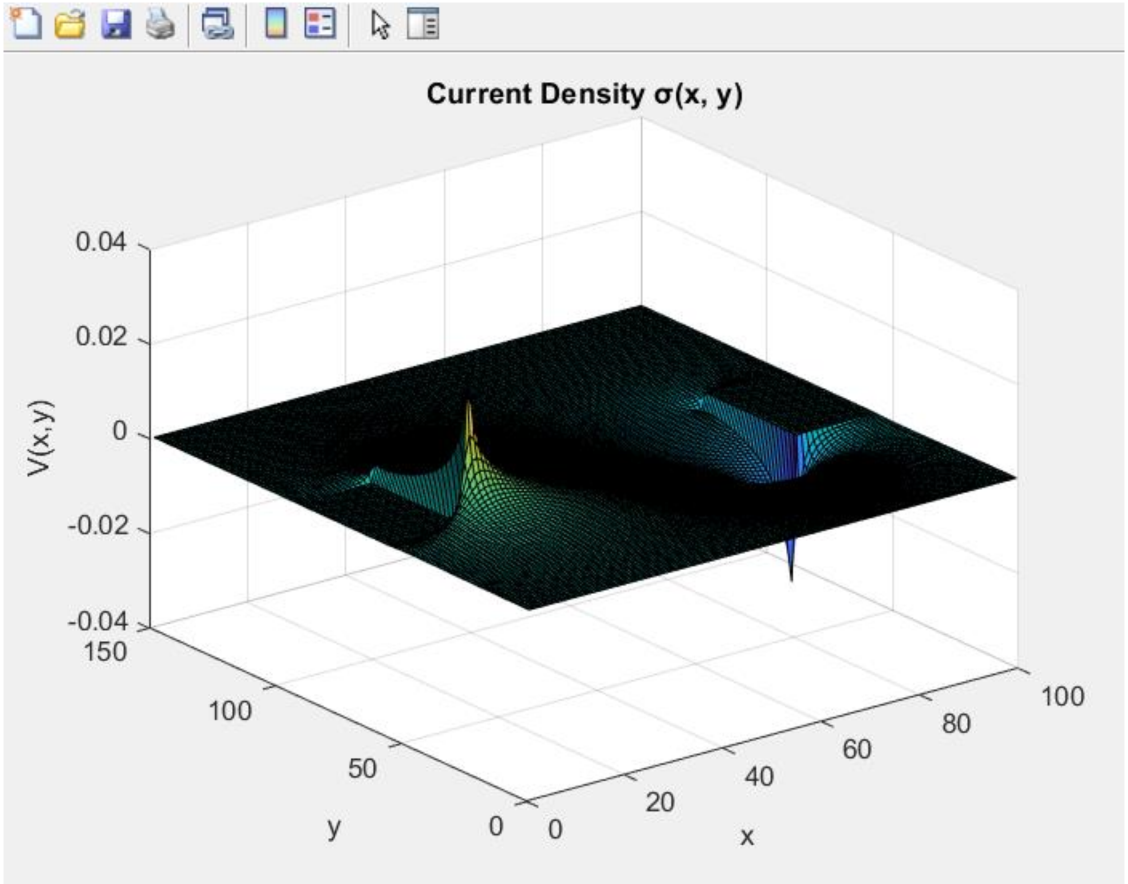


Figure 7: Current Density plot

Code Used for Q2.a:

```
%Assign 2
%Kwabena Gyasi Bawuah
%101048814

% using question 1 a as background

%initializing the dimensions of our matrices, ensuring L is
3/2 times W

W = 100;
L = (3/2)*W;

G = sparse(W*L,W*L);
Op = zeros(L*W,1);
```

```
midX = L/2;
midY = W/2;
boxL = L/4;
boxW = W*(2/3);
sigOut = 1;
sigIn = 10^-2;

leftEdge = midX - boxL/2;
rightEdge = midX + boxL/2;
topEdge = midY + boxW/2;
bottomEdge = midY - boxW/2;



for i=1:L
    for j=1:W
        n=j+(i-1)*W;
        nxms = j+(i-2)*W;
        nxps = j+(i)*W;
        nyms = (j-1)+(i-1)*W;
        nyps = (j+1)+(i-1)*W;
          if i == 1
            G(n,n) = 1;
            Op(n) = 1;
            sigmaMap(i,j) = sigOut;
        elseif i == L
            G(n,n) = 1;
            Op(n) = 0;
            sigmaMap(i,j) = sigOut;
        elseif (j == W)
            G(n,n) = -3;
            if(i>leftEdge && i<rightEdge)
                G(n,nxms) = sigIn;
                G(n,nxps) = sigIn;
                G(n,nyms) = sigIn;
                sigmaMap(i,j) = sigIn;
            else
                G(n,nxms) = sigOut;
                G(n,nxps) = sigOut;
                G(n,nyms) = sigOut;
                sigmaMap(i,j) = sigOut;
            end
        elseif (j == 1)
            G(n,n) = -3;
```

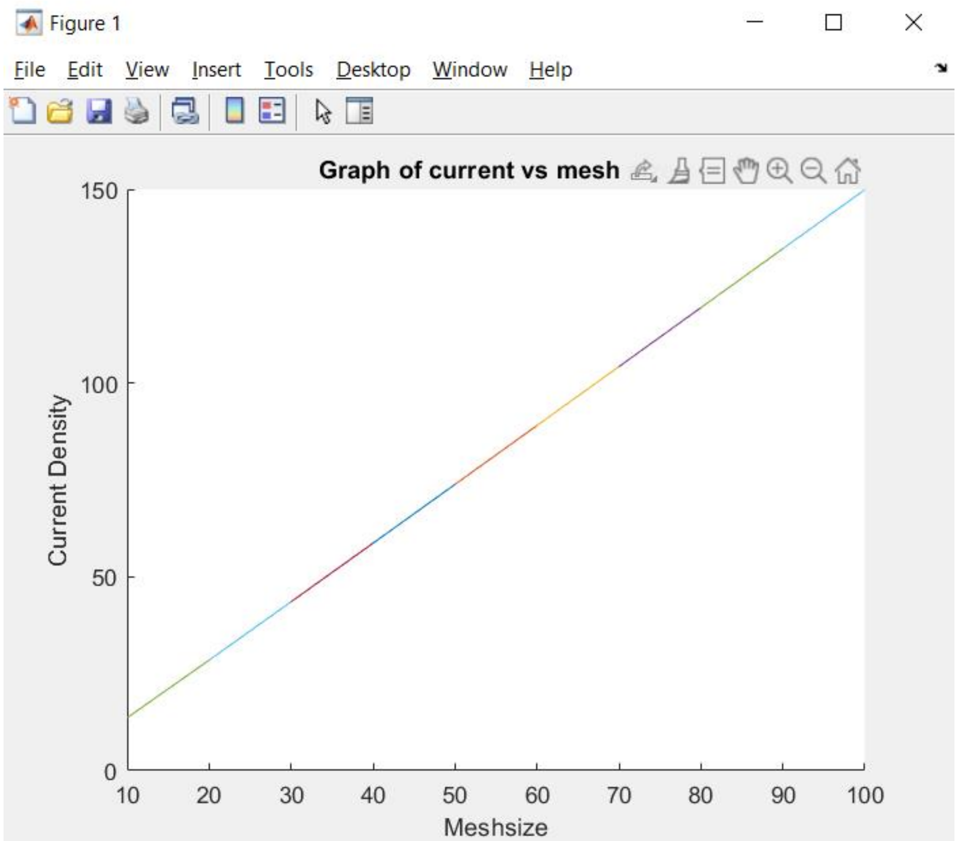
```
            if(i>leftEdge && i<rightEdge)
                G(n,nxms) = sigIn;
                G(n,nxps) = sigIn;
                G(n,nyps) = sigIn;
                sigmaMap(i,j) = sigIn;
            else
                G(n,nxms) = sigOut;
                G(n,nxps) = sigOut;
                G(n,nyps) = sigOut;
                sigmaMap(i,j) = sigOut;
            end
        else
            G(n,n) = -4;
            if( (j>topEdge || j<bottomEdge) && i>leftEdge
&& i<rightEdge)
                G(n,nxps) = sigIn;
                G(n,nxms) = sigIn;
                G(n,nyps) = sigIn;
                G(n,nyms) = sigIn;
                sigmaMap(i,j) = sigIn;
            else
                G(n,nxps) = sigOut;
                G(n,nxms) = sigOut;
                G(n,nyps) = sigOut;
                G(n,nyms) = sigOut;
                sigmaMap(i,j) = sigOut;
            end
        end
    end
end

Voltage = G\Op;
sol = zeros(L,W);

for x=1:L
    for y=1:W
        n = y + (x-1) * W;
        sol(x,y)= Voltage(n);
    end
end

%The electric field can be derived from the surface voltage
using a
%gradient
```

```
[Ey,Ex] = gradient(sol);
%J, the current density, is calculated  a surface plot is
derived by surfing this matrix.
E = gradient(sol);
J = sigmaMap.* E;

%V(x,y) Surface Plot
figure(4)
subplot(2,1,1);
surf(sigmaMap)
xlabel('x');
ylabel('y');
zlabel('V(x,y)')
title('Sigma Charge Density Plot');

subplot(2,1,2);
surf(phi)
xlabel('x');
ylabel('y');
zlabel('V(x,y)')
title('Voltage Plot');

%X component of electric field surface plot
figure(5)
subplot(2,1,1);
surf(Ex)
xlabel('x');
ylabel('y');
zlabel('V(x,y)')
title('Electric Field Plot for x');

%Y component of electric field surface plot
subplot(2,1,2);
surf(Ey)
xlabel('x');
ylabel('y');
zlabel('V(x,y)')
title('Electric Field Plot for y');

figure(6)
surf(J)
xlabel('x');
ylabel('y');
zlabel('V(x,y)')
```

```
title('Current Density ?(x, y)');
```

b) Graph of current vs mesh size

In this part of the assignment, we are to investigating the mesh density. To do this we will start at a mesh size multiple of 10, and incrementally increase this size to observe the effect on the current density.



Analyzing the results of the plot, we see that the meshsize and current density are proportional; an increase in meshsize leads to an increase in current density, which is to be expected.

Code Used for Q2.b:

```
%Assign 2
%Kwabena Gyasi Bawuah
%101048814
%Using q1 b as a background
```

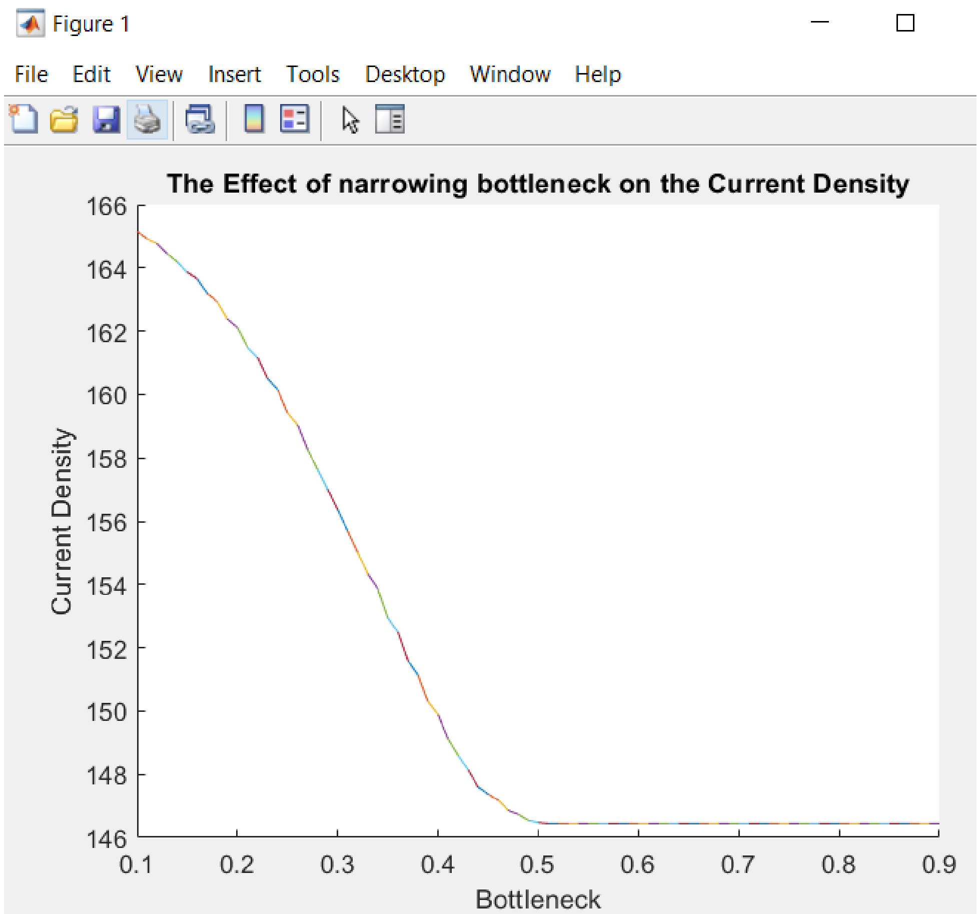
```matlab
%setting up variables just like part 1
for meshsize = 10:10:100

    %multiplying these values by the respective meshsize
    L = (3/2)*meshsize;
    G = sparse(meshsize*L);
    Op = zeros(1, meshsize*L);


    Sigmatrix = zeros(L, meshsize);
    Sig1 = 1;
    Sig2 = 10^-2;


    %bottleneck conditions with meshsize replacing w
    box = [meshsize*2/5 meshsize*3/5 L*2/5 L*3/5];

   %Filling in G matrix
    for x = 1:meshsize
        for y = 1:L
            n = y + (x-1)*L;
            if x == 1
                G(n, :) = 0;
                G(n, n) = 1;
                Op(n) = 1;
            elseif x == meshsize
                G(n, :) = 0;
                G(n, n) = 1;
                Op(n) = 0;
            elseif y == 1
                if x > box(1) && x < box(2)
                    G(n, n) = -3;
                    G(n, n+1) = Sig2;
                    G(n, n+L) = Sig2;
                    G(n, n-L) = Sig2;
                else
                    G(n, n) = -3;
                    G(n, n+1) = Sig1;
                    G(n, n+L) = Sig1;
                    G(n, n-L) = Sig1;
                end
            elseif y == L
                if x > box(1) && x < box(2)
                    G(n, n) = -3;
```

```
                    G(n, n+1) = Sig2;
                    G(n, n+L) = Sig2;
                    G(n, n-L) = Sig2;
                else
                    G(n, n) = -3;
                    G(n, n+1) = Sig1;
                    G(n, n+L) = Sig1;
                    G(n, n-L) = Sig1;
                end
            else
                if x > box(1) && x < box(2) && (y <
box(3)||y > box(4))
                    G(n, n) = -4;
                    G(n, n+1) = Sig2;
                    G(n, n-1) = Sig2;
                    G(n, n+L) = Sig2;
                    G(n, n-L) = Sig2;
                else
                    G(n, n) = -4;
                    G(n, n+1) = Sig1;
                    G(n, n-1) = Sig1;
                    G(n, n+L) = Sig1;
                    G(n, n-L) = Sig1;
                end
            end
        end
    end

    %Just like in part a), except using different meshsizes
    for Length = 1 : meshsize
        for Width = 1 : L
            if Length >= box(1) && Length <= box(2)
                Sigmatrix(Width, Length) = Sig2;
            else
                Sigmatrix(Width, Length) = Sig1;
            end
            if Length >= box(1) && Length <= box(2) &&
Width >= box(3) && Width <= box(4)
                Sigmatrix(Width, Length) = Sig1;
            end
        end
    end

    Voltage = G\Op';
```

```
    sol = zeros(L, meshsize, 1);

    for x = 1:meshsize
        for y = 1:L
            n = y + (x-1)*L;
            sol(y,x) = Voltage(n);
        end
    end

   %electric field found using gradient of voltage
   [elecx, elecy] = gradient(sol);
   %current desntiy is sigma times electric field
    J_x = Sigmatrix.*elecx;
    J_y = Sigmatrix.*elecy;
    J = sqrt(J_x.^2 + J_y.^2);

   %plotting current density vs mesh size
    figure(1)
    hold on
    if meshsize == 10
        Curr = sum(J, 1);
        Currtot = sum(Curr);
        Currold = Currtot;
        plot([meshsize, meshsize], [Currold, Currtot])
    end
    if meshsize > 10
        Currold = Currtot;
        Curr = sum(J, 2);
        Currtot = sum(Curr);
        plot([meshsize-10, meshsize], [Currold, Currtot])
        xlabel("Meshsize")
        ylabel("Current Density")
    end
    title("Graph of current vs mesh size")

end
```

c) Graph or table of current vs various bottlenecks.

In part, we are investigating the narrowing of the bottle neck, this is done by changing the y values of the box that we used in parts a and b. Again, we are going to loop through values to multiply the y values and observe the effects this has on current.



Observing the plot, we see that narrowing the bottleneck incrementally leads to a decrease in the current value, However, after a certain point, when the value of narrowing reaches 0.5, the current stagnates and does not decrease any more and stays fixed at about 71.5. Note that the relationship is not a linear decrease but resembles an exponential decrease before current density stops decreasing.

Code Used for Q2.c:

```
%Assign 2
%Kwabena Gyasi Bawuah
%101048814

for bottleneck = 0.1:0.01:0.9

 %setting up variable matrices like in part 1
    W = 100;
    L = W*3/2;
    G = sparse(W*L);
```

```
    Op = zeros(1, W*L);

    Sigmatrix = zeros(L, W);
    Sig1 = 10^-2;
    Sig2 = 1;


    %The bottleneck is incrementally "narrowed" by
modifying the y values
    %of the box
    box = [W*2/5 W*3/5 L*bottleneck L*(1-bottleneck)];

    %filling in the G matrix
    for i = 1:W

        for j = 1:L

            n = j + (i-1)*L;

            if i == 1

                G(n, :) = 0;
                G(n, n) = 1;
                Op(n) = 1;

            elseif i == W

                G(n, :) = 0;
                G(n, n) = 1;
                Op(n) = 0;

            elseif j == 1

                if i > box(1) && i < box(2)

                    G(n, n) = -3;
                    G(n, n+1) = Sig1;
                    G(n, n+L) = Sig1;
                    G(n, n-L) = Sig1;

                else

                    G(n, n) = -3;
                    G(n, n+1) = Sig2;
                    G(n, n+L) = Sig2;
```

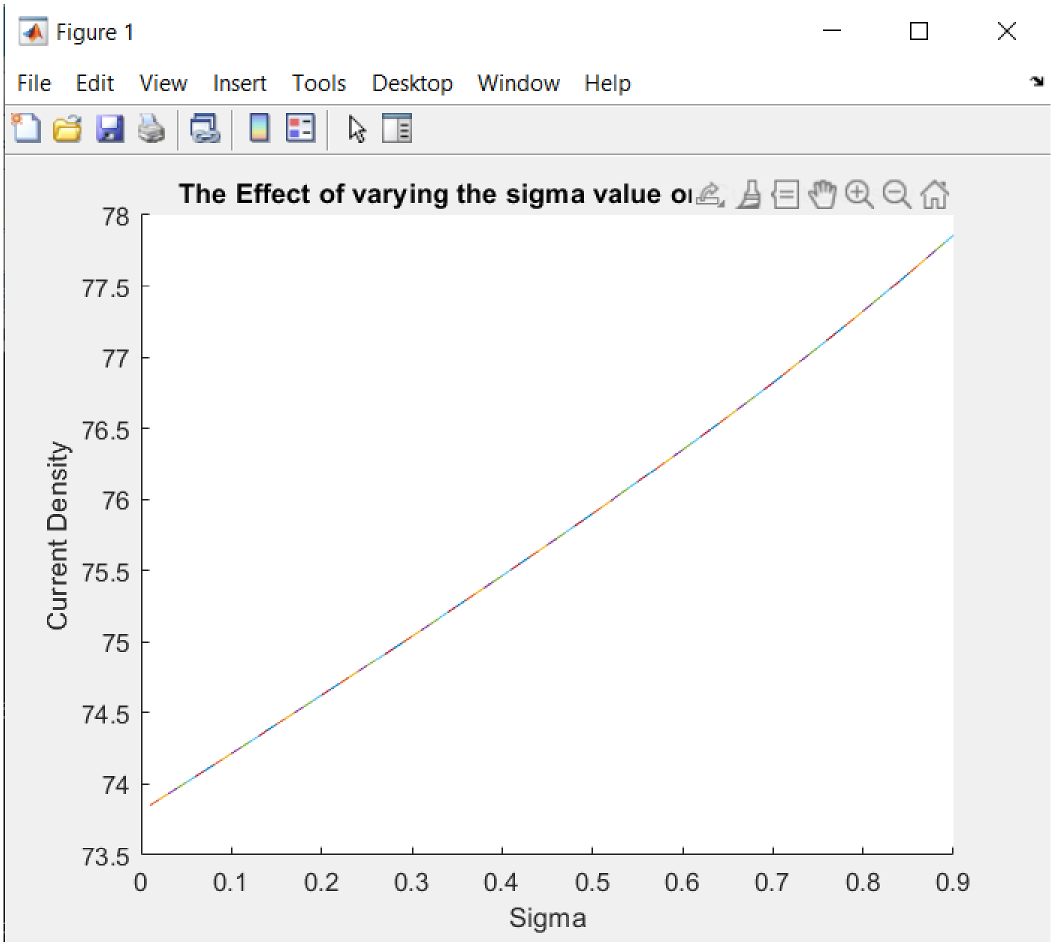
```
                    G(n, n-L) = Sig2;

                end

            elseif j == L

                if i > box(1) && i < box(2)
                    G(n, n) = -3;
                    G(n, n+1) = Sig1;
                    G(n, n+L) = Sig1;
                    G(n, n-L) = Sig1;

                else

                    G(n, n) = -3;
                    G(n, n+1) = Sig2;
                    G(n, n+L) = Sig2;
                    G(n, n-L) = Sig2;

                end

            else

                if i > box(1) && i < box(2) && (j <
box(3)||j > box(4))

                    G(n, n) = -4;
                    G(n, n+1) = Sig1;
                    G(n, n-1) = Sig1;
                    G(n, n+L) = Sig1;
                    G(n, n-L) = Sig1;

                else

                    G(n, n) = -4;
                    G(n, n+1) = Sig2;
                    G(n, n-1) = Sig2;
                    G(n, n+L) = Sig2;
                    G(n, n-L) = Sig2;

                end
            end
        end
    end
```

```
    for Length = 1 : W

        for Width = 1 : L

            if Length >= box(1) && Length <= box(2)
                Sigmatrix(Width, Length) = Sig1;

            else
                Sigmatrix(Width, Length) = Sig2;

            end

            if Length >= box(1) && Length <= box(2) &&
Width >= box(3) && Width <= box(4)

                Sigmatrix(Width, Length) = Sig2;
            end
        end
    end


    Voltage = G\Op';


    sol = zeros(L, W, 1);

    for i = 1:W

        for j = 1:L

            n = j + (i-1)*L;
            sol(j,i) = Voltage(n);

        end
    end

    %The electric field can be derived from the surface
voltage using a
    %gradient
    [elecx, elecy] = gradient(sol);

    %J, the current density, is calculated by multiplying
sigma and the
    %electric field together.
```

```
    J_x = Sigmatrix.*elecx;
    J_y = Sigmatrix.*elecy;
    J = sqrt(J_x.^2 + J_y.^2);

   %plotting bottleneck vs current
    figure(1)
    hold on

    if bottleneck == 0.1

        Curr = sum(J, 2);
        Currtot = sum(Curr);
        Currold = Currtot;
        plot([bottleneck, bottleneck], [Currold, Currtot])

    end

    if bottleneck > 0.1

        Currold = Currtot;
        Curr = sum(J, 2);
        Currtot = sum(Curr);
        plot([bottleneck-0.01, bottleneck], [Currold,
Currtot])
        xlabel("Bottleneck");
        ylabel("Current Density");

    end

    title("The Effect of narrowing bottleneck on the
Current Density")

end
```

d) Graph of current vs σ

In this part we are observing the effect of a varying sigma on the current density. Like in parts b) and c), we are iterating through a loop using different sigma values and plotting sigma vs current density, and then drawing a conclusion from the plot.



From the plot, sigma and current density are proportional; an increase in simga leads to an increase in current density. This relationship is linear, which is to be expected from the formula J = sigma x electric field.

Code Used for Q2.d:

```
%Assign 2
%Kwabena Gyasi Bawuah
%101048814

for sigma = 1e-2:1e-2:0.9

    %setting up variable matrices like in part 1
    W = 50;
    L = W*3/2;

    G = sparse(W*L);
    Op = zeros(1, W*L);
```

```
Sigmatrix = zeros(L, W);
Sig1 = 1;
Sig2 = sigma;

%bottleneck remains the same this time.
box = [W*2/5 W*3/5 L*2/5 L*3/5];
for x = 1:W

    for y = 1:L

        n = y + (x-1)*L;

        if x == 1

            G(n, :) = 0;
            G(n, n) = 1;
            Op(n) = 1;

        elseif x == W

            G(n, :) = 0;
            G(n, n) = 1;
            Op(n) = 0;

        elseif y == 1

            if x > box(1) && x < box(2)

                G(n, n) = -3;
                G(n, n+1) = Sig2;
                G(n, n+L) = Sig2;
                G(n, n-L) = Sig2;

            else

                G(n, n) = -3;
                G(n, n+1) = Sig1;
                G(n, n+L) = Sig1;
                G(n, n-L) = Sig1;

            end

        elseif y == L

            if x > box(1) && x < box(2)
```

```
                G(n, n) = -3;
                G(n, n+1) = Sig2;
                G(n, n+L) = Sig2;
                G(n, n-L) = Sig2;

            else

                G(n, n) = -3;
                G(n, n+1) = Sig1;
                G(n, n+L) = Sig1;
                G(n, n-L) = Sig1;

            end

        else

            if x > box(1) && x < box(2) && (y <
box(3)||y > box(4))

                G(n, n) = -4;
                G(n, n+1) = Sig2;
                G(n, n-1) = Sig2;
                G(n, n+L) = Sig2;
                G(n, n-L) = Sig2;

            else

                G(n, n) = -4;
                G(n, n+1) = Sig1;
                G(n, n-1) = Sig1;
                G(n, n+L) = Sig1;
                G(n, n-L) = Sig1;

            end
        end
    end
end


for Length = 1 : W
    for Width = 1 : L

        if Length >= box(1) && Length <= box(2)
            Sigmatrix(Width, Length) = Sig2;
```

```
            else

                Sigmatrix(Width, Length) = Sig1;

            end

            if Length >= box(1) && Length <= box(2) &&
Width >= box(3) && Width <= box(4)

                Sigmatrix(Width, Length) = Sig1;

            end
        end
    end


    Voltage = G\Op';


    sol = zeros(L, W, 1);

    for x = 1:W

        for y = 1:L

            n = y + (x-1)*L;

            sol(y,x) = Voltage(n);

        end
    end


    [elecx, elecy] = gradient(sol);


    J_x = Sigmatrix.*elecx;
    J_y = Sigmatrix.*elecy;
    J = sqrt(J_x.^2 + J_y.^2);


    figure(1)
    hold on
    if sigma == 0.01
        Curr = sum(J, 2);
```

```
        Currtot = sum(Curr);
        Currold = Currtot;
        plot([sigma, sigma], [Currold, Currtot])
    end
    if sigma > 0.01
        Currold = Currtot;
        Curr = sum(J, 2);
        Currtot = sum(Curr);
        plot([sigma-0.01, sigma], [Currold, Currtot])
        xlabel("Sigma")
        ylabel("Current Density")
    end
    title("The Effect of varying the sigma value on Current
Density")

end
```

Conclusion:

The Finite Difference Method was used to solve for the current flow in certain regions. The results from observations to be as expected. All questions where also as answers with the code used provided in the sections. The codes can be put together in a MATLAB file and run for a complete simulation of the system.