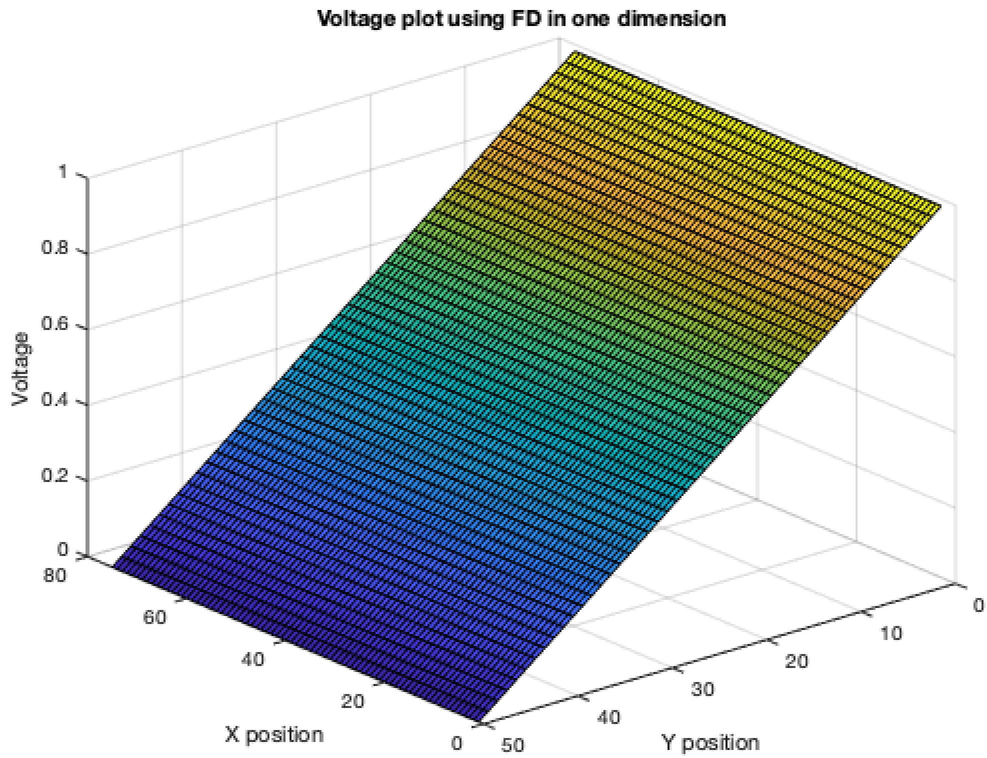
```
%Assignment 2 Part 1 A
%ELEC 4700
%Umeh Ifeanyi
%101045786

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

%we are going to need two matrices for this part.
%Not just a G matrix,
%but a matrix we can use 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.


G = sparse(nx*ny,nx*ny);
Op = zeros(nx*ny,1);

%filling in the G matrix's bulk nodes and BC's using a loop, similar
%to what we did in PA-5 using the Finite Difference method
for x = 1:nx
    for y = 1:ny

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

        if x == 1

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

        elseif x == nx

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

        elseif y == 1

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

        elseif y == ny

            G(n, n) = -3;
```

```
            G(n, n-1) = 1;
            G(n, n+ny) = 1;
            G(n, n-ny) = 1;

        else

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

        end
    end
end

Voltage =     G\Op;



%now, need to create matrix to be surfed (x,y,voltage)

sol = zeros(nx,ny,1);

for x = 1:nx
    for y = 1:ny

        n = y + (x-1)*ny;
        sol(x,y) = Voltage(n);
    end
end

figure(1)
surf(sol)
title("Voltage plot using FD in one dimension")
xlabel("X position")
ylabel("Y position")
zlabel("Voltage")
view(-130,30)
%the end
```

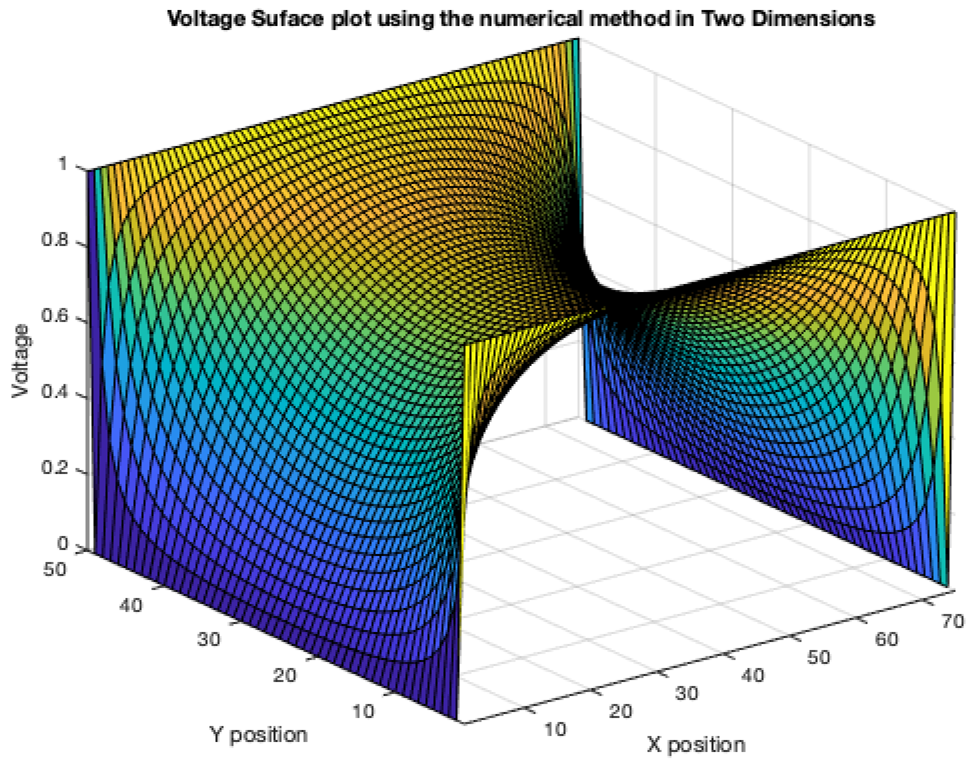


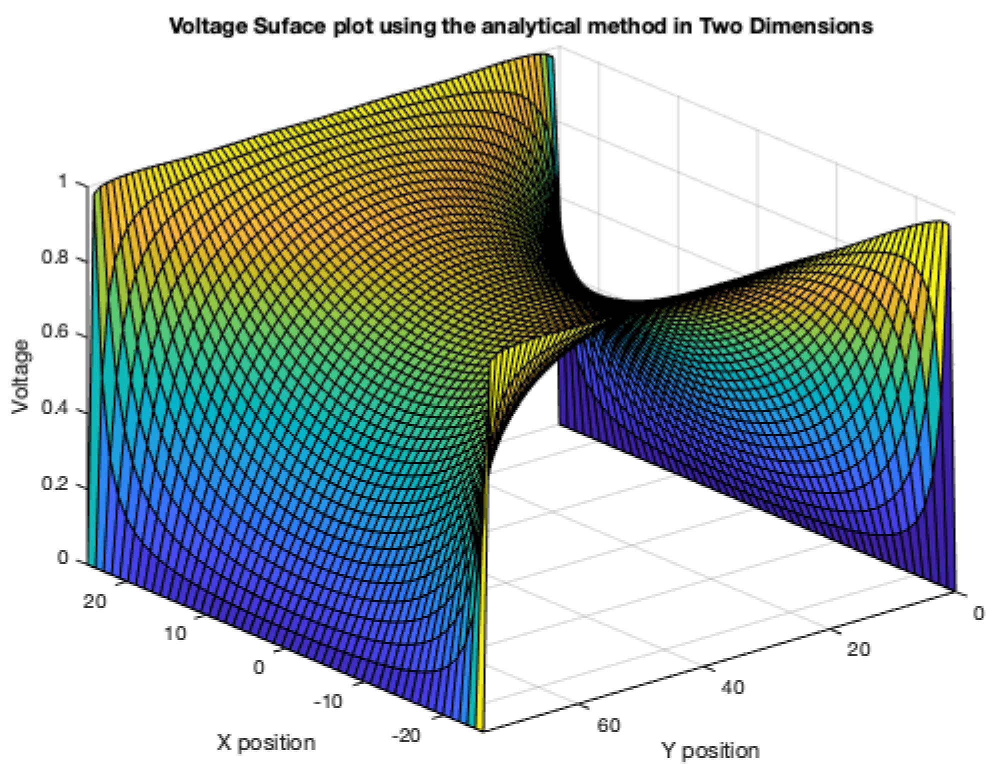
Published with MATLAB® R2019b

```
%Assignment 2 Part 1 B
%ELEC 4700
%Umeh Ifeanyi
%101045786

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

%In Part B), we are preparing and comparing two solutions. 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 won't, however, be
%infinite in this case. I will provide more discussion at the end of
%this code to further explain, and to answer the questions asked in
 the
%assignment outline

G = sparse(nx*ny,nx*ny);
Op = sparse(nx*ny,1);

%filling in the G matrix's bulk nodes and BC's using a loop, similar
%to what we did in PA-5 using the Finite Difference method

for x = 1:nx
    for y = 1:ny
        n = y + (x-1)*ny;

        if x == 1
            G(n, :) = 0;
            G(n, n) = 1;
            Op(n) = 1;
        elseif x == nx
            G(n, :) = 0;
            G(n, n) = 1;
            Op(n) = 1;
        elseif y == 1
            G(n, :) = 0;
            G(n, n) = 1;
        elseif y == ny
            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+ny) = 1;
            G(n, n-ny) = 1;
        end
```

```
    end
end


Voltage = G\Op;



%now, need to create matrix to be surfed (x,y,voltage)

sol = zeros(nx,ny,1);

for x = 1:nx
    for y = 1:ny

        n = y + (x-1)*ny;
        sol(x,y) = Voltage(n);
    end
end

figure(1)
surf(sol)
axis tight
title("Voltage Suface plot using the numerical method in Two
 Dimensions")
xlabel("X position")
ylabel("Y position")
zlabel("Voltage")

%variables to be used in our anayltical solution
a = ny;
b = nx/2;

x2 = linspace(-nx/2,nx/2, 50);
y2 = linspace(0,ny,ny);

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

sol2 = sparse(ny,nx);

%iterating to create a summation of the infinite series (finite in
 this
%case)

for n = 1:2:600

    sol2 = (sol2 + (cosh(n*pi*i/a).*sin(n*pi*j/a))./(n*cosh(n*pi*b/
a)));
    figure(2)
    surf(x2,y2,(4/pi)*sol2)
    title("Voltage Suface plot using the analytical method in Two
 Dimensions")
    xlabel("X position")
    ylabel("Y position")
```

```
    zlabel("Voltage")
    axis tight
    view(-130,30);
    pause(0.001)
end
%the end

%CONCLUSIONS

%The solution from a series does approach the solution that was
 created
%using the FD method. Note that that we were capped at 600 iterations
%because of the fact that this series equation contained the terms
 cosh and sinh.
%This is the maximum mumber of iterations that could be used to
 recreate
%the FD solution. When I iterate above 600 the plot no longer looks
 like
%the true solution. This is because the cosh and sinh values approach
%infinity aroung 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, I would like
 to
%compare and contrast the strengths and weaknesses of 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
 it's
%reliable accuracy. One must understand the limits of the equation to
 avoid
%these possible pitfalls.
```


Voltage Suface plot using the numerical method in Two Dimensions
Voltage
0
0.2
0.4
0.6
0.8
1
50
40
30
20
10
Y position
10
20
30
40
50
60
70
X position


Voltage Suface plot using the analytical method in Two Dimensions
Voltage
0
0.2
0.4
0.6
0.8
1
20
10
0
-10
-20
X position
60
40
20
0
Y position

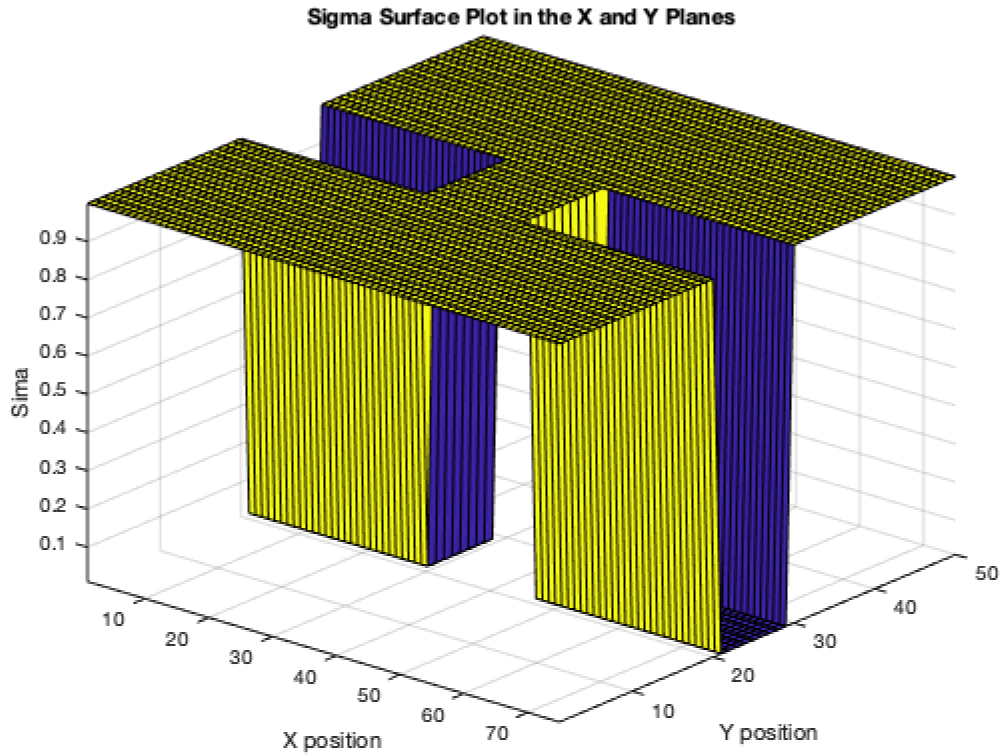
Published with MATLAB® R2019b

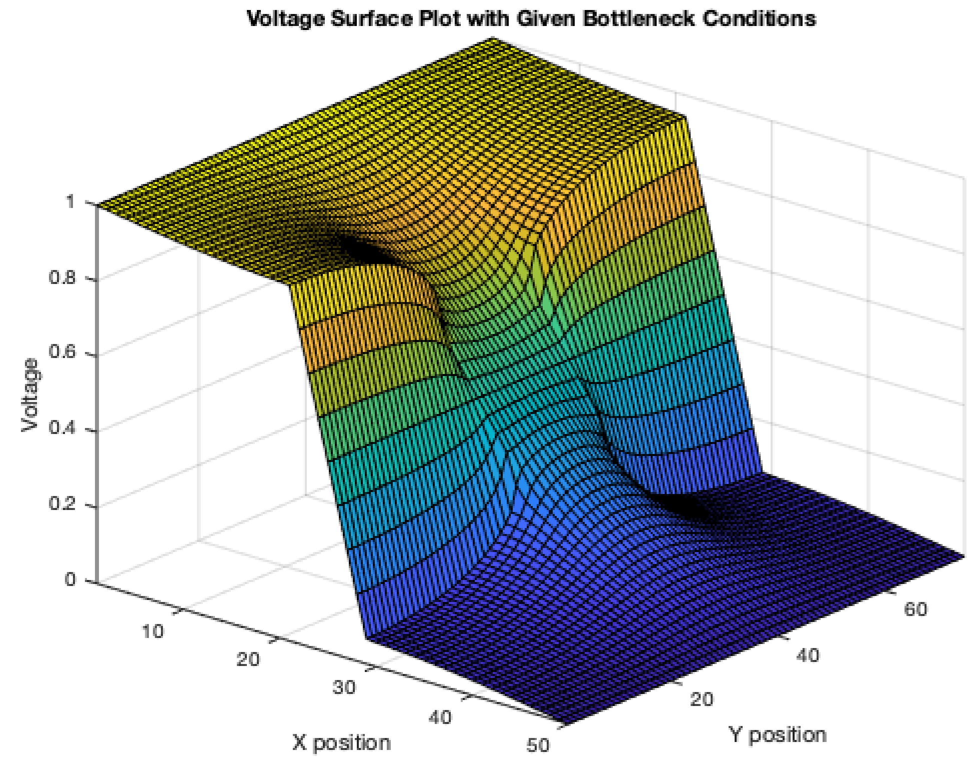
```
%Assignment 2 Part 2 A%
%ELEC 4700
%Umeh Ifeanyi
%101045786


%Part 2 a) in this part of the assignment, we are setting up
%5 surface plots: sigma, voltage, the x and y components of the
 electric
%field and finally the current density plot. Comments are littered
%throughout the code to aid in the understanding in my process.

%setting up variables just like part 1
nx = 50;
ny = (3/2)*nx;
G = sparse(nx*ny);
Op = zeros(1, nx*ny);


Sigmatrix = zeros(nx, ny);     % a sigma matrix is required for this
 part
Sig1 = 1;                      % sigma value given outside the box
Sig2 = 10^-2;                  % sigma value given inside the box

%The box will be difined using a 1x4 matrix containing it's dimensions
box = [nx*2/5 nx*3/5 ny*2/5 ny*3/5];

for i = 1:nx

    for j = 1:ny


        if i > box(1) && i < box(2) && (j < box(3)||j > box(4))
            Sigmatrix(i, j) = Sig2;

        else
            Sigmatrix(i, j) = Sig1;


        end
    end
end

% Filling in G matrix with corresponding bottleneck conditions
for x = 1:nx

    for y = 1:ny

        n = y + (x-1)*ny;
        nposx = y + (x+1-1)*ny;
        nnegx = y + (x-1-1)*ny;
        nposy = y + 1 + (x-1)*ny;
```

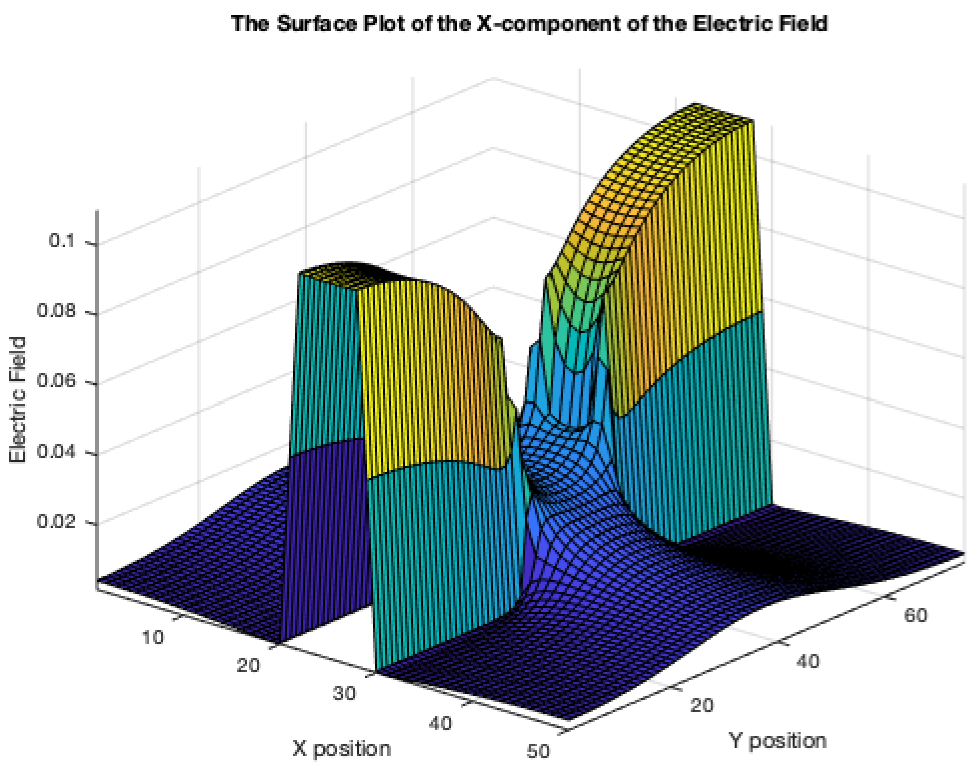
```
        nnegy = y - 1 + (x-1)*ny;

        if x == 1

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

        elseif x == nx

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

        elseif y == 1

            G(n, nposx) = (Sigmatrix(x+1, y) + Sigmatrix(x,y))/2;
            G(n, nnegx) = (Sigmatrix(x-1, y) + Sigmatrix(x,y))/2;
            G(n, nposy) = (Sigmatrix(x, y+1) + Sigmatrix(x,y))/2;
            G(n, n) = -(G(n,nposx)+G(n,nnegx)+G(n,nposy));

        elseif y == ny

            G(n, nposx) = (Sigmatrix(x+1, y) + Sigmatrix(x,y))/2;
            G(n, nnegx) = (Sigmatrix(x-1, y) + Sigmatrix(x,y))/2;
            G(n, nnegy) = (Sigmatrix(x, y-1) + Sigmatrix(x,y))/2;
            G(n, n) = -(G(n,nposx)+G(n,nnegx)+G(n,nnegy));

        else

            G(n, nposx) = (Sigmatrix(x+1, y) + Sigmatrix(x,y))/2;
            G(n, nnegx) = (Sigmatrix(x-1, y) + Sigmatrix(x,y))/2;
            G(n, nposy) = (Sigmatrix(x, y+1) + Sigmatrix(x,y))/2;
            G(n, nnegy) = (Sigmatrix(x, y-1) + Sigmatrix(x,y))/2;
            G(n, n) = -(G(n,nposx)+G(n,nnegx)+G(n,nposy)+G(n,nnegy));

        end
    end
end

% Sigma(x,y) Surface Plot
figure(1)
surf(Sigmatrix);
xlabel("X position")
ylabel("Y position")
zlabel("Sima")
axis tight
view([40 30]);
title("Sigma Surface Plot in the X and Y Planes")


Voltage = G\Op';
```

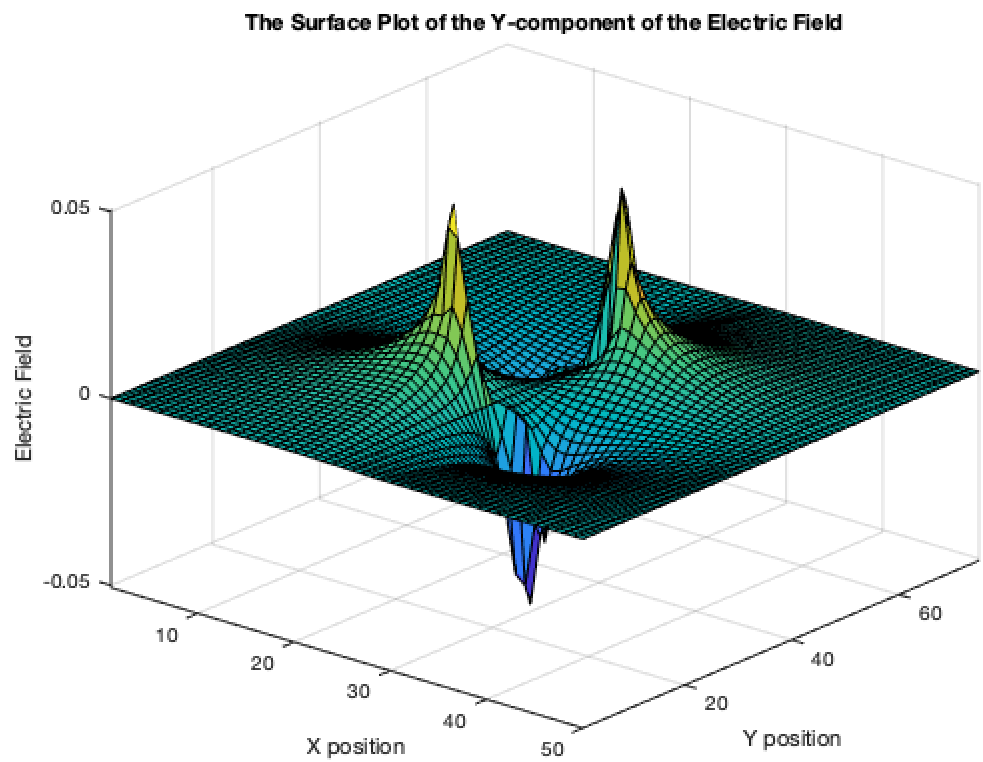
```
sol = zeros(ny, nx, 1);
for i = 1:nx
    for j = 1:ny
        n = j + (i-1)*ny;
        sol(j,i) = Voltage(n);
    end
end

%V(x,y) Surface Plot
figure(2)
surf(sol)
axis tight
xlabel("X position")
ylabel("Y position")
zlabel("Voltage")
view([40 30]);
title("Voltage Surface Plot with Given Bottleneck Conditions")

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

[elecx, elecy] = gradient(sol);

%X component of electric field surface plot
figure(3)
surf(-elecx)
axis tight
xlabel("X position")
ylabel("Y position")
zlabel("Electric Field")
view([40 30]);
title("The Surface Plot of the X-component of the Electric Field")

%Y component of electric field surface plot
figure(4)
surf(-elecy)
axis tight
xlabel("X position")
ylabel("Y position")
zlabel("Electric Field")
view([40 30]);
title("The Surface Plot of the Y-component of the Electric Field")

%J, the current density, is calculated by multiplying sigma and the
%electric field together. Combing the x and y matrices, a surface plot
 is
%derived by surfing this matrix.

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

%J(x,y) Surface Plot
figure(5)
```

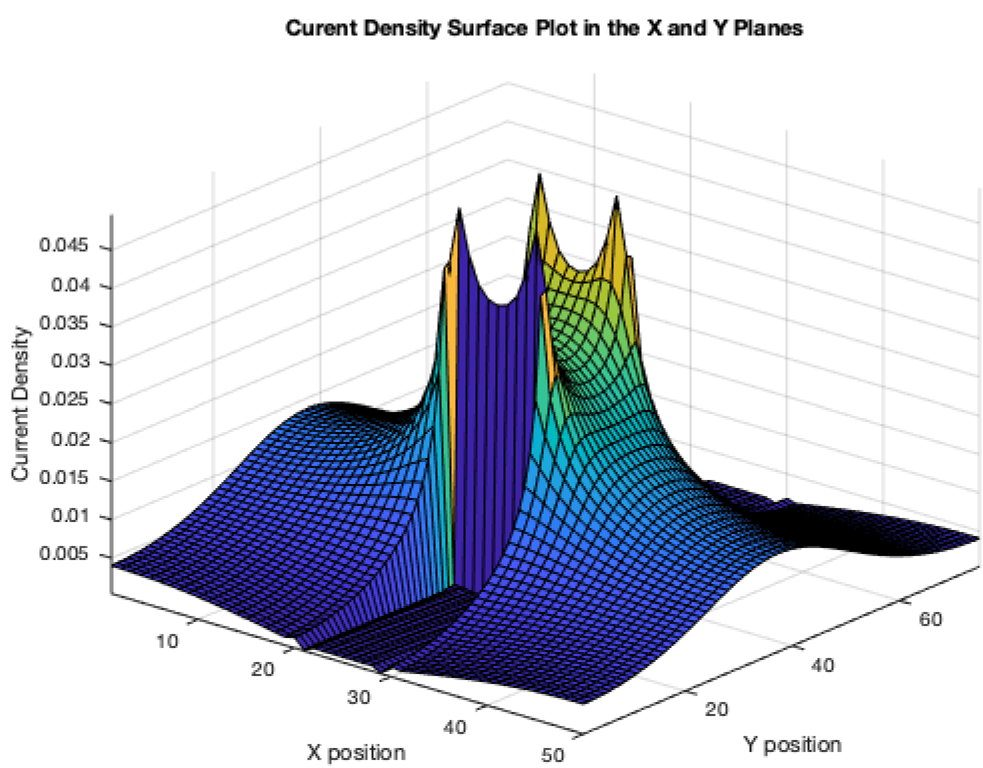
```
surf(J)
axis tight
xlabel("X position")
ylabel("Y position")
zlabel("Current Density")
view([40 30]);
title("Curent Density Surface Plot in the X and Y Planes")

%the end%
```




Voltage Surface Plot with Given Bottleneck Conditions
Voltage
1
0.8
0.6
0.4
0.2
0
10
20
30
40
50
X position
20
40
60
Y position


The Surface Plot of the X-component of the Electric Field
Electric Field
0.1
0.08
0.06
0.04
0.02
10
20
30
40
50
X position
20
40
60
Y position


The Surface Plot of the Y-component of the Electric Field
0.05
0
-0.05
Electric Field
10
20
30
40
50
X position
20
40
60
Y position


Curent Density Surface Plot in the X and Y Planes
0.045
0.04
0.035
0.03
0.025
0.02
0.015
0.01
0.005
Current Density
10
20
30
40
50
X position
20
40
60
Y position

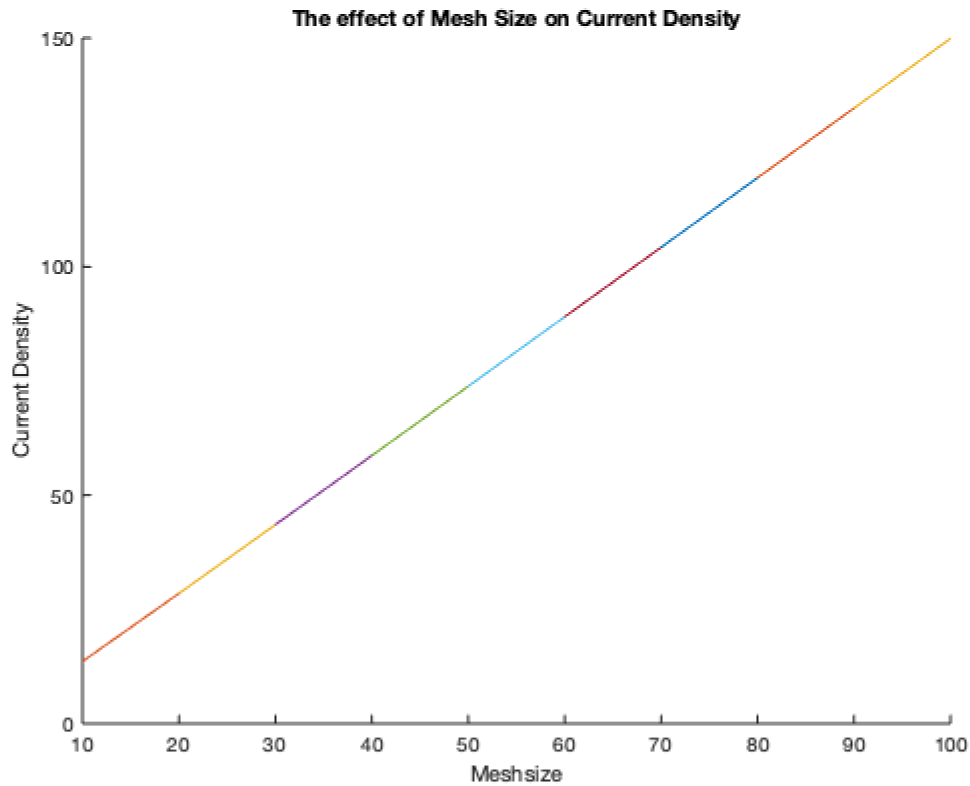
Published with MATLAB® R2019b

```
%Assignment 2 Part 2 B
%ELEC 4700
%Umeh Ifeanyi
%101045786

% In this part of the assignment we are
% investigating 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.

%nx will be incrementally increased using this loop. Note that
 meshsize
%replaces nx in this code.
for meshsize = 10:10:100

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


    Sigmatrix = zeros(ny, meshsize);       % The sigma matrix with nx
 set to meshsize
    Sig1 = 1;                              % sigma value given outside
 the box
    Sig2 = 10^-2;                          % sigma value given inside
 the box


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

   %Filling in G matrix
    for x = 1:meshsize

        for y = 1:ny

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

            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+ny) = Sig2;
                G(n, n-ny) = Sig2;

            else

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

            end

        elseif y == ny

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

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

            else

                G(n, n) = -3;
                G(n, n+1) = Sig1;
                G(n, n+ny) = Sig1;
                G(n, n-ny) = 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+ny) = Sig2;
                G(n, n-ny) = Sig2;

            else

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

            end
```

```
            end
        end
    end

    %Just like in part a), except using different meshsizes
    for Length = 1 : meshsize

        for Width = 1 : ny

            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(ny, meshsize, 1);

    for i = 1:meshsize

        for j = 1:ny

            n = j + (i-1)*ny;
            sol(j,i) = 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("The effect of Mesh Size on Current Density")

end

%the end%

%DISCUSSION%

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



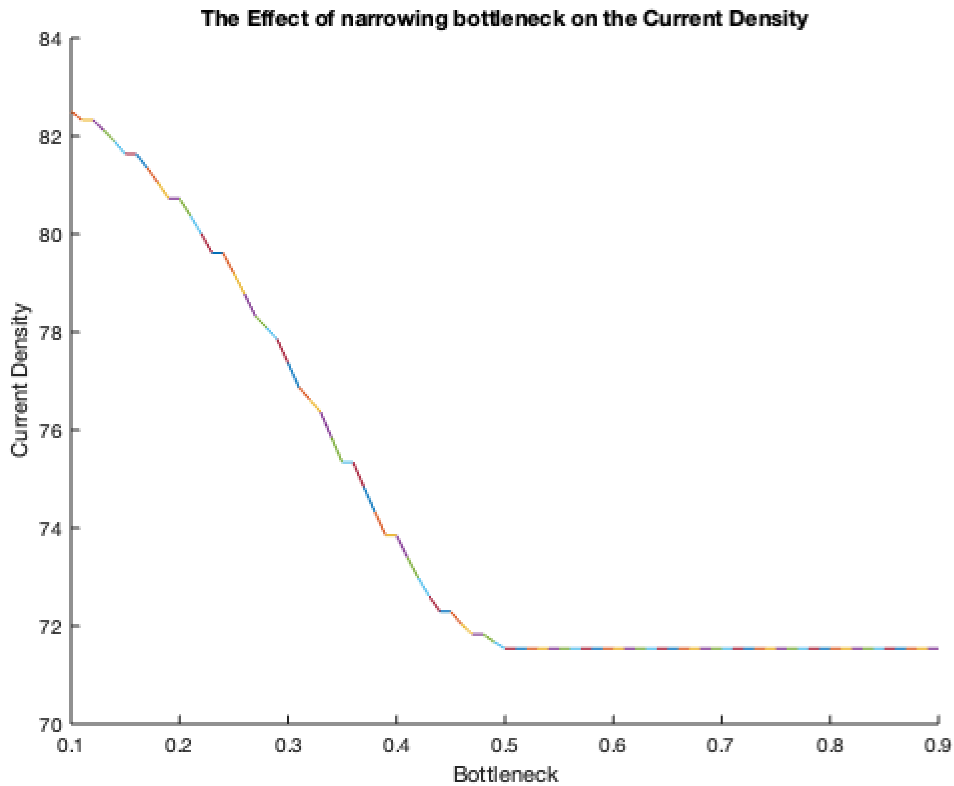
Published with MATLAB® R2019b

```
%Assignment 2 Part 2 B
%ELEC 4700
%Umeh Ifeanyi
%101045786

%In part 2c) we are investigating narrowing of the bottle neck, this
 is
%done by changing the y valyes 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. stay tuned for more

for bottleneck = 0.1:0.01:0.9

 %setting up variable matrices like in part 1
    nx = 50;
    ny = nx*3/2;
    G = sparse(nx*ny);
    Op = zeros(1, nx*ny);

    Sigmatrix = zeros(ny, nx); % a sigma matrix is required for this
 part
    Sig1 = 10^-2;               % sigma value given outside the box
    Sig2 = 1;                    % sigma value given inside the box


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

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

        for j = 1:ny

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

            if i == 1

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

            elseif i == nx

                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+ny) = Sig1;
                G(n, n-ny) = Sig1;

            else

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

            end

        elseif j == ny

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

            else

                G(n, n) = -3;
                G(n, n+1) = Sig2;
                G(n, n+ny) = Sig2;
                G(n, n-ny) = 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+ny) = Sig1;
                G(n, n-ny) = Sig1;

            else

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

            end
```

```
            end
        end
    end


    for Length = 1 : nx

        for Width = 1 : ny

            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(ny, nx, 1);

    for i = 1:nx

        for j = 1:ny

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

%DISCUSSION%

%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.
```



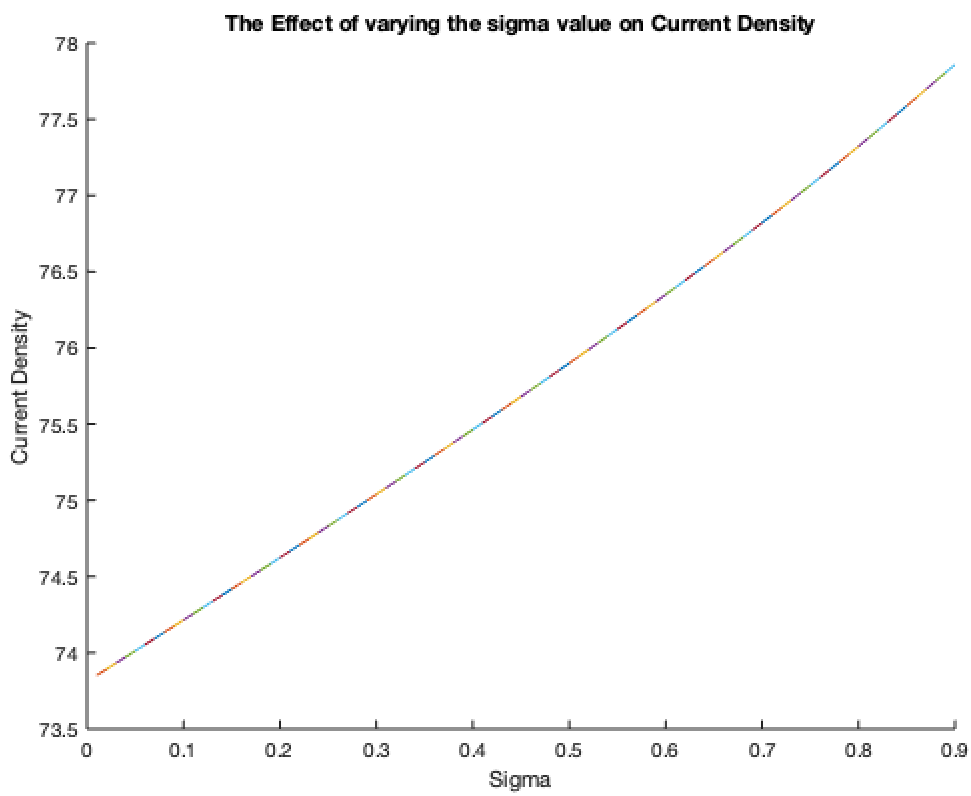
Published with MATLAB® R2019b

```
%In Part 2d) of the assignment, we are observing the effect of varying
%sigma on the current density. Like in parts c) and d), we are
 iterating
%through a loop using different sigma values and plotting sigma vs
 current
%density, and then drawing a conclusion from the plot.


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


    %setting up variable matrices like in part 1
    nx = 50;
    ny = nx*3/2;
    G = sparse(nx*ny);
    Op = zeros(1, nx*ny);

    Sigmatrix = zeros(ny, nx);             % a sigma matrix is
 required for this part
    Sig1 = 1;                              % sigma value given outside
 the box
    Sig2 = sigma;                           % sigma inside box will be
 modified

    %bottleneck remains the same this time.
    box = [nx*2/5 nx*3/5 ny*2/5 ny*3/5];


    for x = 1:nx

        for y = 1:ny

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

            if x == 1

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

            elseif x == nx

                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+ny) = Sig2;
                G(n, n-ny) = Sig2;

            else

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

            end

        elseif y == ny

            if x > box(1) && x < box(2)
                G(n, n) = -3;
                G(n, n+1) = Sig2;
                G(n, n+ny) = Sig2;
                G(n, n-ny) = Sig2;

            else

                G(n, n) = -3;
                G(n, n+1) = Sig1;
                G(n, n+ny) = Sig1;
                G(n, n-ny) = 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+ny) = Sig2;
                G(n, n-ny) = Sig2;

            else

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

            end
        end
    end
end
```

```
    for Length = 1 : nx

        for Width = 1 : ny

            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(ny, nx, 1);

    for x = 1:nx

        for y = 1:ny

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

            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

%the end%

%DISCUSSION%

%From the plot it is noticed that 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.
```



Published with MATLAB® R2019b