Table of Contents


```
clear
close
```

ELEC4700 Assignment 2

Qiushi(Leonard) Chen


```
%101049864
```

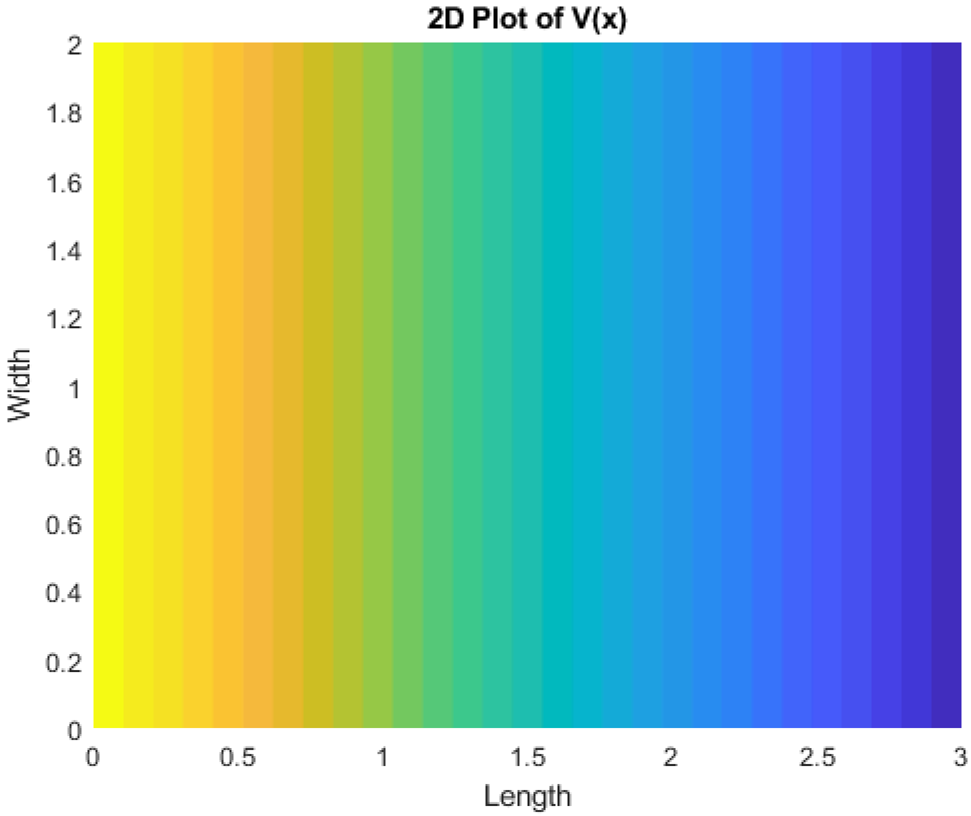
ELEC4700 Assignment 2 Part 1(a)

```
VoL = 1; % Left side of the Area has Boundary Voltage = VoL
VoR = 0; % Right side of the Area has Boundary Voltage = VoR
L = 3; % Area Length
W = 2; % Area Width
GridSize = 0.1; % Each mesh grid has the Length and Width of 0.01
nx = L/GridSize; % Length mesh size
ny = W/GridSize; % Width mesh size
delta = 1; % Delta x and Delta y of FD condition
G = sparse(nx*ny,nx*ny); % G matrix has size(nx*ny,nx*ny)
B = zeros(nx*ny,1); % B is the product of G matrix * V
```

(a) Solve the simple case where V = V0 at x = 0 and V = 0 at x = L.Note that in this case the top/bottom BC are not fixed

```
%
for Horizontal = 1: nx
    for Vertical = 1:ny
        n = Vertical+(Horizontal-1)*ny;
        % Left side Boundary Condition
        if Horizontal == 1
```

```
            G(n,:) = 0;
            G(n,n) = 1/delta;
            B(n) = VoL;
        % Right side Boundary Condition
        elseif Horizontal == nx
            G(n,:) = 0;
            G(n,n) = 1/delta;
            B(n) = VoR;

        else
            nxm = Vertical+((Horizontal-1)-1)*ny;
            nxp = Vertical+((Horizontal+1)-1)*ny;
            G(n,n) = -2/delta;
            G(n,nxm) = 1/delta;
            G(n,nxp) = 1/delta;
        end
    end
end

Vn = G\B; % Find (ny*ny:1), size V = G\B
% Mapping the V to the matrix size of nx*ny
for irow = 1: nx
    for jcolumn = 1:ny
         n = jcolumn+(irow-1)*ny;
         V(irow,jcolumn) = Vn(n);
    end
end
GMatrix = full(G);
figure('name','2D Plot of V(x)')

% Creates a 2D Voltage surface plot
x = linspace(0,L,nx);
y = linspace(0,W,ny);
H = surf(x,y,V');

set(H,'linestyle','none');
title("2D Plot of V(x)")
xlabel("Length")
ylabel("Width")
zlabel("Voltage (V)")
xlim([0 L])
ylim([0 W])
view(2)
```



ELEC4700 Assignment 2 Part 1(b)

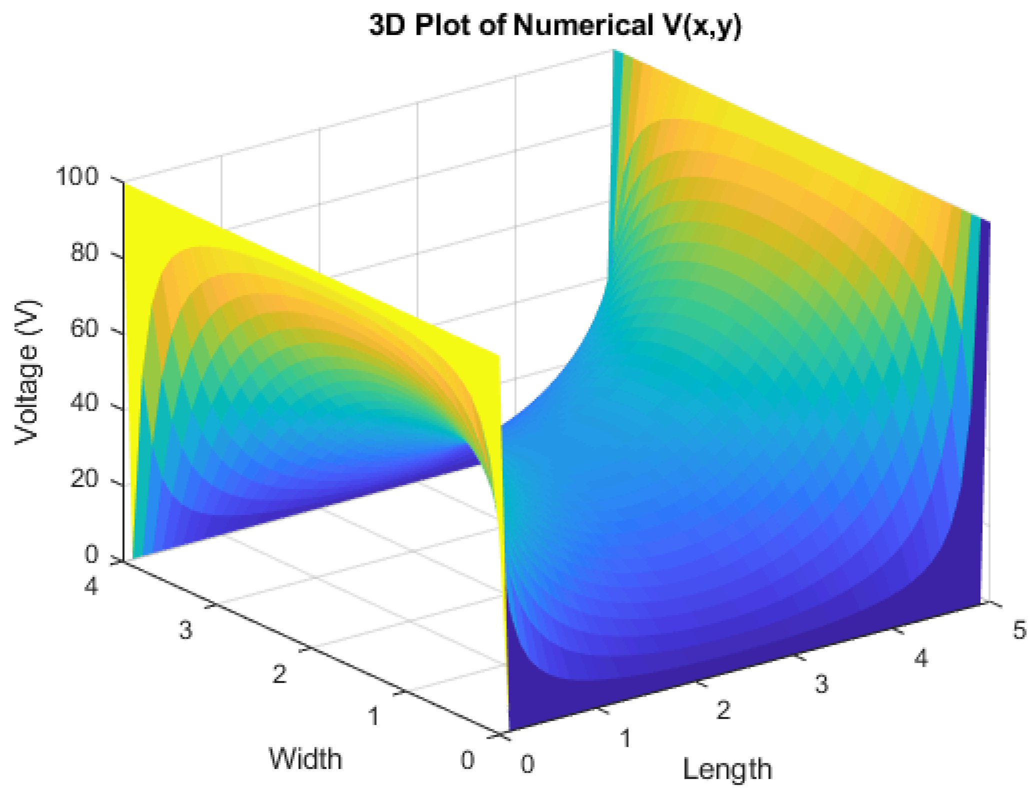
```
VoL = 100; % Left side of the Area has Boundary Voltage = VoL
VoR = 100; % Right side of the Area has Boundary Voltage = VoR
VoT = 0; % Top side of the Area has Boundary Voltage = VoT
VoB = 0; % Bottom side of the Area has Boundary Voltage = VoB

L = 5; % Area Length
W = 4; % Area Width
GridSize = 0.1; % Each mesh grid has the Length and Width of 0.01
nx = L/GridSize; % Length mesh size
ny = W/GridSize; % Width mesh size
index = 150; % index # of analytical series
delta = 1; % Delta x and Delta y of FD condition
G = sparse(nx*ny); % G matrix has size(nx*ny,nx*ny)
B = zeros(nx*ny,1); % B is the product of G matrix * V
VTheory = zeros(nx,ny); % Voltage Analytical series solution
```

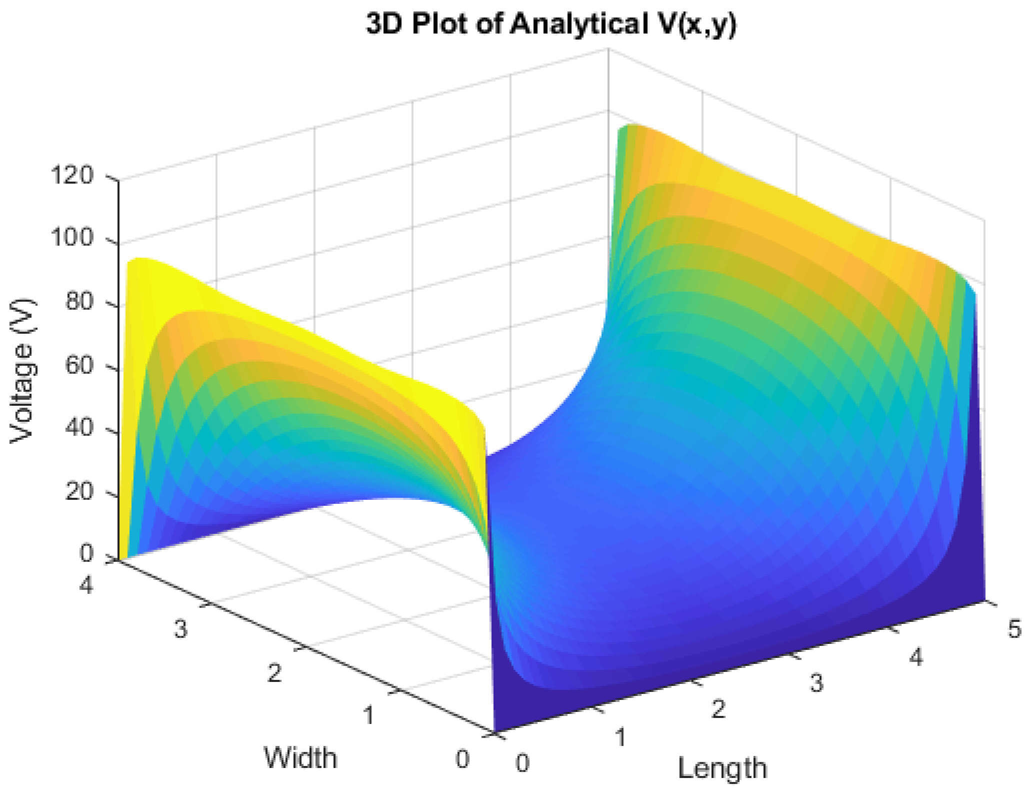
(b) Solve the case where V = V0 at x = 0, x = L and V = 0 at y = 0, y = W. Compare the solution of a bunch of mesh sizes to the analytical series solution

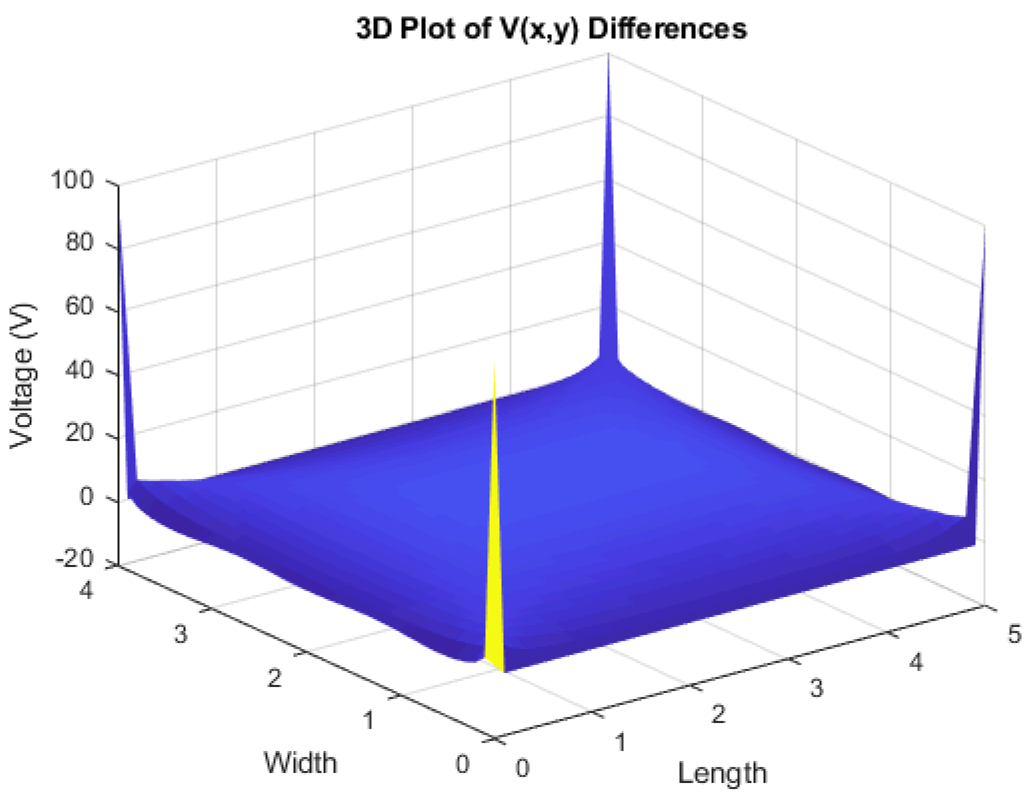
```
for iRow = 1: nx
    for jColumn = 1:ny
        n = jColumn+(iRow-1)*ny;
        % Left side Boundary Condition
        if iRow == 1
            G(n,:) = 0;
            G(n,n) = 1/delta;
            B(n) = VoL;
        % Right side Boundary Condition
        elseif iRow == nx
            G(n,:) = 0;
            G(n,n) = 1/delta;
            B(n) = VoR;

        % Bottom side Boundary Condition
        elseif jColumn == 1
            G(n,n) = 1/delta;
            B(n) = VoB;
        % Top side Boundary Condition
        elseif jColumn == ny
            G(n,n) = 1/delta;
            B(n) = VoT;
        else
            nxm = jColumn+((iRow-1)-1)*ny;
            nxp = jColumn+((iRow+1)-1)*ny;
            nym = (jColumn-1)+(iRow-1)*ny;
            nyp = (jColumn+1)+(iRow-1)*ny;
            G(n,n) = -4/delta;
            G(n,nxm) = 1/delta;
            G(n,nxp) = 1/delta;
            G(n,nym) = 1/delta;
            G(n,nyp) = 1/delta;
        end

    end

end
fullG = full(G);

% Find ny*ny:1 size V = G\B
Vn = G\B;
% Mapping the V to the matrix size of nx*ny
for iRow = 1: nx
    for jColumn = 1:ny
         n = jColumn+(iRow-1)*ny;
```

```
        V(iRow,jColumn) = Vn(n);
    end
end


x = linspace(0,L,nx);
y = linspace(0,W,ny);
[X,Y] = meshgrid(x,y);
% Creates a 3D Voltage numerical surface plot

figure('name','3D Plot of Numerical V(x,y)')
set(surf(X',Y',V),'linestyle','none');
title("3D Plot of Numerical V(x,y)")
xlabel("Length")
ylabel("Width")
zlabel("Voltage (V)")


%
% Gfull = full(G);
%
% figure(1)
% spy(G)

% Analytical solution
a = nx; % set a
b = ny; % set b
x1 = linspace(-b,b,nx); % set x of analytical solution
y1 = linspace(0,a,ny); % set y of analytical solution
for k=1:2:index % odd index only
    for i = 1:nx
        for j = 1:ny
            % Analytical series solution
            VTheory(i,j) = VTheory(i,j) + 4*VoL/pi*1/k*cosh(k*pi*x1(i)/a)/
cosh(k*pi*b/a)*sin(k*pi*y1(j)/a);
        end
    end
end

% Creates a 3D Voltage Analytical surface plot
figure('name','3D Plot of Analytical V(x,y)')
set(surf(X',Y',VTheory),'linestyle','none');
title("3D Plot of Analytical V(x,y)")
xlabel("Length")
ylabel("Width")
zlabel("Voltage (V)")

% Determine the numerical and analytical solution's differences
error = V - VTheory;
figure('name','3D Plot of V(x,y) Differences')
set(surf(X',Y',error),'linestyle','none');
title("3D Plot of V(x,y) Differences")
xlabel("Length")
ylabel("Width")
```

```
zlabel("Voltage (V)")
```




3D Plot of Analytical V(x,y)
Voltage (V)
120
100
80
60
40
20
0
4
3
2
1
0
Width
0
1
2
3
4
5
Length


3D Plot of V(x,y) Differences
Voltage (V)
100
80
60
40
20
0
-20
4
3
2
1
0
Width
0
1
2
3
4
5
Length

Conclusions on meshing and comments on numerical vs analytical

As shown in the plot of the error between the analytical and numerical Voltage solutions, the difference between two solutions are very small at the middle of the area. Howeve, at each corner of the area, the differences are significant. The reason causes this happen is because the corners are relatively hard to converge for the analytical solutions. It usually takes more terms (close to infinite) to obtain the solution with the least errors compare to the numerical one. Numerical and analytical method are both great ways to modeling FD. The numerical solution have the advantage like: easy to calculate, and when there is a condition that there are no analytical equations, the numerical will still be useful. It's disadvantage is: It usually needs high mesh density to get an accurate solution.

```
    % For Analytical method, it is usually more accurate than numerical
    % method, becasue Analytical method are presented as math expressions.
%
```

ELEC4700 Assignment 2 Part 2(a)

```
VoL = 100; % Left side of the Area has Boundary Voltage = VoL
VoR = 0; % Right side of the Area has Boundary Voltage = VoR
VoT = 0; % Top side of the Area has Boundary Voltage = VoT
VoB = 0; % Bottom side of the Area has Boundary Voltage = VoB
Pixel = 10; % Number of mesh per unit length/width

Lb = 2; % Box Length
Wb = 2; % Box Width
xBox = Lb*Pixel; % Box length Pixel Number
yBox = Wb*Pixel; % Box width Pixel Number
L = 9; % Area Length
W = 6; % Area Width
nx = L*Pixel; % Area length Pixel Number
ny = W*Pixel; % Area wdith Pixel Number
sigma = 1; % Outside Box Area Conductivity
BoxSigma = 0.01; % Inside Box Area Conductivity
G = sparse(nx*ny); % G matrix has size(nx*ny,nx*ny)
B = zeros(nx*ny,1); % B is the product of G matrix * V
Conductivity = sigma*ones(nx,ny); % Conductivity of the entire area
```

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

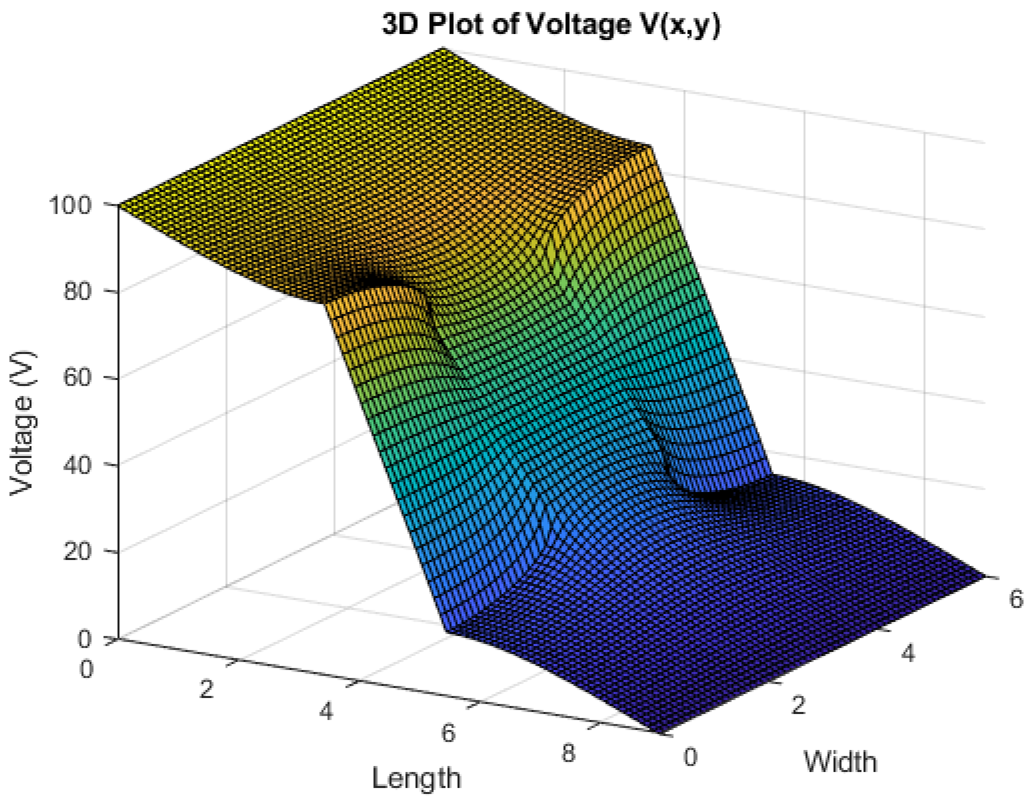
```
% At the left side of the area contact, follow euqation R =
% rho*Length/Area. In this case is a 2D model, so the area will be just
% Width. R = sigma*Length/Width = 1*9/6 = 1.5. Also, Current = Voltage/R =
% VoL/1.5 = 1/1.5 = 0.6667A.

% Set Inside Box area conductivity to variable BoxSigma
for iRow = 1:nx
    for jColumn = 1:ny
```

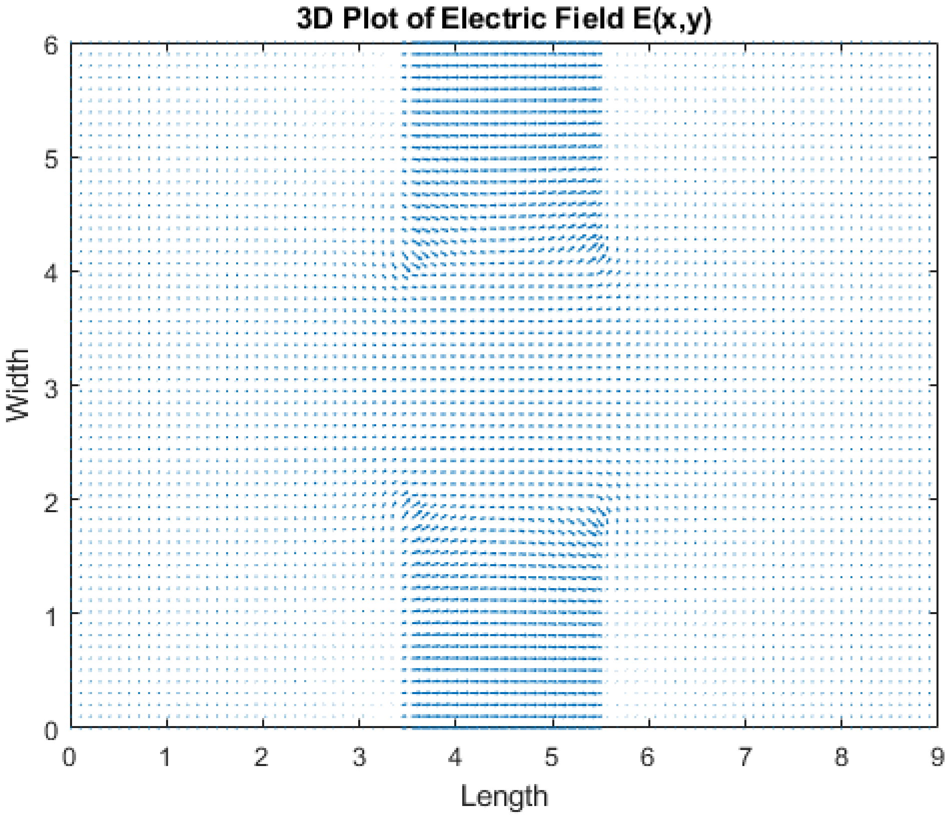
```
        if iRow>=(nx-xBox)/2 && iRow<=((nx+xBox)/2) && jColumn<=yBox
            Conductivity(iRow,jColumn) = BoxSigma;
        elseif iRow>=(nx-xBox)/2 && iRow<=((nx+xBox)/2) && jColumn<=ny &&
 jColumn>ny-yBox
            Conductivity(iRow,jColumn) = BoxSigma;
        end
    end
end

% for jColumn = 1:ny
%     for iRow = 1:nx
for iRow = 1:nx
    for jColumn = 1:ny
        n = jColumn+(iRow-1)*ny;
        % Left side Boundary Condition

        if iRow == 1
            G(n,:) = 0;
            G(n,n) = 1;
            B(n) = VoL;
        % Right side Boundary Condition

        elseif iRow == nx
            G(n,:) = 0;
            G(n,n) = 1;
            B(n) = VoR;

        % Bottom side Boundary Condition
        elseif jColumn == 1
            nxm = jColumn+((iRow-1)-1)*ny;
            nxp = jColumn+((iRow+1)-1)*ny;
            nyp = (jColumn+1)+(iRow-1)*ny;

            rxm = (Conductivity(iRow,jColumn) +
 Conductivity(iRow-1,jColumn))/2;
            rxp = (Conductivity(iRow,jColumn) + Conductivity(iRow
+1,jColumn))/2;
            ryp = (Conductivity(iRow,jColumn) + Conductivity(iRow,jColumn
+1))/2;

            G(n,n) = -(rxm+rxp+ryp);
            G(n,nxm) = rxm;
            G(n,nxp) = rxp;
            G(n,nyp) = ryp;

        % Top side Boundary Condition
        elseif jColumn == ny
            nxm = jColumn+((iRow-1)-1)*ny;
            nxp = jColumn+((iRow+1)-1)*ny;
            nym = (jColumn-1)+(iRow-1)*ny;

            rxm = (Conductivity(iRow,jColumn) +
 Conductivity(iRow-1,jColumn))/2;
```

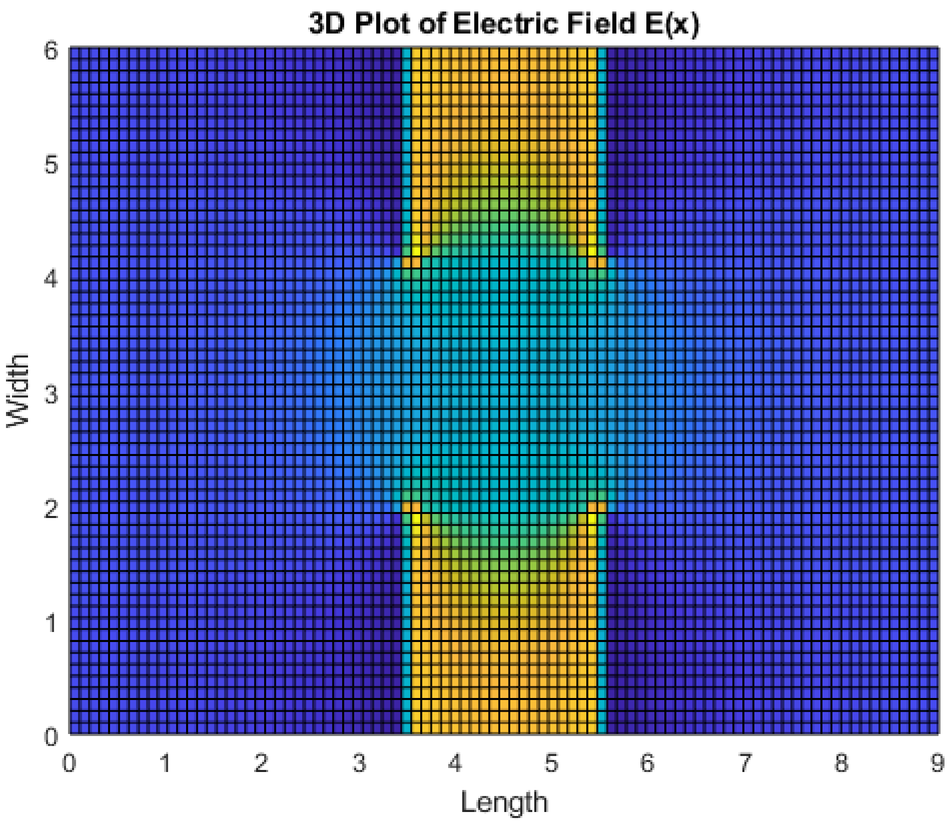
```
            rxp = (Conductivity(iRow,jColumn) + Conductivity(iRow
+1,jColumn))/2;
            rym = (Conductivity(iRow,jColumn) +
 Conductivity(iRow,jColumn-1))/2;

            G(n,n) = -(rxm+rxp+rym);
            G(n,nxm) = rxm;
            G(n,nxp) = rxp;
            G(n,nym) = rym;

        else
            nxm = jColumn+((iRow-1)-1)*ny;
            nxp = jColumn+((iRow+1)-1)*ny;
            nym = (jColumn-1)+(iRow-1)*ny;
            nyp = (jColumn+1)+(iRow-1)*ny;

            rxm = (Conductivity(iRow,jColumn) +
 Conductivity(iRow-1,jColumn))/2;
            rxp = (Conductivity(iRow,jColumn) + Conductivity(iRow
+1,jColumn))/2;
            rym = (Conductivity(iRow,jColumn) +
 Conductivity(iRow,jColumn-1))/2;
            ryp = (Conductivity(iRow,jColumn) + Conductivity(iRow,jColumn
+1))/2;

            G(n,n) = -(rxm+rxp+rym+ryp);
            G(n,nxm) = rxm;
            G(n,nxp) = rxp;
            G(n,nym) = rym;
            G(n,nyp) = ryp;
        end

    end

end


% 3D Plot of Voltage V(x,y)
figure('name','Voltage V(x,y)')

Vn = G\B; % Find (ny*ny:1) size, V = G\B
% Mapping the V to the matrix size of nx*ny
for iRow = 1: nx
    for jColumn = 1:ny
         n = jColumn+(iRow-1)*ny;
         V(iRow,jColumn) = Vn(n);
    end
end

x = linspace(0,L,nx);
y = linspace(0,W,ny);
[X,Y] = meshgrid(x,y);

surf(x,y,V')
```

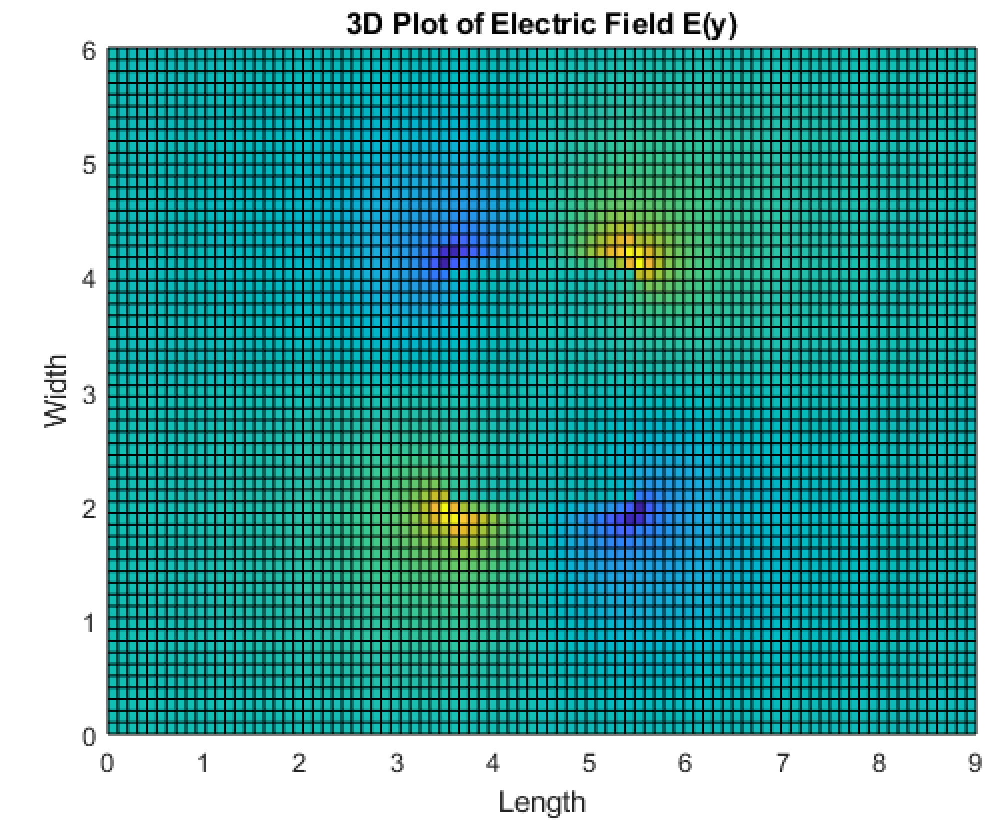
```
view(31,23)
xlim([0 L])
ylim([0 W])
title("3D Plot of Voltage V(x,y)")
xlabel("Length")
ylabel("Width")
zlabel("Voltage (V)")

% 3D Plot of Electric Field E(x,y)
[Ex,Ey] = gradient(V');
figure('name','Electric Field E(x,y)')
quiver(X,Y,-Ex,-Ey)
xlim([0 L])
ylim([0 W])
title("3D Plot of Electric Field E(x,y)")
xlabel("Length")
ylabel("Width")
zlabel("Electric Field (V/m)")

figure('name','Electric Field E(x)')
surf(X,Y,-Ex)
xlim([0 L])
ylim([0 W])
title("3D Plot of Electric Field E(x)")
xlabel("Length")
ylabel("Width")
zlabel("Electric Field (V/m)")
view(2)

figure('name','Electric Field E(y)')
surf(X,Y,-Ey)
xlim([0 L])
ylim([0 W])
title("3D Plot of Electric Field E(y)")
xlabel("Length")
ylabel("Width")
zlabel("Electric Field (V/m)")
view(2)

%E = sqrt((Ex.^2)+ (Ey.^2));

% 3D Plot of Conductivity Cond(x,y)
figure('name','Conductivity Cond(x,y)')
cond = Conductivity';
surf(X,Y,cond)
xlim([0 L])
ylim([0 W])
title("3D Plot of Conductivity Cond(x,y)")
xlabel("Length")
ylabel("Width")
zlabel("Conductivity (ohm*m)")

% 3D Plot of Current Density J(x,y)
figure('name','Current density J(x,y)')
```

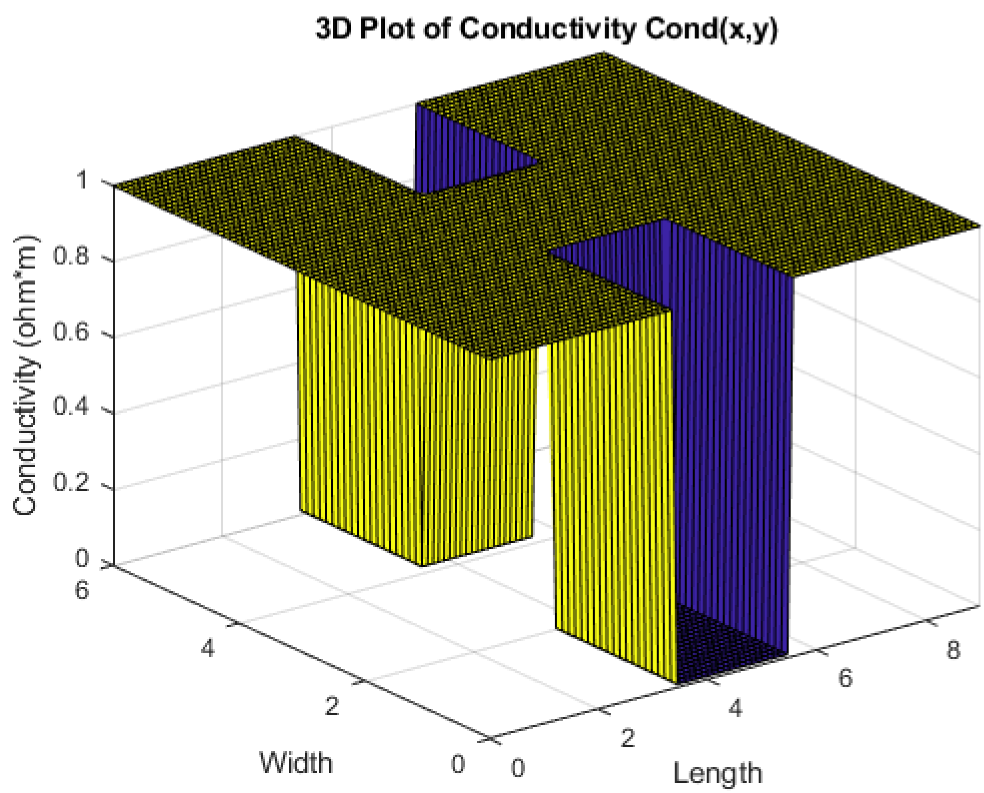
```
Jx = cond.*(-Ex);
Jy = cond.*(-Ey);


quiver(X,Y,Jx,Jy)
xlim([0 L])
ylim([0 W])
title("3D Plot of Current Density J(x,y)")
xlabel("Length")
ylabel("Width")
zlabel("Current Density (A/Area)")
```

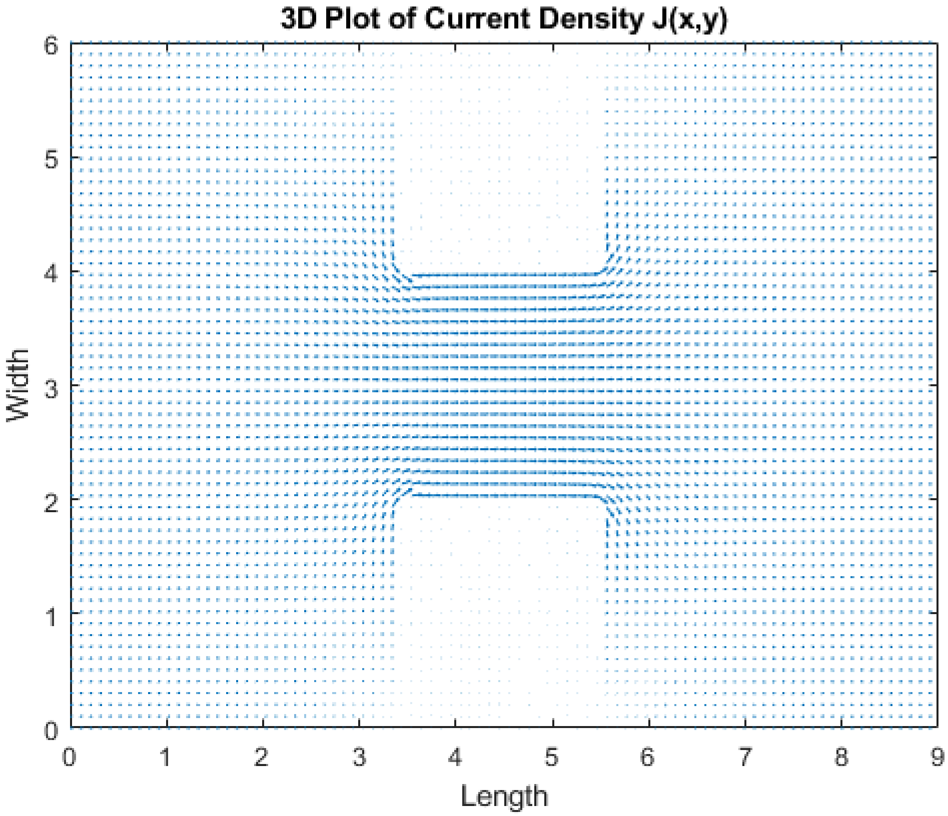



3D Plot of Electric Field E(x,y)
Width
Length
0
1
2
3
4
5
6
7
8
9


3D Plot of Electric Field E(x)
Width
Length
0
1
2
3
4
5
6
7
8
9


3D Plot of Electric Field E(y)
Width
Length
0
1
2
3
4
5
6
7
8
9


3D Plot of Conductivity Cond(x,y)
Conductivity (ohm*m)
0
0.2
0.4
0.6
0.8
1
Width
Length
2
4
6
8



ELEC4700 Assignment 2 Part 2(c)

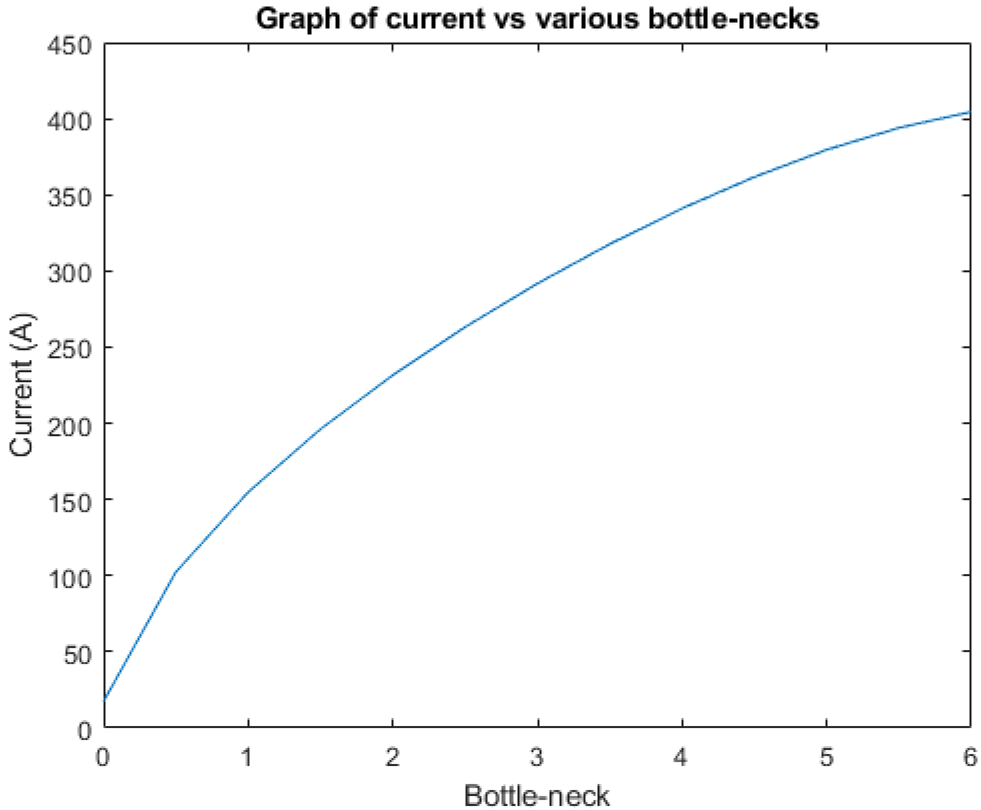
```
VoL = 100; % Left side of the Area has Boundary Voltage = VoL
VoR = 0; % Right side of the Area has Boundary Voltage = VoR
VoT = 0; % Top side of the Area has Boundary Voltage = VoT
VoB = 0; % Bottom side of the Area has Boundary Voltage = VoB
Pixel = 10; % Number of mesh per unit length/width

Lb = 2; % Box Length
Wb = 2; % Box Width
xBox = Lb*Pixel; % Box length Pixel Number
yBox = Wb*Pixel; % Box width Pixel Number
L = 9; % Area Length
W = 6; % Area Width
nx = L*Pixel; % Area length Pixel Number
ny = W*Pixel; % Area wdith Pixel Number
sigma = 1; % Outside Box Area Conductivity
BoxSigma = 0.01; % Inside Box Area Conductivity
G = sparse(nx*ny); % G matrix has size(nx*ny,nx*ny)
B = zeros(nx*ny,1); % B is the product of G matrix * V
Conductivity = sigma*ones(nx,ny); % Conductivity of the entire area

BottleNeckArray = W-[0:0.5:W];
current = zeros(length(BottleNeckArray),1);
```

(c) Graph or table of current vs various bottle-necks

```
for k = 1:length(BottleNeckArray)
    %BottleNeckGrid = BottleNeckArray(1,k)*Pixel;
    Wb = (W-BottleNeckArray(1,k))/2;
    yBox = Wb*Pixel;
    %From Before

    % Set Inside Box area conductivity to variable BoxSigma
    for iRow = 1:nx
        for jColumn = 1:ny
            if iRow>=(nx-xBox)/2 && iRow<=((nx+xBox)/2) && jColumn<=yBox
                Conductivity(iRow,jColumn) = BoxSigma;
            elseif iRow>=(nx-xBox)/2 && iRow<=((nx+xBox)/2) && jColumn<=ny &&
 jColumn>ny-yBox
                Conductivity(iRow,jColumn) = BoxSigma;
            end
        end
    end

    % for jColumn = 1:ny
    %     for iRow = 1:nx
    for iRow = 1:nx
        for jColumn = 1:ny
            n = jColumn+(iRow-1)*ny;
            % Left side Boundary Condition

            if iRow == 1
                G(n,:) = 0;
                G(n,n) = 1;
                B(n) = VoL;
            % Right side Boundary Condition

            elseif iRow == nx
                G(n,:) = 0;
                G(n,n) = 1;
                B(n) = VoR;

            % Bottom side Boundary Condition
            elseif jColumn == 1
                nxm = jColumn+((iRow-1)-1)*ny;
                nxp = jColumn+((iRow+1)-1)*ny;
                nyp = (jColumn+1)+(iRow-1)*ny;

                rxm = (Conductivity(iRow,jColumn) +
 Conductivity(iRow-1,jColumn))/2;
                rxp = (Conductivity(iRow,jColumn) + Conductivity(iRow
+1,jColumn))/2;
                ryp = (Conductivity(iRow,jColumn) + Conductivity(iRow,jColumn
+1))/2;
```

```
                G(n,n) = -(rxm+rxp+ryp);
                G(n,nxm) = rxm;
                G(n,nxp) = rxp;
                G(n,nyp) = ryp;

            % Top side Boundary Condition
            elseif jColumn == ny
                nxm = jColumn+((iRow-1)-1)*ny;
                nxp = jColumn+((iRow+1)-1)*ny;
                nym = (jColumn-1)+(iRow-1)*ny;

                rxm = (Conductivity(iRow,jColumn) +
 Conductivity(iRow-1,jColumn))/2;
                rxp = (Conductivity(iRow,jColumn) + Conductivity(iRow
+1,jColumn))/2;
                rym = (Conductivity(iRow,jColumn) +
 Conductivity(iRow,jColumn-1))/2;

                G(n,n) = -(rxm+rxp+rym);
                G(n,nxm) = rxm;
                G(n,nxp) = rxp;
                G(n,nym) = rym;

            else
                nxm = jColumn+((iRow-1)-1)*ny;
                nxp = jColumn+((iRow+1)-1)*ny;
                nym = (jColumn-1)+(iRow-1)*ny;
                nyp = (jColumn+1)+(iRow-1)*ny;

                rxm = (Conductivity(iRow,jColumn) +
 Conductivity(iRow-1,jColumn))/2;
                rxp = (Conductivity(iRow,jColumn) + Conductivity(iRow
+1,jColumn))/2;
                rym = (Conductivity(iRow,jColumn) +
 Conductivity(iRow,jColumn-1))/2;
                ryp = (Conductivity(iRow,jColumn) + Conductivity(iRow,jColumn
+1))/2;

                G(n,n) = -(rxm+rxp+rym+ryp);
                G(n,nxm) = rxm;
                G(n,nxp) = rxp;
                G(n,nym) = rym;
                G(n,nyp) = ryp;
            end

        end

    end



    Vn = G\B; % Find (ny*ny:1) size, V = G\B
    % Mapping the V to the matrix size of nx*ny
```

```
    for iRow = 1: nx
        for jColumn = 1:ny
             n = jColumn+(iRow-1)*ny;
             V(iRow,jColumn) = Vn(n);
        end
    end
    [Ex,Ey] = gradient(V');
    cond = Conductivity';
    Jx = cond.*(-Ex);
    %J = Conductivity'.*gradient(-(V'));

    a = sum(abs(Jx(:,45))).*W;
    a1 = sum(abs(Jx(:,41))).*W;
    a2 = sum(abs(Jx(:,21))).*W;
    J = Conductivity'.*gradient(-(V'));

    a = sum(abs(Jx(:,45))).*W;


%     current(k,1) = max(J,[],'all');
    current(k,1) = a;


end
%surf(V);
figure('name','Graph of current vs various bottle-necks')
plot(BottleNeckArray,current)
title('Graph of current vs various bottle-necks')
xlabel('Bottle-neck');
ylabel('Current (A)');
```



Published with MATLAB® R2021b