Elec 4700 Assignment 2

Table of Contents

Finite Difference Method for Solving Laplace

Keegan Mauger 101042551

Section 1: Electrostatic Potential in a Rectangular Region

Using Laplace's equation by the finite difference method, an electrostatic potential problem was to be solved. The problem was modelled by an orthogonal resistor network in a region of W by L, choosen to be 90 by 60 for this problem. By using a mesh of resistors, boundary conditions and intrusions became easier to model.

Using the matrix form GV=F, the electrostatic potential in the rectangular region was solved using $del^2 * V = 0$.

Note that, for unknown reasons, the publishing tool does not function as intended with MATLAB 2021b. I ran this code using MATLAB 2020a, and after discussion with Aaron, decided that this was the best method to publish the report. In 2020a, the publisher attaches the figures to the end of the current section, so in order to achieve the desired results, I was forced to break text into sections to compensate for this problem. This had the unfortunate side effect of making the formatting worse than I would have intended.

Section 1a: $V = V_0 @ x = 0$ and $V = 0 @ x = L$

For this problem, the boundary conditions were set such that $V = V_0 = 1$ at the left region boundary, and $V = 0$ at the right region boundary. The y-axis boundary conditions were not set. This case was then solved, with the code and results shown below. Note that the region dimensions are taken to be unitless.

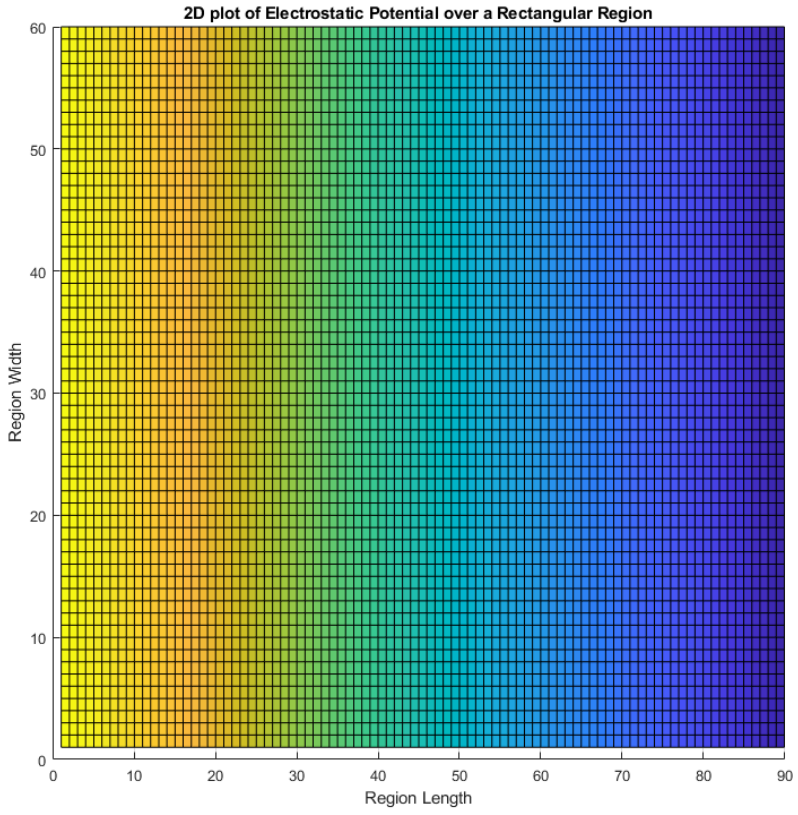
```
clear all
close all
clc
set(0,'DefaultFigureWindowStyle','docked');
```

```
% Solving V=V0 @ x=0 and V=0 @ x=L in region LxW
% Implement funtion 'pbaspect' to fix Z aspect ratio

L = 90;
W = 2/3 * L;
V0 = 1;

fMesh = 1;                  % Mesh factor
nx = fMesh*L;
ny = fMesh*W;
G = sparse(nx*ny);
%V = sparse(nx,ny);
F = sparse(1,nx*ny);

La = linspace(0,L,nx);
Wa = linspace(0,W,ny);

for i = 1:nx                %Iteration through length
    for j = 1:ny            %Iteration through width
        n = j + (i-1)*ny;

        if i == 1          % x=0 BCs
            G(n,:) = 0;
            G(n,n) = 1;
            F(n) = 1;
        elseif i == nx     % x=1 BCs
            G(n,:) = 0;
            G(n,n) = 1;
            F(n) = 0;       %F(n)=0 sets z at final width to 0

        else
            nxm = j + (i-2)*ny;
            nxp = j + (i)*ny;
            nym = j-1 + (i-1)*ny;
            nyp = j+1 + (i-1)*ny;

            G(n,n) = -(4);
            G(n,nxm) = 1;
            G(n,nxp) = 1;
            G(n,nym) = 1;
            G(n,nyp) = 1;
        end

    end
end
% figure(1)
% spy(G)

V = G\F';

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

```
            Vmap(i,j) = V(n);
        end
    end

figure(1)
surf(Vmap)
pbaspect([1 1 0.5])
view(90,270)
title('2D plot of Electrostatic Potential over a Rectangular Region')
xlabel('Region Width')    %Axis swapped due to view change, named
 correctly
ylabel('Region Length')
zlabel('Voltage (V)')
saveas(gcf,'Figure1')
```

Results

On the plot, the colourmap represents the voltage of the region. As expected, the voltage begins in the region at 1V and linearly decreases over the length to zero volts.

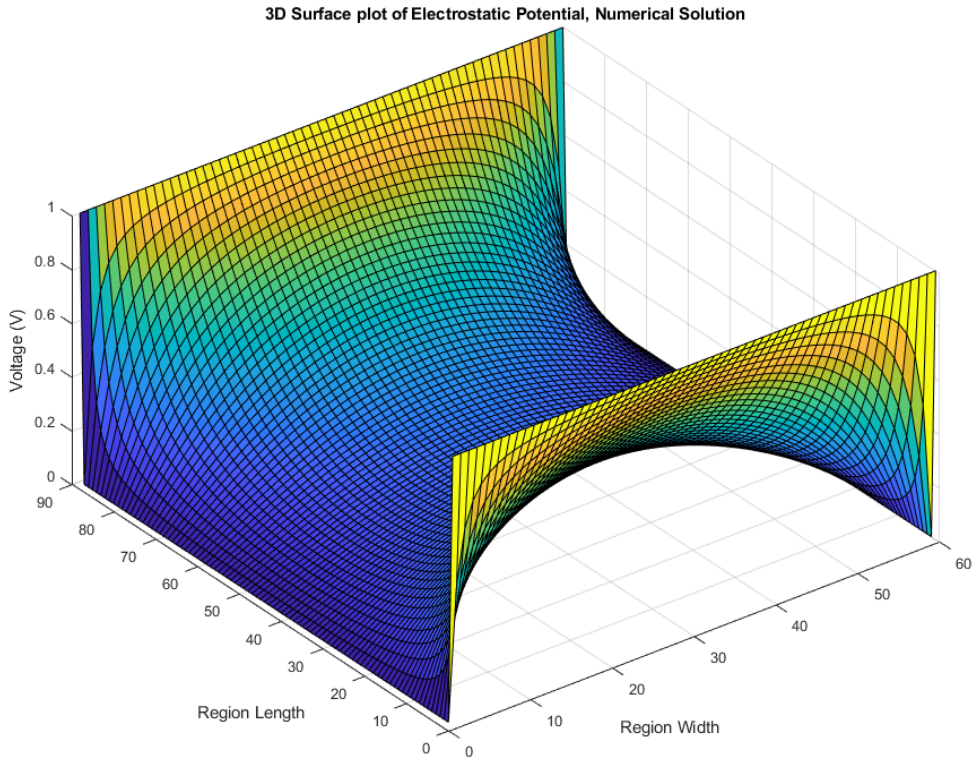


Section 1b: $V = V_0 @ x = 0, L$ and $V = 0 @ y = 0, W$

The problem was solved using both numerical and analytical approaches. First, the problem was solved using the numerical method, shown below.

```
clear all

clc

% Solving V=V0 @ x=0 and V=V0 @ x=L in region LxW
% Implement funtion 'pbaspect' to fix Z aspect ratio

L = 90;
W = 2/3 * L;
V0 = 1;

fMesh = 1;                  % Mesh factor
nx = fMesh*L;
ny = fMesh*W;
G = sparse(nx*ny);
%V = sparse(nx,ny);
F = sparse(1,nx*ny);

La = linspace(0,L,nx);
Wa = linspace(0,W,ny);

for i = 1:nx                %Iteration through length
    for j = 1:ny            %Iteration through width
        n = j + (i-1)*ny;

        if i == 1          % x=0 BCs
            G(n,:) = 0;
            G(n,n) = 1;
            F(n) = 1;
        elseif i == nx     % x=1 BCs
            G(n,:) = 0;
            G(n,n) = 1;
            %F(n) = 0;       %F(n)=0 sets z at final width to 0

% COMMENT BELOW FOR 1a
            F(n) = 1;       %F(n)=1 sets z at final width to 1

        elseif j == 1                % y=0 BCs
            nxm = j + (i-2)*ny;
            nxp = j + (i)*ny;
            nyp = j+1 + (i-1)*ny;

            G(n,n) = 1;
        elseif j == ny               % y=1 BCs
            nxm = j + (i-2)*ny;
            nxp = j + (i)*ny;
            nym = j-1 + (i-1)*ny;

            G(n,n) = 1;


% COMMENT ABOVE FOR 1a

        else
```

```
            nxm = j + (i-2)*ny;
            nxp = j + (i)*ny;
            nym = j-1 + (i-1)*ny;
            nyp = j+1 + (i-1)*ny;

            G(n,n) = -(4);
            G(n,nxm) = 1;
            G(n,nxp) = 1;
            G(n,nym) = 1;
            G(n,nyp) = 1;
        end

    end
end

V = G\F';

Vmap = zeros(nx,ny);
for i = 1:nx
    for j = 1:ny
        n = j + (i-1)*ny;
        Vmap(i,j) = V(n);
    end
end

figure(2)
surf(Vmap)
pbaspect([1 1 0.5])
title('3D Surface plot of Electrostatic Potential, Numerical
 Solution')
xlabel('Region Width')
ylabel('Region Length')
zlabel('Voltage (V)')
saveas(gcf,'Figure2')
```



Discussion

The results above are for a region with a mesh size equal to the length and width, meaning that each increase by 1 in the position axis corresponds to 1 calculation. Increasing or decreasing the mesh size results in a different resolution of equations, which in turn results in a different solution, as discussed below.

Next, the problem was again solved, using an analytical approach. The analytical solution results and discussion are displayed below.

```
clear all
clc

% Solving V=V0 @ x=0 and V=0 @ x=L in region LxW
% Implement funtion 'pbaspect' to fix Z aspect ratio

L = 90;
W = 2/3 * L;
V0 = 1;

nx = 100;
ny = 100;

La = linspace(-L/2,L/2,nx);
Wa = linspace(0, W, ny);
```

```
V = zeros(nx,ny);
Vs = zeros(nx,ny);
n = 1;
f = 0;
figure(3)

for n=1:2:210
    for i=1:nx
        for j = 1:ny
            V(i,j) = (4*V0/pi)*...
            ((1/n)*(cosh(n*pi*La(i)/(W))/cosh(n*pi*(L/2)/(W)))*...
            sin(n*pi*Wa(j)/(W)));
        end
    end
    Vs = Vs + V;
    surf(Wa,La,Vs)
    pbaspect([1 1 0.5])
    pause(0.01)
end
title('3D Surface plot of Electrostatic Potential, Analytical
 Solution')
xlabel('Region Width')
ylabel('Region Length')
zlabel('Voltage (V)')
saveas(gcf,'Figure3')
```

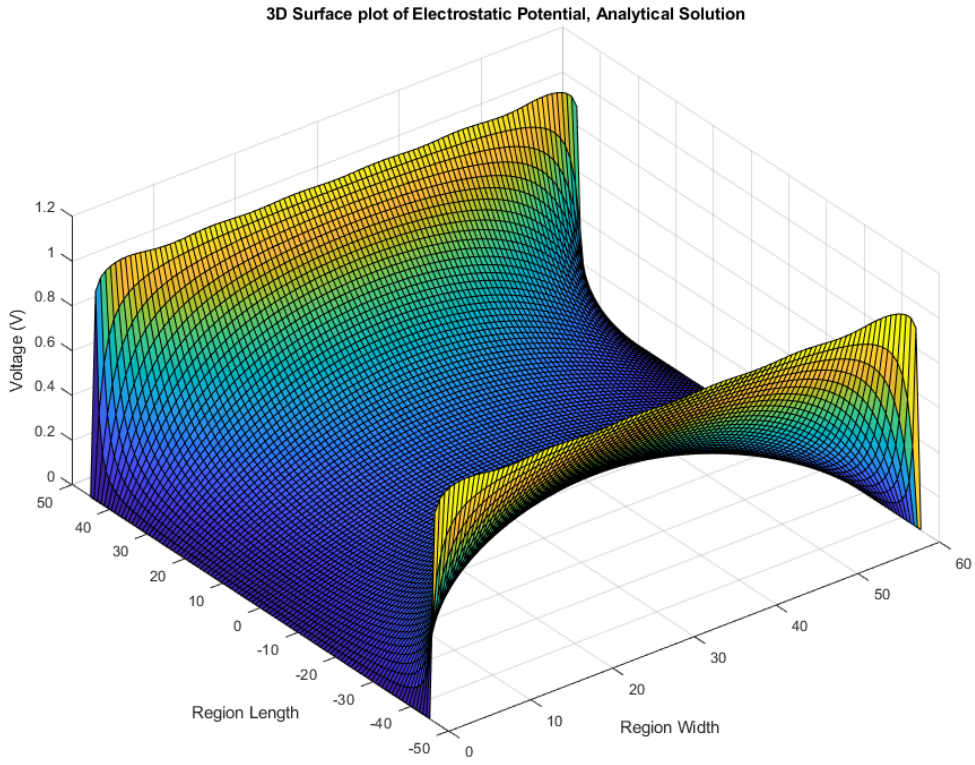
Results

Seen in the analytical plot, the results are similar, but differences do occur at the boundary conditions where the analytical solution produces a "wavy" peak. This similarity is due to the numerical solution having a suffcicent resolution. The optimal point to stop the analytical solution would be based on the difference in errors of the current iteration and the previous iteration, where the error becomes smaller with each successive iteration of the formula. This can then be compared to a user defined accuracy, where when the analytical error becomes smaller than the user defined value, the solution would stop.

As shown in "test.m" provided with this report, if the resolution is decreased the solution becomes increasing distorted, and likely becomes irrelevant. A higher resolution, appropriately, results in a better solution, however a balance occurs between processing time of larger resolutions and increasingly small differences in solution quality.

A major difference between analytical and numerical methods is in computer resourses. The analytical solution, due to being a sum of a potentially complex function, can iterate to a large degree which results in increased computation time. Comparitively, the analytical approach, making use of sparse martricies, is a more efficient use of resources and time.

Additionally, a numerical method can find a solution to a problem for which no analytical solution exists, while an analytical method is exact and thus may have a smaller degree of error.



Section 2a: Current Flow in Inhomogeneous Solids

In section 2, the finite difference method was used to calculate the current and voltage over a rectangle region with varying conductivity. The overall region has a conductivity of 1 seimen, with two interior box regions of 1e-2 seimens. The purpose of this experiemnt was to view how the bottlenecks affected the current flow in the region.

The code below makes use of a conductivity map over the region, used to create the bottleneck. This conductivity map, with the previous modelling of resistors, allows a mesh of known resistor values to be found, and used to calculate the current. The results shown are plots of the conductivity, voltage, electric field, and current density. The regions of lesser conductivity are boxes, 20 by 20 units in area.

```
clear all
clc
set(0,'DefaultFigureWindowStyle','docked');

% Solving V=V0 @ x=0 and V=0 @ x=L in region LxW
% Implement funtion 'pbaspect' to fix Z aspect ratio

L = 90;
W = 2/3 * L;
V0 = 1;
```

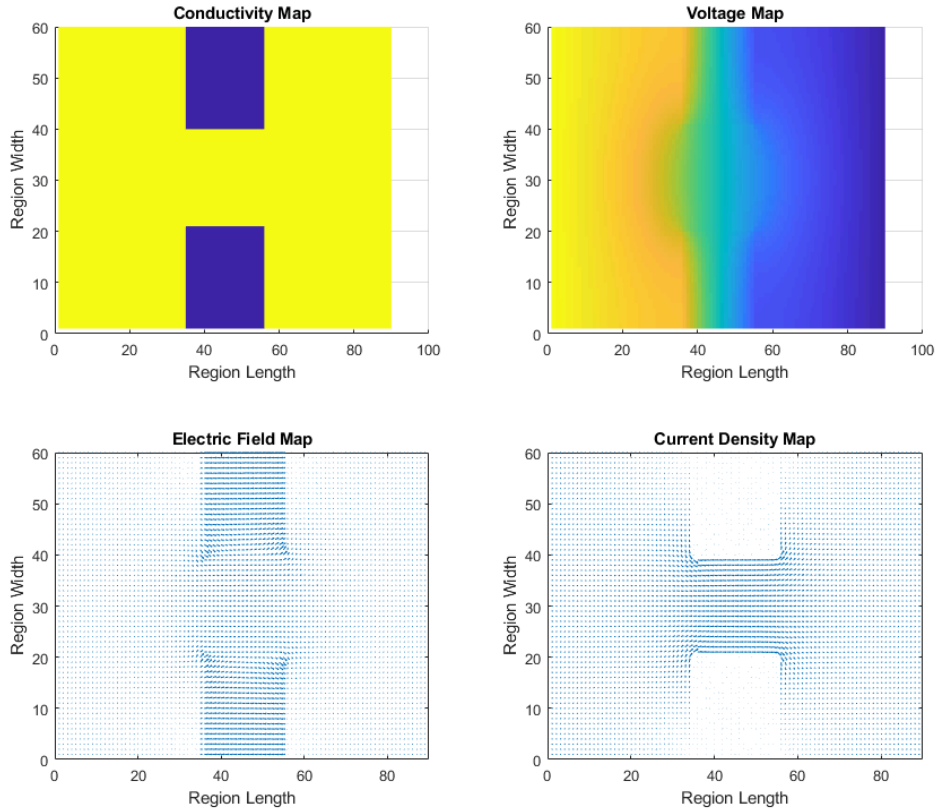
```
nx = L;
ny = W;
G = sparse(nx*ny);
%V = sparse(nx,ny);
F = sparse(1,nx*ny);


Acond = 1;              % background conductivity of region, low
 resistance
Bcond = 1e-2;           % Conductivity of boxes, highly resistive
cMap = zeros(nx,ny);
Lb = 20;
Wb = 20;
for u = 1:nx
    for v = 1:ny
        if (u >= 35 && u <= 55)
            if v >= 0 && v <= 20
                cMap(u,v) = Bcond;
            elseif v >= 40 && v <= 60
                cMap(u,v) = Bcond;
            else
                cMap(u,v) = Acond;
            end
        else
            cMap(u,v) = Acond;
        end
    end
end




for i = 1:nx                %Iteration through length
    for j = 1:ny            %Iteration through width
        n = j + (i-1)*ny;

        if i == 1          % x=0 BCs
            G(n,:) = 0;
            G(n,n) = 1;
            F(n) = 1;
        elseif i == nx     % x=1 BCs
            G(n,:) = 0;
            G(n,n) = 1;
            F(n) = 0;       %F(n)=0 sets z at final width to 0

% COMMENT BELOW FOR 1a
            %F(n) = 1;       %F(n)=1 sets z at final width to 1

        elseif j == 1                  % y=0 BCs
            nxm = j + (i-2)*ny;
            nxp = j + (i)*ny;
            nyp = j+1 + (i-1)*ny;
```

```
            rxm = (cMap(i,j) + cMap(i-1,j))/2;
            rxp = (cMap(i,j) + cMap(i+1,j))/2;
            ryp = (cMap(i,j) + cMap(i,j+1))/2;

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

        elseif j == ny                % y=1 BCs
            nxm = j + (i-2)*ny;
            nxp = j + (i)*ny;
            nym = j-1 + (i-1)*ny;

            rxm = (cMap(i,j) + cMap(i-1,j))/2;
            rxp = (cMap(i,j) + cMap(i+1,j))/2;
            rym = (cMap(i,j) + cMap(i,j-1))/2;

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


% COMMENT ABOVE FOR 1a

        else
            nxm = j + (i-2)*ny;
            nxp = j + (i)*ny;
            nym = j-1 + (i-1)*ny;
            nyp = j+1 + (i-1)*ny;

            rxm = (cMap(i,j) + cMap(i-1,j))/2;
            rxp = (cMap(i,j) + cMap(i+1,j))/2;
            rym = (cMap(i,j) + cMap(i,j-1))/2;
            ryp = (cMap(i,j) + cMap(i,j+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
% figure(1)
% spy(G)

V = G\F';

Vmap = zeros(nx,ny);
for i = 1:nx
```

```
    for j = 1:ny
        n = j + (i-1)*ny;
        Vmap(i,j) = V(n);
    end
end


for i = 1:nx
    for j = 1:ny
        if i == 1
            Ex(i, j) = (Vmap(i + 1, j) - Vmap(i, j));
        elseif i == nx
            Ex(i, j) = (Vmap(i, j) - Vmap(i - 1, j));
        else
            Ex(i, j) = (Vmap(i + 1, j) - Vmap(i - 1, j)) * 0.5;
        end
        if j == 1
            Ey(i, j) = (Vmap(i, j + 1) - Vmap(i, j));
        elseif j == ny
            Ey(i, j) = (Vmap(i, j) - Vmap(i, j - 1));
        else
            Ey(i, j) = (Vmap(i, j + 1) - Vmap(i, j - 1)) * 0.5;
        end
    end
end

Ex = -Ex;
Ey = -Ey;

eFlowx = cMap .* Ex;        %Jx
eFlowy = cMap .* Ey;        %Jy

figure(4)
subplot(2, 2, 1), H = surf(cMap');
set(H, 'linestyle', 'none');
view(0, 90)
title('Conductivity Map')
xlabel('Region Length')
ylabel('Region Width')
subplot(2, 2, 2), H = surf(Vmap);
set(H, 'linestyle', 'none');
view(90, 270)
title('Voltage Map')
xlabel('Region Width')
ylabel('Region Length')
subplot(2, 2, 3), quiver(Ex', Ey');
axis([0 nx 0 ny]);
title('Electric Field Map')
xlabel('Region Length')
ylabel('Region Width')
subplot(2, 2, 4), quiver(eFlowx', eFlowy');
axis([0 nx 0 ny]);
title('Current Density Map')
xlabel('Region Length')
```

```
ylabel('Region Width')
saveas(gcf,'Figure4')
```

```
C0 = sum(eFlowx(1, :));
Cnx = sum(eFlowx(nx, :));
Curr = (C0 + Cnx) * 0.5;
```



Section 2b: Current vs Mesh Size

By applying a mesh factor multiplier in the form of an array to the resolution nx and ny, the resolution was linearly scaled to observe the change in current with varying mesh size. The results are shown below, where it is seen that at small resolutions the program becomes inacurate, having difficulty applying the finite difference method over a low resolution. At larger resolutions, the finite difference method converges towards a solution, at the cost of greater computational resourse use.

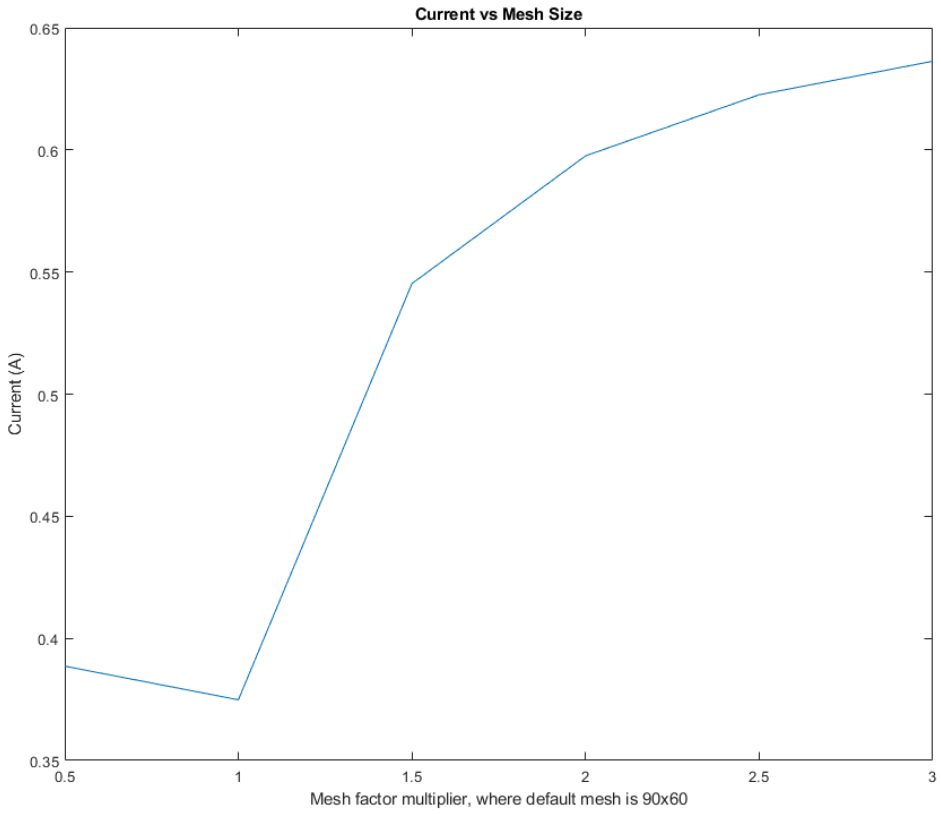
```
clear all
clc
```

```
% Solving V=V0 @ x=0 and V=0 @ x=L in region LxW
% Implement funtion 'pbaspect' to fix Z aspect ratio
```

```
L = 90;
W = 2/3 * L;
V0 = 1;

fMesh = 0.5:0.5:3;
for k = 1:length(fMesh)

    nx = round(fMesh(k)*L);
    ny = round(fMesh(k)*W);
    G = sparse(nx*ny);
    %V = sparse(nx,ny);
    F = sparse(1,nx*ny);


    Acond = 1;              % background conductivity of region, low
 resistance
    Bcond = 1e-2;           % Conductivity of boxes, highly resistive
    BN = 0:1:10;                % Changing bottleneck
    cMap = zeros(nx,ny);
    Lb = 20;
    Wb = 20;

    for u = 1:nx
        for v = 1:ny
            if (u >= 35 && u <= 55)
                if v >= 0 && v <= 20
                    cMap(u,v) = Bcond;
                elseif v >= 40 && v <= 60
                    cMap(u,v) = Bcond;
                else
                    cMap(u,v) = Acond;
                end
            else
                cMap(u,v) = Acond;
            end
        end
    end




    for i = 1:nx                %Iteration through length
        for j = 1:ny            %Iteration through width
            n = j + (i-1)*ny;

            if i == 1          % x=0 BCs
                G(n,:) = 0;
                G(n,n) = 1;
                F(n) = 1;
            elseif i == nx     % x=1 BCs
                G(n,:) = 0;
                G(n,n) = 1;
                F(n) = 0;       %F(n)=0 sets z at final width to 0
```

```
            % COMMENT BELOW FOR 1a
            %F(n) = 1;       %F(n)=1 sets z at final width to 1

        elseif j == 1                   % y=0 BCs
            nxm = j + (i-2)*ny;
            nxp = j + (i)*ny;
            nyp = j+1 + (i-1)*ny;

            rxm = (cMap(i,j) + cMap(i-1,j))/2;
            rxp = (cMap(i,j) + cMap(i+1,j))/2;
            ryp = (cMap(i,j) + cMap(i,j+1))/2;

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

        elseif j == ny                  % y=1 BCs
            nxm = j + (i-2)*ny;
            nxp = j + (i)*ny;
            nym = j-1 + (i-1)*ny;

            rxm = (cMap(i,j) + cMap(i-1,j))/2;
            rxp = (cMap(i,j) + cMap(i+1,j))/2;
            rym = (cMap(i,j) + cMap(i,j-1))/2;

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


            % COMMENT ABOVE FOR 1a

        else
            nxm = j + (i-2)*ny;
            nxp = j + (i)*ny;
            nym = j-1 + (i-1)*ny;
            nyp = j+1 + (i-1)*ny;

            rxm = (cMap(i,j) + cMap(i-1,j))/2;
            rxp = (cMap(i,j) + cMap(i+1,j))/2;
            rym = (cMap(i,j) + cMap(i,j-1))/2;
            ryp = (cMap(i,j) + cMap(i,j+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
    % figure(1)
    % spy(G)

    V = G\F';

    Vmap = zeros(nx,ny);
    for i = 1:nx
        for j = 1:ny
            n = j + (i-1)*ny;
            Vmap(i,j) = V(n);
        end
    end


    for i = 1:nx
        for j = 1:ny
            if i == 1
                Ex(i, j) = (Vmap(i + 1, j) - Vmap(i, j));
            elseif i == nx
                Ex(i, j) = (Vmap(i, j) - Vmap(i - 1, j));
            else
                Ex(i, j) = (Vmap(i + 1, j) - Vmap(i - 1, j)) * 0.5;
            end
            if j == 1
                Ey(i, j) = (Vmap(i, j + 1) - Vmap(i, j));
            elseif j == ny
                Ey(i, j) = (Vmap(i, j) - Vmap(i, j - 1));
            else
                Ey(i, j) = (Vmap(i, j + 1) - Vmap(i, j - 1)) * 0.5;
            end
        end
    end

    Ex = -Ex;
    Ey = -Ey;

    eFlowx = cMap .* Ex;        %Jx
    eFlowy = cMap .* Ey;        %Jy


    C0 = sum(eFlowx(1, :));
    Cnx = sum(eFlowx(nx, :));
    Curr(k) = (C0 + Cnx) * 0.5;
end

figure(5)
plot(fMesh,Curr)
xlabel('Mesh factor multiplier, where default mesh is 90x60')
ylabel('Current (A)')
title('Current vs Mesh Size')
saveas(gcf,'Figure5')
```



Section 2c: Current vs Bottleneck Size

Next, the heights of the bottlenecks were altered to view the effects that a narrower gap would have on the current. This was done by adding a factor BN to the interior-most boundary of both boxes. As the default boxes are 20 units in width, what was plotted for BN is the additional width added to the original dimensions. From the results, it was seen that current is reduced with a narrower bottleneck.

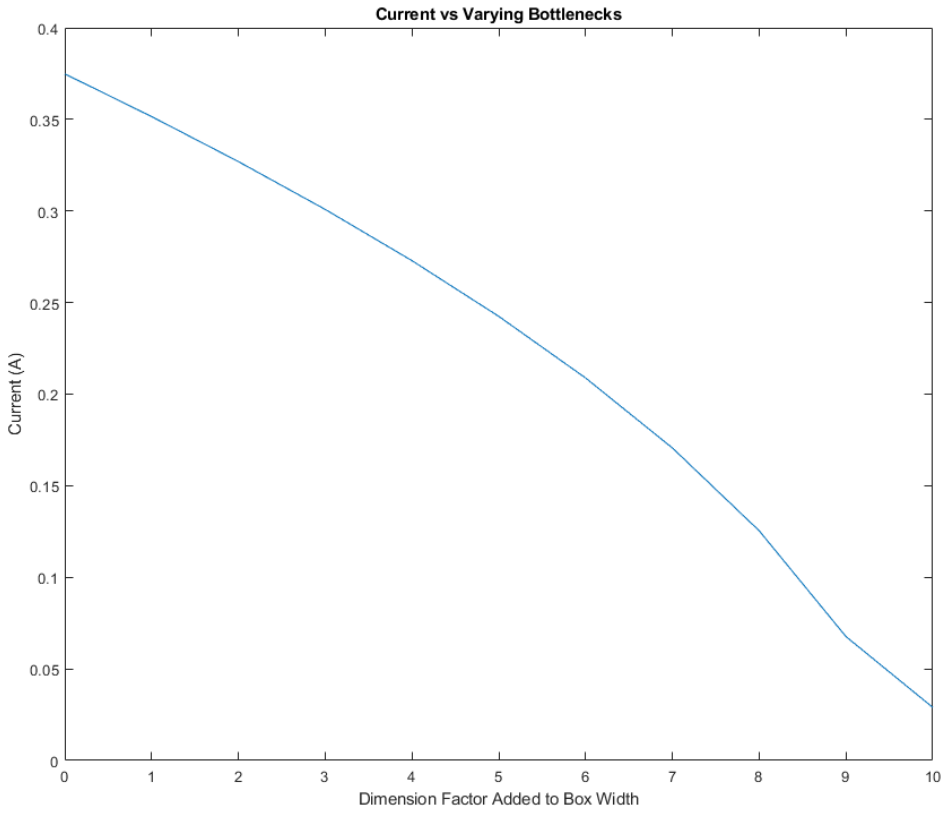
For example, with a BN of 1: the Wb of each box is increased by 1. So for every increase of BN, the gap between boxes is decresed by $2 * BN$.

```
clear all
clc


% Solving V=V0 @ x=0 and V=0 @ x=L in region LxW
% Implement funtion 'pbaspect' to fix Z aspect ratio

L = 90;
W = 2/3 * L;
V0 = 1;

nx = L;
ny = W;
G = sparse(nx*ny);
```

```
%V = sparse(nx,ny);
F = sparse(1,nx*ny);


Acond = 1;              % background conductivity of region, low
 resistance
Bcond = 1e-2;           % Conductivity of boxes, highly resistive
BN = 0:1:10;                % Changing bottleneck
cMap = zeros(nx,ny);
Lb = 20;
Wb = 20;
for k = 1:length(BN)
    for u = 1:nx
        for v = 1:ny
            if (u >= 35 && u <= 55)
                if v >= 0 && v <= (20+BN(k))
                    cMap(u,v) = Bcond;
                elseif v >= (40-BN(k)) && v <= 60
                    cMap(u,v) = Bcond;
                else
                    cMap(u,v) = Acond;
                end
            else
                cMap(u,v) = Acond;
            end
        end
    end




    for i = 1:nx                %Iteration through length
        for j = 1:ny            %Iteration through width
            n = j + (i-1)*ny;

            if i == 1          % x=0 BCs
                G(n,:) = 0;
                G(n,n) = 1;
                F(n) = 1;
            elseif i == nx     % x=1 BCs
                G(n,:) = 0;
                G(n,n) = 1;
                F(n) = 0;       %F(n)=0 sets z at final width to 0

                % COMMENT BELOW FOR 1a
                %F(n) = 1;       %F(n)=1 sets z at final width to 1

            elseif j == 1                   % y=0 BCs
                nxm = j + (i-2)*ny;
                nxp = j + (i)*ny;
                nyp = j+1 + (i-1)*ny;

                rxm = (cMap(i,j) + cMap(i-1,j))/2;
```

```
            rxp = (cMap(i,j) + cMap(i+1,j))/2;
            ryp = (cMap(i,j) + cMap(i,j+1))/2;

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

        elseif j == ny                % y=1 BCs
            nxm = j + (i-2)*ny;
            nxp = j + (i)*ny;
            nym = j-1 + (i-1)*ny;

            rxm = (cMap(i,j) + cMap(i-1,j))/2;
            rxp = (cMap(i,j) + cMap(i+1,j))/2;
            rym = (cMap(i,j) + cMap(i,j-1))/2;

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


            % COMMENT ABOVE FOR 1a

        else
            nxm = j + (i-2)*ny;
            nxp = j + (i)*ny;
            nym = j-1 + (i-1)*ny;
            nyp = j+1 + (i-1)*ny;

            rxm = (cMap(i,j) + cMap(i-1,j))/2;
            rxp = (cMap(i,j) + cMap(i+1,j))/2;
            rym = (cMap(i,j) + cMap(i,j-1))/2;
            ryp = (cMap(i,j) + cMap(i,j+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
% figure(1)
% spy(G)

V = G\F';

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

```
            Vmap(i,j) = V(n);
        end
    end


    for i = 1:nx
        for j = 1:ny
            if i == 1
                Ex(i, j) = (Vmap(i + 1, j) - Vmap(i, j));
            elseif i == nx
                Ex(i, j) = (Vmap(i, j) - Vmap(i - 1, j));
            else
                Ex(i, j) = (Vmap(i + 1, j) - Vmap(i - 1, j)) * 0.5;
            end
            if j == 1
                Ey(i, j) = (Vmap(i, j + 1) - Vmap(i, j));
            elseif j == ny
                Ey(i, j) = (Vmap(i, j) - Vmap(i, j - 1));
            else
                Ey(i, j) = (Vmap(i, j + 1) - Vmap(i, j - 1)) * 0.5;
            end
        end
    end

    Ex = -Ex;
    Ey = -Ey;

    eFlowx = cMap .* Ex;        %Jx
    eFlowy = cMap .* Ey;        %Jy


    C0 = sum(eFlowx(1, :));
    Cnx = sum(eFlowx(nx, :));
    Curr(k) = (C0 + Cnx) * 0.5;
end

figure(6)
plot(BN,Curr)
xlabel('Dimension Factor Added to Box Width')
ylabel('Current (A)')
title('Current vs Varying Bottlenecks')
saveas(gcf,'Figure6')
```



Section 2d: Current vs Conductivity

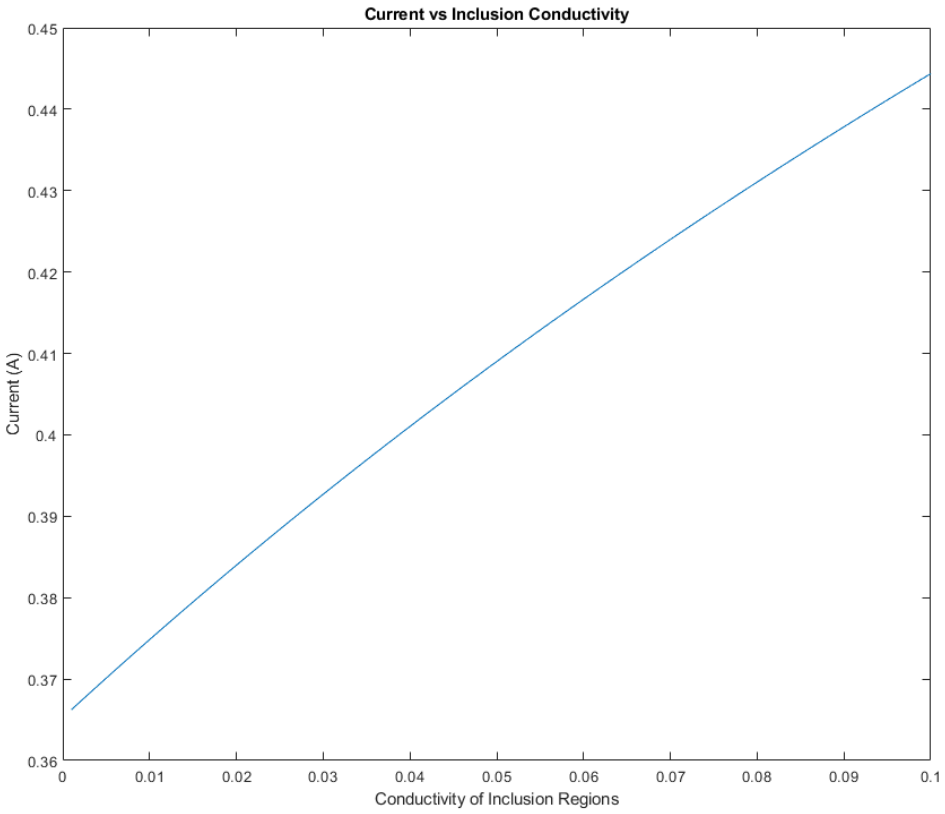
Finally, the conductivity of the box inclusions would be altered by ranging the values from 1e-3 to 1e3. This was then compared to the change in current. From the results, an increase in conductivity leads to a relatively linear increase in current, as was expected.

```
clear all
clc

% Solving V=V0 @ x=0 and V=0 @ x=L in region LxW
% Implement funtion 'pbaspect' to fix Z aspect ratio

L = 90;
W = 2/3 * L;
V0 = 1;

nx = L;
ny = W;
G = sparse(nx*ny);
%V = sparse(nx,ny);
F = sparse(1,nx*ny);
```

```
Acond = 1;              % background conductivity of region, low
 resistance
Bcond = 1e-3:1e-3:1e-1;           % Conductivity of boxes, highly
 resistive
cMap = zeros(nx,ny);
Lb = 20;
Wb = 20;
for k = 1:length(Bcond)
    for u = 1:nx
        for v = 1:ny
            if (u >= 35 && u <= 55)
                if v >= 0 && v <= 20
                    cMap(u,v) = Bcond(k);
                elseif v >= 40 && v <= 60
                    cMap(u,v) = Bcond(k);
                else
                    cMap(u,v) = Acond;
                end
            else
                cMap(u,v) = Acond;
            end
        end
    end




    for i = 1:nx                %Iteration through length
        for j = 1:ny            %Iteration through width
            n = j + (i-1)*ny;

            if i == 1          % x=0 BCs
                G(n,:) = 0;
                G(n,n) = 1;
                F(n) = 1;
            elseif i == nx     % x=1 BCs
                G(n,:) = 0;
                G(n,n) = 1;
                F(n) = 0;       %F(n)=0 sets z at final width to 0

                % COMMENT BELOW FOR 1a
                %F(n) = 1;       %F(n)=1 sets z at final width to 1

            elseif j == 1                % y=0 BCs
                nxm = j + (i-2)*ny;
                nxp = j + (i)*ny;
                nyp = j+1 + (i-1)*ny;

                rxm = (cMap(i,j) + cMap(i-1,j))/2;
                rxp = (cMap(i,j) + cMap(i+1,j))/2;
                ryp = (cMap(i,j) + cMap(i,j+1))/2;

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

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

        elseif j == ny                  % y=1 BCs
            nxm = j + (i-2)*ny;
            nxp = j + (i)*ny;
            nym = j-1 + (i-1)*ny;

            rxm = (cMap(i,j) + cMap(i-1,j))/2;
            rxp = (cMap(i,j) + cMap(i+1,j))/2;
            rym = (cMap(i,j) + cMap(i,j-1))/2;

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


            % COMMENT ABOVE FOR 1a

        else
            nxm = j + (i-2)*ny;
            nxp = j + (i)*ny;
            nym = j-1 + (i-1)*ny;
            nyp = j+1 + (i-1)*ny;

            rxm = (cMap(i,j) + cMap(i-1,j))/2;
            rxp = (cMap(i,j) + cMap(i+1,j))/2;
            rym = (cMap(i,j) + cMap(i,j-1))/2;
            ryp = (cMap(i,j) + cMap(i,j+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
% figure(1)
% spy(G)

V = G\F';

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

```
    for i = 1:nx
        for j = 1:ny
            if i == 1
                Ex(i, j) = (Vmap(i + 1, j) - Vmap(i, j));
            elseif i == nx
                Ex(i, j) = (Vmap(i, j) - Vmap(i - 1, j));
            else
                Ex(i, j) = (Vmap(i + 1, j) - Vmap(i - 1, j)) * 0.5;
            end
            if j == 1
                Ey(i, j) = (Vmap(i, j + 1) - Vmap(i, j));
            elseif j == ny
                Ey(i, j) = (Vmap(i, j) - Vmap(i, j - 1));
            else
                Ey(i, j) = (Vmap(i, j + 1) - Vmap(i, j - 1)) * 0.5;
            end
        end
    end

    Ex = -Ex;
    Ey = -Ey;

    eFlowx = cMap .* Ex;        %Jx
    eFlowy = cMap .* Ey;        %Jy

    C0 = sum(eFlowx(1, :));
    Cnx = sum(eFlowx(nx, :));
    Curr(k) = (C0 + Cnx) * 0.5;
end

figure(7)
plot(Bcond,Curr)
xlabel('Conductivity of Inclusion Regions')
ylabel('Current (A)')
title('Current vs Inclusion Conductivity')
saveas(gcf,'Figure7')
```



Published with MATLAB® R2020a