Part 1: Electrostatic Potential in Rectangular Region

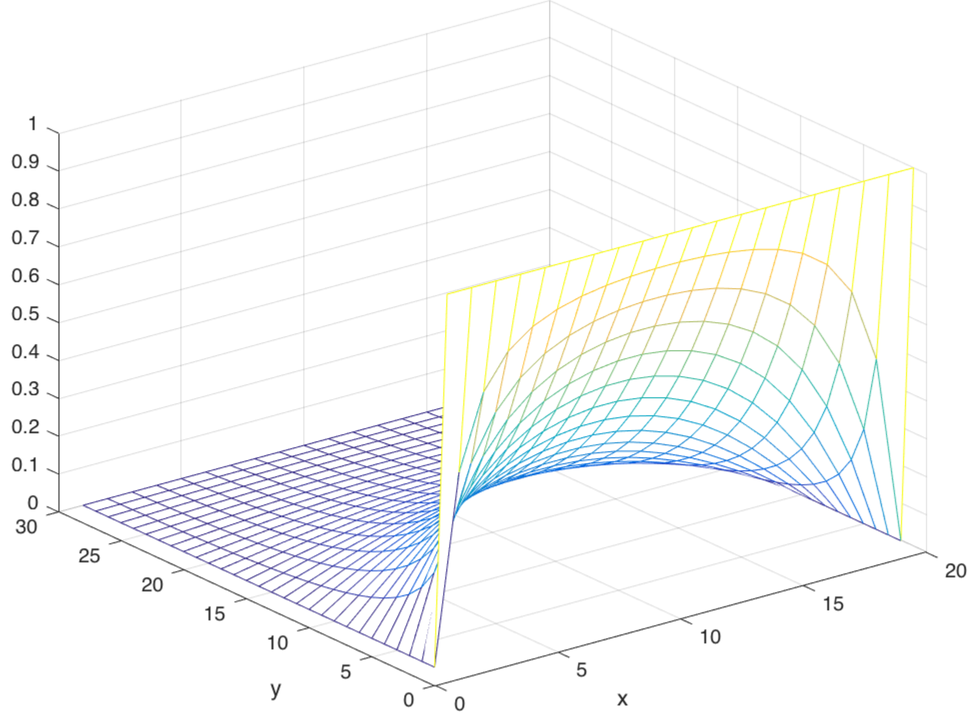
This section provides code that applies the finite difference method to solving for the electrostatic potential in the rectangular region defined by L and W.

```
clear all
close all

W = 20;
L = 30;

v0 = 1;
dx = 1;
dy = 1;
nx = L / dx;
ny = W / dy;

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

for i = 1:nx
    for j = 1:ny

        n = i + (j - 1) * nx;
        nym = i + (j - 2) * nx;
        nyp = i + j * nx;
        nxm = i - 1 + (j - 1) * nx;
        nxp = i + 1 + (j - 1) * nx;

        if i == 1
            G(n, n) = 1;
            F(n) = v0;
        elseif i == nx
            G(n, n) = 1;
            F(n) = 0;
        elseif j == 1
            G(n, n) = 1;
        elseif j == ny
            G(n, n) = 1;
        else
            G(n, n) = -4;
            G(n, nxm) = 1;
            G(n, nxp) = 1;
            G(n, nym) = 1;
            G(n, nyp) = 1;
            F(n) = 0;
        end
    end
end
```

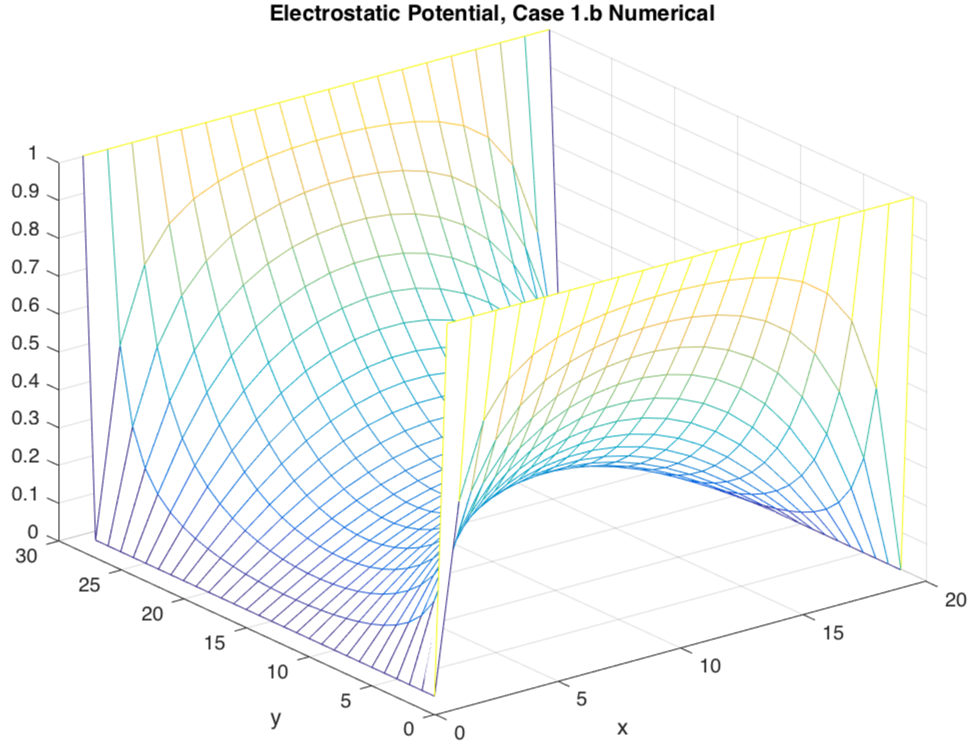
```
V2 = zeros(nx, ny);
V = G\F;
for i = 1:nx
    for j = 1:ny
        n = i + (j - 1)*nx;
        V2(i, j) = V(n);
    end
end
[X, Y] = meshgrid(1:1:ny, 1:1:nx);
figure(1)
mesh(X, Y, V2)
title('Electrostatic Potential, Case 1.a')
xlabel('x')
ylabel('y')




%Part 1 b)

for i = 1:nx
    for j = 1:ny

        n = i + (j - 1) * nx;
        nym = i + (j - 2) * nx;
        nyp = i + j * nx;
        nxm = i - 1 + (j - 1) * nx;
        nxp = i + 1 + (j - 1) * nx;

        if i == 1
            G(n, n) = 1;
            F(n) = v0;
        elseif i == nx
            G(n, n) = 1;
            F(n) = v0;
        elseif j == 1
            G(n, n) = 1;
        elseif j == ny
            G(n, n) = 1;
        else
            G(n, n) = -4;
            G(n, nxm) = 1;
            G(n, nxp) = 1;
            G(n, nym) = 1;
            G(n, nyp) = 1;
            F(n) = 0;
        end
    end
end

V2 = zeros(nx, ny);
V = G\F;
```

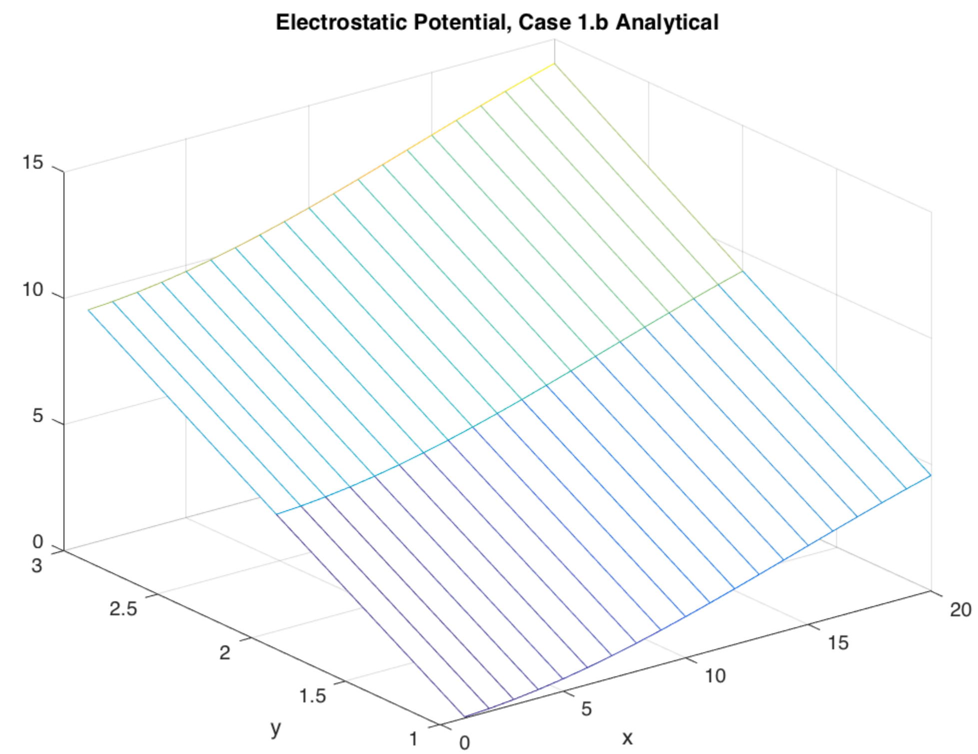
```
for i = 1:nx
    for j = 1:ny
        n = i + (j - 1)*nx;
        V2(i, j) = V(n);
    end
end

[X, Y] = meshgrid(1:1:ny, 1:1:nx);
figure(2)
mesh(X, Y, V2)
title('Electrostatic Potential, Case 1.b Numerical')
xlabel('x')
ylabel('y')




a = 30;
b = 20;
sum = 0;

for i = 1:nx
    for j = 1:ny
        for n = 1:2:1999
            sum = sum + 1/n * cosh(n*pi*i/a) * sin(n*pi*j/a) /
 cosh(n*pi*b/a);
        end
        V2(i, j) = 4/pi * sum;
    end
end

figure(3)
mesh(V2)
title('Electrostatic Potential, Case 1.b Analytical')
xlabel('x')
ylabel('y')
```

Electrostatic Potential, Case 1.a



Electrostatic Potential, Case 1.b Numerical





The analytical solution significantly differs from the numerical solution and seems to be very innaccurate. This error can possibly be accounted for in the number of iterations of the sum that were performed (represented by variable 'n' in the code). The analytical solution requires the sum to be iterated an infinite amount of times. Since that solution is impossible to realize, a very large number of iterations (just under 2000) was used. Therefore, for this problem under these specific conditions, the numerical solution is more accurate and shows less error.

Part 2: Current Flow in Rectangular Region

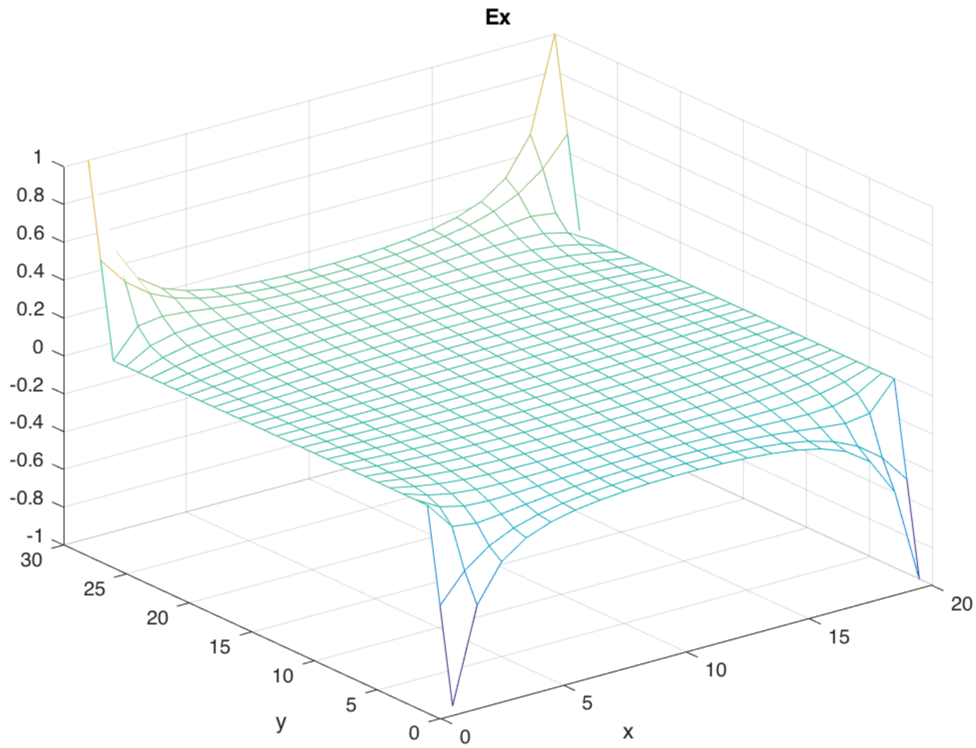
This section provides code that applies the finite difference method to solving for the current flow in the rectangular region defined by L and W.

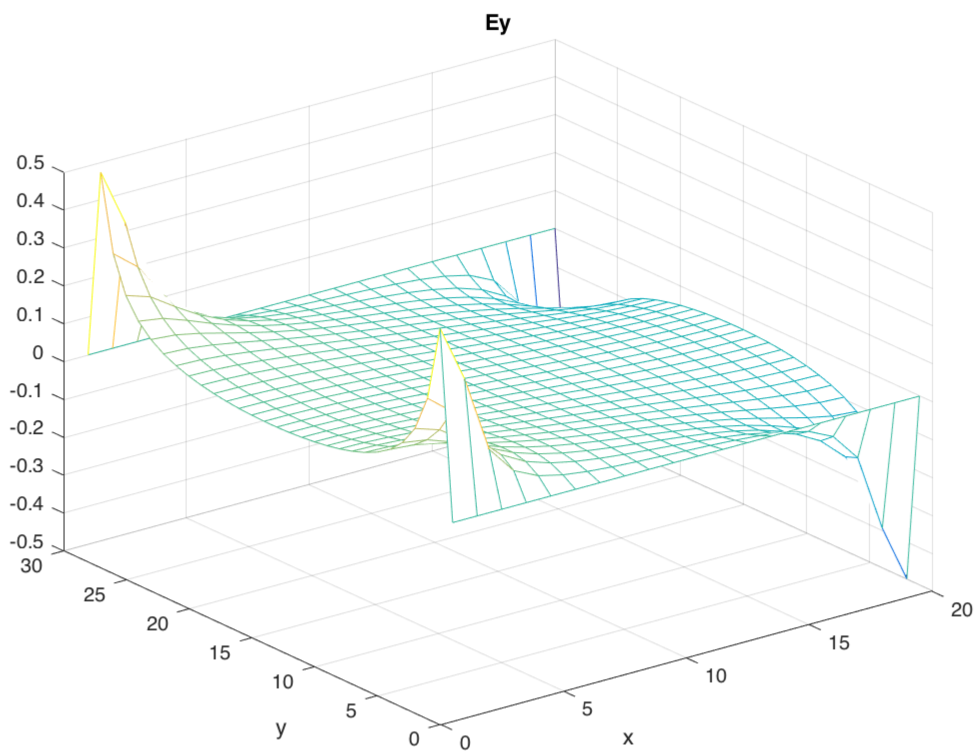
```
W = 20;
L = 30;

v0 = 1;
dx = 1;
dy = 1;
nx = L / dx;
ny = W / dy;

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

for i = 1:nx
    for j = 1:ny
```

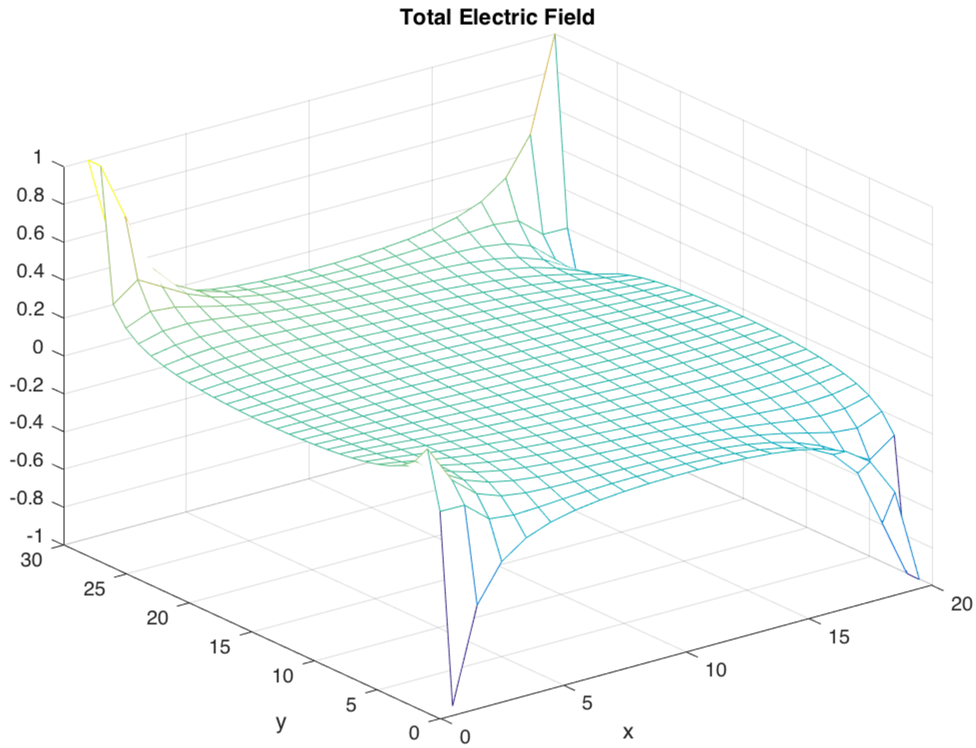
```
        n = i + (j - 1) * nx;
        nym = i + (j - 2) * nx;
        nyp = i + j * nx;
        nxm = i - 1 + (j - 1) * nx;
        nxp = i + 1 + (j - 1) * nx;

        if i == 1
            G(n, n) = 1;
            F(n) = v0;
        elseif i == nx
            G(n, n) = 1;
            F(n) = v0;
        elseif j == 1
            G(n, n) = 1;
        elseif j == ny
            G(n, n) = 1;
        else
            G(n, n) = -4;
            G(n, nxm) = 1;
            G(n, nxp) = 1;
            G(n, nym) = 1;
            G(n, nyp) = 1;
            F(n) = 0;
        end
    end
end

V2 = zeros(nx, ny);
Ex = zeros(nx, ny);
Ey = zeros(nx, ny);
V = G\F;

for i = 1:nx
    for j = 1:ny
        n = i + (j - 1)*nx;
        V2(i, j) = V(n);
    end
end

for i = 1:nx
    for j = 1:ny
        if i == 1
            Ex(i, j) = V2(i+1, j) - V2(i, j);
        elseif i == nx
            Ex(i, j) = V2(i, j) - V2(i-1, j);
        else
            Ex(i, j) = (V2(i+1, j) - V2(i-1, j))/2.0;
        end

        if j == 1
            Ey(i, j) = V2(i, j+1) - V2(i, j);
        elseif j == ny
            Ey(i, j) = V2(i, j) - V2(i, j-1);
```

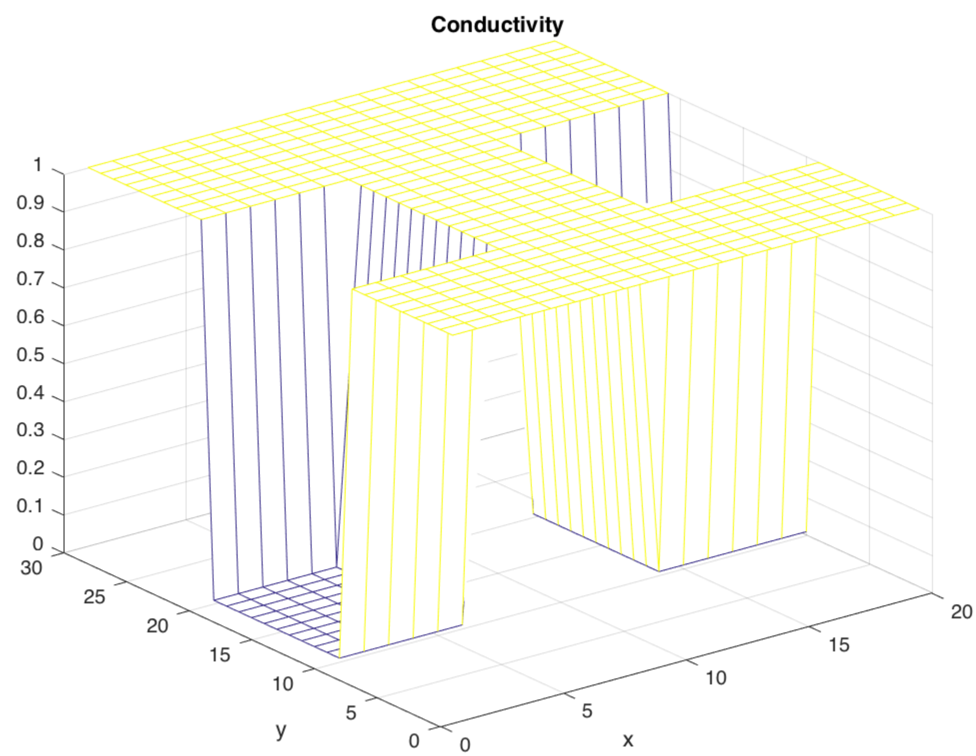
```
        else
            Ey(i, j) = (V2(i, j+1) - V2(i, j-1))/2.0;
        end
    end
end

figure(4)
mesh(Ex)
title('Ex')
xlabel('x')
ylabel('y')

figure(5)
mesh(Ey)
title('Ey')
xlabel('x')
ylabel('y')

figure(6)
mesh(Ex + Ey)
title('Total Electric Field')
xlabel('x')
ylabel('y')

sigma = ones(nx, ny);
sigmaLarge = ones(nx, ny);
sigmaSmall = ones(nx, ny);
sigmaLargeBN = ones(nx, ny);
sigmaSmallBN = ones (nx, ny);

for i = 1:nx
    for j = 1:ny
        if j <= (ny/3.0) || j >= (ny*2/3.0)
            if i >= (nx/3.0) && i <= (nx*2/3.0)
                sigma(i,j) = 10^(-2);
                sigmaLarge(i, j) = 100;
                sigmaSmall(i, j) = 10^(-10);
            end
        end
        if j <= (ny/4.0) || j >= (ny*3/4.0)
            if i >= (nx/3.0) && i <= (nx*2/3.0)
                sigmaSmallBN(i,j) = 10^(-2);
            end
        end
        if j <= (ny/2.25) || j >= (ny - ny/2.25)
            if i >= (nx/3.0) && i <= (nx*2/3.0)
                sigmaLargeBN(i,j) = 10^(-2);
            end
        end
    end
end

figure(7)
mesh(sigma)
```

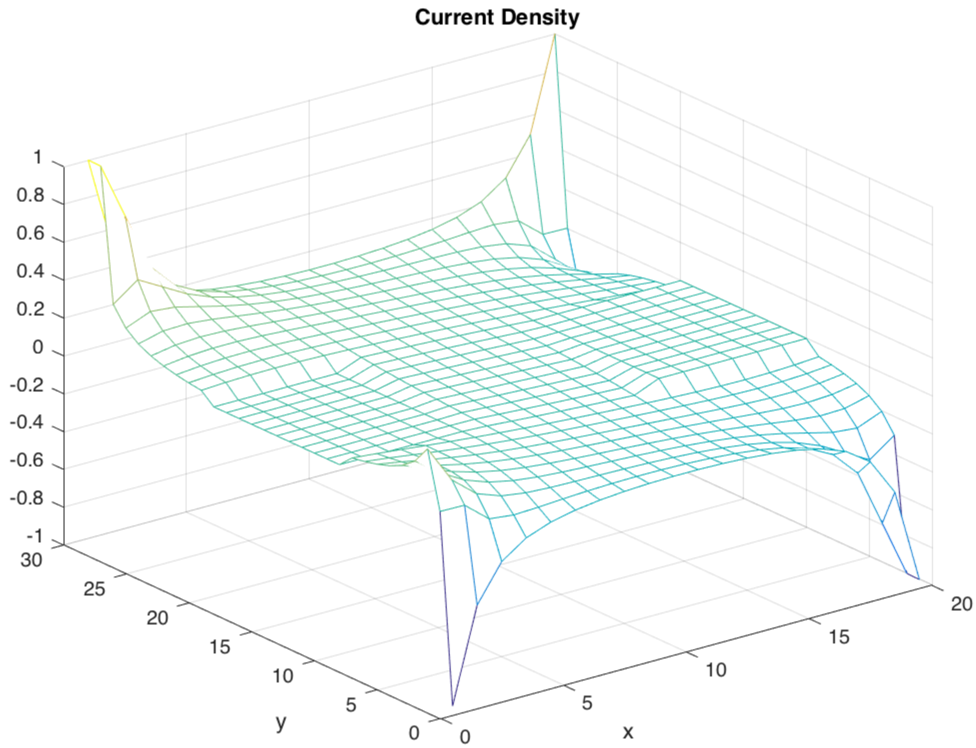
```
title('Conductivity')
xlabel('x')
ylabel('y')

E = Ex + Ey;
J = sigma .* E;
figure(8)
mesh(J)
title('Current Density')
xlabel('x')
ylabel('y')

rho = 1./sigma;

I = V2 ./ rho;
figure(9)
mesh(I)
title('Current')
xlabel('x')
ylabel('y')

%Part 2 c)
rhoLargeBN = 1./sigmaLargeBN;
rhoSmallBN = 1./sigmaSmallBN;
figure(10)
subplot(3, 1, 1)
mesh(I)
title('Current vs. Moderate Bottleneck Size')
subplot(3, 1, 2)
mesh(V2 ./ rhoLargeBN)
title('Current vs. Large Bottleneck Size')
subplot(3, 1, 3)
mesh(V2 ./ rhoSmallBN)
title('Current vs. Small Bottleneck Size')

%Part 2 d)
rhoLarge = 1./sigmaLarge;
rhoSmall = 1./sigmaSmall;
figure(11)
subplot(3, 1, 1)
mesh(I)
title('Current vs. Sigma = 10^-2')
subplot(3, 1, 2)
mesh(V2 ./ rhoLarge)
title('Current vs. Sigma = 100')
subplot(3, 1, 3)
mesh(V2 ./ rhoSmall)
title('Current vs. Sigma = 10^-10')
```

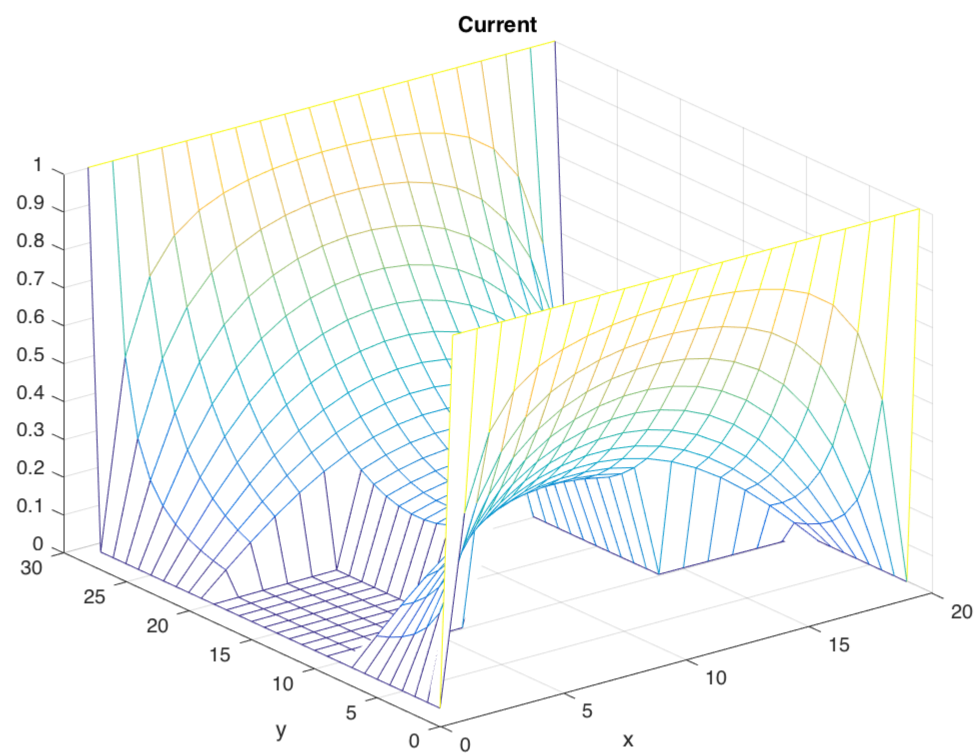

Ex
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
30
25
20
15
10
5
0
y
0
5
10
15
20
x


Ey
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
30
25
20
15
10
5
0
y
0
5
10
15
20
x

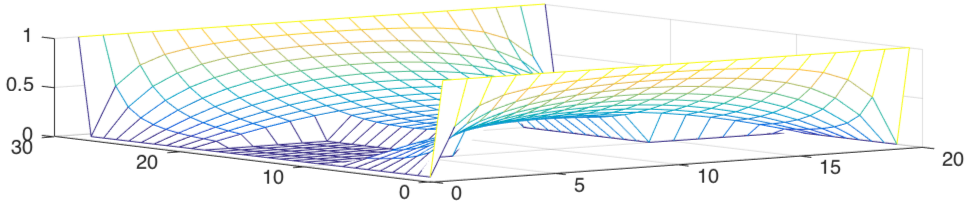

Total Electric Field
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
30
25
20
15
10
5
0
y
0
5
10
15
20
x


Conductivity
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
30
25
20
15
10
5
0
y
0
5
10
15
20
x

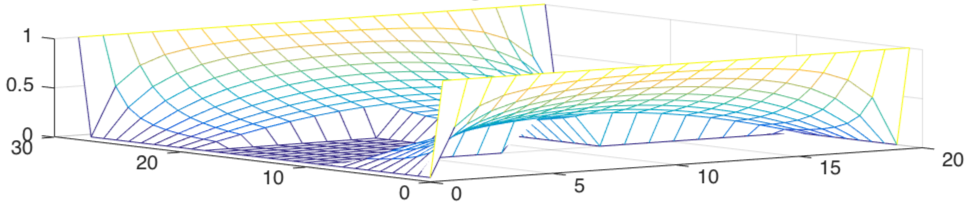

Current Density
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
30
25
20
15
10
5
0
y
0
5
10
15
20
x


Current
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
30
25
20
15
10
5
0
y
0
5
10
15
20
x

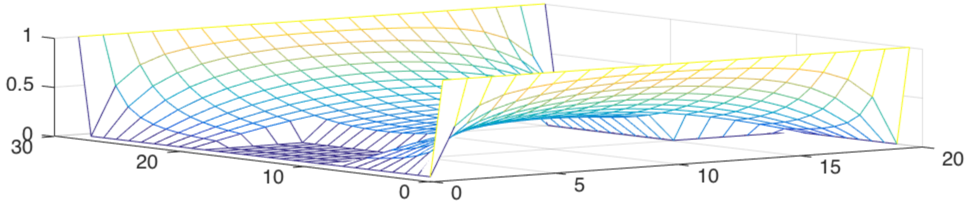
Current vs. Moderate Bottleneck Size



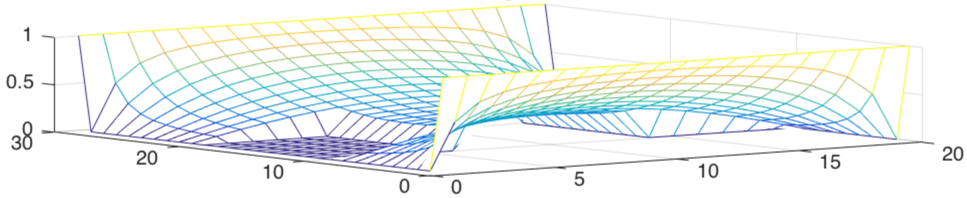
Current vs. Large Bottleneck Size



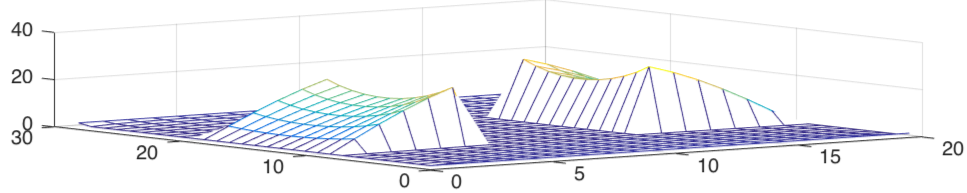
Current vs. Small Bottleneck Size



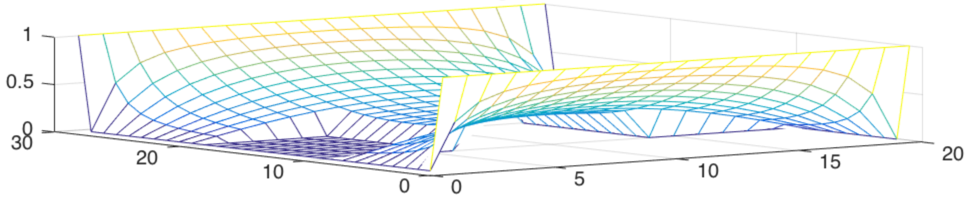
Current vs. Sigma = 10ˆ2



Current vs. Sigma = 100



Current vs. Sigma = 10ˆ10



Published with MATLAB® R2015b