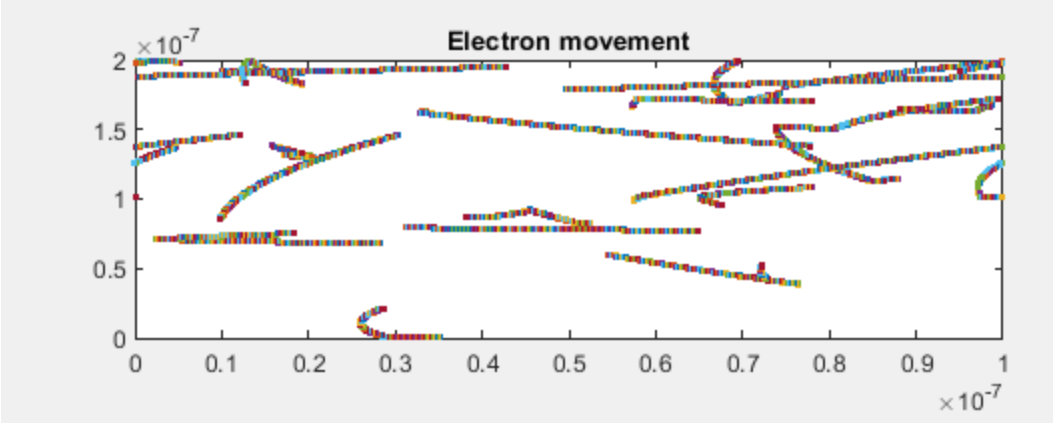
Question 1

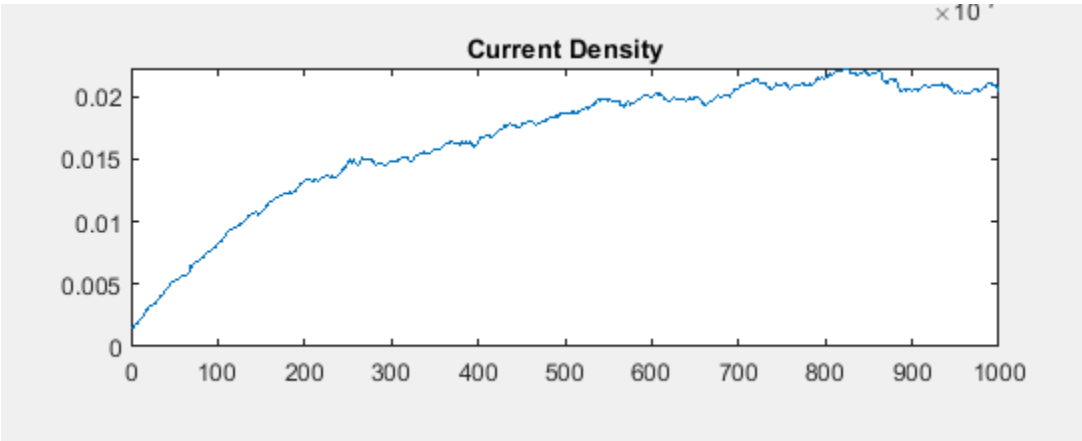
a) Ex = 5000 N/C
b) Fx = 8.0100e-14 N
c) accelerationX = 3.3821e+17 m/s^2

The code used to plot the trajectories with curved movement is shown below:

```
for i=1:1000

    %3. new Xvelocity with acceleration
    xVelocity=xVelocity+accelerationX*deltaT;

    %boundary conditions in vertical array
    IT=(verticalArray>=w);
    yVelocity(IT)=-yVelocity(IT);
    verticalArray(IT)=(verticalArray(IT)-2*(verticalArray(IT)-w));

    IT=(verticalArray<=0);
    yVelocity(IT)=-yVelocity(IT);
    verticalArray(IT)=(verticalArray(IT)+2*(0-verticalArray(IT)));

    %Boundary conditions in horriziontal array
    horrizArray(horrizArray>=l)=horrizArray(horrizArray>=l) - l;
    horrizArray(horrizArray<=0)=horrizArray(horrizArray<=0)+l;

    %Update position with velocity
    horrizArray=horrizArray+xVelocity.*deltaT;
    verticalArray=verticalArray+yVelocity.*deltaT;

    %scattering of the electrons:
    pscat=1-exp(-deltaT/(0.2*10^-12));
    a=rand(numParticles,1);

    xVelocity(a < pscat)=randn(sum(a < pscat),1).*thermalV/sqrt(2);
    yVelocity(a < pscat)=randn(sum(a < pscat),1).*thermalV/sqrt(2);
    VoltageRms=sqrt(xVelocity.^2+yVelocity.^2);

    %Temperature vs time
    subplot(2,1,2)
    currentx(i)=mean(xVelocity)*-q*w*1e15*(100^2);
    plot(linspace(1,1000,1000),currentx);
    title(['Current Density'])
    subplot(2,1,1)
    plot(horrizArray(partplot),verticalArray(partplot),'.')
    title(['Electron movement']);
    xlim([0 l])
    ylim([0 w])
    hold on
    pause(.01)
end
```



d) drift current density formula is: J = N*(charge/(density))*mean(Vn)
the code used to plot is shown above with the electron movement. The plot below shows the drift current density is curved upward whilst the electrons keep moving.



The plot of the current density shows that the results are converging as time goes on. The larger the current density is the lower the velocity is required.

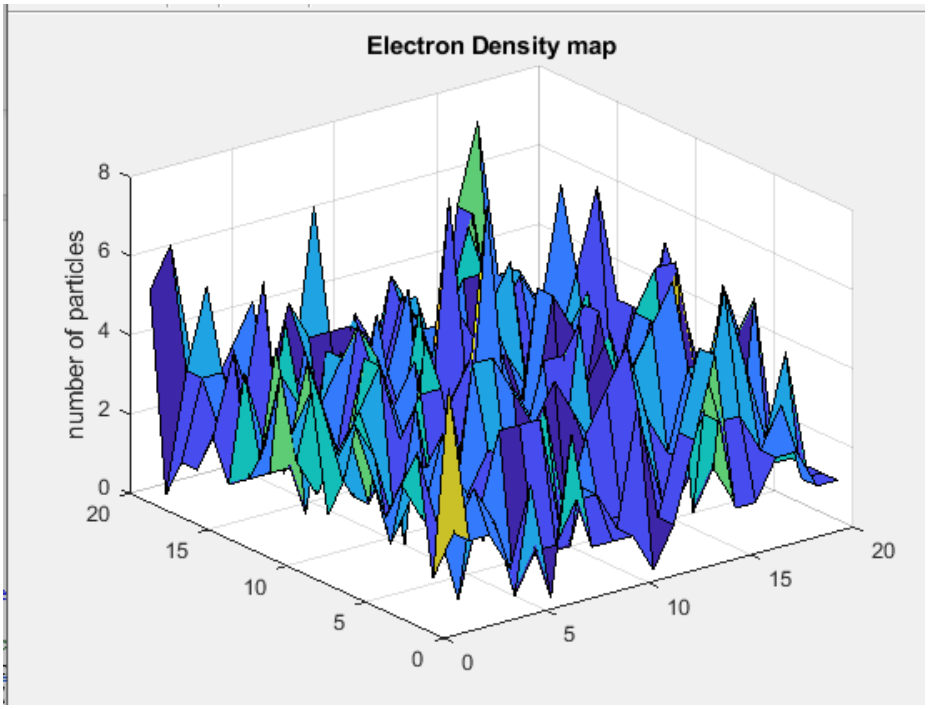
e)

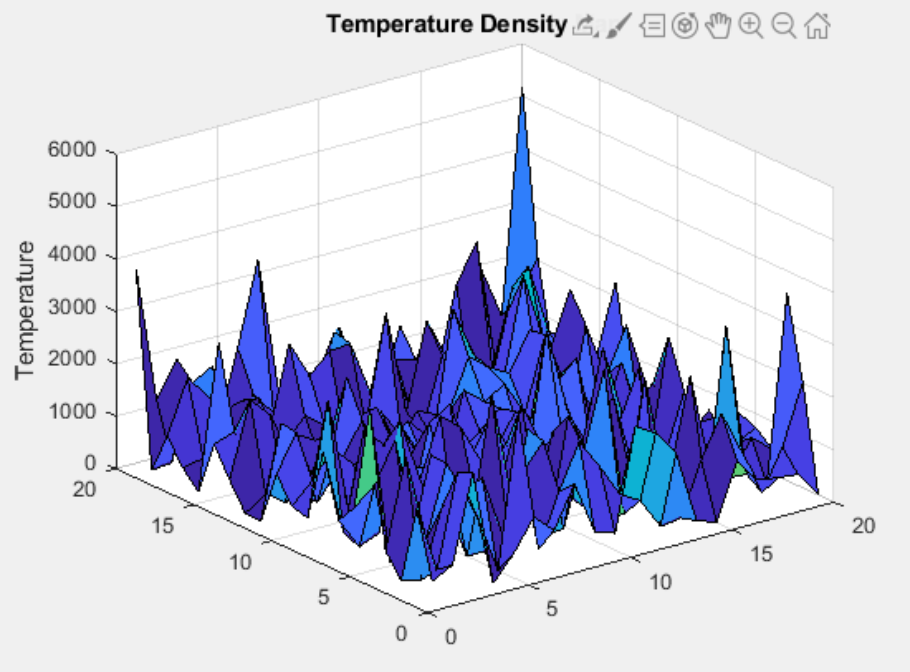
The code below shows how the density and temperature maps were plotted:

```
[Ex,Ey]=meshgrid(0:l/20:l,0:w/20:w);
countPart=zeros(20,20);
tempcheck=zeros(20,20);
counter=0;
totalV=0;
for i=1:20
    txmn=Ex(1,i);
    txmx=Ex(1,i+1);
    for y =1:20
        tymn=Ey(y,1);
        tymx=Ey(y+1,1);
        for j=1:numParticles
            if(horrizArray(j)>txmn && horrizArray(j)<txmx &&
verticalArray(j)<tymx && verticalArray(j)>tymn)
                counter=counter+1;
                countPart(y,i)=countPart(y,i)+1;
```

```
                totalV=totalV+sqrt(xVelocity(j)^2+yVelocity(j)^2);
                if(counter~=0)
                    tempcheck(y,i)=m*(totalV^2)/(counter*k);
                end
            end
        end
        totalV=0;
        counter=0;
    end
end
%electron density map
figure(2)
surf(flipud(countPart))
title('Electron Density map')
zlabel('number of particles')

%temperature map
figure(3)
surf(flipud(tempcheck))
title('Temperature Density Map')
zlabel('Temperature')
```

The resulting plots are shown below:





Question 2

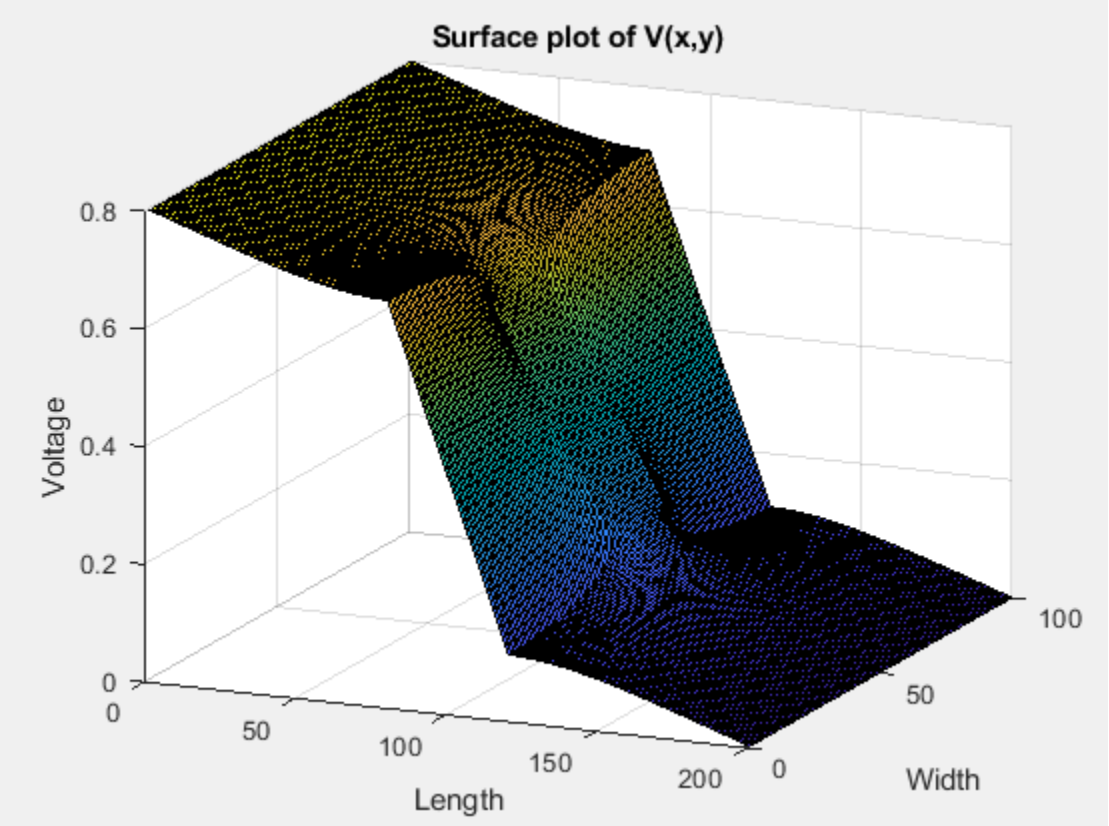
a)

The code to plot the surface plot and electric field vector are shown below:

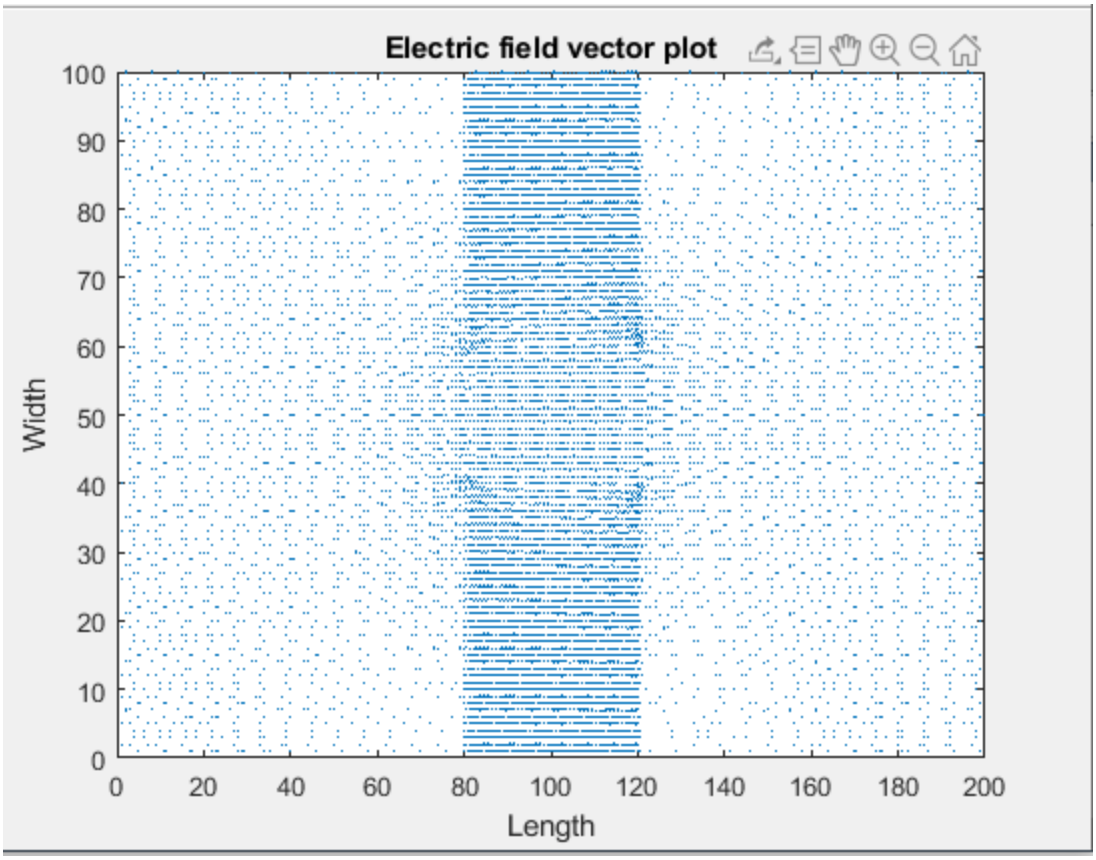
```
kb = 1.380649e-23;
avmeanTime = 0.2e-12;
numSim = 10000;
Length = 2;
Width = 1;
Vo = 0.8;
bottleW = 0.4;
bottleL = 0.4;
nx = 100*Length;
ny = 100*Width;
sigma = zeros(ny,nx);

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


for x = 1:ny
    for y= 1:nx
        if(y >= nx*bottleW && y <= nx-nx*bottleW && (x >= ny-ny*bottleL || x
<= ny*bottleL))
            sigma(x,y) = 1e-2;
        else
            sigma(x,y) = 1;
        end
    end
end
for y = 1:nx
```

```
    for x = 1:ny
        n = x + (y - 1)*ny;
        nxx = x + (y - 1-1)*ny;
        nxp = x + y*ny;
        nyy = x - 1+ (y - 1)*ny;
        nyp = x + 1+ (y - 1)*ny;

        if y == 1
            G(n,n) = 1;
        elseif y == nx
            G(n,n) = 1;
        elseif (x == 1)
            upperY = (sigma(x,y)+sigma(x+1,y))/2;
            upperX = (sigma(x,y)+sigma(x,y+1))/2;
            lowerX = (sigma(x,y)+sigma(x,y-1))/2;
            G(n,nyp) = upperY;
            G(n,nxp) = upperX;
            G(n,nxx) = lowerX;
            G(n,n) = -(upperY + upperX + lowerX);

        elseif (x == ny)
            lowerY = (sigma(x,y)+sigma(x-1,y))/2;
            upperX = (sigma(x,y)+sigma(x,y+1))/2;
            lowerX = (sigma(x,y)+sigma(x,y-1))/2;
            G(n,nyy) = lowerY;
            G(n,nxp) = upperX;
            G(n,nxx) = lowerX;
            G(n,n) = -(lowerY + upperX + lowerX);

        else
            upperY = (sigma(x,y)+sigma(x+1,y))/2;
            lowerY = (sigma(x,y)+sigma(x-1,y))/2;
            upperX = (sigma(x,y)+sigma(x,y+1))/2;
            lowerX = (sigma(x,y)+sigma(x,y-1))/2;

            G(n,nyp) = upperY;
            G(n,nyy) = lowerY;
            G(n,nxp) = upperX;
            G(n,nxx) = lowerX;
            G(n,n) = -(upperY + lowerY + upperX + lowerX);

        end
    end
end

for y = 1:nx
    for x = 1:ny
        n = x + (y - 1)*ny;
        if y == 1
            F(n) = Vo;
        end
    end
end

V = G\F;
```

```
for y = 1:nx
    for x = 1:ny
        n = x + (y - 1)*ny;
        Emap(x,y) = V(n);
    end
end

figure(1)
surf(Emap)
xlabel('Length')
ylabel('Width')
zlabel('Voltage')
title('Surface plot of V(x,y)')
pause (1)
[Ex,Ey] = gradient(-Emap,1e-9);
figure(2)
quiver(Ex,Ey)
xlabel('Length')
ylabel('Width')
title('Electric field vector plot')
ylim([0,100])
xlim([0,200])
pause (1)
```



b)



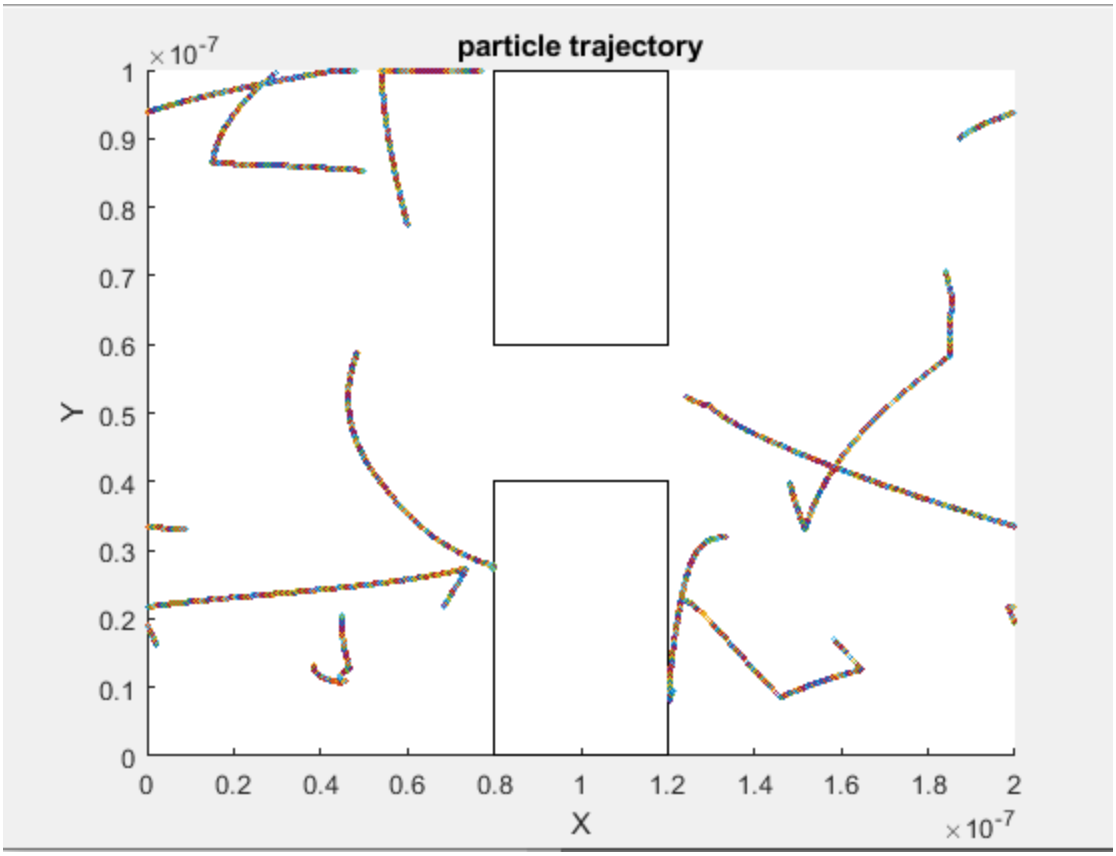
c) The trajectory of electrons under this new field for min of 1000 electrons for a min of 1000 steps code is shown below, the trajectory is also shown in the figure below:

```
for t = 1:1000
    for n = 1:numSim
        if(round(1e9*xPosition(n,1)) <= 0 || round(yPosition(n,1)*1e9) <= 0)
            Forcex =
Ex(ceil(yPosition(n,1)*1e9),ceil(1e9*xPosition(n,1))).*qCharge;
            Forcey =
Ey(ceil(yPosition(n,1)*1e9),ceil(1e9*xPosition(n,1))).*qCharge;
        elseif(round(yPosition(n,1)*1e9) > 100 || round(1e9*xPosition(n,1)) >
200)
            Forcex =
Ex(floor(yPosition(n,1)*1e9),floor(1e9*xPosition(n,1))).*qCharge;
            Forcey =
Ey(floor(yPosition(n,1)*1e9),floor(1e9*xPosition(n,1))).*qCharge;
        else
            Forcex =
Ex(round(yPosition(n,1)*1e9),round(1e9*xPosition(n,1))).*qCharge;
            Forcey =
Ey(round(yPosition(n,1)*1e9),round(1e9*xPosition(n,1))).*qCharge;
        end
        % calcualte the acclerations now with the forces
        accelerationx = Forcex/massE;
        accelerationy = Forcey/massE;
```

```
        % get the velocities
        xVelocity(n,1) = xVelocity(n,1) + accelerationx*tStep;
        yVelocity(n,1) = yVelocity(n,1) + accelerationy*tStep;

        if (rand < PScat)
            xVelocity(n,:) = randn(1)*sqrt((kb*Temperature)/massE);
            yVelocity(n,:) = randn(1)*sqrt((kb*Temperature)/massE);
            newXPos(n,:) = tStep*xVelocity(n,1);
            newYPos(n,:) = tStep*yVelocity(n,1);
            XoldPosition = xPosition(:,1);
            YoldPosition = yPosition(:,1);
        else
            newXPos(n,:) = tStep*xVelocity(n,1);
            newYPos(n,:) = tStep*yVelocity(n,1);
        end
        if (xPosition(n,1) + newXPos(n) > 2e-7)
            xPosition(n,2) = newXPos(n) + xPosition(n,1) - 2e-7;
        elseif (xPosition(n,1) + newXPos(n) < 0)
            xPosition(n,2) = newXPos(n) + xPosition(n,1) + 2e-7;
        else
            xPosition(n,2) = newXPos(n) + xPosition(n,1);
        end

        if ((yPosition(n,1) + newYPos(n) > 1e-7)|| (yPosition(n,1) +
newYPos(n) < 0))
            newYPos(n) = -newYPos(n);
            yPosition(n,2) = newYPos(n) + yPosition(n,1);
        elseif(xPosition(n,1) > 1.2e-7 && xPosition(n,1) < 2e-7 &&
yPosition(n,1) > 0.6e-7 && yPosition(n,1) < 1e-7 &&(xPosition(n,1) +
newXPos(n) < 1.2e-7))
            newXPos(n) = -newXPos(n);
            xPosition(n,2) = newXPos(n) + xPosition(n,1);
        elseif (xPosition(n,1) > 0 && xPosition(n,1) < 0.8e-7 &&
yPosition(n,1) > 0.6e-7 && yPosition(n,1) < 1e-7 && (xPosition(n,1) +
newXPos(n) > 0.8e-7))
            newXPos(n) = -newXPos(n);
            xPosition(n,2) = newXPos(n) + xPosition(n,1);
        elseif (xPosition(n,1) > 0 && xPosition(n,1) < 0.8e-7 &&
(xPosition(n,1) + newXPos(n) > 0.8e-7) && yPosition(n,1)> 0 && yPosition(n,1)
< 0.4e-7)
            newXPos(n) = -newXPos(n);
            xPosition(n,2) = newXPos(n) + xPosition(n,1);
        elseif(xPosition(n,1) > 1.2e-7 && xPosition(n,1) < 2e-7 &&
yPosition(n,1) > 0 && yPosition(n,1) < 0.4e-7 &&(xPosition(n,1) + newXPos(n)
< 1.2e-7))
            newXPos(n) = -newXPos(n);
            xPosition(n,2) = newXPos(n) + xPosition(n,1);
        elseif(xPosition(n,1) > 0.8e-7 && xPosition(n,1) < 1.2e-7 &&
yPosition(n,1) > 0.4e-7 && yPosition(n,1) < 0.6e-7 && (yPosition(n,1) +
newYPos(n) < 0.4e-7))
            newYPos(n) = -newYPos(n);
            yPosition(n,2) = newYPos(n) + yPosition(n,1);
        elseif(xPosition(n,1) > 0.8e-7 && xPosition(n,1) < 1.2e-7 &&
yPosition(n,1) > 0.4e-7 && yPosition(n,1) < 0.6e-7 &&(yPosition(n,1) +
newYPos(n) > 0.6e-7))
```

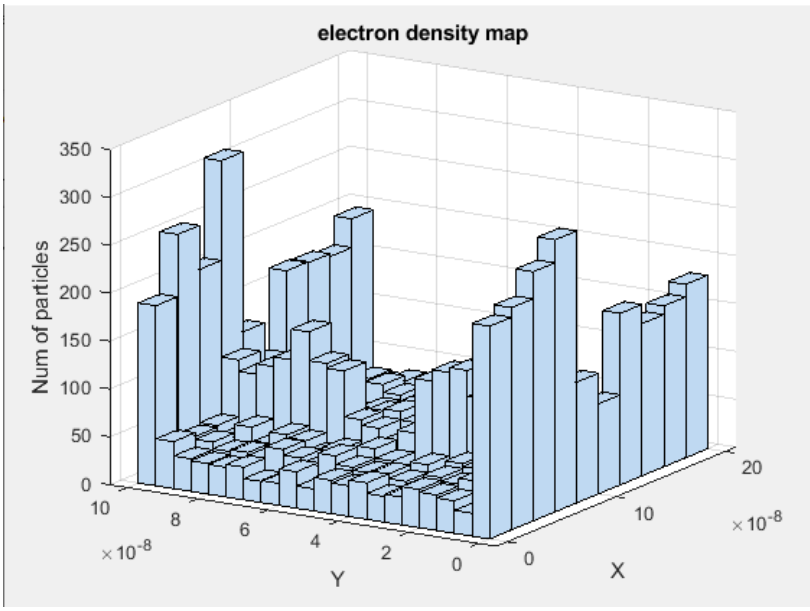
```
            newYPos(n) = -newYPos(n);
            yPosition(n,2) = newYPos(n) + yPosition(n,1);
        else
            yPosition(n,2) = newYPos(n) + yPosition(n,1);
        end
    end
    timeVector(t) = tStep*t;

    if (counter == 0)
        figure(3)
        scatter(xPosition([1:10],2),yPosition([1:10],2),1)
        hold on
        plot([0.8e-7,0.8e-7],[0,0.4e-7],'k',[0.8e-7,1.2e-7],[0.4e-7,0.4e-
7],'k',[1.2e-7,1.2e-7],[0.4e-7,0],'k',[0.8e-7,1.2e-7],[0,0],'k')
        plot([0.8e-7,0.8e-7],[1e-7,0.6e-7],'k',[0.8e-7,1.2e-7],[0.6e-7,0.6e-
7],'k',[1.2e-7,1.2e-7],[0.6e-7,1e-7],'k',[0.8e-7,1.2e-7],[1e-7,1e-7],'k')

        title(['particle trajector'])
        xlabel('X')
        ylabel('Y')
        xlim([0 200e-9])
        ylim([0 100e-9])
    elseif (counter > 0 && counter < 1000)
        title(['particle trajectory'])
        scatter(xPosition([1:10],2),yPosition([1:10],2),1)
        hold on
    else
        title(['particle trajector'])
        scatter(xPosition([1:10],2),yPosition([1:10],2),1)
        hold off
    end
    pause(0.0001)
    xPosition(:,1) = xPosition(:,2);
    yPosition(:,1) = yPosition(:,2);

    counter = counter + 1;
end
```

*** I only plotted couple of electrons since computer keeps freezing.



Question 3

a) density map is shown below:



Most of the electrons moving around the top and bottom of the device and a little on the sides. Since the y electric field would cause some electrons to stay at top and bottom. The density plot shows no electrons present in the bottleneck.

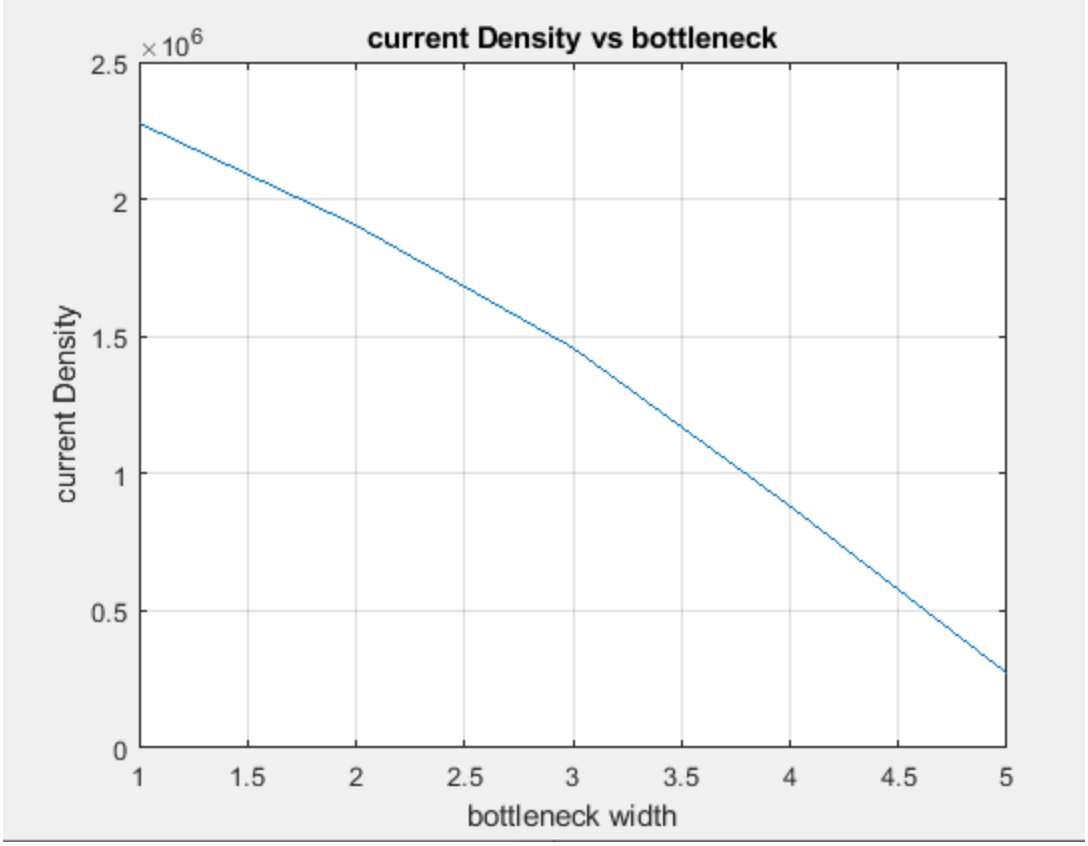
b)

the code for average current at different bottleneck width is shown below:

```
for numberSolution = 1:5
    bottleW = 0.4;
    bottleL = 0.4*(1+(numberSolution/20));
    nx = 100*Length;
    ny = 100*Width;
    sigma = zeros(ny,nx);
    for x = 1:ny
        for y = 1:nx
            if(y >= nx*bottleW && y <= nx-nx*bottleW &&  (x >= ny-ny*bottleL
|| x <= ny*bottleL))
                sigma(x,y) = 1e-2;
            else
                sigma(x,y) = 1;
            end
        end
    end
    G = sparse(nx*ny,nx*ny);
    F = zeros(nx*ny,1);
    for y = 1:nx
        for x = 1:ny
            n = x + (y - 1)*ny;
            nxx = x + (y - 1-1)*ny;
            nxp = x + y*ny;
            nyy = x - 1 + (y - 1)*ny;
            nyp = x + 1 + (y - 1)*ny;

            if y == 1
                G(n,n) = 1;
            elseif y == nx
                G(n,n) = 1;
            elseif (x == 1)
                upperY = (sigma(x,y)+sigma(x+1,y))/2;
                upperX = (sigma(x,y)+sigma(x,y+1))/2;
                lowerX = (sigma(x,y)+sigma(x,y-1))/2;

                G(n,n) = -(upperY + upperX + lowerX);
                G(n,nyp) = upperY;
                G(n,nxp) = upperX;
                G(n,nxx) = lowerX;

            elseif (x == ny)
                lowerY = (sigma(x,y)+sigma(x-1,y))/2;
                upperX = (sigma(x,y)+sigma(x,y+1))/2;
                lowerX = (sigma(x,y)+sigma(x,y-1))/2;

                G(n,n) = -(lowerY + upperX + lowerX);
                G(n,nyy) = lowerY;
                G(n,nxp) = upperX;
                G(n,nxx) = lowerX;
```

```
            else
                upperY = (sigma(x,y)+sigma(x+1,y))/2;
                lowerY = (sigma(x,y)+sigma(x-1,y))/2;
                upperX = (sigma(x,y)+sigma(x,y+1))/2;
                lowerX = (sigma(x,y)+sigma(x,y-1))/2;

                G(n,n) = -(upperY + lowerY + upperX + lowerX);
                G(n,nyp) = upperY;
                G(n,nyy) = lowerY;
                G(n,nxp) = upperX;
                G(n,nxx) = lowerX;
            end
        end
    end
    for y = 1:nx
        for x = 1:ny
            n = x + (y - 1)*ny;
            if (y == 1)
                F(n) = Vo;
            end
        end
    end
    V = G\F;
    for y = 1:nx
        for x = 1:ny
            n = x + (y - 1)*ny;
            Emap(x,y) = V(n);
        end
    end
    [Ex,Ey] = gradient(-Emap,1e-9);
    currentY(numberSolution) = mean(sigma(:,1).*Ex(:,1));
    currentX(numberSolution) = mean(sigma(:,1).*Ex(:,1));
    currentDensity= sqrt(currentX.^2 + currentY.^2);
end
figure(6)
plot(linspace(1,5,5),current)
xlabel('bottleneck width')
ylabel('current Density')
title('current Density vs bottleneck')
grid on
```

The plot is shown below:



IT is seen as the bottleneck width increase the current density drops. That is because the resistance of the device is being increased. And, when the width is larger, the electrons do not have much space to move to acquire a high current density.

c)

In order to increase the accuracy a smaller size of the device can be used. Also, more particles to test and smaller step time. Also, increasing the mesh when calculating the temperature and density. Another was is letting the simulation run in 3-D so particles are not constricted to x-y plane only.