ELEC 4700

Assignment - 1

Monte-Carlo Modeling of Electron Transport

Written by:

Matthew Janok

101036060

1. Electron Modelling (40)

a. vth = sqrt((kb*T)/mn)

solving this equation in Matlab where T = 300 and kb = 1.381e-23 we get:

$$vth = 1.3225e+05\ m/s$$

b. If the mean time between collisions is τmn = 0.2ps, then the mean free path can be found by multiplying the time by average velocity to get the distance of the mean free path. Using the average velocity as vth = 1.3225e+05 m/s, we find:

$$MFP = vth * \tau mn = 2.6e-08\ m$$

c. For code see appendix A

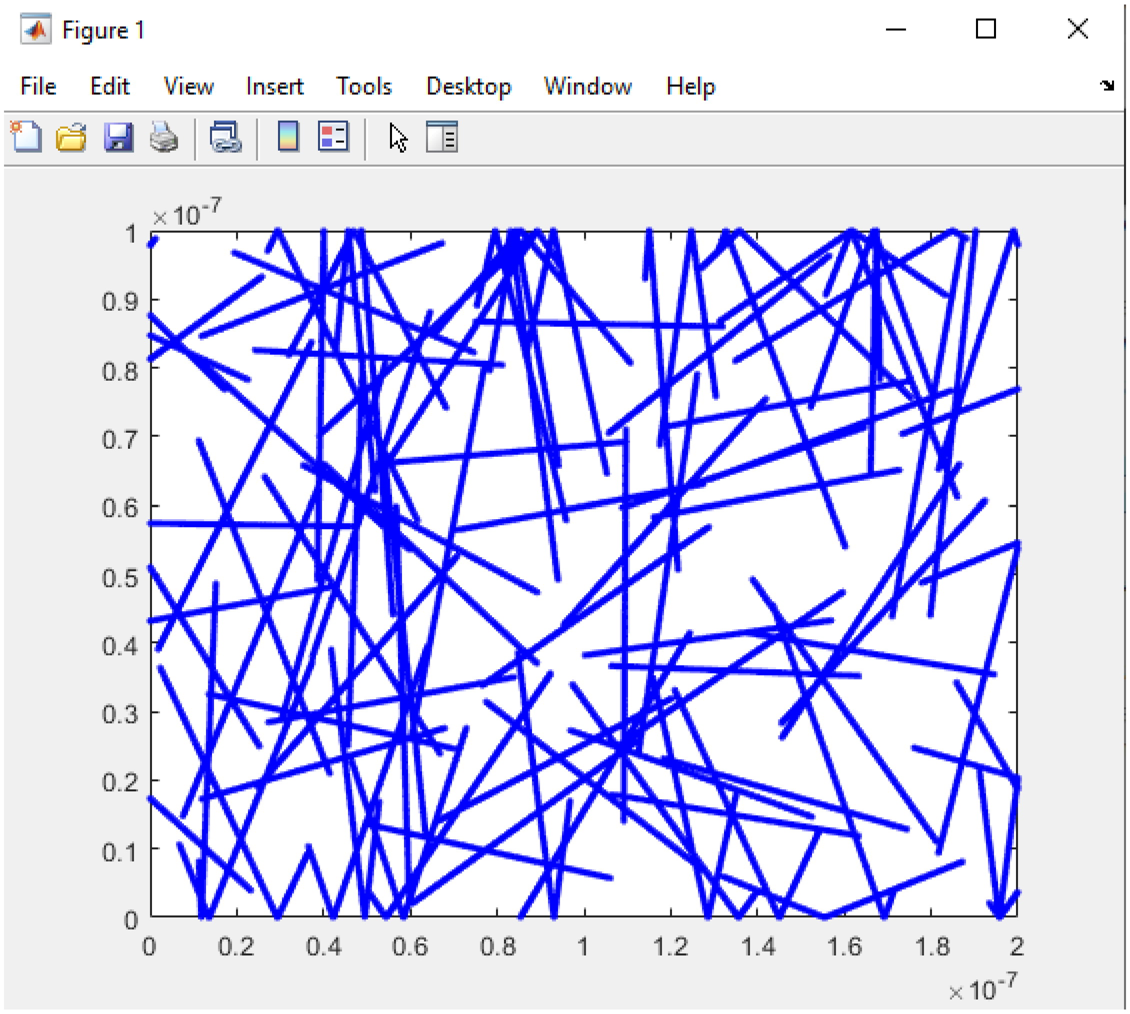

Figure 1: 2D Plot of Particle Trajectories

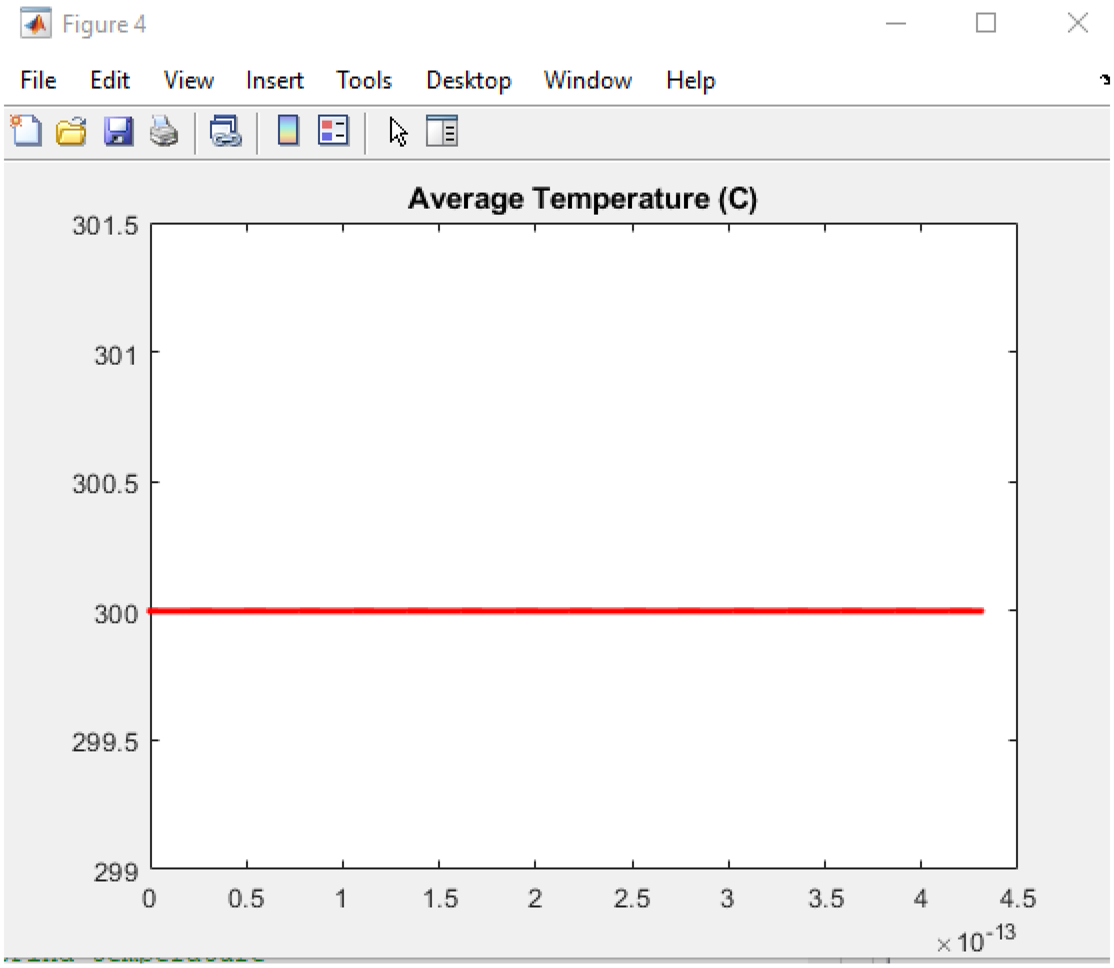

Figure 2: Average temperature remains constant

2. Collisions with Mean Free Path (MFP) (25)

a.

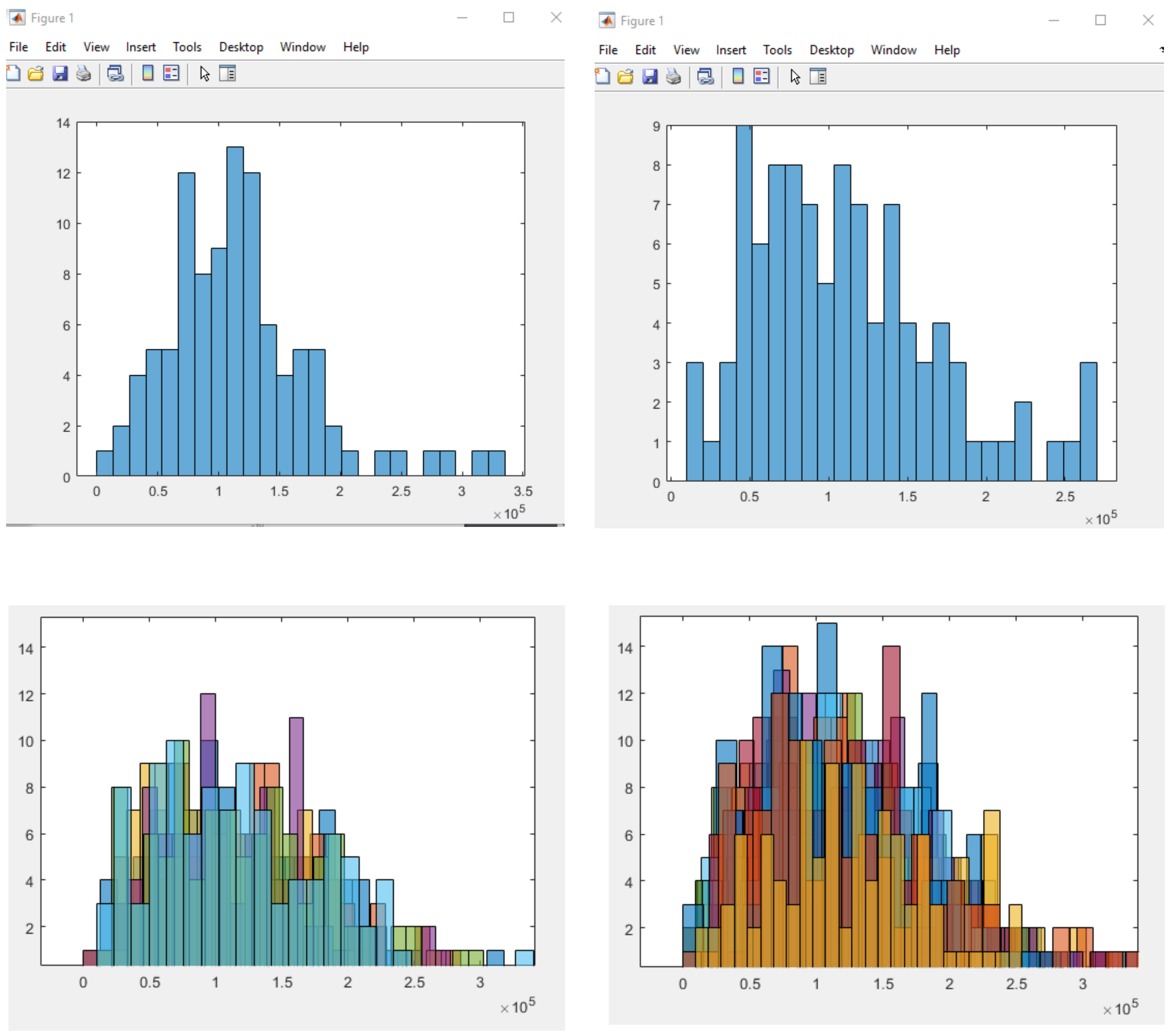

Figure 3: Histogram plots showing a Maxwell-Boltzmann distribution

The first two plots show the types of random velocities we see using the Maxwell-Boltzmann distribution. The two figures on the bottom show the random velocities as they iterate for new calculations. We can see from many iterations that this is indeed a Maxwell-Boltzmann distribution centered at Vth which was found to be $vth = 1.3225e + 05\ m/s$.

b.

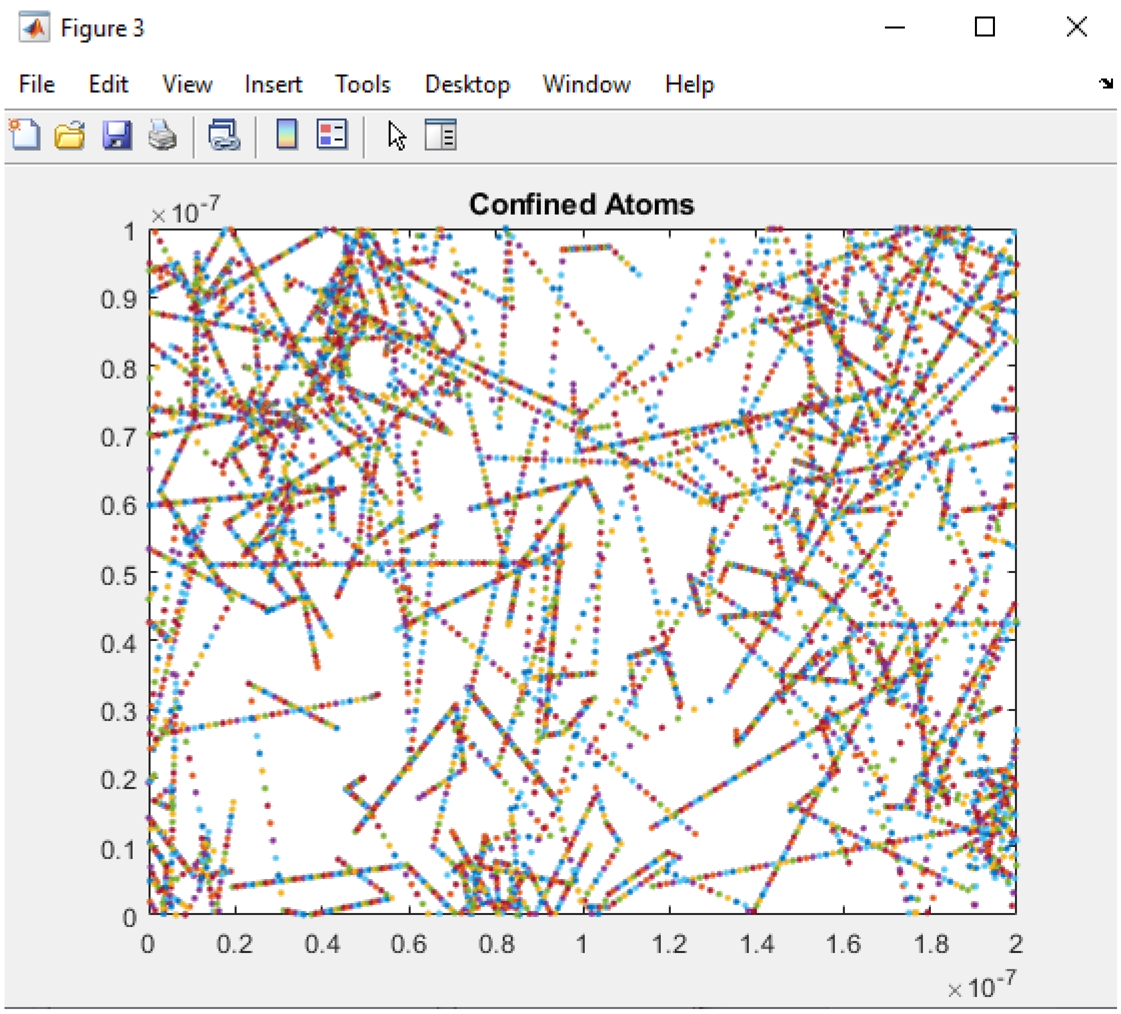

Figure 4: 2D plot of particle trajectories for part 2

We can see in Figure 4 above that the particle scattering was implemented where the scattering causes the particles to re-thermalize and change direction and velocity. The new velocity is based on the Maxwell-Boltzmann distribution and the probability of scattering is based on $1 - e^{-\frac{dt}{\tau mn}}$ where dt was the timestep of 1E-14 sec and τmn of 0.2ps.

c.

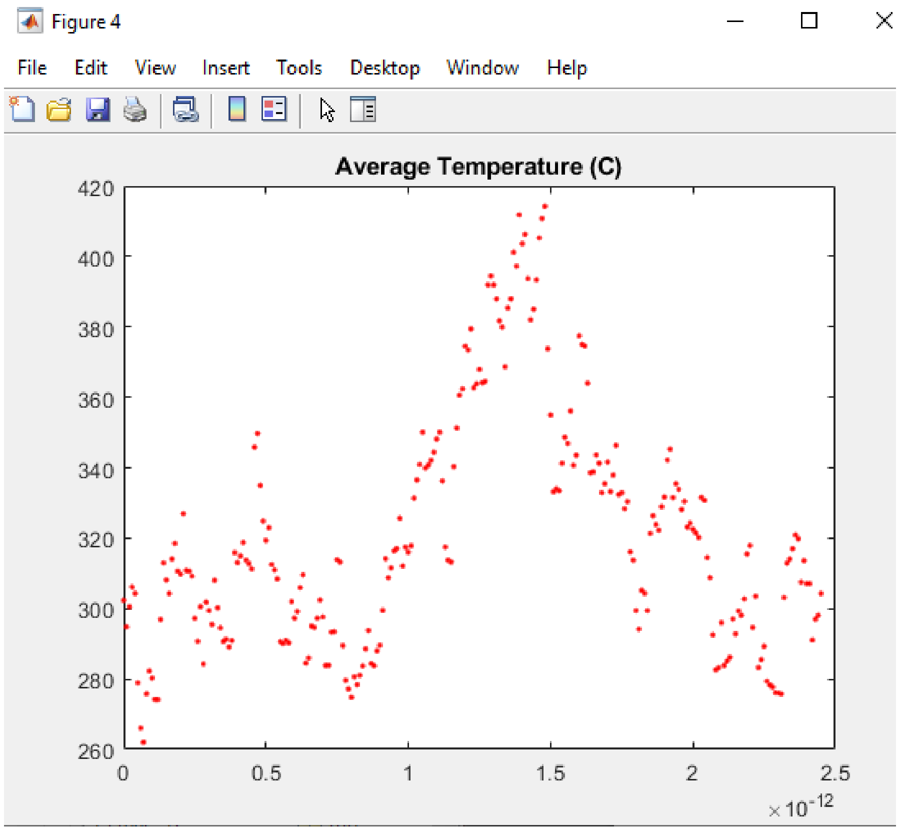

Figure 5: Temperature plot of the system centered around 300k

This temperature plot is based on the average velocity found from the Maxwell-Boltzmann distribution. Because this is a random distribution, we can see that the temperature rises and falls about 300k over time. There happened to be a large spike in temperature from this simulation which can also be seen through Figure 6 in the mean free path plot since higher temperatures means the particles move faster increasing the mean free path.

d.

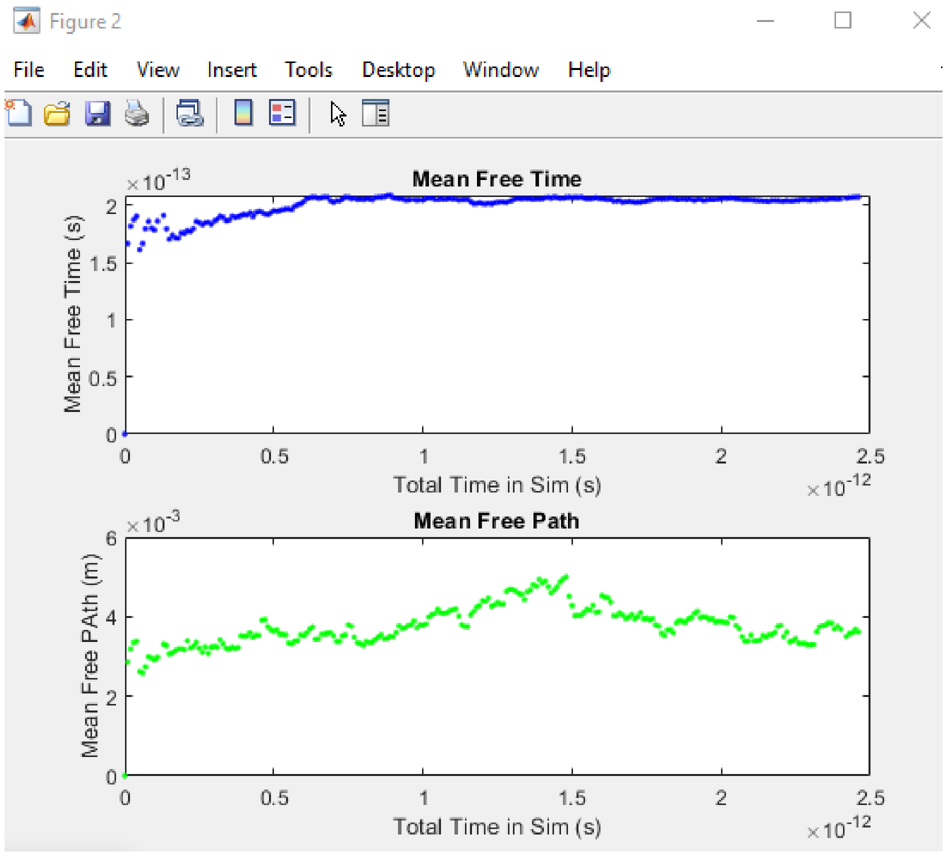

Figure 6: Mean free time and Mean free path plots

We can see in Figure 6 that the actual mean free time plot finds a steady state at 0.2ps as we go through time. This mean free time was used in the calculation to find the actual mean free path which can be seen to average at about 3.5E-3m. We can see the effect the temperature from Figure 5 has on the mean free path because from 1ps to 1.5ps we find a maximum in temperature and in the mean free path.

3. Collisions with Mean Free Path (MFP) (25)

a.

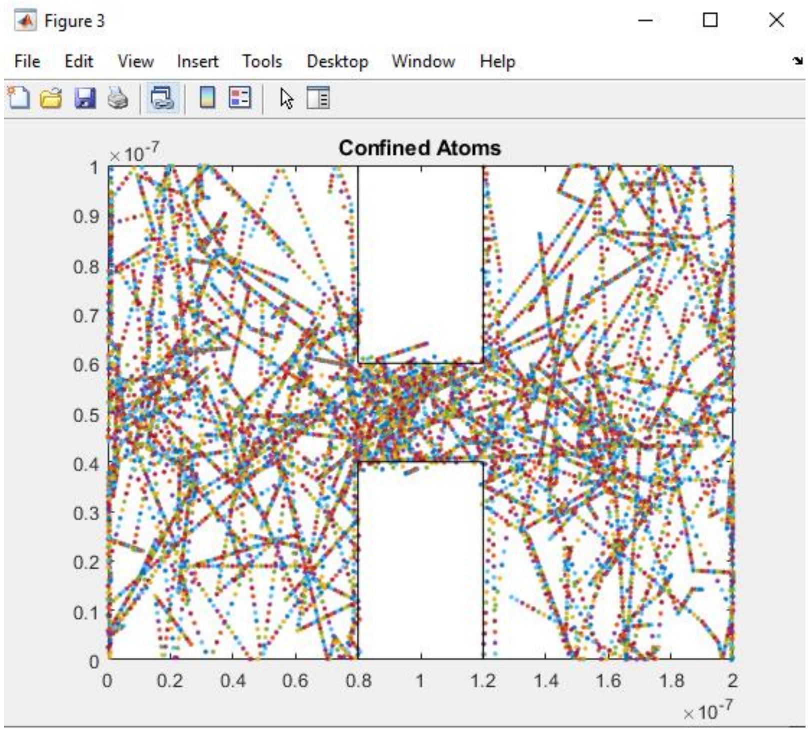

Figure 7: 2D plot of particle trajectories in a “bottleneck”

We can see the trajectories from Figure 7 deflect much more inside the bottleneck than they do outside that region. We also notice the particles are still scattering at a random rate and the particles are still reflecting and transmitting through the X and Y bounds.

b.

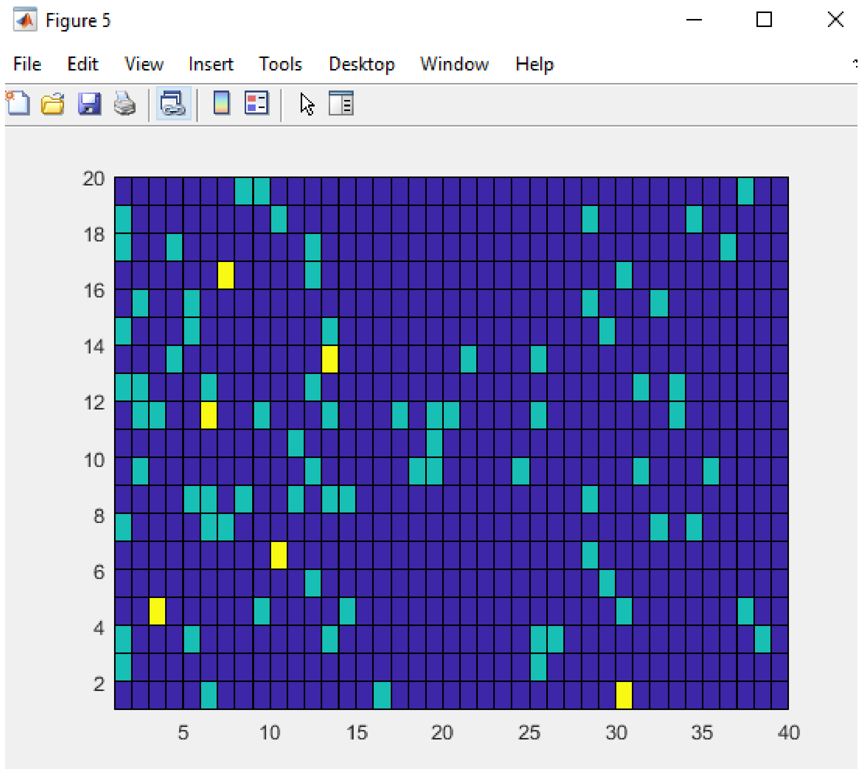

Figure 8: Electron Density Map

The electron density map in Figure 8 shows us where the final electron positions are located when the simulation ends. We can see from the map that there are more electrons concentrated on the left side of the system than on the right side. We can also notice that there are two large spaces in the middle of the map where the bottleneck prevents the electrons from entering.

c.

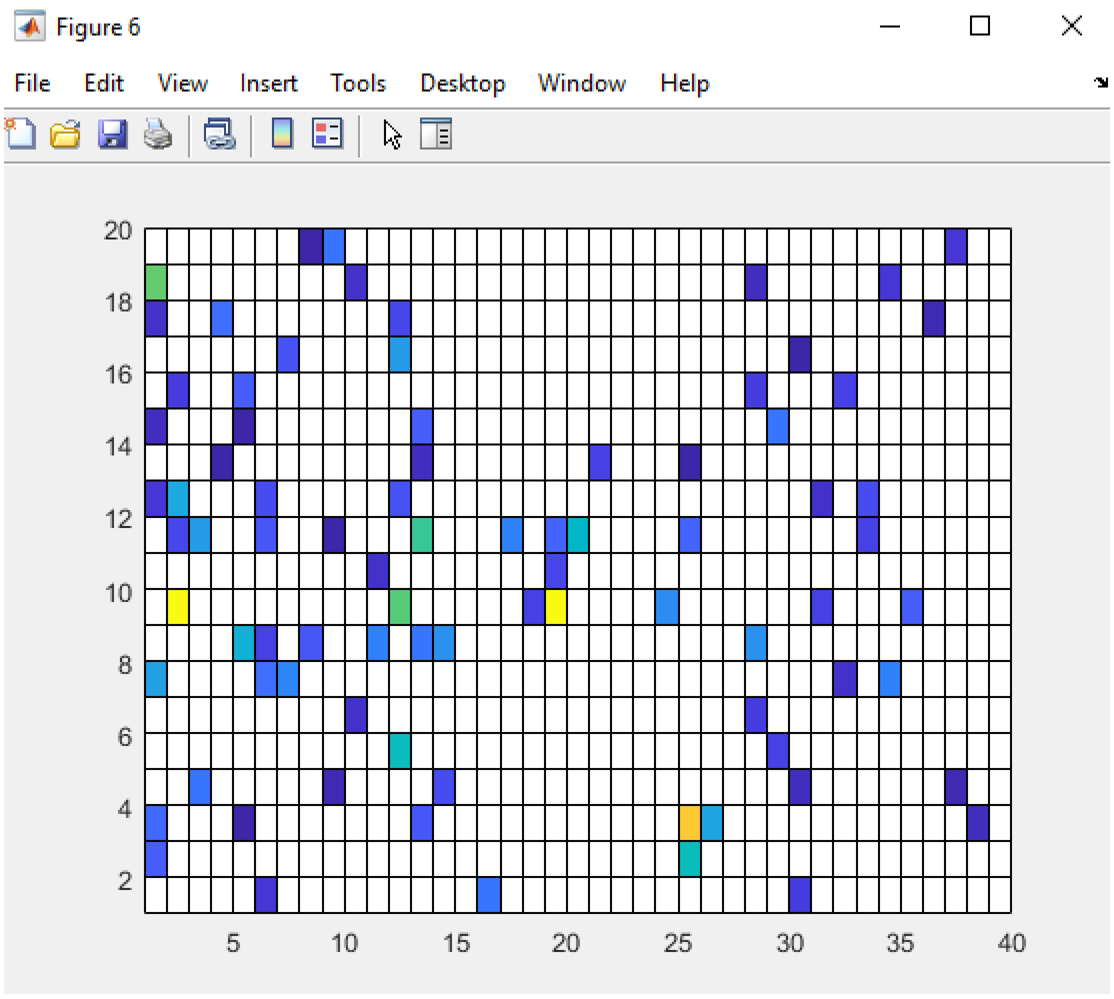

Figure 9: Temperature Map

Figure 9 relies on the electron positions in Figure 8 and trajectories from Figure 7 to calculate the average velocity of the particles in each section of the plane. Once we have the average velocity of a given discretized space, we find the average temperature of that space and map it. Using Figure 8 as a reference, we can see that all the temperature squares share the same discretized space as the Electron Density squares. However, we can make a notable comparison that the spaces with higher ED don't necessarily have the highest temperature because the electrons could be moving slow compared to single electrons which could be moving fast, making them have a larger temperature.

APPENDIX A

Code for Part 1 graphs

```
clc
clear

mo = 9.11e-31;
mn = 0.26*mo;
kb = 1.381e-23;
T = 300;

%Initialise the particles
initialX = 200e-9*rand(100,1);
initialY = 100e-9*rand(100,1);
axis ([0 200e-9 0 100e-9])

%Initialise angles
angleRad = 2*pi*rand(100,1);

%Set velocity
vth = sqrt((kb*T)/mn);
velocityX = vth.*cos(angleRad);
velocityY = vth.*sin(angleRad);

for time = 0:1e-15:0.01
    %Find new positions
    newX = initialX + velocityX*1e-15;
    newY = initialY + velocityY*1e-15;


    %Find temperature
    Vavg = mean((velocityX.^2) + (velocityY.^2));
    T = (mn*Vavg)/(kb);


    %Check X boundary conditions
    [NH,IH] = max(newX);
    [NL,IL] = min(newX);

    upperX = newX > 200e-9;
    newX(upperX)= newX(upperX)-200;

    lowX = newX < 0;
    newX(lowX) = newX(lowX)+200;



    %Check Y boundary conditions
    [NumH,IndexH] = max(newY);
    [NumL,IndexL] = min(newY);

    upperY = newY > 100e-9;
    velocityY(upperY)= -velocityY(upperY);

    lowY = newY < 0;
    velocityY(lowY) = -velocityY(lowY);


    initialX = newX;
    initialY = newY;

    figure(1)
    plot(newX,newY, 'b.')
    hold on

    axis ([0 200e-9 0 100e-9])


    figure(4)
    title('Average Temperature (C)');
    plot(time, T, 'r.')
    hold on


    pause(0.01)
    time

end
```

APPENDIX B

Code for Part 2 graphs

```
clc

        clear

        mo = 9.11e-31;
        mn = 0.26*mo;
        kb = 1.381e-23;
        T = 300;
        Pscat = 1-exp(-1e-14/0.2e-12);
        tauMN = 0;

        %Initialize Bottleneck
        % BLX = [0.8e-7 0.8e-7; 0.8e-7 1.2e-7; 1.2e-7 1.2e-7];
        % BLY = [0 0.4e-7; 0.4e-7 0.4e-7; 0.4e-7 0];
        % BHX = [0.8e-7 0.8e-7; 0.8e-7 1.2e-7; 1.2e-7 1.2e-7];
        % BHY = [1e-7 0.6e-7; 0.6e-7 0.6e-7; 0.6e-7 1e-7];
        % figure(3)
        % plot(BLX,BLY,'k',BHX,BHY,'k')
        % hold on

        %Initialise the particles
        initialX = 200e-9*rand(100,1);
        initialY = 100e-9*rand(100,1);
        axis ([0 200e-9 0 100e-9])

        %Initialise angles
        angleRad = 2*pi*rand(100,1);

        %Set velocity
        vth = sqrt((kb*T)/mn);

        %Maxwell Boltzman Inital Velocity
        MD1 = randn(100,1).*(vth/sqrt(2));
        MD2 = randn(100,1).*(vth/sqrt(2));
        MaxwellBoltzman = sqrt((MD1).^2+(MD2).^2);
        initialRV = MaxwellBoltzman;

        % figure(1)
% velocity = histogram(initialRV,20);

scat = 0;


velocityX = initialRV.*cos(angleRad);
velocityY = initialRV.*sin(angleRad);


for time = 0:1e-14:0.01


    %Find new positions
    newX = initialX + velocityX*1e-14;
    newY = initialY + velocityY*1e-14;

    %Check for Scatter
    Escat = rand(100,1) < Pscat;
     if newX(Escat) > 0
        %Rethermalize
        MD1 = randn(100,1).*(vth/sqrt(2));
        MD2 = randn(100,1).*(vth/sqrt(2));
        MaxwellBoltzman = sqrt((MD1).^2+(MD2).^2);
        initialRV = MaxwellBoltzman;

        %Find New Velocities
        angleRad = 2*pi*rand(100,1);
        velocityX(Escat) = initialRV(Escat).*cos(angleRad(Escat));
        velocityY(Escat) = initialRV(Escat).*sin(angleRad(Escat));
        figure(1)
        velocity = histogram(initialRV,25);

        %Mean Free Path/Time Between Collisions

        scat = scat+ sum(Escat);
        tauMN = (time*100)/scat;
        Vavg = mean((velocityX.^2) + (velocityY.^2));

        MFP = tauMN*Vavg;


        figure(2)
        subplot(2,1,1)
    title('Mean Free Time')
    xlabel('Total Time in Sim (s)')
    ylabel('Mean Free Time (s)')
    plot(time, tauMN, 'b.')
    hold on

    figure(2)
    subplot(2,1,2)
    xlabel('Total Time in Sim (s)')
    ylabel('Mean Free PAth (m)')
    title('Mean Free Path')
    plot(time, MFP, 'g.')
    hold on

end



%Find temperature
Vavg = mean((velocityX.^2) + (velocityY.^2));
T = (mn*Vavg)/(kb);



%Check X boundary conditions
[NH,IH] = max(newX);
[NL,IL] = min(newX);

upperX = newX > 200e-9;
newX(upperX)= newX(upperX)-200e-9;

lowX = newX < 0;
newX(lowX) = newX(lowX)+200e-9;



%Check Y boundary conditions
[NumH,IndexH] = max(newY);
[NumL,IndexL] = min(newY);

upperY = newY > 100e-9;
velocityY(upperY)= -velocityY(upperY);
```

```
    lowY = newY < 0;
    velocityY(lowY) = -velocityY(lowY);




    xv = [initialX newX];
    yv = [initialY newY];
%     Color =  colormap(hsv(100));




    figure(3)
    title('Confined Atoms');
    plot(xv.', yv.');
%     , BLX,BLY,'k',BHX,BHY,'k');
    hold on

    axis ([0 200e-9 0 100e-9])


    figure(4)
    title('Average Temperature (C)');
    plot(time, T, 'r.')
    hold on


    initialX = newX;
    initialY = newY;


    pause(0.001)
    time

end

hold off
```

APPENDIX C

Code for Part 3 graphs

```
clc
clear

mo = 9.11e-31;
mn = 0.26*mo;
kb = 1.381e-23;
T = 300;
Pscat = 1-exp(-1e-14/0.2e-12);
tauMN = 0;

%Initialize Bottleneck
BLX = [0.8e-7 0.8e-7; 0.8e-7 1.2e-7; 1.2e-7 1.2e-7];
BLY = [0 0.4e-7; 0.4e-7 0.4e-7; 0.4e-7 0];
BHX = [0.8e-7 0.8e-7; 0.8e-7 1.2e-7; 1.2e-7 1.2e-7];
BHY = [1e-7 0.6e-7; 0.6e-7 0.6e-7; 0.6e-7 1e-7];


%Initialise the particles
initialX = 200e-9*rand(100,1);
initialY = 0.2e-7.*rand(100,1) + 0.4e-7;
axis ([0 200e-9 0 100e-9])

%Initialise random angles
angleRad = 2*pi*rand(100,1);

%Set velocity
vth = sqrt((kb*T)/mn);

%Maxwell Boltzman Inital Velocity
MD1 = randn(100,1).*(vth/sqrt(2));
MD2 = randn(100,1).*(vth/sqrt(2));
MaxwellBoltzman = sqrt((MD1).^2+(MD2).^2);
initialRV = MaxwellBoltzman;

% figure(1)
% velocity = histogram(initialRV,20);

scat = 0;


velocityX = initialRV.*cos(angleRad);
velocityY = initialRV.*sin(angleRad);


for time = 0:1e-14: 1e-12
```

```
%Find new positions
newX = initialX + velocityX*1e-14;
newY = initialY + velocityY*1e-14;

%Check for Scatter
Escat = rand(100,1) < Pscat;
 if newX(Escat) > 0
    %Rethermalize
    MD1 = randn(100,1).*(vth/sqrt(2));
    MD2 = randn(100,1).*(vth/sqrt(2));
    MaxwellBoltzman = sqrt((MD1).^2+(MD2).^2);
    initialRV = MaxwellBoltzman;

    %Find New Velocities
    angleRad = 2*pi*rand(100,1);
    velocityX(Escat) = initialRV(Escat).*cos(angleRad(Escat));
    velocityY(Escat) = initialRV(Escat).*sin(angleRad(Escat));
    figure(1)
    velocity = histogram(initialRV,25);

    %Mean Free Path/Time Between Collisions

    scat = scat+ sum(Escat);
    tauMN = (time*100)/scat;
    Vavg = mean((velocityX.^2) + (velocityY.^2));

    MFP = tauMN*Vavg;


    figure(2)
    subplot(2,1,1)
    title('Mean Free Time')
    xlabel('Total Time in Sim (s)')
    ylabel('Mean Free Time (s)')
    plot(time, tauMN, 'b.')
    hold on

    figure(2)
    subplot(2,1,2)
    xlabel('Total Time in Sim (s)')
    ylabel('Mean Free PAth (m)')
    title('Mean Free Path')
    plot(time, MFP, 'g.')
    hold on

end



%Find temperature
Vavg = mean((velocityX.^2) + (velocityY.^2));
T = (mn*Vavg)/(kb);
```

```
%Check X boundary conditions
[NH,IH] = max(newX);
[NL,IL] = min(newX);

upperX = newX > 2e-7;
newX(upperX)= newX(upperX)-2e-7;

lowX = newX < 0;
newX(lowX) = newX(lowX)+200e-9;



%Check Y boundary conditions
[NumH,IndexH] = max(newY);
[NumL,IndexL] = min(newY);

upperY = newY > 100e-9;
velocityY(upperY)= -velocityY(upperY);

lowY = newY < 0;
velocityY(lowY) = -velocityY(lowY);



%Check Upper Box Conditions
%Left Condition
UpperBoxLeftX0 = initialX < 0.8e-7;
UpperBoxLeftX1 = newX > 0.8e-7;
UpperBoxLeftX = UpperBoxLeftX0>0 & UpperBoxLeftX1>0;

UpperBoxLeftY = newY > 0.6e-7;

bouncebackL = UpperBoxLeftX>0 & UpperBoxLeftY>0;
velocityX(bouncebackL) = -velocityX(bouncebackL);


%Center Condition
UpperBoxCenterX1 = newX > 0.8e-7 ;
UpperBoxCenterX2 = newX < 1.2e-7;
UpperBoxCenterX = UpperBoxCenterX1>0 & UpperBoxCenterX2>0;

UpperBoxCenterY0 = initialY < 0.6e-7;
UpperBoxCenterY1 = newY > 0.6e-7;
UpperBoxCenterY = UpperBoxCenterY0>0 & UpperBoxCenterY1>0;


bouncebackC = UpperBoxCenterX>0 & UpperBoxCenterY>0;
velocityY(bouncebackC) = -velocityY(bouncebackC);

%Right Condition
```

```matlab
UpperBoxRightX0 = initialX > 1.2e-7;
UpperBoxRightX1 = newX < 1.2e-7;
UpperBoxRightX = UpperBoxRightX0>0 & UpperBoxRightX1>0;

UpperBoxRightY = newY > 0.6e-7;

bouncebackR = UpperBoxRightX>0 & UpperBoxRightY>0;
velocityX(bouncebackR) = -velocityX(bouncebackR);







%Check Lower Box Conditions
%Left Condition
LowerBoxLeftX0 = initialX < 0.8e-7;
LowerBoxLeftX1 = newX > 0.8e-7;

LowerBoxLeftX = LowerBoxLeftX0>0 & LowerBoxLeftX1>0;
LowerBoxLeftY = newY < 0.4e-7;

bouncebackL = LowerBoxLeftX>0 & LowerBoxLeftY>0;
velocityX(bouncebackL) = -velocityX(bouncebackL);


%Center Condition
LowerBoxCenterX1 = newX > 0.8e-7 ;
LowerBoxCenterX2 = newX < 1.2e-7;
LowerBoxCenterX = LowerBoxCenterX1>0 & LowerBoxCenterX2>0;

LowerBoxCenterY0 = initialY > 0.4e-7;
LowerBoxCenterY1 = newY < 0.4e-7;
LowerBoxCenterY = LowerBoxCenterY0>0 & LowerBoxCenterY1>0;


bouncebackC = LowerBoxCenterX>0 & LowerBoxCenterY>0;
velocityY(bouncebackC) = -velocityY(bouncebackC);

%Right Condition
LowerBoxRightX0 = initialX > 1.2e-7;
LowerBoxRightX1 = newX < 1.2e-7;

LowerBoxRightX = LowerBoxRightX0>0 & LowerBoxRightX1>0;
LowerBoxRightY = newY < 0.4e-7;

bouncebackR = LowerBoxRightX>0 & LowerBoxRightY>0;
velocityX(bouncebackR) = -velocityX(bouncebackR);
```

```
    xv = [initialX newX];
    yv = [initialY newY];
%     Color =  colormap(hsv(100));





    figure(3)
    title('Confined Atoms');
    plot(xv.', yv.','.', BLX,BLY,'k',BHX,BHY,'k');
    hold on

    axis ([0 200e-9 0 100e-9])


    figure(4)
    title('Average Temperature (C)');
    plot(time, T, 'r.')
    hold on


    initialX = newX;
    initialY = newY;


    pause(0.001)
    time

end

hold off


%Electron Density at final positions
del = 0.05e-7;
nx = 40;
ny = 20;
ED = zeros(ny,nx);
Temp = zeros(ny,nx);

for i = 1:nx
    for j = 1:ny
        LowerBoxLeftX1 = newX < (i*del);
        LowerBoxLeftX2 = newX > (i*del - del);
        LowerBoxLeftX = LowerBoxLeftX1>0 & LowerBoxLeftX2>0;


        LowerBoxLeftY1 = newY < (j*del);
        LowerBoxLeftY2 = newY > (j*del - del);
        LowerBoxLeftY = LowerBoxLeftY1>0 & LowerBoxLeftY2>0;

        particles = LowerBoxLeftX>0 & LowerBoxLeftY>0;
        Xvelocity = velocityX(particles);
        Yvelocity = velocityY(particles);


        TotalElectrons = LowerBoxLeftX>0 & LowerBoxLeftY>0;
        Vavg = mean((Xvelocity.^2) + (Yvelocity.^2));


        Temp(j,i) = (mn*Vavg)/(kb);
        ED(j,i) = sum(TotalElectrons);

    end
end

figure(5)
title('Electron Density')
xlabel('X Distance In Plane (1square = 0.05e-7m')
ylabel('Y Distance In Plane (1square = 0.05e-7m')
pcolor(ED)




figure(6)
title('Temperature Map')
xlabel('X Distance In Plane (1square = 0.05e-7m')
ylabel('Y Distance In Plane (1square = 0.05e-7m')
pcolor(Temp)






done = string("DONE")
```