Mary Oji 101036761

Part 1: Electron Modelling(2000 electrons)

```
%Using Maxwell's principle of equipartition of energy and this is
 given by;
%
%
```

$$\overline{KE} = \frac{1}{2}kT = 2(\frac{1}{2}m\overline{v^2}) \Rightarrow \overline{v^2} = \frac{2kT}{m}$$

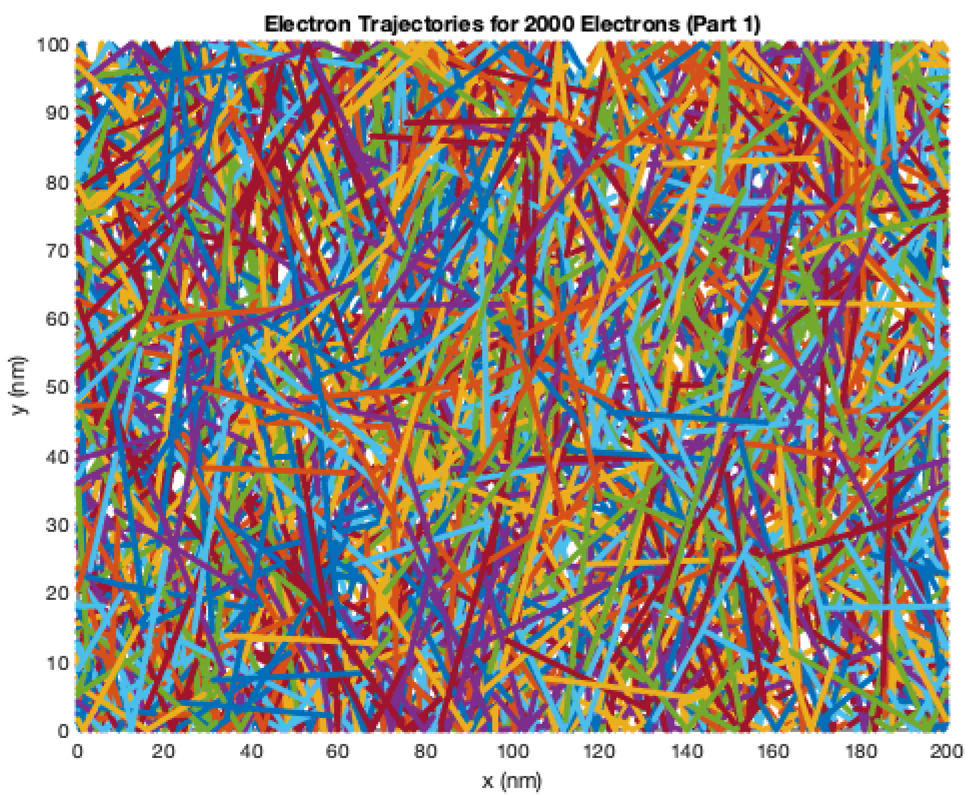
```
mo = 9.1e-31; %kg
mn = 0.26 * mo; %effective mass
T = 300; %K
k = 1.28e-23; %J/K??
vth = sqrt((2*k*T)/mn); %thermal velocity
tmn = 0.2e-12; %seconds(mean time between collisions)

%Mean of free path
meanFP = vth * tmn
```

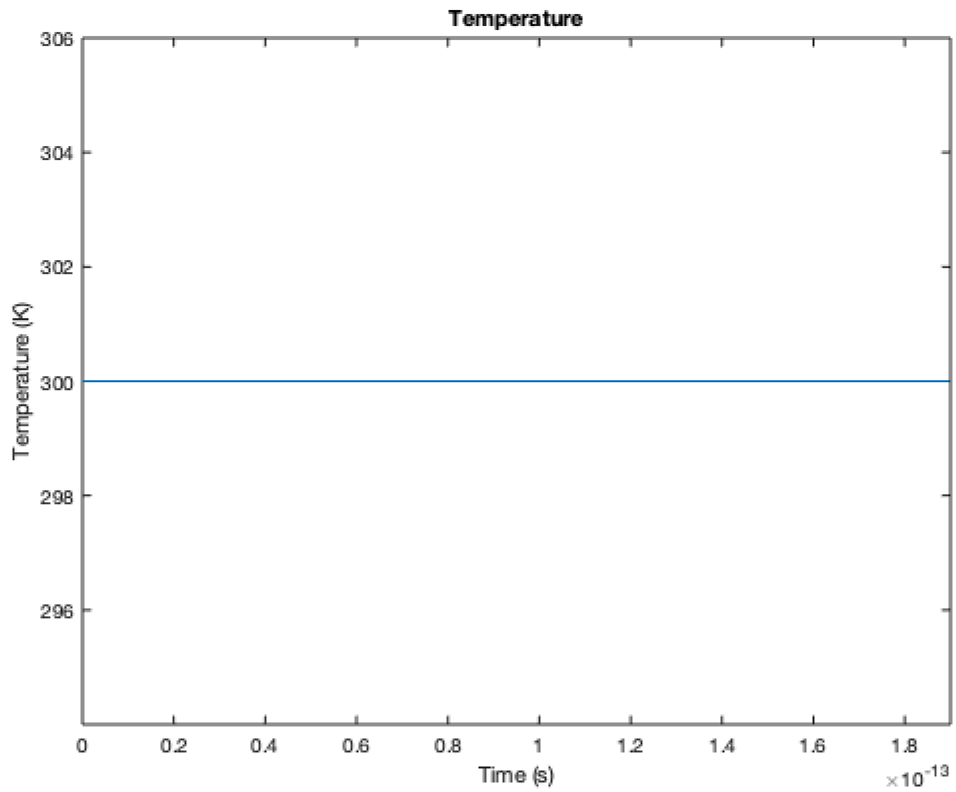
```
meanFP =

   3.6033e-08
```

```
%The mean free path is 36 nm

%normal size of region
l = 200e-9; %metres
w = 100e-9; %metres

%spacial step
t_step0 = 0.01 * 2e-14; %2e-14 ia the area of the region
%ideal spacial step
t_step = t_step0 - 0.1e-16; % smaller than 1/100 of region

%electrons
eplot = 2000;
loop = 1000;
movie = 0;

%Information is stored in arrays. This includes the position, velocity
 and temperature
%
pos = zeros(eplot,4);
traj = zeros(loop,eplot*2);
temp = zeros(loop,1);

%initial population
%
for i = 1:eplot
```

```
    ang = rand*2*pi;
    pos(i,:) = [l*rand w*rand vth*cos(ang) vth*sin(ang)];
end

%iterate and update position
%

for i = 1:loop
    pos(:,1:2) = pos(:,1:2) + t_step*pos(:,3:4);

    %side collision
    j = pos(:,1) > l;
    pos(j,1) = pos(j,1) - l;

    j = pos(:,1) < 0;
    pos(j,1) = pos(j,1) + l;

    %bottom and top colission
    j = pos(:,2) > w;
    pos(j,2) = 2*w - pos(j,2);
    pos(j,4) = -pos(j,4);

    j = pos(:,2) < 0;
    pos(j,2) = -pos(j,2);
    pos(j,4) = -pos(j,4);

    temp(i) = (sum(pos(:,3).^2) + sum(pos(:,4).^2))*mn/k/2/eplot;

    %trajectory
    for j = 1:eplot
        traj(i, (2*j):(2*j+1)) = pos(j, 1:2);
    end

    %update movie after some iterations
    if movie && mod(i,10) == 0
        figure(1);
        hold off;
        plot(pos(1:eplot,1)./1e-9, pos(1:eplot,2)./1e-9, 'o');
        axis([0 l/1e-9 0 w/1e-9]);
        title(sprintf('Trajectories for %d Electrons (Part 1)',...
        eplot));
        xlabel('x (nm)');
        ylabel('y (nm)');

        if i > 1
            figure (2);
            hold off;
            plot(t_step*(0:i-1), temp(1:i));
            axis([0 t_step*loop min(temp)*0.98 max(temp)*1.02]);
            title('Semiconductor Temperature');
            xlabel('Time (s)');
            ylabel('Temperature (K)');
        end
        pause(0.05);
```

```
        end
    end

     %trajectory after movie
     figure (1);
     title(sprintf('Electron Trajectories for %d Electrons (Part 1)',...
     eplot));
     xlabel('x (nm)');
     ylabel('y (nm)');
     axis([0 l/1e-9 0 w/1e-9]);
     hold on;

     for i=1:eplot
        plot(traj(:,i*2)./1e-9, traj(:,i*2+1)./1e-9, '.');
     end

     if (~movie)
         figure(2);
         hold off;
         plot(t_step*(0:loop-1), temp);
         axis([0 t_step*loop min(temp)*0.98 max(temp)*1.02]);
         title('Temperature');
         xlabel('Time (s)');
         ylabel('Temperature (K)');
     end
```





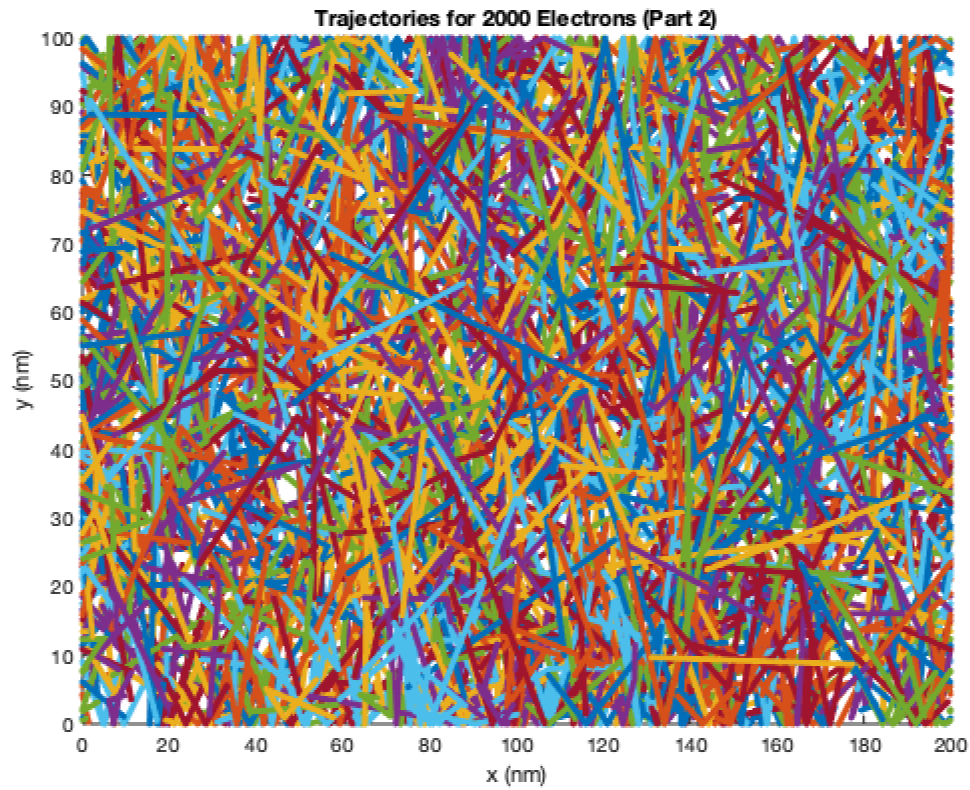
Part 2:Collisions with Mean Free Path

```
%Here, initial velocity is based on Maxwell-Boltzmann distribution.
%probability of scattering in a time step
%

p_Scat = 1 - exp(-t_step/tmn);

%Velecity in x and y is gaussian
%therefore the overall is a Maxwell-Boltzman distribution
v_o = makedist('Normal', 'mu', 0, 'sigma', sqrt(k*T/mn));

%initial polpulation
for i = 1:eplot
   ang = rand*2*pi;
   pos(i,:) = [l*rand w*rand random(v_o) random(v_o)];
end

%average velocity
avg_vel = sqrt(sum(pos(:,3).^2)/eplot + ...
    sum(pos(:,4).^2)/eplot)
```

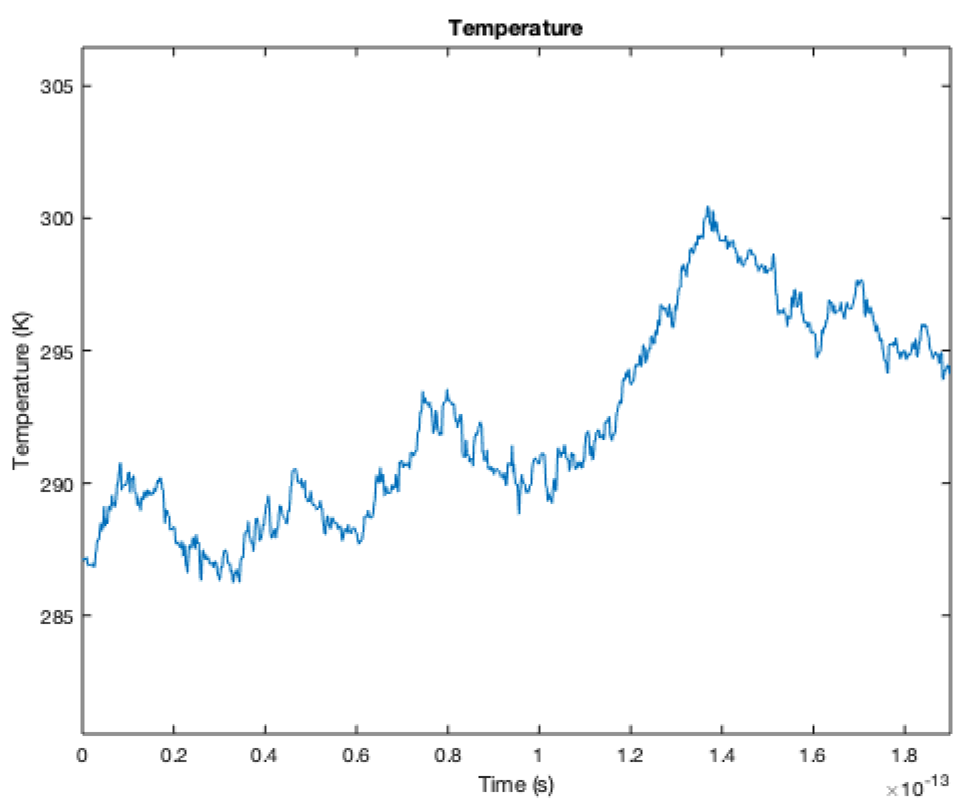
```
avg_vel =

   1.7627e+05
```

```
%The average velocity is given as 1.8027e+05 m/s

%iterate and update position
for i = 1:loop
  pos(:,1:2) = pos(:,1:2) + t_step.*pos(:,3:4);

    j = pos(:,1) > l;
    pos(j,1) = pos(j,1) - l;

    j = pos(:,1) < 0;
    pos(j,1) = pos(j,1) + l;

    j = pos(:,2) > w;
    pos(j,2) = 2*w - pos(j,2);
    pos(j,4) = -pos(j,4);

    j = pos(:,2) < 0;
    pos(j,2) = -pos(j,2);
    pos(j,4) = -pos(j,4);

    %scatter
    j = rand(eplot, 1) < p_Scat;
    pos(j,3:4) = random(v_o, [sum(j),2]);

    temp(i) = (sum(pos(:,3).^2) + sum(pos(:,4).^2))*mn/k/2/eplot;

    %Trajectory
    for j=1:eplot
        traj(i, (2*j):(2*j+1)) = pos(j, 1:2);
    end

    %update movie after some iterations
    if movie && mod(i,10) == 0
        figure(3);
        hold off;
        plot(pos(1:eplot,1)./1e-9, pos(1:eplot,2)./1e-9, 'o');
        axis([0 l/1e-9 0 w/1e-9]);
        title(sprintf('Trajectories for %d Electrons (Part 2)',...
        eplot));
        xlabel('x (nm)');
        ylabel('y (nm)');
        if i > 1
            figure (4);
            hold off;
            plot(t_step*(0:i-1), temp(1:i));
            axis([0 t_step*loop min(temp)*0.98 max(temp)*1.02]);
            title('Temperature');
            xlabel('Time (s)');
            ylabel('Temperature (K)');
        end

        % histogram
        figure (5);
        vel = sqrt(pos(:,3).^2 + pos(:,4).^2);
```

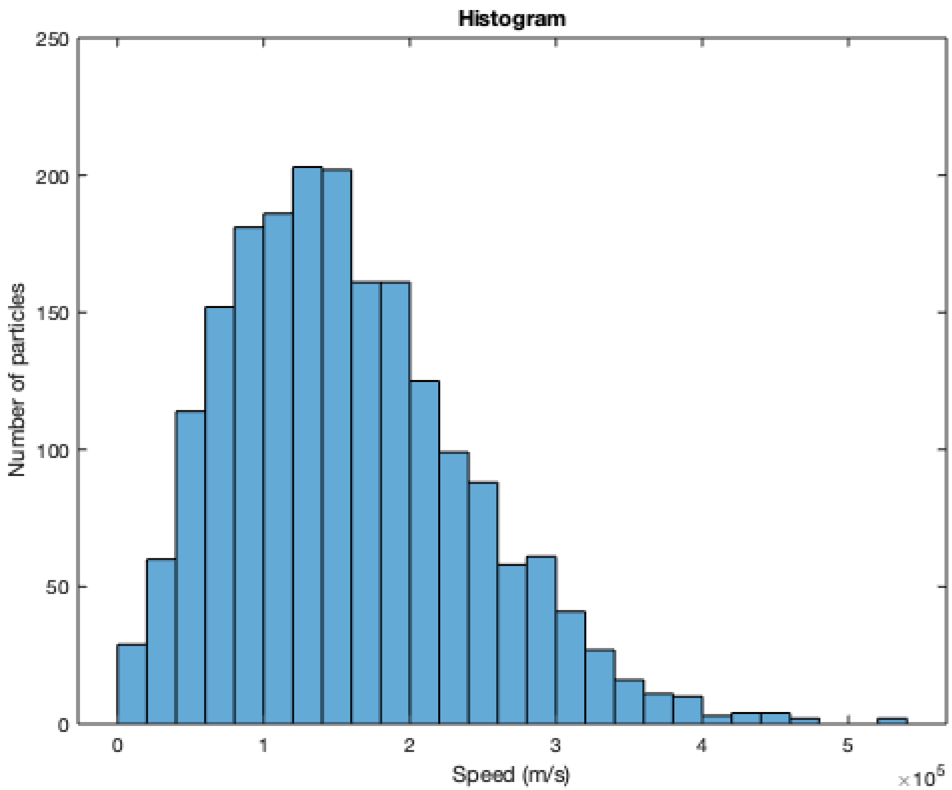
```
        title('Histogram of Electron Speeds');
        histogram(vel);
        xlabel('Speed (m/s)');
        ylabel('Number of particles');

        pause(0.05);
    end
end

%trajectory after movie
figure(3);
title(sprintf('Trajectories for %d Electrons (Part 2)',...
    eplot));
xlabel('x (nm)');
ylabel('y (nm)');
axis([0 l/1e-9 0 w/1e-9]);
hold on;

for i=1:eplot
    plot(traj(:,i*2)./1e-9, traj(:,i*2+1)./1e-9, '.');
end

figure (4);
hold off;
plot(t_step*(0:loop-1), temp);
axis([0 t_step*loop min(temp)*0.98 max(temp)*1.02]);
title('Temperature');
xlabel('Time (s)');
ylabel('Temperature (K)');

%histogram
figure (5);
vel = sqrt(pos(:,3).^2 + pos(:,4).^2);
title('Histogram of Electron Speeds');
histogram(vel);
title('Histogram');
xlabel('Speed (m/s)');
ylabel('Number of particles');
```



Trajectories for 2000 Electrons (Part 2)
y (nm)
x (nm)
0
10
20
30
40
50
60
70
80
90
100
0
20
40
60
80
100
120
140
160
180
200



Temperature
Temperature (K)
Time (s)
285
290
295
300
305
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
$\times 10^{-13}$



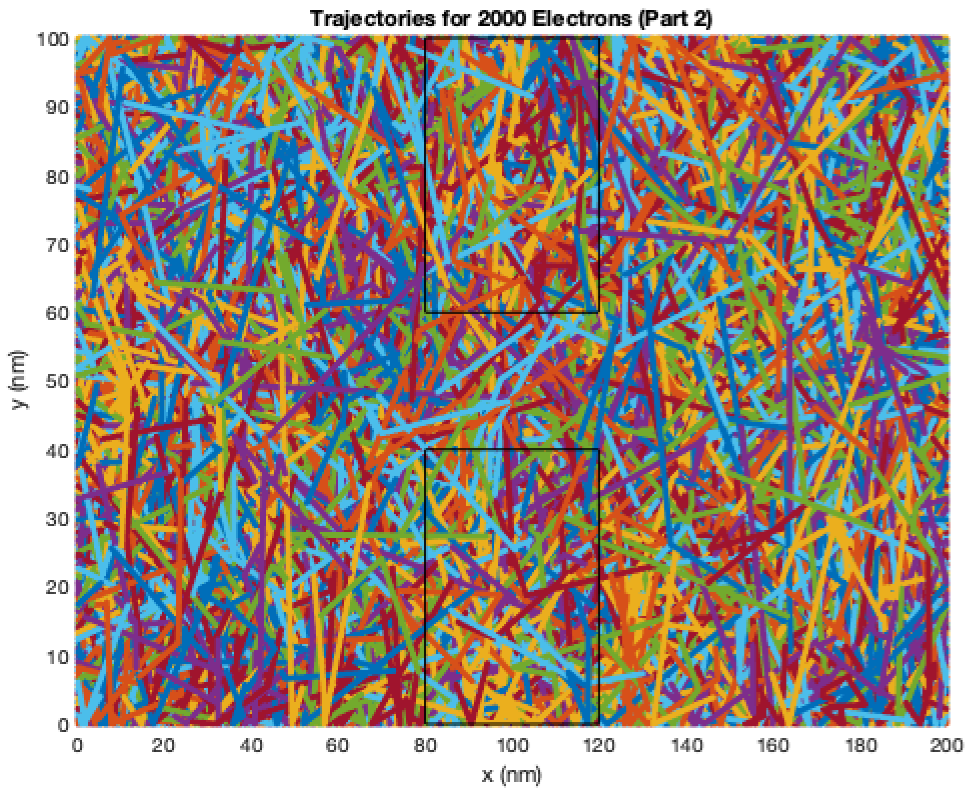
Part3: Enhancements

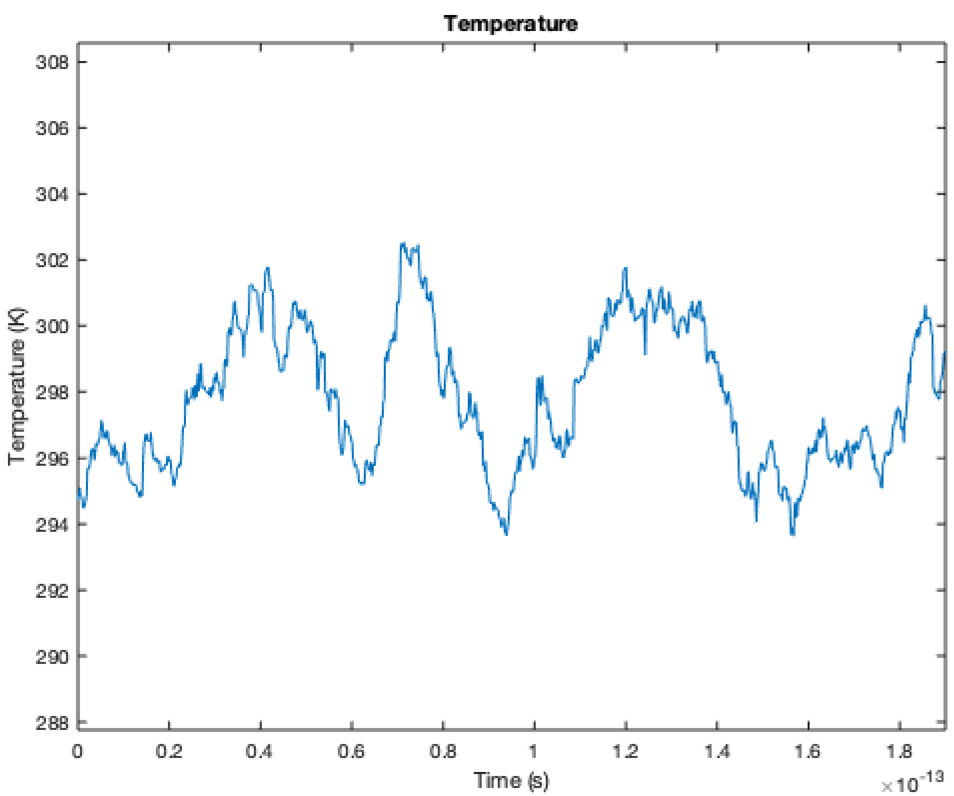
```
%Here, boundaries are either specular or diffusive and this simulation
%includes obstacles(boxes)

%Specular or diffusive boundaries
%diffusive = 0
%specular = 1
top = 0;
bottom = 0;
box = [0 1];
box1 = [80 120 0 40];
box2 = [80 120 60 100];
boxes = 1e-9 .* [box1; box2];

%initial population
for i = 1:eplot
    ang = rand*2*pi;
    pos(i,:) = [l*rand w*rand random(v_o) random(v_o)];

    %no particles start in a box
    %while(pos(i,1:2) == boxes)
    %    pos(i,1:2) = [l*rand w*rand];
    %end
end
```

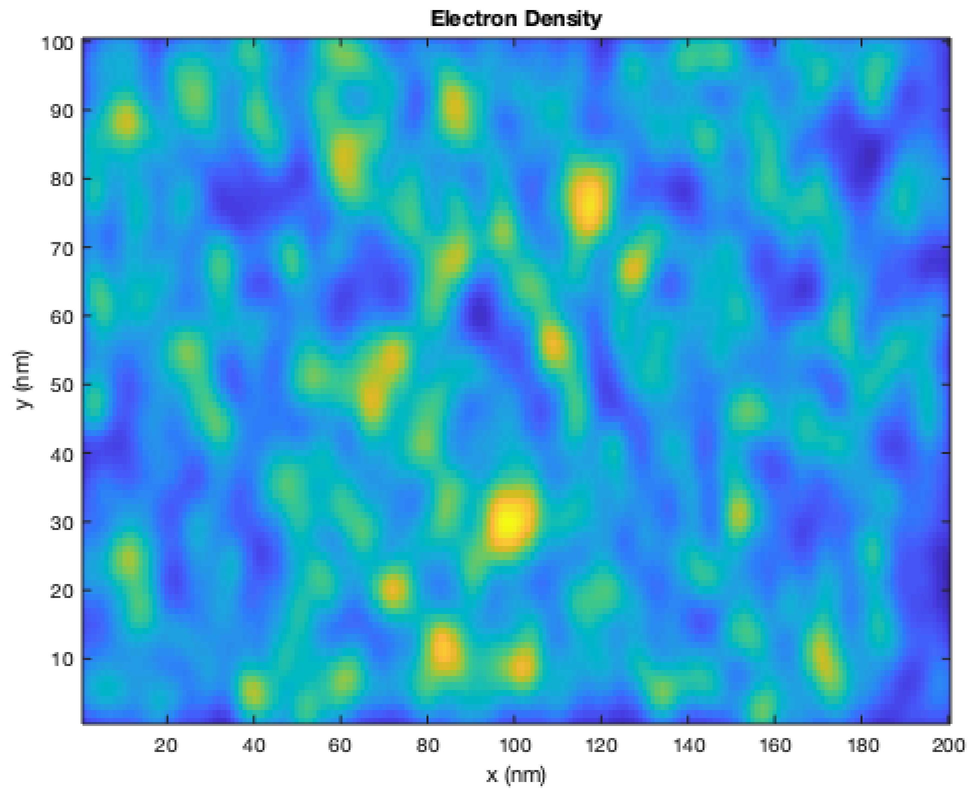
```
%third simulation
for i = 1:loop
    pos(:,1:2) = pos(:,1:2) + t_step.*pos(:,3:4);

    j = pos(:,1) > l;
    pos(j,1) = pos(j,1) - l;

    j = pos(:,1) < 0;
    pos(j,1) = pos(j,1) + l;

    j = pos(:,2) > w;

    if(top)
        pos(j,2) = 2*w - pos(j,2);
        pos(j,4) = -pos(j,4);
    else
        % Diffusive and electron bounce off at a random angle
        pos(j,2) = w;
        v = sqrt(pos(j,3).^2 + pos(j,4).^2);
        ang = rand([sum(j),1])*2*pi;
        pos(j,3) = v.*cos(ang);
        pos(j,4) = -abs(v.*sin(ang));
    end

    j = pos(:,2) < 0;

    if(bottom)
        pos(j,2) = -pos(j,2);
        pos(j,4) = -pos(j,4);
    else
        % Diffusive and electron bounce off at a random angle
        pos(j,2) = 0;
        v = sqrt(pos(j,3).^2 + pos(j,4).^2);
        ang = rand([sum(j),1])*2*pi;
        pos(j,3) = v.*cos(ang);
        pos(j,4) = abs(v.*sin(ang));
    end

    %scatter
    j = rand(eplot, 1) < p_Scat;
    pos(j,3:4) = random(v_o, [sum(j),2]);

    temp(i) = (sum(pos(:,3).^2) + sum(pos(:,4).^2))*mn/k/2/eplot;

    %Trajectory
    for j=1:eplot
        traj(i, (2*j):(2*j+1)) = pos(j, 1:2);
    end

    %update movie after some iterations
    if movie && mod(i,10) == 0
        figure(6);
        hold off;
        plot(pos(1:eplot,1)./1e-9, pos(1:eplot,2)./1e-9, 'o');
```

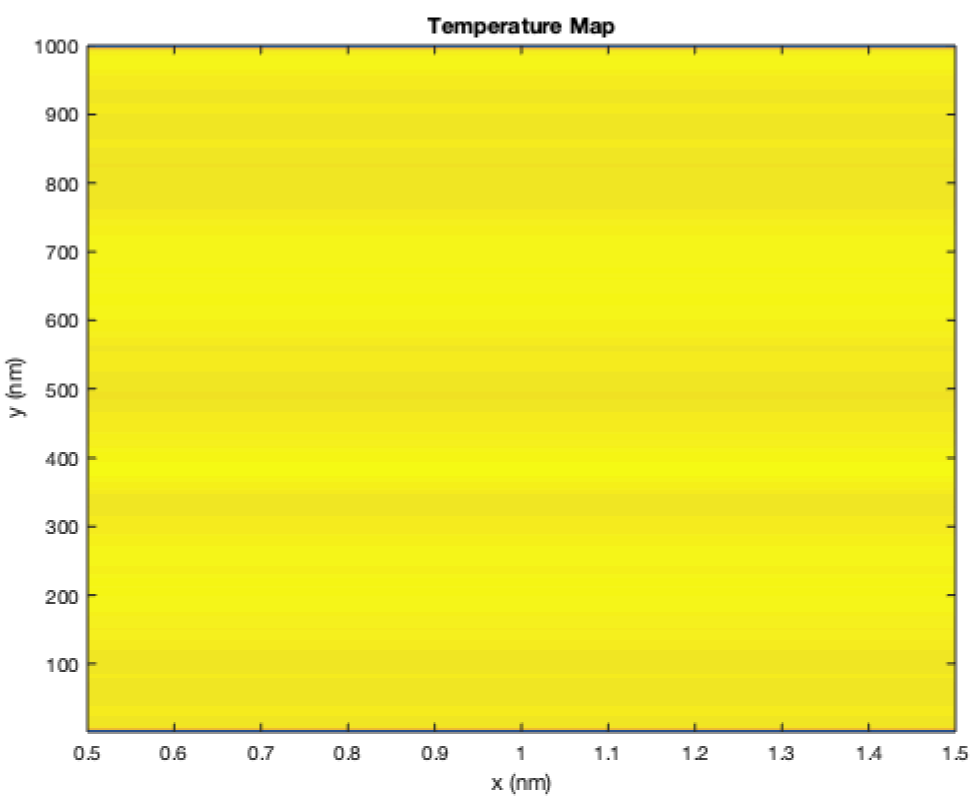
```
        %plot boxes
        for j=1:size(boxes,1)
           plot([boxes(j, 1) boxes(j, 1) boxes(j, 2) boxes(j, 2)
 boxes(j, 1)]./1e-9,...
               [boxes(j, 3) boxes(j, 4) boxes(j, 4) boxes(j, 3)
 boxes(j, 3)]./1e-9, 'k-');
        end

        axis([0 l/1e-9 0 w/1e-9]);
        title(sprintf('Trajectories for %d Electrons (Part 3)',...
        eplot));
        xlabel('x (nm)');
        ylabel('y (nm)');
        if i > 1
            figure (7);
            hold off;
            plot(t_step*(0:i-1), temp(1:i));
            axis([0 t_step*loop min(temp)*0.98 max(temp)*1.02]);
            title('Temperature');
            xlabel('Time (s)');
            ylabel('Temperature (K)');
        end
        pause(0.05);
    end
end

%trajectory after movie
figure(6);
title(sprintf('Trajectories for %d Electrons (Part 2)',...
    eplot));
xlabel('x (nm)');
ylabel('y (nm)');
axis([0 l/1e-9 0 w/1e-9]);
hold on;

for i=1:eplot
    plot(traj(:,i*2)./1e-9, traj(:,i*2+1)./1e-9, '.');
end

for j=1:size(boxes,1)
   plot([boxes(j, 1) boxes(j, 1) boxes(j, 2) boxes(j, 2) boxes(j,
 1)]./1e-9,...
       [boxes(j, 3) boxes(j, 4) boxes(j, 4) boxes(j, 3) boxes(j,
 3)]./1e-9, 'k-');
end

figure (7);
hold off;
plot(t_step*(0:loop-1), temp);
axis([0 t_step*loop min(temp)*0.98 max(temp)*1.02]);
title('Temperature');
xlabel('Time (s)');
ylabel('Temperature (K)');
```

```
%Electron density map using a histogram
density = hist3(pos(:,1:2),[200 100])';

% Smooth out the electron density map
N = 20;
sigma = 3;
[x, y]=meshgrid(round(-N/2):round(N/2), round(-N/2):round(N/2));
f=exp(-x.^2/(2*sigma^2)-y.^2/(2*sigma^2));
f=f./sum(f(:));
figure(8);
imagesc(conv2(density,f,'same'));
set(gca,'YDir','normal');
title('Electron Density');
xlabel('x (nm)');
ylabel('y (nm)');

%Temperature Map
N = 20;
sigma = 3;
[x y]=meshgrid(round(-N/2):round(N/2), round(-N/2):round(N/2));
f=exp(-x.^2/(2*sigma^2)-y.^2/(2*sigma^2));
f=f./sum(f(:));
figure(9);
imagesc(conv2(temp,f,'same'));
set(gca,'YDir','normal');
title('Temperature Map');
xlabel('x (nm)');
ylabel('y (nm)');
```


Trajectories for 2000 Electrons (Part 2)
y (nm)
x (nm)
0
20
40
60
80
100
120
140
160
180
200
10
30
50
70
90


Temperature
Temperature (K)
Time (s)
$\times 10^{-13}$
288
290
292
294
296
298
300
302
304
306
308
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8


Electron Density
y (nm)
x (nm)


Temperature Map
y (nm)
x (nm)

Published with MATLAB® R2019b