Elec 4700 Assignment 3

Table of Contents

Steven Cramp 101030294

Introduction:

The purpose of this Assignment is to investigate the effects of an electric feild on the trajectories of the electrons in a semiconductor. The assignment will first look at the effects of a constant electric field on the trajectory, then the effect of a changing electric feild.

Part 1:

This segment looks at the trajectories of electorns in a constant electric feild. A constant electric feild causes the trajectories of the electrons to curve due to the force that it applies. The code sagment below was used to calculate the the trajectories of the ellectrons with in the semiconductor as well as the current, dencity and temperatuer of the electrons in the semiconductor. The fallowing equation is used to calculate current that flows through the semiconductor. Where V is the volocity of the electon, q is the charge, is the electron dencity, and A is the cross section of the semiconductor in the y direction.

$$Jdrift = -V * N * q * A$$

The electric feild acting across the semiconductor can be calculated by dividing the applied voltage by the length of the semiconductor. Since the voltage is only applied in the x direction the y component of the electic field is zero. From this the force on the electons can be found by multipling the charge of the electon by the electric feild. The acceleration of the electrons can also be found by dividing the fouce on the electon by its effective mass in the semiconductor. These values can be seen below.

```
clc;clear;close all
global C

C.q_0 = 1.60217653e-19;             % electron charge
C.hb = 1.054571596e-34;             % Dirac constant
C.h = C.hb * 2 * pi;                % Planck constant
C.m_0 = 9.10938215e-31;             % electron mass
C.kb = 1.3806504e-23;               % Boltzmann constant
C.eps_0 = 8.854187817e-12;          % vacuum permittivity
C.mu_0 = 1.2566370614e-6;           % vacuum permeability
C.c = 299792458;                    % speed of light
C.g = 9.80665; %metres (32.1740 ft) per sÂ²
a3p1 =1;
```

```
a3p2 = 0;

partical_colission = 1;% part 2
boxes =0;%part 3
specular = 0 ;% part 3 one is on zero is off

T= 300;%k temperature
Eme= 0.26* C.m_0;% kg effective mass of electron
Tmn = 0.2e-12;%s mean time between colissions
vth = sqrt(2*C.kb*T/Eme);% m/s thermal volocity
MFP = vth*Tmn; %m
m= 500;% length of sim
dt = 5e-15;% time step
N = 10000; %number of electons
dimx =200e-9;%m
dimy =100e-9;%m
c =zeros(1,N);

Fx=0;
Fy=0;

% create boxes
if (boxes)
    box1 =
 [dimx-125e-9,dimx-75e-9,dimx-75e-9,dimx-125e-9,dimx-125e-9;dimy,dimy,
 dimy-40e-9,dimy-40e-9,dimy];
    box2 =
 [dimx-125e-9,dimx-75e-9,dimx-75e-9,dimx-125e-9,dimx-125e-9;40e-9,40e-9,
 0,0,40e-9];
else
    box1 =[0 0 0 0 0 ; 0 0 0 0 0];
    box2 =[0 0 0 0 0 ; 0 0 0 0 0];
end
% initiates points and ensures that they donot spawn ouside the
 boundries
if ~boxes
    xpos = (randi((dimx*1e9)+1,1,N)-1)/1e9;% m electron position x
    ypos = (randi((dimy*1e9)+1,1,N)-1)/1e9;% m electron position y
    xpos(xpos ==dimx)=xpos(xpos ==dimx)-1e-9;
    xpos(xpos ==0)=xpos(xpos ==0)+1e-9;
    ypos(ypos ==dimy)=ypos(ypos ==dimy)-1e-9;
    ypos(ypos ==0)=ypos(ypos ==0)+1e-9;
elseif(boxes)
    for l =1:N
        xpos(1,l) =(randi((dimx*1e9)+1,1,1)-1)/1e9 ;
        ypos(1,l) =(randi((dimy*1e9)+1,1,1)-1)/1e9;
        while (ypos(1,l)<=0|ypos(1,l)>=dimy|
xpos(1,l)>=box1(1,1)-1e-9&xpos(1,l)<=box1(1,2)+1e-9&(ypos(1,l)<=box2(2,1)+1e-9|
ypos(1,l)>=box1(2,3)-1e-9))
            xpos(1,l) =(randi((dimx*1e9)+1,1,1)-1)/1e9 ;
            ypos(1,l) =(randi((dimy*1e9)+1,1,1)-1)/1e9;
        end
    end
end
```

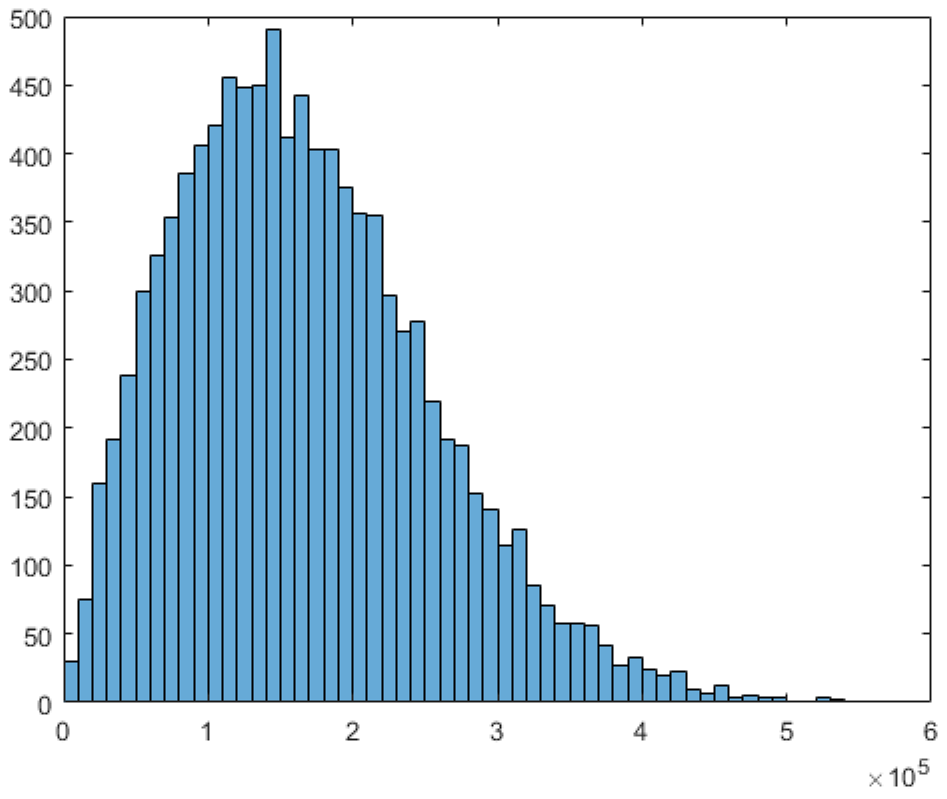
```
% while(sum ((xpos>=box1(1,1)&xpos<= box1(1,1)& ypos>= box1(2,3))|
(xpos>=box2(1,1)&xpos<= box2(1,1)& ypos>= box2(2,3)))>=1)
%     xpos = (randi((dimx*1e9)+1,1,N)-1)/1e9;% m electron position x
%     ypos = (randi((dimy*1e9)+1,1,N)-1)/1e9;% m electron position y
% end

vx   = zeros(1,N);%m/s velocity in x
vy   = zeros(1,N);%m/s velocity in y
colision_count = zeros(1,N);
dtraveled=zeros(1,N);
colourL = ["R", "G", "B","LB", "P", "y", "BLk", "DB", "O","M","GG"];
colour = [[1 0 0];[0 1 0];[0 0 1];[0 1 1];[1 0 1];[1 1 0];[0 0 0];
[0 0.447 0.741];[0.85 0.325 0.098];[0.929 0.694 0.125];[0.466 0.674
 0.188]];
Temp = T;

 dx2 =1e-9;
    dy2=1e-9;
    nx2 =round( dimx/dx2);
    ny2 = round(dimy/dy2);
    Ex=zeros(ny2,nx2);
    Ey=zeros(ny2,nx2);
% initiated the partical velocities
if ~partical_colission
    angle = rand(1,N);
    vx = vth* cos(angle*2*pi);
    vy = vth* sin(angle*2*pi);
else
    vx =randn(1,N)*sqrt(C.kb*T/Eme);
    vy=randn(1,N)*sqrt(C.kb*T/Eme);
end
v = sqrt(vx.^2+vy.^2);



if boxes
    figure (2)
    subplot(2,1,1);
    plot (box1(1,:),box1(2,:),'-k')
    hold on
    plot (box2(1,:),box2(2,:),'-k')
end
if partical_colission
    figure
    histogram(v)
    p =1-exp(-dt/Tmn);
else
    p=0;
end


if a3p1 ==1

    econcentration =1e15;
    J_driftx = mean(vx)*econcentration*C.q_0*dimy*100;
    Vapplied =0.1;
    Ex = Vapplied/dimx
    Ey =0
    Fx = (-C.q_0)*Ex
    Fy = (-C.q_0)*Ey
    AccelerationX =(Fx./(Eme))
    AccelerationY =(Fy./(Eme))
end
if a3p2 == 1
     econcentration =1e15;
   boundy = [-1 nan 0 nan];
   J_driftx = mean(vx)*econcentration*C.q_0*dimy*100;
    [Ex, Ey]= calcEfeild (dimx,dimy,dx2,dy2,boundy,1, box1, box2);

end

for l=1:m
    if a3p2 ==1
    lincordx = floor(xpos(1,:)/dx2)+1;
    lincordy = floor(ypos(1,:)/dy2)+1;
    lincordx(lincordx==round(dimx/dx2))=1;
    lincordy(lincordy==round(dimy/dy2))=1;
    for k = 1:N
    ExatPos=
 (Ex(lincordy(k),lincordx(k))+Ex(lincordy(k)+1,lincordx(k))+Ex(lincordy(k),lincord
    EyatPos=
 (Ey(lincordy(k),lincordx(k))+Ey(lincordy(k)+1,lincordx(k))+Ey(lincordy(k),lincord
    end

        Fx = (C.q_0)*ExatPos(1,:);
        Fy = (C.q_0)*EyatPos(1,:);

    end

    %updates position
    vx = vx+(Fx./(Eme))*dt;
    vy =vy+(Fy./(Eme))*dt;
    v = sqrt(vx.^2+vy.^2);
    J_driftx =[J_driftx, -C.q_0*econcentration *mean(vx)*dimy*100];
    xpos=[xpos;xpos(l,:)+(vx)*dt];
    ypos=[ypos;ypos(l,:)+(vy)*dt];
    % fineds the distance traveled by each partical
    dtraveled = dtraveled + sqrt ((xpos(l,:)-xpos(l
+1,:)).^2+(ypos(l,:)-ypos(l+1,:)).^2);
    slope =(vy./vx);
    % sets up colision detection by determining if the particals have
 the
    % distance to the edges

    dtt = sqrt(((dimy -ypos(l+1,:)).^2)+((((dimy-ypos(l+1,:))./
slope)).^2));
    dtb = sqrt(((0 -ypos(l+1,:)).^2)+((((-ypos(l+1,:))./slope)).^2));
    if(boxes)
```

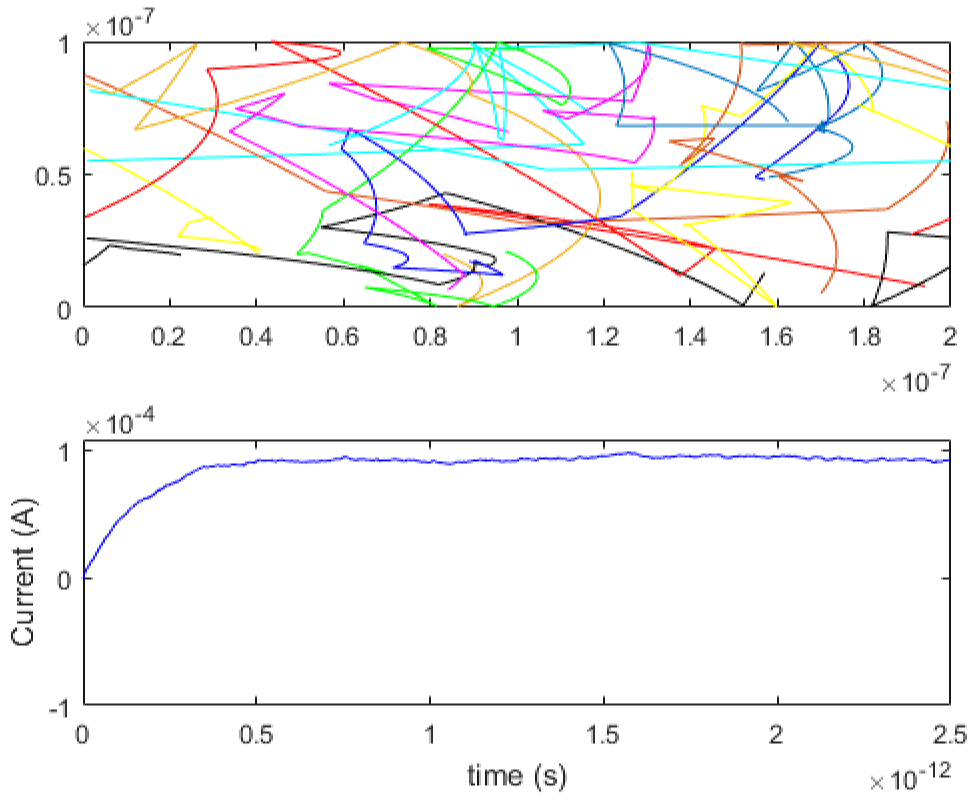
```
        dttbf =((xpos(l+1,:)>=box1(1,1)&xpos(l+1,:)<=box1(1,2))).*
 sqrt(((box1(2,3) -ypos(l+1,:)).^2)+((((box1(2,3)-ypos(l+1,:))./
slope)).^2))+~((xpos(l+1,:)>=box1(1,1)&xpos(l+1,:)<=box1(1,2))).*100;
        dtbbf = ((xpos(l+1,:)>=box1(1,1)&xpos(l
+1,:)<=box1(1,2))).*sqrt(((box2(2,1) -ypos(l+1,:)).^2)+((((box2(2,1)-
ypos(l+1,:))./slope)).^2))+~((xpos(l+1,:)>=box1(1,1)&xpos(l
+1,:)<=box1(1,2))).*100;
        dts1 =( ypos(l+1,:)<=box2(2,1)|ypos(l+1,:)>=box1(2,3)).*
 sqrt((slope.*(box1(1,1)-xpos(l+1,:))).^2+(box1(1,1)-xpos(l+1,:)).^2)
 +~( ypos(l+1,:)<=box2(2,1)|ypos(l+1,:)>=box1(2,3)).*100;
        dts2=( ypos(l+1,:)<=box2(2,1)|ypos(l+1,:)>=box1(2,3)).*
 sqrt((slope.*(box1(1,2)-xpos(l+1,:))).^2+(box1(1,2)-xpos(l+1,:)).^2)
 +~( ypos(l+1,:)<=box2(2,1)|ypos(l+1,:)>=box1(2,3)).*100;
    else
        dttbf =ones(1,N).*100;
        dtbbf = ones(1,N).*100;
        dts1 =ones(1,N).*100;
        dts2=ones(1,N).*100;

    end
    %counts the number of colissions that have occured and
    pc =c;
    c=(((dts1<1e-9|dts2<1e-9)|((dtt<1e-9|dtb<1e-9|dttbf<1e-9|
dtbbf<1e-9))));
    colision_count = colision_count+(((dts1<1e-9|dts2<1e-9)|
((dtt<1e-9|dtb<1e-9|dttbf<1e-9|dtbbf<1e-9))));
    if specular
        % basic colission part one
        vy= -((dtt<1e-9|dtb<1e-9|dttbf<1e-9|dtbbf<1e-9).*2-1).*vy;
        vx= -((dts1<1e-9|dts2<1e-9).*2-1).*vx;
    else
        % re thermalized velocities for part 3
        %if rethermalized volocity is in the same direection as
 previouse
        %than flip signs
        signx=sign(vx);
        signy=sign(vy);



        vx= ((((dts1<1e-9|dts2<1e-9)|((dtt<1e-9|dtb<1e-9|dttbf<1e-9|
dtbbf<1e-9))))&~pc).*(randn(1,N)*sqrt(C.kb*T/Eme)) +(~((((dts1<1e-9|
dts2<1e-9))|((dtt<1e-9|dtb<1e-9|dttbf<1e-9|dtbbf<1e-9)))&~pc)).*vx;
        vy= ((((dts1<1e-9|dts2<1e-9)|((dtt<1e-9|dtb<1e-9|dttbf<1e-9|
dtbbf<1e-9))))&~pc).*(randn(1,N)*sqrt(C.kb*T/Eme)) +(~((((dts1<1e-9|
dts2<1e-9))|((dtt<1e-9|dtb<1e-9|dttbf<1e-9|dtbbf<1e-9)))&~pc)).*vy;
        vx = (((dts1<1e-9&sign(vx)==1)|(dts2<1e-9
 &sign(vx)==-1))&~pc).*-1.*vx+(~(((dts1<1e-9&sign(vx)==1)|(dts2<1e-9
 &sign(vx)==-1))&~pc)).*vx;
        vy = (((((dtt<1e-9&sign(vy)==1)|(dtb<1e-9&sign(vy)==-1)|
(dttbf<1e-9&sign(vy)==1)|(dtbbf<1e-9&sign(vy)==-1)))&~pc)).*-1.*vy
+(~(((((dtt<1e-9&sign(vy)==1)|(dtb<1e-9&sign(vy)==-1)|
(dttbf<1e-9&sign(vy)==1)|(dtbbf<1e-9&sign(vy)==-1)))&~pc))).*vy;
```

```
    end
    % loop condition for end boundries
    xpos(l+1,:) = (xpos(l+1,:)>dimx).*0+(xpos(l+1,:)<0).*dimx+~(xpos(l
+1,:)>=dimx|xpos(l+1,:)<=0).*xpos(l+1,:);
    % colisions with other particals are only alouwed when partical is
 away
    % from the edges and has not colided with an edge Part 2 and 3
    colision = p>rand(1,N)&~(((dts1<30e-9|dts2<30e-9)|((dtt<30e-9|
dtb<10e-9|dttbf<30e-9|dtbbf<30e-9))))&~c;
    colision_count = colision_count+colision;
    vy=colision.*(randn(1,N)*sqrt(C.kb*T/Eme))+(~colision).*vy;
    vx=colision.*(randn(1,N)*sqrt(C.kb*T/Eme))+(~colision).*vx;



    % skips the plot of the x boundry transition
    skip = (xpos(l+1,:)>=dimx|xpos(l+1,:)<=0);
    % progress = (l/m)*100
    v = sqrt(vx.^2+vy.^2);
    c=0;
    %finds the current temperature
    Temp =[Temp, mean((v.^2)*Eme/(2*C.kb))];
    % plots the electrons
    figure (2)
    for k =1:10
        if skip(k)==0
            subplot(2,1,1);
            plot([xpos(l,k),xpos(l+1,k)],[ypos(l,k),ypos(l
+1,k)],'-','color',colour(k,:))
        end
        xlim([0,dimx])
        ylim([0,dimy])
        hold on
        %quiver(xpos(:,k),ypos(:,k),vx+xpos(:,k),vy+ypos(:,k),0.0001)
        drawnow limitrate

    end
    subplot(2,1,2)
    plot([dt*(l-1),dt*(l)],J_driftx(l:l+1),'b-')
    hold on
    xlabel('time (s)')
    ylabel('Current (A)')

    xlim([0,m*dt])
    ylim([-1e-4,max(J_driftx)+10e-6])
end
%finds the mean free path and mean time between colission for part 2
 and
%plots a final velocity hystegram
% if (partical_colission)
%     colision_count(colision_count<=0) = 1;
%     MFP2 = mean (dtraveled./(colision_count));
%     Tmn2= mean( m*dt./(colision_count));
% %     figure(3)
% %     histogram(v)
% %     xlabel('volocity (m/s)')
% %     ylabel('probablility')
% end
% %findes the dencity and temperature of the electrons  part 3
% squarcount= zeros(round(dimy/1e-9),round(dimx/1e-9));
% temps= zeros(round(dimy/1e-9),round(dimx/1e-9));
% for k =1:N
%
%     yindex = min((round(dimy/1e-9))-ceil(ypos(m
+1,k)/1e-9)+1,round(dimy/1e-9));
%     xindex = min(ceil(xpos(m+1,k)/1e-9)+1,round(dimx/1e-9));
%     % excludes particals that violate boundry conditions.
%     if ((yindex>0&xindex>0) & ~((((ypos(m+1,k)>=box1(2,3))|(ypos(m
+1,k)<=box2(2,1))))&(xpos(m+1,k)>= box1(1,1)&xpos(m+1,k)<=box1(1,2))))
%         squarcount(yindex,xindex) =squarcount(yindex,xindex) +1;
%         temps(yindex,xindex)=((v(k))^2)*Eme/
(2*C.kb)+temps(yindex,xindex);
%     end
% end
% temps=temps./squarcount;
% squarcount =squarcount./((1e-9)*(1e-9));

% figure
% subplot(1,2,1)
% bar3(squarcount)
% xlabel('x position (nm)')
% ylabel('y position (nm)')
% zlabel('Density of Electrons (electron/m)')
% subplot(1,2,2)
%
% bar3(temps)
% xlabel('x position (nm)')
% ylabel('y position (nm)')
% zlabel('Temperature (k)')
```

```
Ex =

   5.0000e+05


Ey =

     0


Fx =

  -8.0109e-14
```

```
Fy =

     0


AccelerationX =

  -3.3823e+17


AccelerationY =

     0
```





The plots above show trajectories of the electrons in the feild and the curret through the semiconductor. From the current plot it can be seen that as the volocity of the electrons increases due to the acceleration of the electric field the current through the feild also increases to a maximum. This is due to the limiting effects that the colissions have on the volocity of the electons. The current in this case is positive as it shows the conventional current as aposed to the electron flow. Positive current is represents a conventional flow to the left.

Part 2:

The Code below was used to calculate the trajectories on electons in a varying electric feild. The code below first calculates the electric feild across the semiconductor, then it uses the feild to plot the trajectories of the electrons.

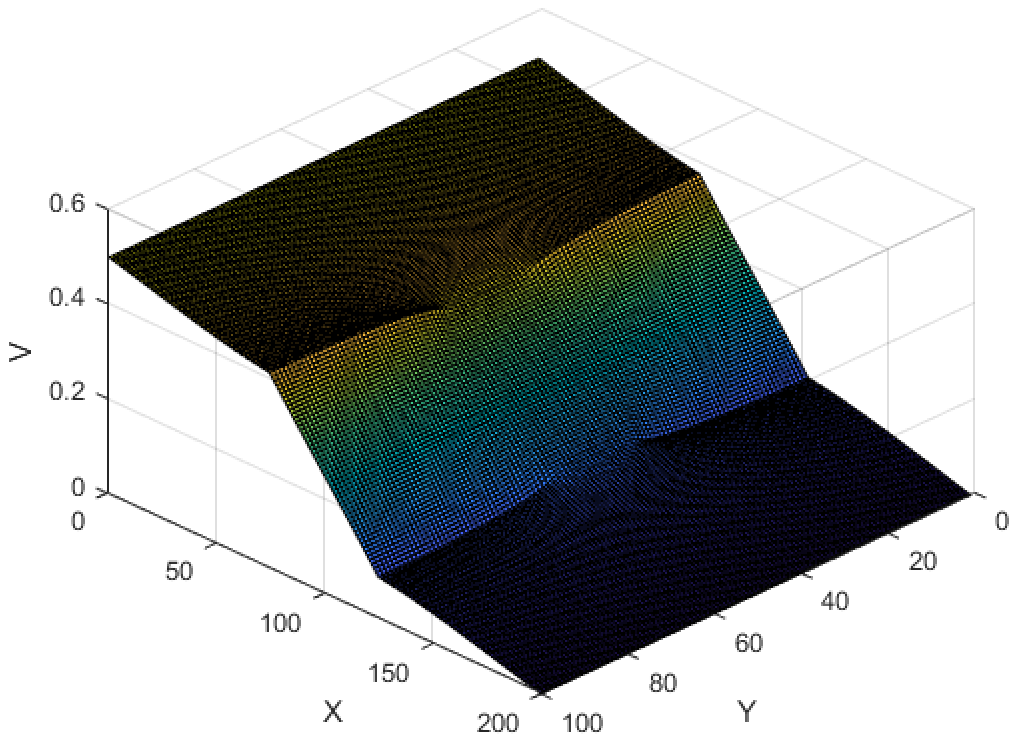
```
 clc;clear;close all


C.q_0 = 1.60217653e-19;             % electron charge
C.hb = 1.054571596e-34;             % Dirac constant
C.h = C.hb * 2 * pi;                % Planck constant
C.m_0 = 9.10938215e-31;             % electron mass
C.kb = 1.3806504e-23;               % Boltzmann constant
C.eps_0 = 8.854187817e-12;          % vacuum permittivity
C.mu_0 = 1.2566370614e-6;           % vacuum permeability
C.c = 299792458;                    % speed of light
C.g = 9.80665; %metres (32.1740 ft) per sÂ²
a3p1 =0;
```

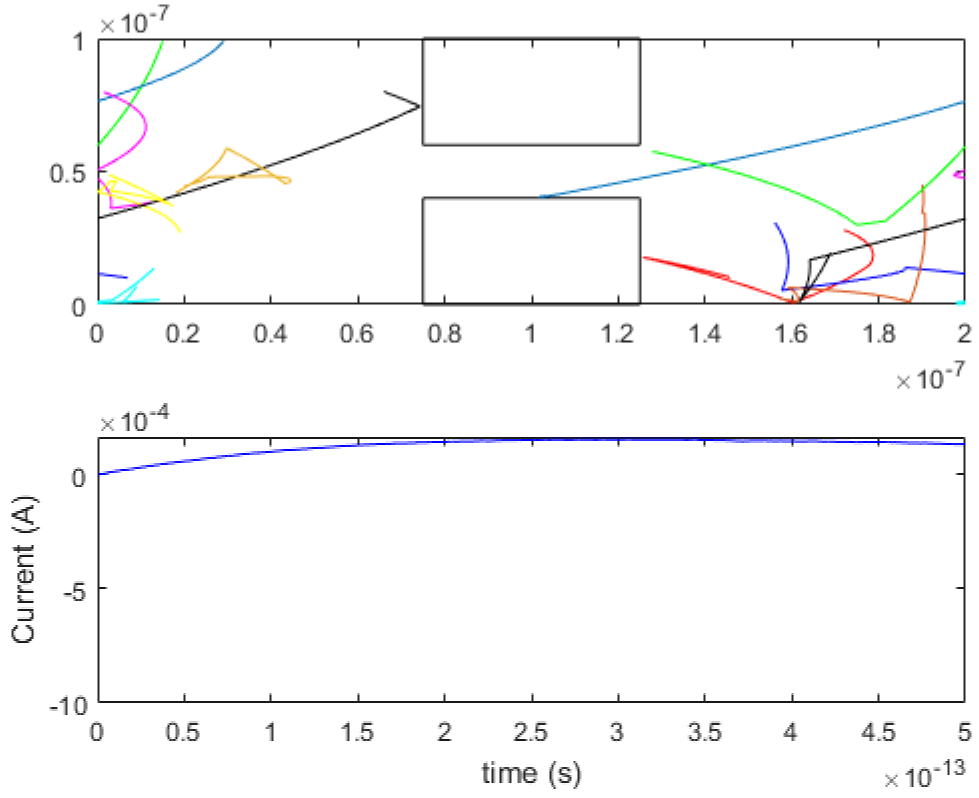
```
a3p2 = 1;

partical_colission = 1;% part 2
boxes =1;%part 3
specular = 0 ;% part 3 one is on zero is off

T= 300;%k temperature
Eme= 0.26* C.m_0;% kg effective mass of electron
Tmn = 0.2e-12;%s mean time between colissions
vth = sqrt(2*C.kb*T/Eme);% m/s thermal volocity
MFP = vth*Tmn; %m
m= 500;% length of sim
dt = 1e-15;% time step
N = 10000; %number of electons
dimx =200e-9;%m
dimy =100e-9;%m
c =zeros(1,N);

Fx=0;
Fy=0;

% create boxes
if (boxes)
    box1 =
 [dimx-125e-9,dimx-75e-9,dimx-75e-9,dimx-125e-9,dimx-125e-9;dimy,dimy,
 dimy-40e-9,dimy-40e-9,dimy];
    box2 =
 [dimx-125e-9,dimx-75e-9,dimx-75e-9,dimx-125e-9,dimx-125e-9;40e-9,40e-9,
 0,0,40e-9];
else
    box1 =[0 0 0 0 0 ; 0 0 0 0 0];
    box2 =[0 0 0 0 0 ; 0 0 0 0 0];
end
% initiates points and ensures that they donot spawn ouside the
 boundries
if ~boxes
    xpos = (randi((dimx*1e9)+1,1,N)-1)/1e9;% m electron position x
    ypos = (randi((dimy*1e9)+1,1,N)-1)/1e9;% m electron position y
    xpos(xpos ==dimx)=xpos(xpos ==dimx)-1e-9;
    xpos(xpos ==0)=xpos(xpos ==0)+1e-9;
    ypos(ypos ==dimy)=ypos(ypos ==dimy)-1e-9;
    ypos(ypos ==0)=ypos(ypos ==0)+1e-9;
elseif(boxes)
    for l =1:N
        xpos(1,l) =(randi((dimx*1e9)+1,1,1)-1)/1e9 ;
        ypos(1,l) =(randi((dimy*1e9)+1,1,1)-1)/1e9;
        while (ypos(1,l)<=0|ypos(1,l)>=dimy|
xpos(1,l)>=box1(1,1)-1e-9&xpos(1,l)<=box1(1,2)+1e-9&(ypos(1,l)<=box2(2,1)+1e-9|
ypos(1,l)>=box1(2,3)-1e-9))
            xpos(1,l) =(randi((dimx*1e9)+1,1,1)-1)/1e9 ;
            ypos(1,l) =(randi((dimy*1e9)+1,1,1)-1)/1e9;
        end
    end
end
```

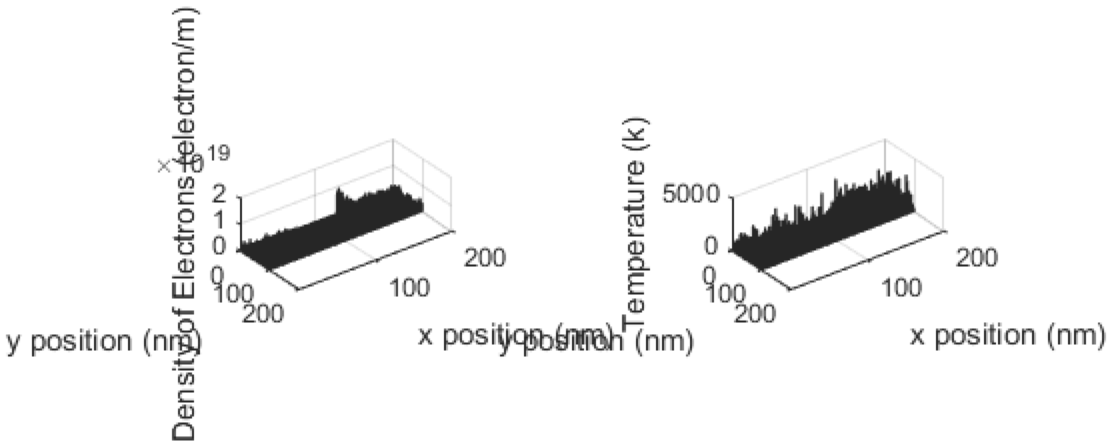
```
% while(sum ((xpos>=box1(1,1)&xpos<= box1(1,1)& ypos>= box1(2,3))|
(xpos>=box2(1,1)&xpos<= box2(1,1)& ypos>= box2(2,3)))>=1)
%     xpos = (randi((dimx*1e9)+1,1,N)-1)/1e9;% m electron position x
%     ypos = (randi((dimy*1e9)+1,1,N)-1)/1e9;% m electron position y
% end

vx   = zeros(1,N);%m/s velocity in x
vy   = zeros(1,N);%m/s velocity in y
colision_count = zeros(1,N);
dtraveled=zeros(1,N);
colourL = ["R", "G", "B","LB", "P", "y", "BLk", "DB", "O","M","GG"];
colour = [[1 0 0];[0 1 0];[0 0 1];[0 1 1];[1 0 1];[1 1 0];[0 0 0];
[0 0.447 0.741];[0.85 0.325 0.098];[0.929 0.694 0.125];[0.466 0.674
 0.188]];
Temp = T;

 dx2 =1e-9;
    dy2=1e-9;
    nx2 =round( dimx/dx2);
    ny2 = round(dimy/dy2);
    Ex=zeros(ny2,nx2);
    Ey=zeros(ny2,nx2);
% initiated the partical velocities
if ~partical_colission
    angle = rand(1,N);
    vx = vth* cos(angle*2*pi);
    vy = vth* sin(angle*2*pi);
else
    vx =randn(1,N)*sqrt(C.kb*T/Eme);
    vy=randn(1,N)*sqrt(C.kb*T/Eme);
end
v = sqrt(vx.^2+vy.^2);



if boxes
    figure (2)
    subplot(2,1,1);
    plot (box1(1,:),box1(2,:),'-k')
    hold on
    plot (box2(1,:),box2(2,:),'-k')
end
if partical_colission
%     figure
%     histogram(v)
    p =1-exp(-dt/Tmn);
else
    p=0;
end


if a3p1 ==1

    econcentration =1e15;
    J_driftx = mean(vx)*econcentration*(-C.q_0)*dimy*100;
    Vapplied =0.1;
    Ex = -Vapplied/dimx
    Ey =0
    Fx = (-C.q_0)*Ex
    Fy = (-C.q_0)*Ey
    AccelerationX =(Fx./(Eme))
    AccelerationY =(Fy./(Eme))
end
if a3p2 == 1
     econcentration =1e15;
   boundy = [0.5 nan 0 nan];
   J_driftx = mean(vx)*econcentration*(-C.q_0)*dimy*100;
    [Ex, Ey]= calcEfeild (dimx,dimy,dx2,dy2,boundy,1, box1, box2);

end

for l=1:m
    if a3p2 ==1
    lincordx = floor(xpos(1,:)/dx2)+1;
    lincordy = floor(ypos(1,:)/dy2)+1;
    lincordx(lincordx==round(dimx/dx2))=1;
    lincordy(lincordy==round(dimy/dy2))=1;
    for k = 1:N
    ExatPos=
 (Ex(lincordy(k),lincordx(k))+Ex(lincordy(k)+1,lincordx(k))+Ex(lincordy(k),lincord
    EyatPos=
 (Ey(lincordy(k),lincordx(k))+Ey(lincordy(k)+1,lincordx(k))+Ey(lincordy(k),lincord
    end

        Fx = (-C.q_0)*ExatPos(1,:);
        Fy = (-C.q_0)*EyatPos(1,:);

    end

    %updates position
    vx = vx+(Fx./(Eme))*dt;
    vy =vy+(Fy./(Eme))*dt;
    v = sqrt(vx.^2+vy.^2);
    J_driftx =[J_driftx, -C.q_0*econcentration *mean(vx)*dimy*100];
    xpos=[xpos;xpos(l,:)+(vx)*dt];
    ypos=[ypos;ypos(l,:)+(vy)*dt];
    % fineds the distance traveled by each partical
    dtraveled = dtraveled + sqrt ((xpos(l,:)-xpos(l
+1,:)).^2+(ypos(l,:)-ypos(l+1,:)).^2);
    slope =(vy./vx);
    % sets up colision detection by determining if the particals have
 the
    % distance to the edges

    dtt = sqrt(((dimy -ypos(l+1,:)).^2)+((((dimy-ypos(l+1,:))./
slope)).^2));
    dtb = sqrt(((0 -ypos(l+1,:)).^2)+((((-ypos(l+1,:))./slope)).^2));
    if(boxes)
```

```
        dttbf =((xpos(l+1,:)>=box1(1,1)&xpos(l+1,:)<=box1(1,2))).*
 sqrt(((box1(2,3) -ypos(l+1,:)).^2)+((((box1(2,3)-ypos(l+1,:))./
slope)).^2))+~((xpos(l+1,:)>=box1(1,1)&xpos(l+1,:)<=box1(1,2))).*100;
        dtbbf = ((xpos(l+1,:)>=box1(1,1)&xpos(l
+1,:)<=box1(1,2))).*sqrt(((box2(2,1) -ypos(l+1,:)).^2)+((((box2(2,1)-
ypos(l+1,:))./slope)).^2))+~((xpos(l+1,:)>=box1(1,1)&xpos(l
+1,:)<=box1(1,2))).*100;
        dts1 =( ypos(l+1,:)<=box2(2,1)|ypos(l+1,:)>=box1(2,3)).*
 sqrt((slope.*(box1(1,1)-xpos(l+1,:))).^2+(box1(1,1)-xpos(l+1,:)).^2)
 +~( ypos(l+1,:)<=box2(2,1)|ypos(l+1,:)>=box1(2,3)).*100;
        dts2=( ypos(l+1,:)<=box2(2,1)|ypos(l+1,:)>=box1(2,3)).*
 sqrt((slope.*(box1(1,2)-xpos(l+1,:))).^2+(box1(1,2)-xpos(l+1,:)).^2)
 +~( ypos(l+1,:)<=box2(2,1)|ypos(l+1,:)>=box1(2,3)).*100;
    else
        dttbf =ones(1,N).*100;
        dtbbf = ones(1,N).*100;
        dts1 =ones(1,N).*100;
        dts2=ones(1,N).*100;

    end
    %counts the number of colissions that have occured and
    pc =c;
    c=(((dts1<1e-9|dts2<1e-9)|((dtt<1e-9|dtb<1e-9|dttbf<1e-9|
dtbbf<1e-9))));
    colision_count = colision_count+(((dts1<1e-9|dts2<1e-9)|
((dtt<1e-9|dtb<1e-9|dttbf<1e-9|dtbbf<1e-9))));
    if specular
        % basic colission part one
        vy= -((dtt<1e-9|dtb<1e-9|dttbf<1e-9|dtbbf<1e-9).*2-1).*vy;
        vx= -((dts1<1e-9|dts2<1e-9).*2-1).*vx;
    else
        % re thermalized velocities for part 3
        %if rethermalized volocity is in the same direction as
 previouse
        %than flip signs
        signx=sign(vx);
        signy=sign(vy);



        vx= ((((dts1<1e-9|dts2<1e-9)|((dtt<1e-9|dtb<1e-9|dttbf<1e-9|
dtbbf<1e-9))))&~pc).*(randn(1,N)*sqrt(C.kb*T/Eme)) +(~((((dts1<1e-9|
dts2<1e-9))|((dtt<1e-9|dtb<1e-9|dttbf<1e-9|dtbbf<1e-9)))&~pc)).*vx;
        vy= ((((dts1<1e-9|dts2<1e-9)|((dtt<1e-9|dtb<1e-9|dttbf<1e-9|
dtbbf<1e-9))))&~pc).*(randn(1,N)*sqrt(C.kb*T/Eme)) +(~((((dts1<1e-9|
dts2<1e-9))|((dtt<1e-9|dtb<1e-9|dttbf<1e-9|dtbbf<1e-9)))&~pc)).*vy;
        vx = (((dts1<1e-9&sign(vx)==1)|(dts2<1e-9
 &sign(vx)==-1))&~pc).*-1.*vx+(~(((dts1<1e-9&sign(vx)==1)|(dts2<1e-9
 &sign(vx)==-1))&~pc)).*vx;
        vy = (((((dtt<1e-9&sign(vy)==1)|(dtb<1e-9&sign(vy)==-1)|
(dttbf<1e-9&sign(vy)==1)|(dtbbf<1e-9&sign(vy)==-1)))&~pc)).*-1.*vy
+(~(((((dtt<1e-9&sign(vy)==1)|(dtb<1e-9&sign(vy)==-1)|
(dttbf<1e-9&sign(vy)==1)|(dtbbf<1e-9&sign(vy)==-1)))&~pc))).*vy;
```

```
    end
    % loop condition for end boundries
    xpos(l+1,:) = (xpos(l+1,:)>dimx).*0+(xpos(l+1,:)<0).*dimx+~(xpos(l
+1,:)>=dimx|xpos(l+1,:)<=0).*xpos(l+1,:);
    % colisions with other particals are only alouwed when partical is
 away
    % from the edges and has not colided with an edge Part 2 and 3
    colision = p>rand(1,N)&~(((dts1<30e-9|dts2<30e-9)|((dtt<30e-9|
dtb<10e-9|dttbf<30e-9|dtbbf<30e-9))))&~c;
    colision_count = colision_count+colision;
    vy=colision.*(randn(1,N)*sqrt(C.kb*T/Eme))+(~colision).*vy;
    vx=colision.*(randn(1,N)*sqrt(C.kb*T/Eme))+(~colision).*vx;



    % skips the plot of the x boundry transition
    skip = (xpos(l+1,:)>=dimx|xpos(l+1,:)<=0);
    % progress = (l/m)*100
    v = sqrt(vx.^2+vy.^2);
    c=0;
    %finds the current temperature
    Temp =[Temp, mean((v.^2)*Eme/(2*C.kb))];
    % plots the electrons
    figure (2)
    for k =1:10
        if skip(k)==0
            subplot(2,1,1);
            plot([xpos(l,k),xpos(l+1,k)],[ypos(l,k),ypos(l
+1,k)],'-','color',colour(k,:))
        end
        xlim([0,dimx])
        ylim([0,dimy])
        hold on
        %quiver(xpos(:,k),ypos(:,k),vx+xpos(:,k),vy+ypos(:,k),0.0001)
        drawnow limitrate

    end
    figure (2)
    subplot(2,1,2)
    plot([dt*(l-1),dt*(l)],J_driftx(l:l+1),'b-')
    hold on
    xlabel('time (s)')
    ylabel('Current (A)')

    xlim([0,m*dt])
    ylim([-10e-4,max(J_driftx)+10e-6])
end
%finds the mean free path and mean time between colission for part 2
 and
%plots a final velocity hystegram
% if (partical_colission)
%     colision_count(colision_count<=0) = 1;
%     MFP2 = mean (dtraveled./(colision_count));
%     Tmn2= mean( m*dt./(colision_count));
% %     figure(3)
% %     histogram(v)
% %     xlabel('volocity (m/s)')
% %     ylabel('probablility')
% end
% %findes the dencity and temperature of the electrons  part 3
squarcount= zeros(round(dimy/1e-9),round(dimx/1e-9));
temps= zeros(round(dimy/1e-9),round(dimx/1e-9));
for k =1:N

    yindex = min((round(dimy/1e-9))-ceil(ypos(m
+1,k)/1e-9)+1,round(dimy/1e-9));
    xindex = min(ceil(xpos(m+1,k)/1e-9)+1,round(dimx/1e-9));
    % excludes particals that violate boundry conditions.
    if ((yindex>0&xindex>0) & ~((((ypos(m+1,k)>=box1(2,3))|(ypos(m
+1,k)<=box2(2,1))))&(xpos(m+1,k)>= box1(1,1)&xpos(m+1,k)<=box1(1,2))))
        squarcount(yindex,xindex) =squarcount(yindex,xindex) +1;
        temps(yindex,xindex)=((v(k))^2)*Eme/
(2*C.kb)+temps(yindex,xindex);
    end
end
temps=temps./squarcount;
squarcount =squarcount./((1e-9)*(1e-9));

figure
subplot(1,2,1)
bar3(squarcount)
xlabel('x position (nm)')
ylabel('y position (nm)')
zlabel('Density of Electrons (electron/m)')
subplot(1,2,2)

bar3(temps)
xlabel('x position (nm)')
ylabel('y position (nm)')
zlabel('Temperature (k)')
```


V
0.6
0.4
0.2
0
X
0
50
100
150
200
Y
0
20
40
60
80
100


×10^-7
1
0.5
0
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
×10^-7
×10^-4
0
-5
-10
Current (A)
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
time (s)
×10^-13



The plots above show the Voltage and electric feild at each point in the semiconductor, the trajectories of the electrons and the current through semiconductor. as well as the dencity and temperatur of the electons in the semiconductor. From these plots it can be seen that the the current through the semiconductor increases to a maximum and then decreases down to a steady value. This is due to a combination of the colisions that occure and the electrons being blocked by the barriors. This can also be seen by the dencity plot above where the dencity of the elecrons is greater on the left side on the side of the smiconductor with the applied voltage. This is because the discontinuites in the semiconductor restrict the flow of the electon through the semiconductor.

Part 3:

This section looks at an extraction of the paramiters of the semiconductor. The code sagment below was used to find the effect of the bottle neck size on the current.

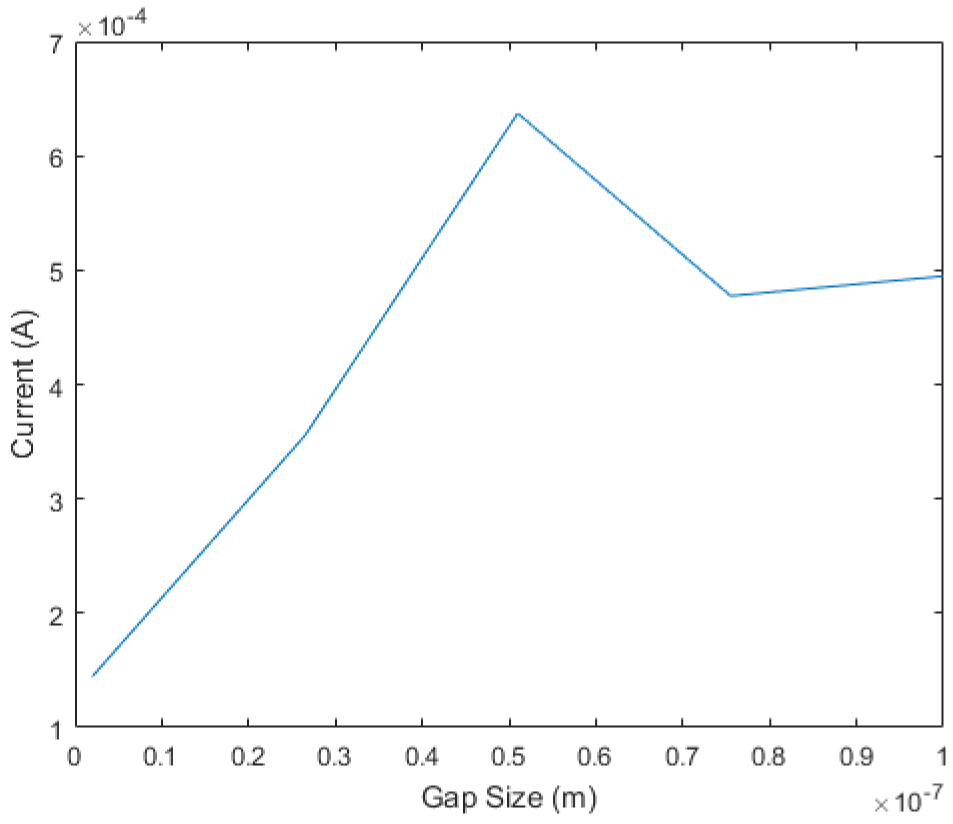
```
  clc;clear;close all

for w= 1:5

C.q_0 = 1.60217653e-19;             % electron charge
C.hb = 1.054571596e-34;             % Dirac constant
C.h = C.hb * 2 * pi;                % Planck constant
C.m_0 = 9.10938215e-31;             % electron mass
C.kb = 1.3806504e-23;               % Boltzmann constant
C.eps_0 = 8.854187817e-12;          % vacuum permittivity
C.mu_0 = 1.2566370614e-6;           % vacuum permeability
C.c = 299792458;                    % speed of light
```

```
C.g = 9.80665; %metres (32.1740 ft) per sÂ²
a3p1 =0;
a3p2 = 1;

partical_colission = 1;% part 2
boxes =1;%part 3
specular = 0 ;% part 3 one is on zero is off

T= 300;%k temperature
Eme= 0.26* C.m_0;% kg effective mass of electron
Tmn = 0.2e-12;%s mean time between colissions
vth = sqrt(2*C.kb*T/Eme);% m/s thermal volocity
MFP = vth*Tmn; %m
m= 1000;% length of sim
dt = 0.5e-15;% time step
N = 10000; %number of electons
dimx =200e-9;%m
dimy =100e-9;%m
c =zeros(1,N);

Fx=0;
Fy=0;

blocksize = linspace(0,(dimy/2)-1e-9,5);

% create boxes
if (boxes)
    box1 =
 [dimx-125e-9,dimx-75e-9,dimx-75e-9,dimx-125e-9,dimx-125e-9;dimy,dimy,
 dimy-blocksize(w),dimy-blocksize(w),dimy];
    box2 =
 [dimx-125e-9,dimx-75e-9,dimx-75e-9,dimx-125e-9,dimx-125e-9;blocksize(w),blocksize
 0,0,blocksize(w)];
else
    box1 =[0 0 0 0 0 ; 0 0 0 0 0];
    box2 =[0 0 0 0 0 ; 0 0 0 0 0];
end
% initiates points and ensures that they donot spawn ouside the
 boundries
if ~boxes
    xpos = (randi((dimx*1e9)+1,1,N)-1)/1e9;% m electron position x
    ypos = (randi((dimy*1e9)+1,1,N)-1)/1e9;% m electron position y
    xpos(xpos ==dimx)=xpos(xpos ==dimx)-1e-9;
    xpos(xpos ==0)=xpos(xpos ==0)+1e-9;
    ypos(ypos ==dimy)=ypos(ypos ==dimy)-1e-9;
    ypos(ypos ==0)=ypos(ypos ==0)+1e-9;
elseif(boxes)
    for l =1:N
        xpos(1,l) =(randi((dimx*1e9)+1,1,1)-1)/1e9 ;
        ypos(1,l) =(randi((dimy*1e9)+1,1,1)-1)/1e9;
        while (ypos(1,l)<=0|ypos(1,l)>=dimy|
xpos(1,l)>=box1(1,1)-1e-9&xpos(1,l)<=box1(1,2)+1e-9&(ypos(1,l)<=box2(2,1)+1e-9|
ypos(1,l)>=box1(2,3)-1e-9))
            xpos(1,l) =(randi((dimx*1e9)+1,1,1)-1)/1e9 ;
            ypos(1,l) =(randi((dimy*1e9)+1,1,1)-1)/1e9;
        end
    end
end
% while(sum ((xpos>=box1(1,1)&xpos<= box1(1,1)& ypos>= box1(2,3))|
(xpos>=box2(1,1)&xpos<= box2(1,1)& ypos>= box2(2,3)))>=1)
%     xpos = (randi((dimx*1e9)+1,1,N)-1)/1e9;% m electron position x
%     ypos = (randi((dimy*1e9)+1,1,N)-1)/1e9;% m electron position y
% end

vx   = zeros(1,N);%m/s velocity in x
vy   = zeros(1,N);%m/s velocity in y
colision_count = zeros(1,N);
dtraveled=zeros(1,N);
colourL = ["R", "G", "B","LB", "P", "y", "BLk", "DB", "O","M","GG"];
colour = [[1 0 0];[0 1 0];[0 0 1];[0 1 1];[1 0 1];[1 1 0];[0 0 0];
[0 0.447 0.741];[0.85 0.325 0.098];[0.929 0.694 0.125];[0.466 0.674
 0.188]];
Temp = T;

 dx2 =1e-9;
    dy2=1e-9;
    nx2 =round( dimx/dx2);
    ny2 = round(dimy/dy2);
    Ex=zeros(ny2,nx2);
    Ey=zeros(ny2,nx2);
% initiated the partical velocities
if ~partical_colission
    angle = rand(1,N);
    vx = vth* cos(angle*2*pi);
    vy = vth* sin(angle*2*pi);
else
    vx =randn(1,N)*sqrt(C.kb*T/Eme);
    vy=randn(1,N)*sqrt(C.kb*T/Eme);
end
v = sqrt(vx.^2+vy.^2);



if boxes
%     figure (2)
%     subplot(2,1,1);
%     plot (box1(1,:),box1(2,:),'-k')
%     hold on
%     plot (box2(1,:),box2(2,:),'-k')
end
if partical_colission
%     figure
%     histogram(v)
    p =1-exp(-dt/Tmn);
else
    p=0;
end
```

```
if a3p1 ==1

    econcentration =1e15;
    J_driftx = mean(vx)*econcentration*(-C.q_0)*dimy*100;
    Vapplied =0.1;
    Ex = Vapplied/dimx
    Ey =0
    Fx = (C.q_0)*Ex
    Fy = (C.q_0)*Ey
    AccelerationX =(Fx./(Eme))
    AccelerationY =(Fy./(Eme))
end
if a3p2 == 1
     econcentration =1e15;
   boundy = [0.8 nan 0 nan];
   J_driftx = mean(vx)*econcentration*(-C.q_0)*dimy*100;

    [Ex, Ey]= calcEfeild (dimx,dimy,dx2,dy2,boundy,0, box1, box2);

end

for l=1:m
    if a3p2 ==1
    lincordx = floor(xpos(1,:)/dx2)+1;
    lincordy = floor(ypos(1,:)/dy2)+1;
    lincordx(lincordx==round(dimx/dx2))=1;
    lincordy(lincordy==round(dimy/dy2))=1;
    for k = 1:N
    ExatPos=
 (Ex(lincordy(k),lincordx(k))+Ex(lincordy(k)+1,lincordx(k))+Ex(lincordy(k),lincord
    EyatPos=
 (Ey(lincordy(k),lincordx(k))+Ey(lincordy(k)+1,lincordx(k))+Ey(lincordy(k),lincord
    end

        Fx = (-C.q_0)*ExatPos(1,:);
        Fy = (-C.q_0)*EyatPos(1,:);

    end

    %updates position
    vx = vx+(Fx./(Eme))*dt;
    vy =vy+(Fy./(Eme))*dt;
    v = sqrt(vx.^2+vy.^2);
    J_driftx =[J_driftx, -C.q_0*econcentration *mean(vx)*dimy*100];
    xpos=[xpos;xpos(l,:)+(vx)*dt];
    ypos=[ypos;ypos(l,:)+(vy)*dt];
    % fineds the distance traveled by each partical
    dtraveled = dtraveled + sqrt ((xpos(l,:)-xpos(l
+1,:)).^2+(ypos(l,:)-ypos(l+1,:)).^2);
    slope =(vy./vx);
    % sets up colision detection by determining if the particals have
 the
    % distance to the edges
    dtt = sqrt(((dimy -ypos(l+1,:)).^2)+((((dimy-ypos(l+1,:))./
slope)).^2));
    dtb = sqrt(((0 -ypos(l+1,:)).^2)+((((-ypos(l+1,:))./slope)).^2));
    if(boxes)
        dttbf =((xpos(l+1,:)>=box1(1,1)&xpos(l+1,:)<=box1(1,2))).*
 sqrt(((box1(2,3) -ypos(l+1,:)).^2)+((((box1(2,3)-ypos(l+1,:))./
slope)).^2))+~((xpos(l+1,:)>=box1(1,1)&xpos(l+1,:)<=box1(1,2))).*100;
        dtbbf = ((xpos(l+1,:)>=box1(1,1)&xpos(l
+1,:)<=box1(1,2))).*sqrt(((box2(2,1) -ypos(l+1,:)).^2)+((((box2(2,1)-
ypos(l+1,:))./slope)).^2))+~((xpos(l+1,:)>=box1(1,1)&xpos(l
+1,:)<=box1(1,2))).*100;
        dts1 =( ypos(l+1,:)<=box2(2,1)|ypos(l+1,:)>=box1(2,3)).*
 sqrt((slope.*(box1(1,1)-xpos(l+1,:))).^2+(box1(1,1)-xpos(l+1,:)).^2)
 +~( ypos(l+1,:)<=box2(2,1)|ypos(l+1,:)>=box1(2,3)).*100;
        dts2=( ypos(l+1,:)<=box2(2,1)|ypos(l+1,:)>=box1(2,3)).*
 sqrt((slope.*(box1(1,2)-xpos(l+1,:))).^2+(box1(1,2)-xpos(l+1,:)).^2)
 +~( ypos(l+1,:)<=box2(2,1)|ypos(l+1,:)>=box1(2,3)).*100;
    else
        dttbf =ones(1,N).*100;
        dtbbf = ones(1,N).*100;
        dts1 =ones(1,N).*100;
        dts2=ones(1,N).*100;

    end
    %counts the number of colissions that have occured and
    pc =c;
    c=(((dts1<1e-9|dts2<1e-9)|((dtt<1e-9|dtb<1e-9|dttbf<1e-9|
dtbbf<1e-9))));
    colision_count = colision_count+(((dts1<1e-9|dts2<1e-9)|
((dtt<1e-9|dtb<1e-9|dttbf<1e-9|dtbbf<1e-9))));
    if specular
        % basic colission part one
        vy= -((dtt<1e-9|dtb<1e-9|dttbf<1e-9|dtbbf<1e-9).*2-1).*vy;
        vx= -((dts1<1e-9|dts2<1e-9).*2-1).*vx;
    else
        % re thermalized velocities for part 3
        %if rethermalized volocity is in the same direection as
 previouse
        %than flip signs
        signx=sign(vx);
        signy=sign(vy);



        vx= ((((dts1<1e-9|dts2<1e-9)|((dtt<1e-9|dtb<1e-9|dttbf<1e-9|
dtbbf<1e-9))))&~pc).*(randn(1,N)*sqrt(C.kb*T/Eme)) +(~((((dts1<1e-9|
dts2<1e-9))|((dtt<1e-9|dtb<1e-9|dttbf<1e-9|dtbbf<1e-9)))&~pc)).*vx;
        vy= ((((dts1<1e-9|dts2<1e-9)|((dtt<1e-9|dtb<1e-9|dttbf<1e-9|
dtbbf<1e-9))))&~pc).*(randn(1,N)*sqrt(C.kb*T/Eme)) +(~((((dts1<1e-9|
dts2<1e-9))|((dtt<1e-9|dtb<1e-9|dttbf<1e-9|dtbbf<1e-9)))&~pc)).*vy;
        vx = (((dts1<1e-9&sign(vx)==1)|(dts2<1e-9
 &sign(vx)==-1))&~pc).*-1.*vx+(~(((dts1<1e-9&sign(vx)==1)|(dts2<1e-9
 &sign(vx)==-1))&~pc)).*vx;
```

```
        vy = (((((dtt<1e-9&sign(vy)==1)|(dtb<1e-9&sign(vy)==-1)|
(dttbf<1e-9&sign(vy)==1)|(dtbbf<1e-9&sign(vy)==-1)))&~pc)).*-1.*vy
+(~(((((dtt<1e-9&sign(vy)==1)|(dtb<1e-9&sign(vy)==-1)|
(dttbf<1e-9&sign(vy)==1)|(dtbbf<1e-9&sign(vy)==-1)))&~pc))).*vy;



    end
    % loop condition for end boundries
    xpos(l+1,:) = (xpos(l+1,:)>dimx).*0+(xpos(l+1,:)<0).*dimx+~(xpos(l
+1,:)>=dimx|xpos(l+1,:)<=0).*xpos(l+1,:);
    % colisions with other particals are only alouwed when partical is
 away
    % from the edges and has not colided with an edge Part 2 and 3
    colision = p>rand(1,N)&~(((dts1<30e-9|dts2<30e-9)|((dtt<30e-9|
dtb<10e-9|dttbf<30e-9|dtbbf<30e-9))))&~c;
    colision_count = colision_count+colision;
    vy=colision.*(randn(1,N)*sqrt(C.kb*T/Eme))+(~colision).*vy;
    vx=colision.*(randn(1,N)*sqrt(C.kb*T/Eme))+(~colision).*vx;



    % skips the plot of the x boundry transition
    skip = (xpos(l+1,:)>=dimx|xpos(l+1,:)<=0);
    % progress = (l/m)*100
    v = sqrt(vx.^2+vy.^2);
    c=0;
    %finds the current temperature
    Temp =[Temp, mean((v.^2)*Eme/(2*C.kb))];
    % plots the electrons
%     figure (2)
%     for k =1:10
%         if skip(k)==0
%             subplot(2,1,1);
%             plot([xpos(l,k),xpos(l+1,k)],[ypos(l,k),ypos(l
+1,k)],'-','color',colour(k,:))
%         end
%         xlim([0,dimx])
%         ylim([0,dimy])
%         hold on
%         %quiver(xpos(:,k),ypos(:,k),vx+xpos(:,k),vy
+ypos(:,k),0.0001)
%         drawnow limitrate
%
%     end
%     figure (2)
%     subplot(2,1,2)
%     plot([dt*(l-1),dt*(l)],J_driftx(l:l+1),'b-')
%     hold on
%     xlabel('time (s)')
%     ylabel('Current (A)')
%
%     xlim([0,m*dt])
%     ylim([-1e-4,max(J_driftx)+10e-6])
```

```
end
%finds the mean free path and mean time between colission for part 2
 and
%plots a final velocity hystegram
% if (partical_colission)
%     colision_count(colision_count<=0) = 1;
%     MFP2 = mean (dtraveled./(colision_count));
%     Tmn2= mean( m*dt./(colision_count));
% %     figure(3)
% %     histogram(v)
% %     xlabel('volocity (m/s)')
% %     ylabel('probablility')
% end
% %findes the dencity and temperature of the electrons  part 3
squarcount= zeros(round(dimy/1e-9),round(dimx/1e-9));
temps= zeros(round(dimy/1e-9),round(dimx/1e-9));
% for k =1:N
%
%     yindex = min((round(dimy/1e-9))-ceil(ypos(m
+1,k)/1e-9)+1,round(dimy/1e-9));
%     xindex = min(ceil(xpos(m+1,k)/1e-9)+1,round(dimx/1e-9));
%     % excludes particals that violate boundry conditions.
%     if ((yindex>0&xindex>0) & ~((((ypos(m+1,k)>=box1(2,3))|(ypos(m
+1,k)<=box2(2,1))))&(xpos(m+1,k)>= box1(1,1)&xpos(m+1,k)<=box1(1,2))))
%         squarcount(yindex,xindex) =squarcount(yindex,xindex) +1;
%         temps(yindex,xindex)=((v(k))^2)*Eme/
(2*C.kb)+temps(yindex,xindex);
%     end
% end
% temps=temps./squarcount;
% squarcount =squarcount./((1e-9)*(1e-9));

% figure
% subplot(1,2,1)
% bar3(squarcount)
% xlabel('x position (nm)')
% ylabel('y position (nm)')
% zlabel('Density of Electrons (electron/m)')
% subplot(1,2,2)
%
% bar3(temps)
% xlabel('x position (nm)')
% ylabel('y position (nm)')
% zlabel('Temperature (k)')

current(w) = mean(J_driftx);
close all
end

figure
plot(dimy-blocksize*2,current)
xlabel('Gap Size (m)')
ylabel('Current (A)')
```



From the above plot it can be seen that as the gap size decreases the current through the semiconductor current deceases. This is because as the size of the gap decreases more of the electons are blocked and cannot travel through the gap.

Conclution

In conclution the this assignment sucsessfuly analysed the effects of an electric feild on the trajectories of electrons in a semiconductor. It was found that the addition of an electric feild has a dromatic effect on the trajectories by causeing them to curve in the direction of the applied voltage. This simulator is a very basic simulator, one posable addition to make the simulator more accuret, electron emission can be added. Electron emission would cause electons to be removed from the material if a sificent energy is reached.

Appendix 1: E-Feild solver

```
function [Ex, Ey]= calcEfeild (dimx,dimy,Dx,Dy,boundy,pr, box1, box2,
 box3)


sig = 1;
sigbox = 10e-2;

dx =Dx;
dy=Dy;

box1pos= [round(box1(1,1)/dx),round(box1(2,1)/dy)];
```

```
box2pos= [round(box2(1,1)/dx),round(box2(2,1)/dy)];
box1dim= [round((box1(1,2)-box1(1,1))/dx),round((box1(2,1)-box1(2,3))/
dy)];
box2dim= [round((box2(1,2)-box2(1,1))/dx),round((box2(2,1)-box2(2,3))/
dy)];


nx = round(dimx/dx);
ny = round(dimy/dy);
V3 = zeros(ny,nx);
G2= sparse(nx*ny,nx*ny);
V02=1;
BC2= boundy;
B2 = zeros(1,nx*ny);
cond = zeros (ny,nx);
% sets up conduction map

for p = 1:ny
    for m = 1:nx

        if ((m >=box1pos(1)&m <=box1pos(1)+box1dim(1))&p
 <=box1pos(2)&p >=box1pos(2)-box1dim(2))|((m >=box2pos(1)&m
 <=box2pos(1)+box2dim(1))&p <=box2pos(2)&p >=box2pos(2)-box2dim(2))
            cond(p,m) =sigbox;
        else
           cond(p,m) =sig;
        end
    end
end


%sets boundry conditions
for p = 1:size(B2,2)

    if (p== 1)
        if isnan(BC2(1))

        else
            B2(p)=BC2(1);
        end
    elseif (p == nx)
        if isnan(BC2(3))

        else
            B2(p)=BC2(3);
        end
    elseif (p == (1+(ny-1)*nx))
        if isnan(BC2(1))

        else
            B2(p)=BC2(1);
        end
    elseif(p == nx*ny)
        if isnan(BC2(3))
```

```
            else
                B2(p)=BC2(3);
            end
        elseif(mod(p,nx)==0)
            if isnan(BC2(3))

            else
                B2(p)=BC2(3);
            end
        elseif(mod(p-1,nx)==0)
            if isnan(BC2(1))

            else
                B2(p) = BC2(1);
            end
        elseif(1<p&p<nx)
            if isnan(BC2(4))

            else
                B2(p) =BC2(4);
            end
        elseif((1+(ny-1)*nx)<p&p<nx*ny)
            if isnan(BC2(2))

            else
                B2(p) =BC2(2);
            end
        else

        end

    end
    % creates conduction matrix
    for p = 1:ny
        for m = 1:nx
            n = m+(p-1)*nx;
            nxm = (m-1)+(p-1)*nx;
            nxp = (m+1)+(p-1)*nx;
            nym = (m)+(p-2)*nx;
            nyp = m+(p)*nx;
            nxm2 = (m-2)+(p-1)*nx;
            nxp2=(m+2)+(p-1)*nx;
            nym2 = (m)+(p-3)*nx;
            nyp2 = m+(p+1)*nx;
            nxp3 = (m+3)+(p-1)*nx;
            nxm3 = (m-3)+(p-1)*nx;

            if m == 1
                G2(n,:)=0;
                G2(n,n)=1;

            elseif m==nx
                G2(n,:)=0;
                G2(n,n)=1;

            elseif p ==1

                ryp = (cond(p,m)+cond(p+1,m))/2;
                rxp = (cond(p,m)+cond(p,m+1))/2;
                rxm = (cond(p,m)+cond(p,m-1))/2;

                G2(n,n) = -(ryp+rxp+rxm);
                G2(n,nyp)= ryp;
                G2(n,nxm)= rxm;
                G2(n,nxp) = rxp;

            elseif p==ny
                rym = (cond(p,m)+cond(p-1,m))/2;
                rxp = (cond(p,m)+cond(p,m+1))/2;
                rxm = (cond(p,m)+cond(p,m-1))/2;

                G2(n,n) = -(rym+rxp+rxm);
                G2(n,nym)= rym;
                G2(n,nxm)= rxm;
                G2(n,nxp) = rxp;
            else
                ryp = (cond(p,m)+cond(p+1,m))/2;
                rym = (cond(p,m)+cond(p-1,m))/2;
                rxp = (cond(p,m)+cond(p,m+1))/2;
                rxm = (cond(p,m)+cond(p,m-1))/2;

                G2(n,n) = -(ryp+rym+rxp+rxm);
                G2(n,nyp)= ryp;
                G2(n,nym)= rym;
                G2(n,nxm)= rxm;
                G2(n,nxp) = rxp;
            end

    end
end
%create voltage map
V3 = G2\B2';
Vout2 = zeros(ny, nx);
for p = 1:ny
    for m = 1:nx
        n = m+(p-1)*nx;
        Vout2(p,m) =V3(n);
    end
end
if pr==1
% figure
% surf(cond)
% title('Sigma')
% xlabel('X')
% ylabel('Y')
% zlabel('conductance')
```

```
figure
title('Voltage')
surf(Vout2)
xlabel('X')
ylabel('Y')
zlabel('V')
view(-45, -45)
end
Ex= size (Vout2);
Ey = size (Vout2);
for p = 1:ny
    for m = 1:nx

        if (m==1)
            Ex(p,m) = (Vout2(p,m+1)-Vout2(p,m))/dx;
        elseif m == nx
             Ex(p,m) = (Vout2(p,m)-Vout2(p,m-1))/dx;
        else
             Ex(p,m) = (Vout2(p,m+1)-Vout2(p,m-1))/(dx*2);

        end
         if (p==1)
            Ey(p,m) = (Vout2(p+1,m)-Vout2(p,m))/dy;
        elseif p == ny
             Ey(p,m) = (Vout2(p,m)-Vout2(p-1,m))/dy;
        else
             Ey(p,m) = (Vout2(p+1,m)-Vout2(p-1,m))/(dy*2);

        end
    end
end
Ex=-Ex;
Ey=-Ey;




end
```

Published with MATLAB® R2017a