Assignment 3 Monte-Carlo/ Finite Difference Method

Table of Contents

Question 1

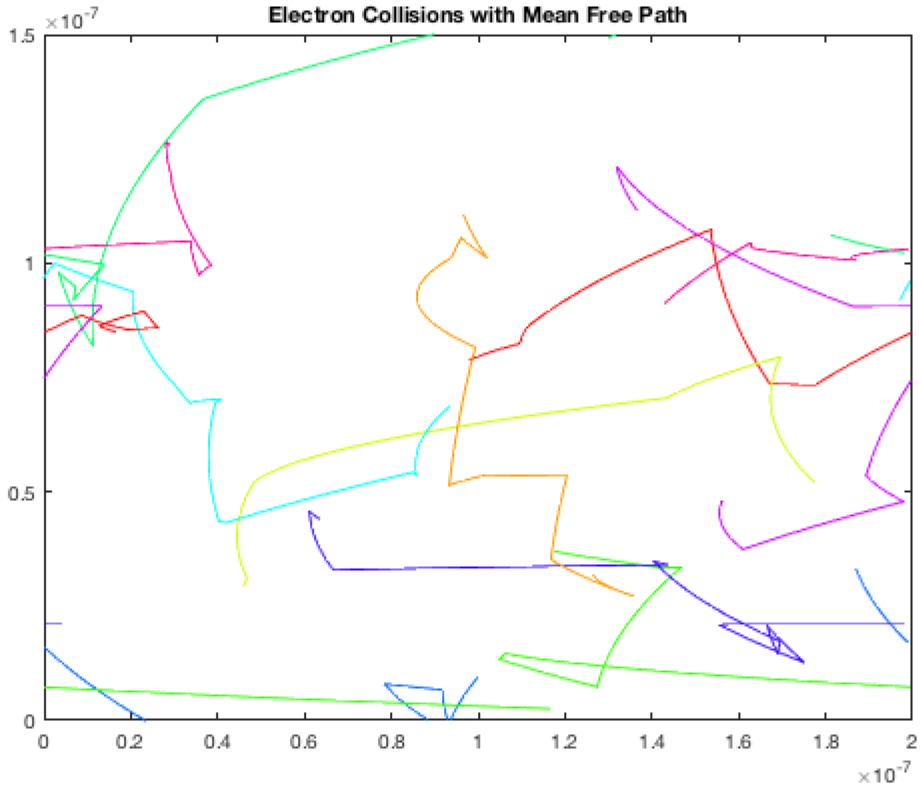
Monte-Carlo Simulator with voltage difference in the x-direction

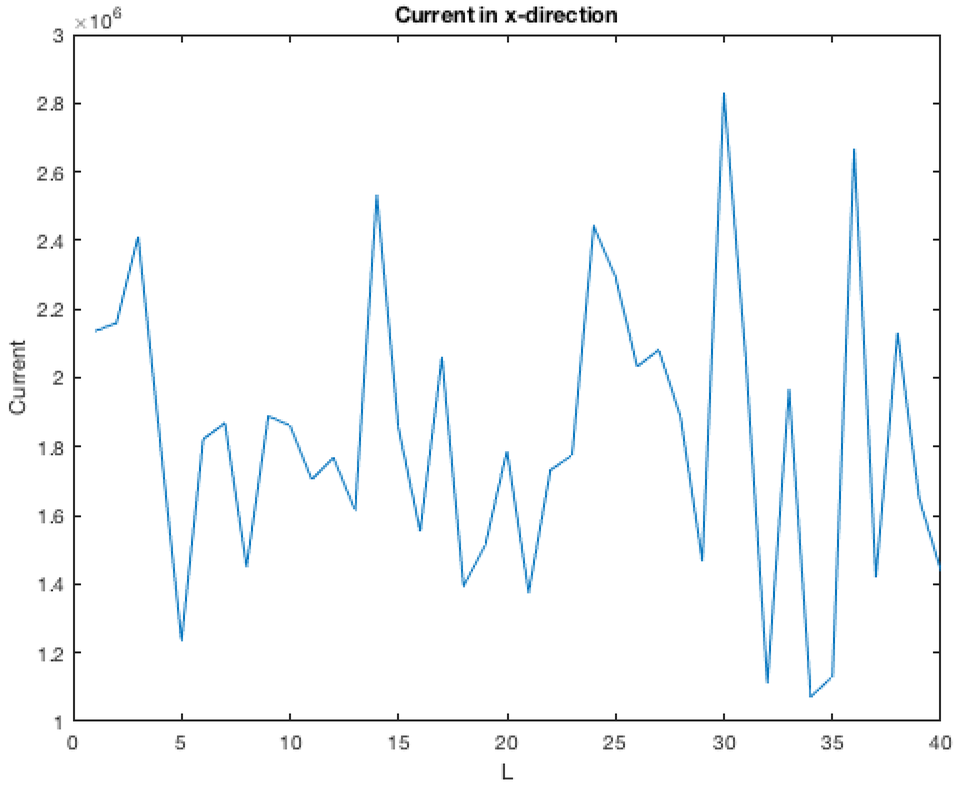
```
close all
clear all

L=200e-9;
W=150e-9;
n=1000; %change
nsteps =100; %change

tau_mn=0.2e-12; %seconds

m0=9.109382e-31; %electron mass
mn=0.26*m0;
T=300; %Kelvin
k=physconst('Boltzman');
Va=0.1;
E=Va/L;
q = 1.60217662e-19;
F=q*E;
ac = F/mn;

vth = sqrt(k*T/mn);
mfp = tau_mn*vth ;% meters

%inititalize particle locations
x=rand(1,n)*L;
y=rand(1,n)*W;

%initialize previous location as first location just to get the plot
%started
xp = x;
yp = y;

%initialize random velocities
vx=vth*randn(1,n)/sqrt(2);
vy=vth*randn(1,n)/sqrt(2);
speed=sqrt(vx.*vx +vy.*vy);
```

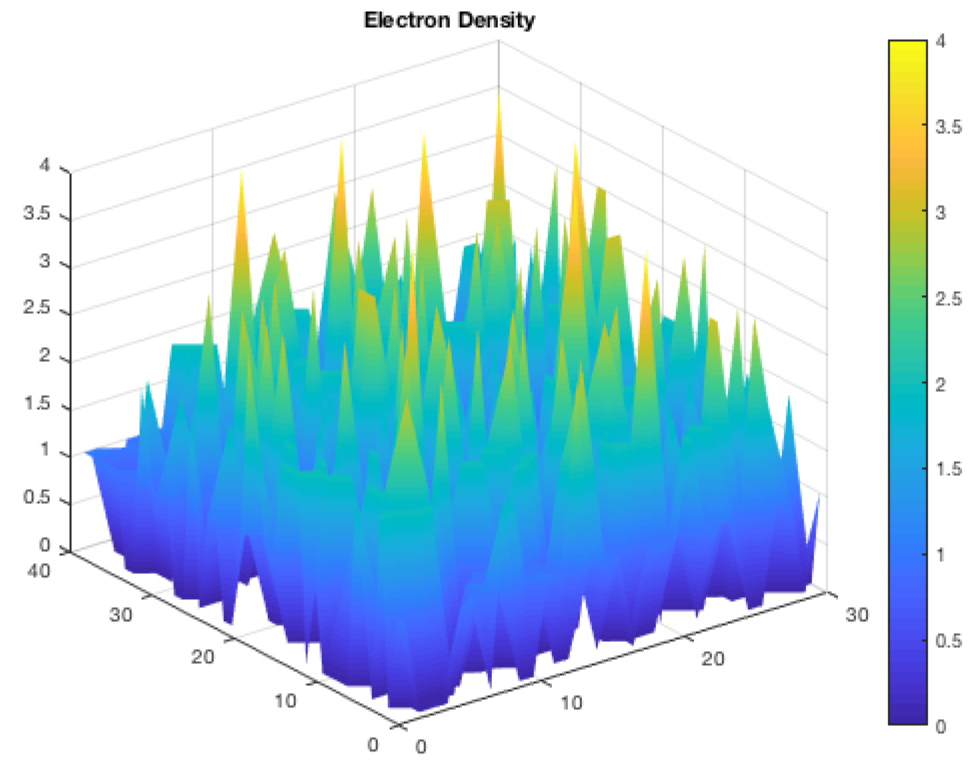
```
%f1 = figure;
f2 = figure;
%f3=figure;

% set(0, 'CurrentFigure', f1)
% subplot(2,2,1);
% histogram (vx);
% title ('Distribution-Vx')
% subplot(2,2,2);
% histogram (vy);
% title ('Distribution-vy')
% subplot(2,2,[3,4]);
% histogram (speed);
% title ('Distribution-speed');

dt=(L/vth)/100;
av_temp = zeros(1,nsteps);

%vector of colours for particle trajectories
col=hsv(10);
set(0, 'CurrentFigure', f2)
  for p = 1:10
        plot([x(p); xp(p)],[y(p); yp(p)],'color',col(p,:));  hold on
  end
    xlim([0 L])
    ylim([0 W])


    %main timeloop
for aa=1:nsteps
    xp=x;
    yp=y;

    %scattering
    pscat=1-exp(-dt/tau_mn);
    scatCount= 0;

    for bb=1:n
        if (pscat > rand())
            vx(bb)=vth*randn()/sqrt(2);
            vy(bb)=vth*randn()/sqrt(2);
            scatCount = scatCount+1;
        end
    end

    vx = vx + ac*dt;

    dx=vx*dt;
    dy=vy*dt;

    %increment every particle over dt
    x=x+dx;
    y=y+dy;
    %xpath calc before boundary adjustment
    xpath=abs(x-xp);

    %travelling restrictions (WALL)
     for a=1:n
        %periodic boundaries at x=0 and x=L
         if (xp(a)< L && x(a)>=L)
            x(a)=x(a)-L;
            xp(a)=xp(a)-L;
         elseif (xp(a)< 0 && x(a)<0)
            x(a) = x(a)+L;
            xp(a)=xp(a)+L;
         end

        %specular boundaries at y=0 and y=W
        if (y(a)>=W || y(a)<=0)
            vy(a) = -vy(a);
        elseif y(a)<=0
          vy(a) = -vy(a);
        end
     end %end travelling restrictions loop


     %ypath calc after boundary adjustment
    ypath=abs(y-yp);

    %calculate path
    path = sqrt(xpath.*xpath + ypath.*ypath);



% plot(x,y,'o');hold on
  set(0, 'CurrentFigure', f2)
    %plot trajectories
    for p = 1:10
        plot([x(p); xp(p)],[y(p); yp(p)],'color',col(p,:));  hold on
    end
    xlim([0 L])
    ylim([0 W])
    title ('Electron Collisions with Mean Free Path')
    pause(0.01);


    velx = mean(vx.^2);
    vely = mean(vy.^2);
    v_inst=sqrt(velx+vely);

    Temp= v_inst*v_inst*mn/k ;
%    set(0, 'CurrentFigure', f3)
%     plot(aa,Temp, 'o')
%     hold on
%     title(sprintf('Semiconductor Temperature = %s', Temp))
```

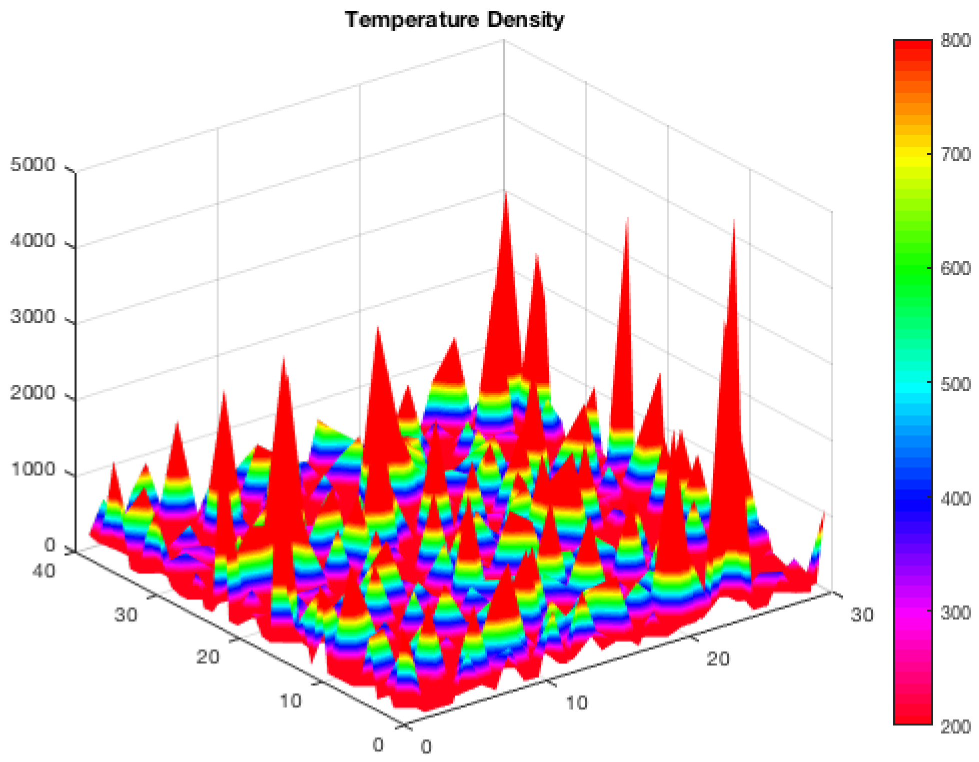
```
    av_temp(aa) = Temp;
end


% AverageTemperature = mean(av_temp)
% meanfreepath = mean(path)
% meanfreetime = meanfreepath/(mean(v_inst))
%

delta = 5e-9;
ne = 10e19;
counta=0;
countelectrons=0;
hi=W/delta;
Nmap = zeros(round(L/delta),round(W/delta)); %number of electrons in
 region
Vmap = zeros(round(L/delta),round(W/delta)); %average velocity of
 electrons per region
Tmap = zeros(round(L/delta),round(W/delta))*300; %average temperature
 of region
Vel = zeros(1,round(L/delta));
curr = zeros(1,round(L/delta));

%populate density maps
for  aa = delta:delta:L
    counta=counta+1;
    countb=0;
    for bb=delta:delta:W
        countb=countb+1;
        for cc=1:n
            %Populate Electron Density Map
            if (x(cc)<(counta*delta) & x(cc)>=((counta-1)*delta) &
 y(cc)<(countb*delta) & y(cc)>=((countb-1)*delta))
              Nmap(counta,countb) = Nmap(counta,countb)+1;
              Vmap(counta,countb) =
 Vmap(counta,countb)+sqrt(vx(cc)*vx(cc)+vy(cc)*vy(cc));
              map(counta,countb)=Vmap(counta,countb)/
Nmap(counta,countb);
              Tmap(counta,countb) =
 map(counta,countb)*map(counta,countb)*mn/k;
              countelectrons = countelectrons +1;
            elseif(x(cc)== L)
                Nmap(counta,countb) = Nmap(counta,countb)+1;
                Vmap(counta,countb) =
 Vmap(counta,countb)+sqrt(vx(cc)*vx(cc)+vy(cc)*vy(cc));
                map(counta,countb)=Vmap(counta,countb)/
Nmap(counta,countb);
               Tmap(counta,countb) =
 map(counta,countb)*map(counta,countb)*mn/k;
                countelectrons = countelectrons +1;
             elseif(y(cc)==W)
                Nmap(counta,countb) = Nmap(counta,countb)+1;
                Vmap(counta,countb) =
 Vmap(counta,countb)+sqrt(vx(cc)*vx(cc)+vy(cc)*vy(cc));
```

```
                map(counta,countb)=Vmap(counta,countb)/
Nmap(counta,countb);
               Tmap(counta,countb) =
 map(counta,countb)*map(counta,countb)*mn/k;
                countelectrons = countelectrons +1;
            elseif(x(cc)== L & y(cc)==W)
                Nmap(counta,countb) = Nmap(counta,countb)+1;
                Vmap(counta,countb) =
 Vmap(counta,countb)+sqrt(vx(cc)*vx(cc)+vy(cc)*vy(cc));
                map(counta,countb)=Vmap(counta,countb)/
Nmap(counta,countb);
               Tmap(counta,countb) =
 map(counta,countb)*map(counta,countb)*mn/k;
                countelectrons = countelectrons +1;
            end
        end
    end

    Vel(counta) = Vel(counta) + mean(abs(Vmap(counta,:)));
    curr(counta)=mean(Vel(counta))*ne*q;
end

figure(2)
plot(curr)
hold on;
title('Current in x-direction')
ylabel('Current')
xlabel('L')


figure(3)
surf(Nmap)
title('Electron Density')
colormap('parula')
colorbar
shading interp;

figure(4)
surf(Tmap)
title('Temperature Density')
h=flipud(hsv);
colormap(h)
caxis([200 800])
colorbar
shading interp;
```


Electron Collisions with Mean Free Path
×10⁻⁷
1.5
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


Current in x-direction
×10⁶
3
2.8
2.6
2.4
2.2
2
1.8
1.6
1.4
1.2
1
Current
0
5
10
15
20
25
30
35
40
L


Electron Density
4
3.5
3
2.5
2
1.5
1
0.5
0
40
30
20
10
0
0
10
20
30


Temperature Density
5000
4000
3000
2000
1000
0
40
30
20
10
0
0
10
20
30
800
700
600
500
400
300
200

Question 2

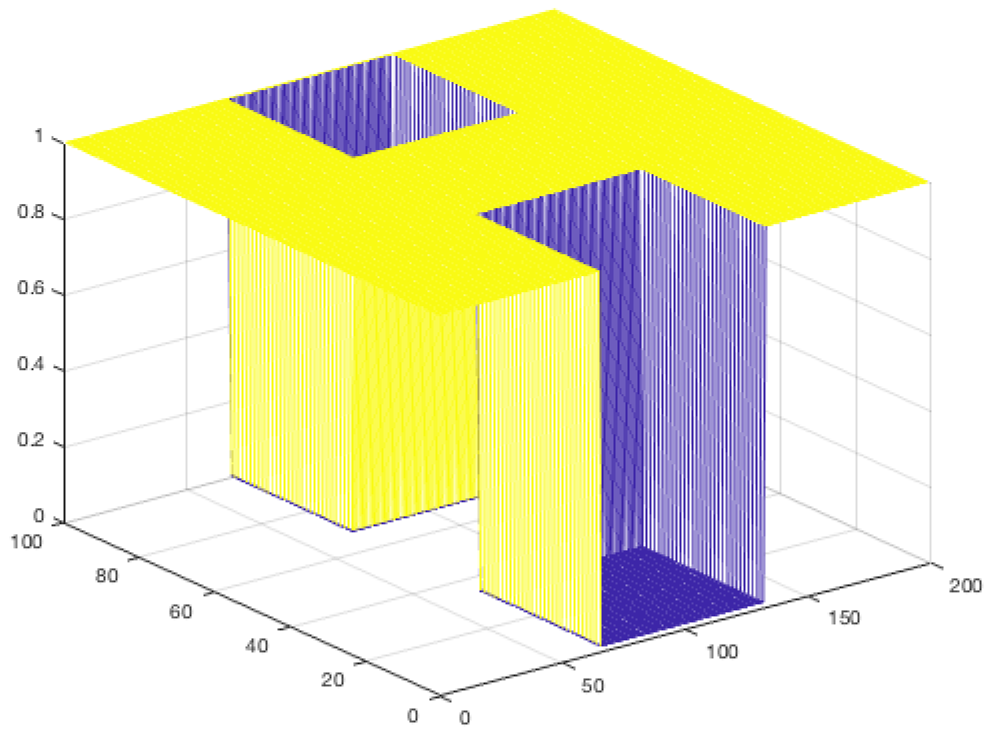
Finite Difference to calculate the electric field.

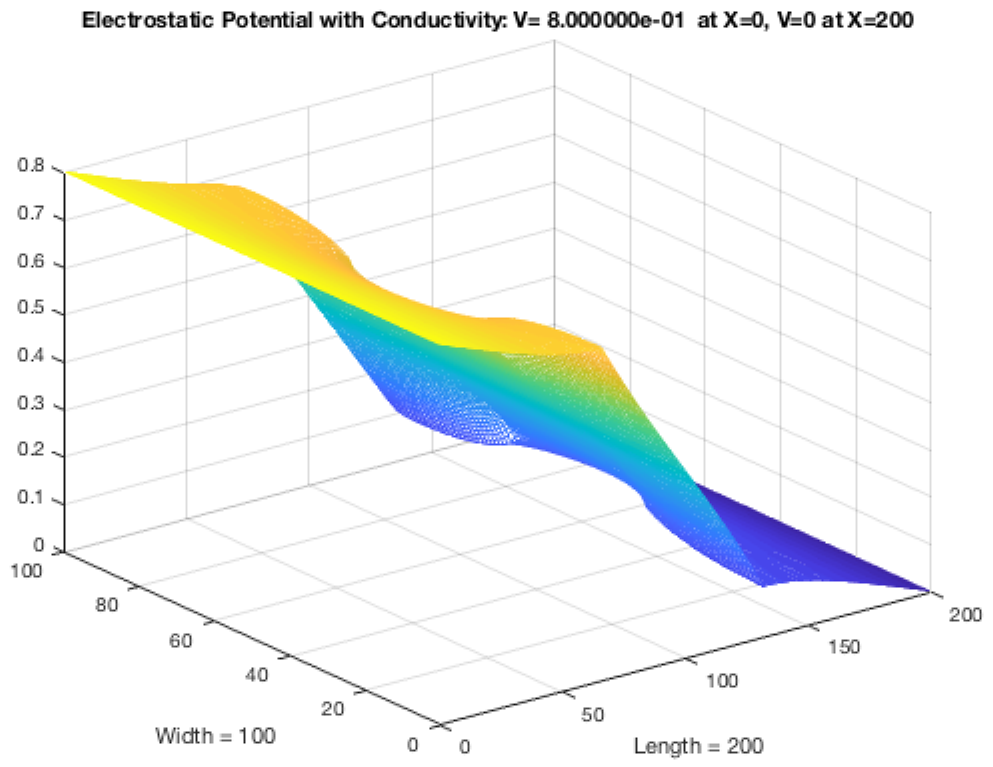
```
nx=200; %L
ny=100; %W
L=nx;
W=ny;
BC_left=0.8;
BC_right=0;
BC_top=0;
BC_bottom=0;

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

Llow = round(L/3);
Lhigh= round(L-L/3);
Wlow =round(W/3);
Whigh=round(W-W/3);

sig = ones(W,L);

% set conductivity in boxes
for a = 1:Wlow
    for b=Llow:Lhigh
        sig(a,b) = 0.01;
    end
end
for a = Whigh:W-1
    for b=Llow:Lhigh
        sig(a,b) = 0.01;
    end
end
figure(5)
mesh(sig)


for a=1:ny
    for b=1:nx
        n=a+(b-1)*nx;

        if b==1
            %Left Side
            G(n,:) = 0;
            G(n,n) = 1;
            B(n)=BC_left;
        elseif b==nx
            %Right Side
            G(n,:) = 0;
            G(n,n) = 1;
            B(n)=BC_right;
        elseif a==ny
```

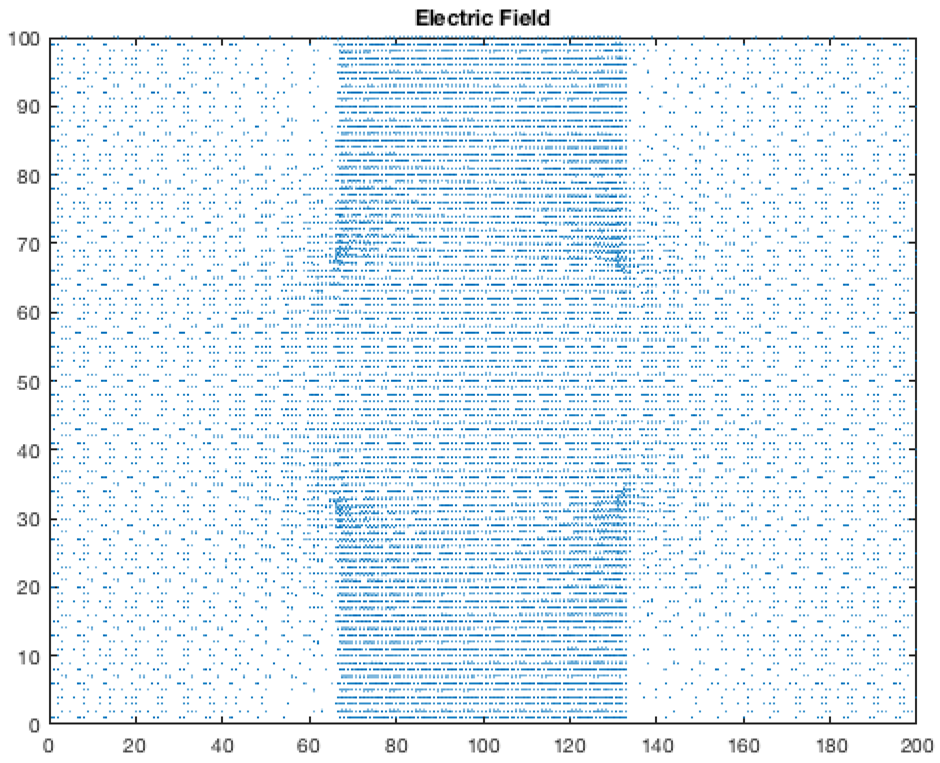
```
            % Top
            nxm= a-1 +(b-1)*nx;
            nyp= a +(b-2)*nx;
            nym= a +(b)*nx;

            ym = (sig(a,b)+sig(a,b+1))/2;
            yp = (sig(a,b)+sig(a,b-1))/2;
            xm = (sig(a-1,b)+sig(a,b))/2;

            G(n,n) =-(ym+yp+xm);
            G(n,nxm) = xm;
            G(n,nyp) = yp;
            G(n,nym) = ym;
        elseif a==1
            %Bottom
            nxp= a+1 +(b-1)*nx;
            nyp= a +(b-2)*nx;
            nym= a +(b)*nx;

            ym = (sig(a,b)+sig(a,b+1))/2;
            yp = (sig(a,b)+sig(a,b-1))/2;
            xp = (sig(a+1,b)+sig(a,b))/2;

            G(n,n) =-(yp+xp+ym);
            G(n,nym) = ym;
            G(n,nxp) = xp;
            G(n,nyp) = yp;
        else
            %All Central Nodes
            nxm= a-1 +(b-1)*nx;
            nxp= a+1 +(b-1)*nx;
            nyp= a +(b-2)*nx;
            nym= a +(b)*nx;

            ym = (sig(a,b)+sig(a,b+1))/2;
            yp = (sig(a,b)+sig(a,b-1))/2;
            xm = (sig(a-1,b)+sig(a,b))/2;
            xp = (sig(a+1,b)+sig(a,b))/2;

            G(n,n) =-(yp+ym+xp+xm);
            G(n,nxm) = xm;
            G(n,nxp) = xp;
            G(n,nym) = ym;
            G(n,nyp) = yp;
        end
    end
end


V=G\B';

Vmap = zeros(ny,nx);
for a=1:ny
    for b=1:nx
```

```
            n=a+(b-1)*nx;
        Vmap(a,b) = V(n);
    end
end

figure(6)
mesh(Vmap)
    xlim([0 L])
    ylim([0 W])
xlabel(sprintf('Length = %d', nx))
ylabel(sprintf('Width = %d', ny))
title(sprintf('Electrostatic Potential with Conductivity: V= %d  at
 X=0, V=%d at X=%d',BC_left, BC_right,nx))

[Ex,Ey] = gradient(Vmap);

figure(7)
quiver(Ex,Ey)
    xlim([0 L])
    ylim([0 W])
title('Electric Field')
```

```
Warning: Matrix is singular to working precision.
```




Electrostatic Potential with Conductivity: V= 8.000000e-01 at X=0, V=0 at X=200
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
100
80
60
40
20
0
Width = 100
0
50
100
150
200
Length = 200

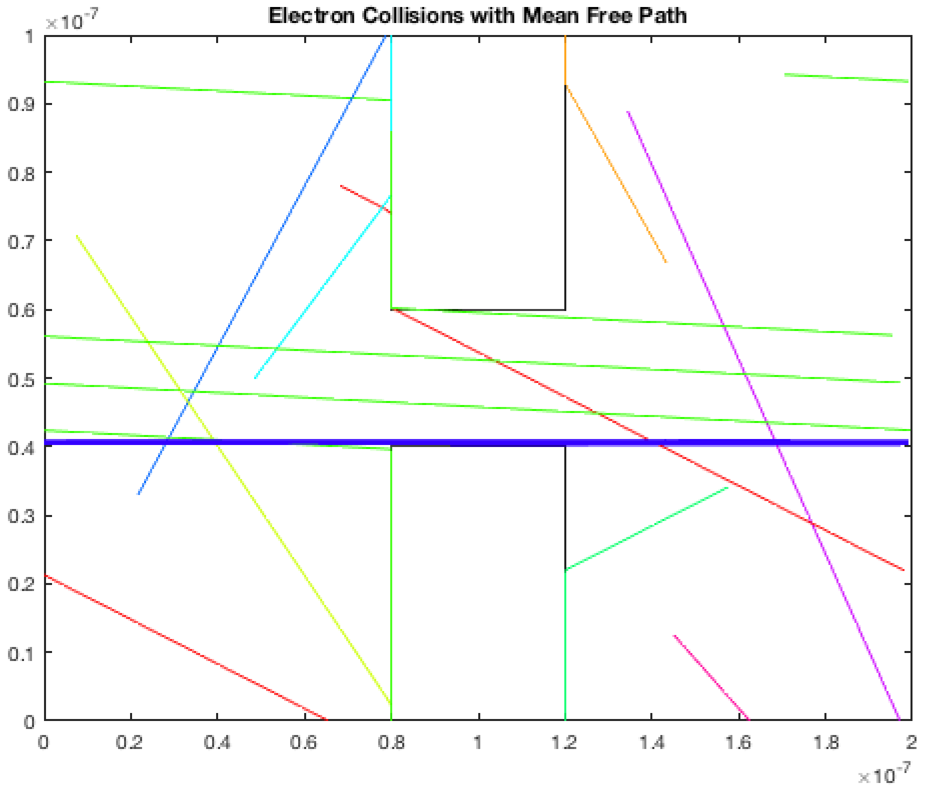

Electric Field
100
90
80
70
60
50
40
30
20
10
0
0
20
40
60
80
100
120
140
160
180
200

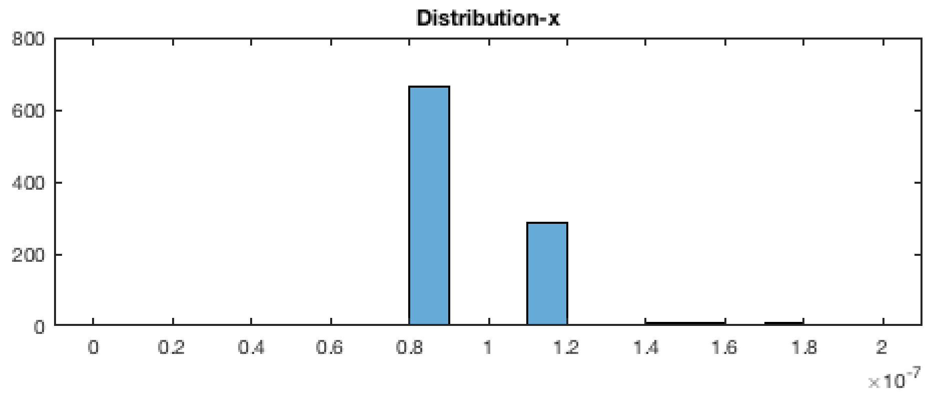
Question 3

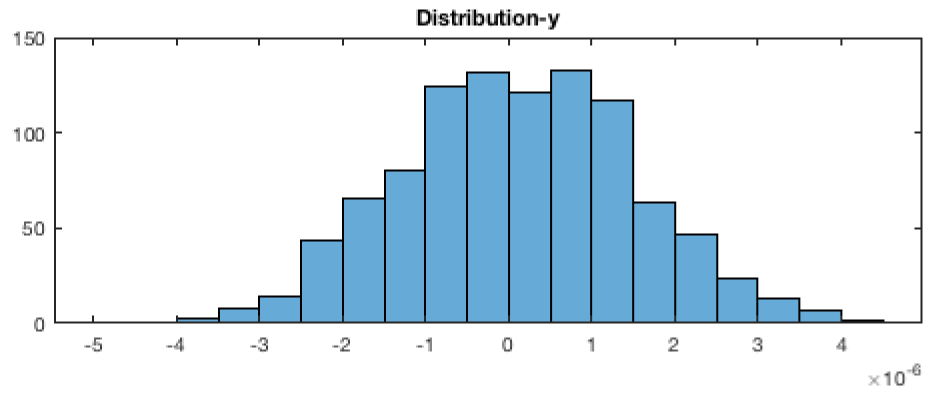
Finite Difference to provide field for the Monte-Carlo Simulation Using the electric field calculated in the previous part as an input, we re-run the Monte-Carlo simulation

```
nx=200e-9; %L
ny=100e-9; %W
L=nx;
W=ny;
n=1000; %change
nsteps =1000; %change

tau_mn=0.2e-12; %seconds

m0=9.109382e-31; %electron mass
mn=0.26*m0;
T=300; %Kelvin
k=physconst('Boltzman');
q = 1.60217662e-19;
Fx=q*Ex;
Fy=q*Ey;
ax = Fx/mn;
ay = Fy/mn;

vth = sqrt(k*T/mn);
mfp = tau_mn*vth ;% meters

%inititalize particle locations
x=rand(1,n)*L;
y=rand(1,n)*W;

% create a bunch of electrons not in the boxes
% box 1  190e-9<x<210e-9 60e-9<y<100e-9
% box 2  190e-9<x<210e-9 0<y<40e-9
Cxlow = 80e-9;
Cxhigh= 120e-9;
Cylow =40e-9;
Cyhigh=60e-9;
Ibox = (y>Cyhigh | y<Cylow) & x<Cxhigh & x>Cxlow;

countrestarts =0;
% no starting in boxes
for a = 1:n
while (x(a)<Cxhigh && x(a)>Cxlow && (y(a)>Cyhigh || y(a)<Cylow))
    x(a) = rand()*L;
    y(a) = rand()*W;
    countrestarts = countrestarts+1;
end
end

%initialize previous location as first location just to get the plot
%started
xp = x;
```

```
yp = y;

%initialize random velocities
vx=vth*randn(1,n)/sqrt(2);
vy=vth*randn(1,n)/sqrt(2);
speed=sqrt(vx.*vx +vy.*vy);

f8 = figure;


dt=(L/vth)/100;
av_temp = zeros(1,nsteps);

%vector of colours for particle trajectories
col=hsv(10);
set(0, 'CurrentFigure', f8)
  for p = 1:10
        plot([x(p); xp(p)],[y(p); yp(p)],'color',col(p,:));  hold on
  end
    xlim([0 L])
    ylim([0 W])

  %display boxes
line([Cxlow,Cxlow,Cxhigh,Cxhigh], [0,Cylow,Cylow,0], 'color', 'k');
line([Cxlow,Cxlow,Cxhigh,Cxhigh], [W,Cyhigh,Cyhigh,W], 'color', 'k');

    %main timeloop
for aa=1:nsteps
    xp=x;
    yp=y;

    %scattering
    pscat=1-exp(-dt/tau_mn);
    scatCount= 0;

    for bb=1:n
        if (pscat > rand())
            vx(bb)=vth*randn()/sqrt(2);
            vy(bb)=vth*randn()/sqrt(2);
            scatCount = scatCount+1;
        end
    end

    for bb=1:n
       for ee=1:L
          for ff=1:W
            if (x(n)>= ee && x(n)< (ee+1) && y(n) >=ff &&y(n) < (ff
+1))
                vx(n) = vx(n) + ax(ff,ee)*dt;
                vy(n) = vy(n) + ay(ff,ee)*dt;

                dx(n) = vx(n)*dt;
                dy(n) = vy(n)*dt;
                end
              end
           end
        end


        %increment every particle over dt
        x=x+dx;
        y=y+dy;

        %xpath calc before boundary adjustment
        xpath=abs(x-xp);

        %travelling restrictions (WALL)
         for a=1:n
             %no travelling through boxes
            if ( xp(a)<=Cxlow && x(a)>=Cxlow &&(y(a)>=Cyhigh ||
    y(a)<=Cylow))
                x(a)=Cxlow ;
                vx(a)=-vx(a);
            elseif (xp(a)>=Cxhigh && x(a)<=Cxhigh&&(y(a)>=Cyhigh ||
    y(a)<=Cylow))
                x(a)=Cxhigh;
                vx(a)=-vx(a);
            elseif (yp(a)<=Cyhigh && y(a)>=Cyhigh&&(x(a)>=Cxlow &&
     x(a)<=Cxhigh))
                y(a) = Cyhigh;
                vy(a) = -vy(a);
            elseif (yp(a)>=Cylow && y(a)<=Cylow&&(x(a)>=Cxlow &&
     x(a)<=Cxhigh))
                  y(a) = Cylow;
                vy(a) = -vy(a);
            end

            %periodic boundaries at x=0 and x=L
             if (xp(a)< L && x(a)>=L)
                x(a)=x(a)-L;
                xp(a)=xp(a)-L;
             elseif (xp(a)< 0 && x(a)<0)
                x(a) = x(a)+L;
                xp(a)=xp(a)+L;
             end

            %specular boundaries at y=0 and y=W
            if (y(a)>=W || y(a)<=0)
                vy(a) = -vy(a);
            elseif y(a)<=0
              vy(a) = -vy(a);
            end
         end %end travelling restrictions loop


         %ypath calc after boundary adjustment
        ypath=abs(y-yp);
        %calculate path
        path = sqrt(xpath.*xpath + ypath.*ypath);



    % plot(x,y,'o');hold on
      set(0, 'CurrentFigure', f8)
        %plot trajectories
        for p = 1:10
            plot([x(p); xp(p)],[y(p); yp(p)],'color',col(p,:));  hold on
        end
        xlim([0 L])
        ylim([0 W])
        title ('Electron Collisions with Mean Free Path')
        pause(0.01);

    end
    figure(9)
    subplot(2,1,1);
    histogram (x);
    title ('Distribution-x')
    subplot(2,1,2);
    histogram (y);
    title ('Distribution-y')
```







Published with MATLAB® R2017b