**ELEC 4700**

**ASSIGNMENT 3**

**Precious Usoroh**

**101031041**

**Introduction:**

The objective of this experiment involved using previous knowledge gotten from assignment1&2(Monte Carlo simulation and Finite difference method). Also, for this experiment we observed the 2D representation for the Density and temperature plot. Additionally, observation of the electrons.

Text

Description automatically generated

Chart, line chart

Description automatically generated

Graphical user interface, chart, line chart

Description automatically generated

Established on the current plot the performance of the current can be related to scattering of electrons. Also based on observation if the number of time steps is increased the current plot will be better.

Chart, histogram

Description automatically generated

Fundamentally J = V\*n\*q\*Ny. This is because drift velocity of the electrons and its current density is directly proportional to each other**.**

**PART2**

Chart, surface chart

Description automatically generated

Chart

Description automatically generated

**Conclusion:**

The knowledge gotten from the rework of assignment1&2 was key in doing this experiment.

**CODE**

clearvars

clearvars -GLOBAL

close all

set(0,'DefaultFigureWindowStyle', 'docked')

q = 1.60217662e-19;

electron\_conc = 1e15\*100^2;

m0 = 9.10938356e-31;

effective\_m = 0.26\*m0;

Temperature = 300;

Boltz\_const = 1.38064852e-23;

Thermal\_v = sqrt(2\*Boltz\_const\*Temperature/effective\_m);

widithx = 200e-9; % width

lengthy = 100e-9; % length

time\_step = lengthy/Thermal\_v/100;

Voltageinx = 0.1; %

%question a

electricfield = Voltageinx/widithx;

fprintf('The electric field of the charge is %0.2f V/m.\n',electricfield);

%question b

force= electricfield\*q;

fprintf('The force on each electron is %0.2E. \n ',force);

%question c a=f/m

acc= force/effective\_m;

fprintf('The acceleration on each electron is %f .\n',acc);

timestep=100;

t\_mn = 0.2e-12;

dt=1e-14;

BoundaryX = 200e-9; % X boundary

BoundaryY = 100e-9; % Y boundary

n=1000;

T=300;

C.kb = 1.3806504e-23;

C.m\_0 = 9.10938215e-31; % electron mass

Em = 0.26 \* C.m\_0; % Mass of the Electron

vel= sqrt(2\*C.kb\*T/Em);

Pscat = 1 - exp(-(dt/t\_mn));

ppx=rand(n,1)\*BoundaryX;

ppy=rand(n,1)\*BoundaryY;

vx=randn(n,1)\*vel/sqrt(2);

vy=randn(n,1)\*vel/sqrt(2);

randomvalue= randi(n,[10,1]);

J = zeros(1,timestep-1);

for i=2:timestep

% pxold=ppx;

%pyold=ppy;

st=Pscat> rand(n,1);

vx(st)= randn(sum(st),1)\*vel/sqrt(2);

vy(st)=randn(sum(st),1)\*vel/sqrt(2);

pxold=ppx;

pyold=ppy;

ppx=pxold+ vx\*dt;

ppy=pyold+vy\*dt;

m=ppx<0;

m1=ppx>BoundaryX;

ppx(m) = ppx(m)+ BoundaryX;

pxold(m)=BoundaryX;

ppx(m1)=ppx(m1)- BoundaryX;

pxold(m1)=0;

k=ppy<0;

k1=ppy>BoundaryY;

vy(k)=-vy(k);

vy(k1)=-vy(k1);

%set colour

myColors = ['r' 'b' 'g' 'y' 'm' ];

myColorTyp = j;

for j=1:10

subplot(3,1,1);

%title('The 2-D plot of particle trajectories');

plot([pxold(randomvalue(j)) ppx(randomvalue(j))], [pyold(randomvalue(j)) ppy(randomvalue(j))]);

hold on

end

pause(0.1)

title('The 2-D plot of particle trajectories');

average=(mean(sqrt((vx.^2) + (vy.^2))));

%Tavgold=TemperatureAvg;

TemperatureAvg = ((average.^2) \* Em)/(2 \* C.kb);

Tavgold=TemperatureAvg;

%TAvgp1 = TemperatureAvg;

subplot(3,1,2);

plot([i-1 i],[Tavgold TemperatureAvg],'r');

xlim([0 timestep]);

%ylim([0 400]);

pause(0.1)

hold on

title('The Average Temperature');

% J(i) = numElectrons \* mean((average)) \* q;

J(i) = n \* mean((vx)) \* q;

end

%

%question d

%current formula is J = vnqNy

figure(2)

plot(linspace(2,timestep,timestep),J);

title('The Current Plot');

ylabel('J');

xlabel('TimeSteps');

%question e

newaverage = sqrt(vx.^2 + vy.^2);

TemperatureAvgnew = (Em \* (newaverage.^2))./(2 \* C.kb);

TheppX = linspace(min(ppx), max(ppx), 100);

TheppY = linspace(min(ppy), max(ppy), 50);

[X,Y] = meshgrid(TheppX, TheppY);

Tsurf = griddata(ppx,ppy,TemperatureAvgnew,X,Y);

figure(3)

surf(Tsurf);%Tsurf

title('Temperature Map');

**PART2 CODE**

clearvars

clearvars -GLOBAL

close all

set(0,'DefaultFigureWindowStyle', 'docked')

%

nx = 40;

ny = 40;

max=3;

Lb = round(nx/max); % rounds the elements of X to the nearest integers

Wb = round(ny/max);

inside = 1;

outside = 10e-2;

%Conductivity map

conductivityMap = zeros(nx,ny);

% mapping of inside

for i = 1:nx

for j = 1: ny

conductivityMap(i,j) = inside;

end

end

%mapping of outside

% for i = 1:nx

% for j = 1: ny

% conductivityMap(i,j) = outside;

% end

% end

max1=2;

for i = 1:nx

for j = 1:ny

if (i>=1 && i<=Wb && j>Lb && j<=(max1\*Lb) )||(i<=ny && i>=(ny-Wb) && j>Lb && j<=(max1\*Lb));

%max1\*Lb

conductivityMap(i,j) = outside;

end

%

% if (i<=ny && i>=(ny-Wb) && j>Lb && j<=(max1\*Lb))

% conductivityMap(i,j) = outside;

%end

end

end

conductivityMap=conductivityMap';

G = sparse(nx\*ny);

b = zeros(1, nx\*ny);

%

% G - Matrix Formulation from slides

for i = 1:nx

for j = 1:ny

n = j + (i - 1) \* ny;

if i == 1

G(n,:) = 0;

G(n,n) = 1;

b(n) = 1;

elseif i == nx %(i == 1 && i > 1 && i < nx)

G(n,:) = 0;

G(n,n) = 1;

elseif j == 1

nxm = j + (i-2) \* ny;

nxp = j + (i) \* ny;

nyp = (j+1) + (i-1) \* ny;

rxm = (conductivityMap(i,j) + conductivityMap(i-1,j))/2.0;

rxp = (conductivityMap(i,j) + conductivityMap(i+1,j))/2.0;

ryp = (conductivityMap(i,j) + conductivityMap(i,j+1))/2.0;

G(n,n) = -(rxm + rxp + ryp);

G(n,nxm) = rxm;

G(n,nxp) = rxp;

G(n, nyp) = ryp;

elseif j == ny %(j == ny && i > 1 && i < nx)

nxm = j + (i-2) \* ny;

nxp = j + (i) \* ny;

nym = (j-1) + (i-1) \* ny;

rxm = (conductivityMap(i,j) + conductivityMap(i-1,j))/2.0;

rxp = (conductivityMap(i,j) + conductivityMap(i+1,j))/2.0;

rym = (conductivityMap(i,j) + conductivityMap(i,j-1))/2.0;

G(n,n) = -(rxm + rxp + rym);

G(n,nxm) = rxm;

G(n,nxp) = rxp;

G(n, nym) = rym;

else

nxm = j + (i-2) \* ny;

nxp = j + (i) \* ny;

nym = (j-1) + (i-1) \* ny;

nyp = (j+1) + (i-1) \* ny;

rxm = (conductivityMap(i,j) + conductivityMap(i-1,j))/2.0;

rxp = (conductivityMap(i,j) + conductivityMap(i+1,j))/2.0;

rym = (conductivityMap(i,j) + conductivityMap(i,j-1))/2.0;

ryp = (conductivityMap(i,j) + conductivityMap(i,j+1))/2.0;

G(n,n) = -(rxm + rxp + rym + ryp);

G(n,nxm) = rxm;

G(n,nxp) = rxp;

G(n, nym) = rym;

G(n, nyp) = ryp;

end

end

end

V = G\b';

%vmap= zeros (nx,ny);

for i = 1:nx %Converting V to matrix to plot

for j = 1:ny

n = j + (i-1) \* ny;

VG(i,j) = V(n);

end

end

for i = 1:nx

for j = 1:ny

if i == 1

Ex(i, j) = (VG(i + 1, j) - VG(i, j));

elseif i == nx

Ex(i, j) = (VG(i, j) - VG(i - 1, j));

else

Ex(i, j) = (VG(i + 1, j) - VG(i - 1, j)) \* 0.5;

end

if j == 1

Ey(i, j) = (VG(i, j + 1) - VG(i, j));

elseif j == ny

Ey(i, j) = (VG(i, j) - VG(i, j - 1));

else

Ey(i, j) = (VG(i, j + 1) - VG(i, j - 1)) \* 0.5;

end

end

end

Ex = -Ex;

Ey = -Ey;

figure(2);

surf(VG);

title('Voltage Map with Bottleneck');

figure(3);

quiver(Ex', Ey');

title('Electric field Plot')