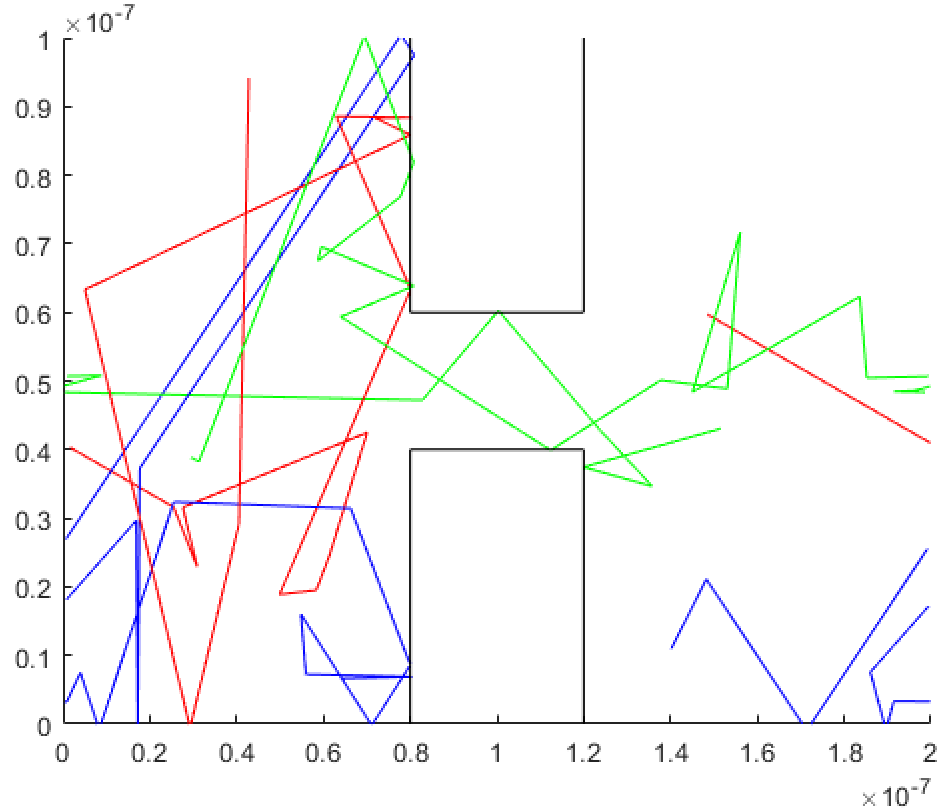
Section 3 - Enhancements

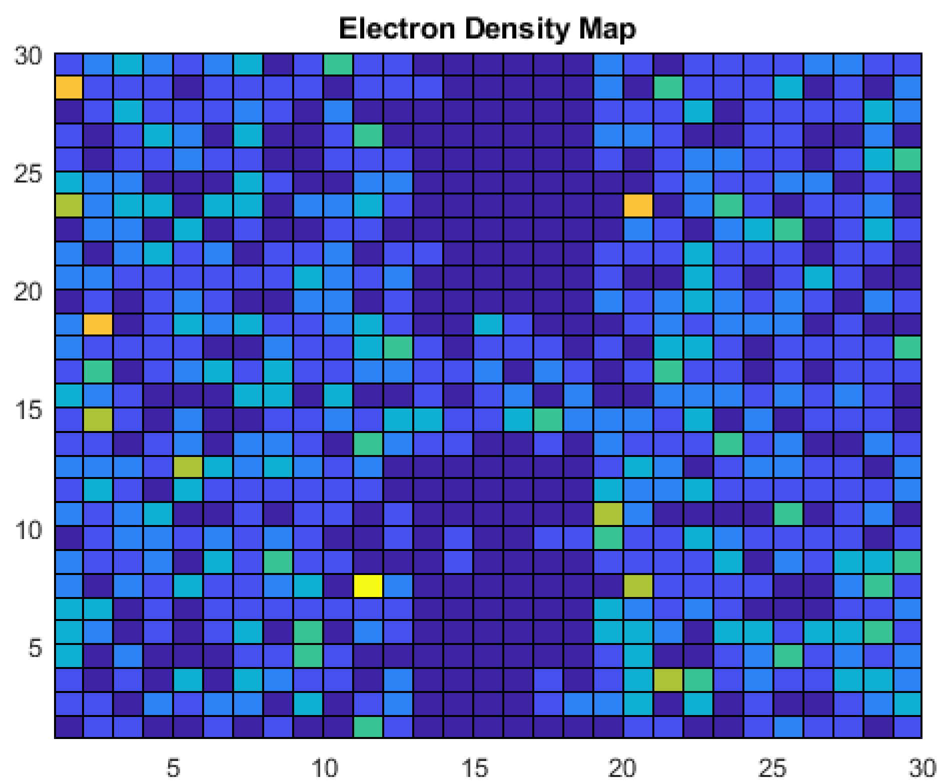
This section was fairly challenging, and certainly made sure I understood how to use logical indexing for the boundary conditions. The trajectory plot seems to work as intended, but the electron density map reveals that my boundary conditions are not perfect. Before the particle reflects off the boundary, it penetrates into the boxes just slightly, and it shows on the map. Other than this issue, the electron density map and temperature map are both looking proper.

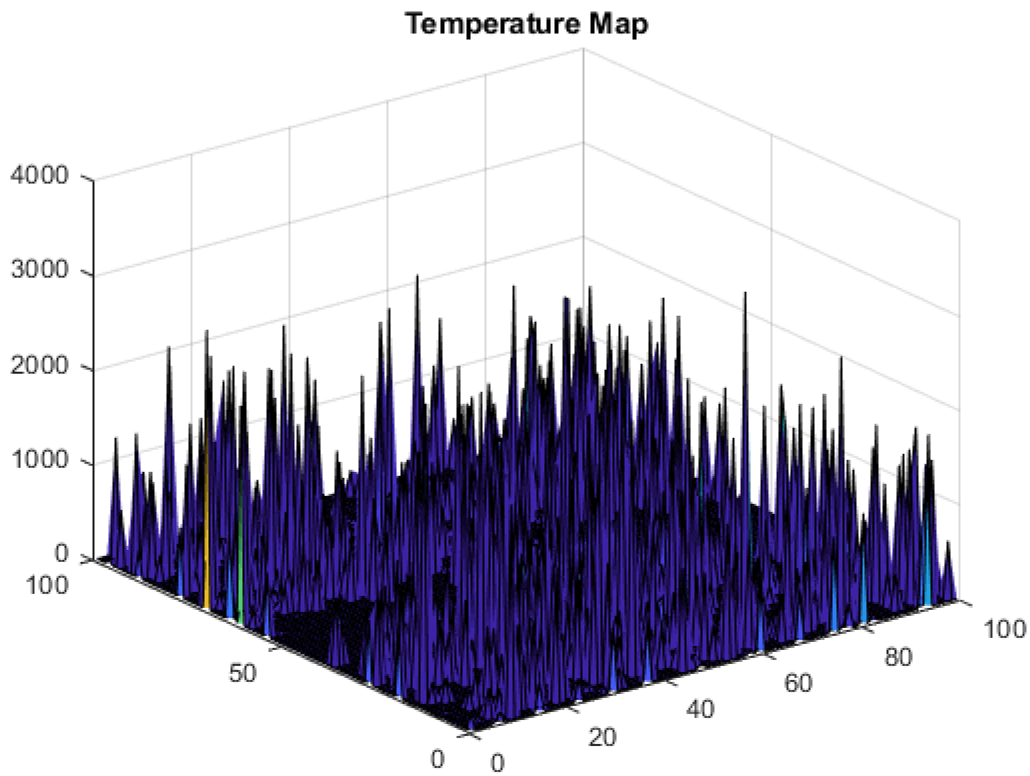
```
%constants
clear
C.q_0 = 1.60217653e-19;
C.m_0 = 9.10938215e-31;
C.kb = 1.3806504e-23;
C.T = 300;
frameWidth = 200e-9;
frameHeight = 100e-9;
nAtoms = 1000;
bins = nAtoms / 10;
Vth = sqrt(2*C.kb*C.T /(0.26*C.m_0));
dt = frameHeight/Vth/100;
Tstop = 500*dt;
t = 0;
freepath = 0.2e-12;
Pscatter = 1 - exp(-dt/freepath);

%initializing vectors
Xnext = zeros(1,nAtoms);
Ynext = zeros(1,nAtoms);
VX = Vth * randn(1,nAtoms);
VY = Vth * randn(1,nAtoms);
V = sqrt(VY.*VY+VX.*VX);
X = frameWidth * rand(1, nAtoms);
Y = frameHeight * rand(1, nAtoms);
R = zeros(1, nAtoms);
Temperature = zeros(1, 100);
iteration = 1;

%defining box boundaries
boxleft = X>0.8e-7;
boxright = X<1.2e-7;
box1bottom = Y>0.6e-7;
box2top = Y<0.4e-7;
boxedin = (boxleft&boxright&box1bottom)|(boxleft&boxright&box2top);
%removing particles from the boxes
while sum(boxedin)>0
X(boxedin) = rand*frameWidth;
Y(boxedin) = rand*frameHeight;
boxleft = X>0.8e-7;
boxright = X<1.2e-7;
box1bottom = Y>0.6e-7;
box2top = Y<0.4e-7;
boxedin = (boxleft&boxright&box1bottom)|(boxleft&boxright&box2top);
```

```
end

while t < Tstop
    %determines which particles scatter and performs calculations on
 them
    %to determine mean free path and time between collisions
    R = rand(1,nAtoms);
    VX(R<Pscatter) = Vth*randn(1);
    VY(R<Pscatter) = Vth*randn(1);
    V = sqrt(VY.*VY+VX.*VX);
    Xnext = X + VX*dt;
    Ynext = Y + VY*dt;
    %set X boundary conditions
    right = Xnext>frameWidth;
    left = Xnext<0;
    Xnext(right) = Xnext(right)-frameWidth;
    Xnext(left) = Xnext(left) + frameWidth;
    %set Y boundary conditions
    top = Ynext > frameHeight;
    bottom = Ynext < 0;
    VY(top | bottom) = VY(top | bottom) * -1;
    %set boundary for sides of boxes
    box1sides = (Ynext>0.6e-7)&(Xnext>0.8e-7)&(Xnext<1.2e-7);
    box2sides = (Ynext<0.4e-7)&(Xnext>0.8e-7)&(Xnext<1.2e-7);
    VX(box1sides|box2sides) = VX(box1sides|box2sides) * -1;
    topbox = (Y<0.6e-7)&(Ynext>0.6e-7)&(Xnext>0.8e-7)&(Xnext<1.2e-7);
    bottombox =
 (Y>0.4e-7)&(Ynext<0.4e-7)&(Xnext>0.8e-7)&(Xnext<1.2e-7);
    VY(topbox|bottombox) = VY(topbox|bottombox) * -1;
    VX(topbox|bottombox) = VX(topbox|bottombox) * -1;

    %calculations for temperature
    Temperature(iteration) = 0.26*C.m_0*mean(V.^2)/4/C.kb;
    %setting up figure, drawing both boxes
    figure(6)
    xlim([0 frameWidth])
    ylim([0 frameHeight])
    hold on
    plot([0.8e-7 0.8e-7],[0 0.4e-7], 'black')
    plot([1.2e-7 1.2e-7],[0 0.4e-7], 'black')
    plot([0.8e-7 1.2e-7],[0.4e-7 0.4e-7], 'black')
    plot([0.8e-7 0.8e-7],[1e-7 0.6e-7], 'black')
    plot([1.2e-7 1.2e-7],[1e-7 0.6e-7], 'black')
    plot([0.8e-7 1.2e-7],[0.6e-7 0.6e-7], 'black')

    %plotting, but avoid plotting the full horizontal jump from right
 and
    %left boundaries
    if abs(Xnext(1) - X(1)) < 2*abs(VX(1))*dt
        figure(6)
        plot([Xnext(1) X(1)], [Ynext(1) Y(1)], 'blue')
    end
    if abs(Xnext(2) - X(2)) < 2*abs(VX(2))*dt
        figure(6)
```

```
        plot([Xnext(2) X(2)], [Ynext(2) Y(2)], 'red')
    end
    if abs(Xnext(3) - X(3)) < 2*abs(VX(3))*dt
        figure(6)
        plot([Xnext(3) X(3)], [Ynext(3) Y(3)], 'green')
    end
    %optional plotting for 2 more particles
    %if abs(Xnext(4) - X(4)) < 2*abs(VX(4))*dt
    %    figure(2)
    %    plot([Xnext(4) X(4)], [Ynext(4) Y(4)], 'cyan')
    %end
    %if abs(Xnext(5) - X(5)) < 2*abs(VX(5))*dt
    %    figure(2)
    %    plot([Xnext(5) X(5)], [Ynext(5) Y(5)], 'magenta')
   % end

    %updating positions, and advancing time a step forward so the
 while
    %loop works
    X = Xnext;
    Y = Ynext;
    t = t+dt;
    iteration = iteration + 1;
   % pause(0.0001);
end

%electron density map
figure(7)
EDM = hist3([X',Y'],[30,30]);
pcolor(EDM')
title('Electron Density Map')

%temperature map
xLim = linspace(0,frameWidth,100);
yLim = linspace(0,frameHeight,100);
xTempReg = discretize(X,xLim);
yTempReg = discretize(Y,yLim);
for q=1:1:100
    for w=1:1:100
        %Temperature contained in defined region
        tempReg = (q == xTempReg) & (w == yTempReg);

        %Total velocities in region
        vxTot=sum(VX(tempReg));
        vyTot=sum(VY(tempReg));
        vTot = sqrt((vxTot)^2+(vyTot)^2);

        %Calculate Temperature
        tempMap(q,w) = C.m_0*0.26*(vTot)^2/(2*C.kb);
    end
end
figure(8)
surf(tempMap)
title('Temperature Map')
```


×10^-7
×10^-7


Electron Density Map



Published with MATLAB® R2018a