PHYS 225 - Rutherford Lab (sample report)

FirstName LastName

Contents

- Purpose
- Answers to handout questions
- Computation: Matlab code for simulating Rutherford's experiment
- Results and analysis
- Conclusion
- Feedback


```
disp(date)
```

```
27-Jan-2011
```

Purpose

The purpose of this lab is to create a simulation of Rutherford's experiment using Matlab; this will allow us to estimate the fraction of alpha particles that are scattered from a gold atom at steep angles.

Answers to handout questions

1. Given Rutherford's findings about the structure of the atom, if you bombarded a gold atom with a spray of alpha particles, what fraction would you estimate to be scattered at steep angles? (We are looking for an estimate, not a complicated calculation, but you should think about the geometry of the situation. Note: look up appropriate quantities you need for your estimate). Briefly explain what quantities went into your estimate and how you chose them.

We can estimate the fraction of alpha particles that are scattered by finding the ratio between the area of the cross-section of the gold nucleus to the area of the entire atom's cross-section:

```
r = 7.3e-15;      % nuclear radius in meters
R = 0.1441e-9;    % atomic radius in meters
fs = pi*r^2/(pi*R^2); % fraction scattered
disp('Fraction scattered:')
disp(fs)
```

```
Fraction scattered:
   2.5664e-09
```

We find that the number of scattered alpha particles is $2.566x10^{-9}$ times the total number of particles incident on the gold atom.

2. It can be shown that the projectiles follow hyperbolic paths and that the relationship between a projectile's impact parameter b and its scattering angle θ is given by: $b = \frac{zZ}{2K}\frac{e^2}{4\pi\epsilon_0}cot(\theta/2)$ where z and Z are the charge of the alpha particle and the gold nucleus, respectively, e is the electron's charge, ϵ_0 is the permittivity of vacuum, and K is the kinetic

energy of the alpha particle. Estimate b for an alpha particle with kinetic energy 8.0 MeV that is scattered at an angle of 90°. Use this value of b to refine your estimate for question 1.

```
clear all
z = 2;  % charge of alpha particle
Z = 79; % charge of gold nucleus
q = 1.602e-19; % electron charge in Coulombs
e0 = 8.85e-12; % permittivity of vacuum in F/m
K = 8e6*q; % kinetic energy in eV, transform to joules by multiplying by q
b = z*Z*q^2*cot(pi/4)/(2*K*4*pi*e0);
disp('Impact parameter:')
disp(b)
R = 0.1441e-9;   % atomic radius in meters
disp('Fraction scattered:')
disp(b^2/R^2)
```

```
Impact parameter:
   1.4225e-14

Fraction scattered:
   9.7446e-09
```

Using the formula, we find that $b = 1.423x10^{-14}m$ and the fraction of alpha particles that are scattered is $9.745x10^{-9}$.

Computation: Matlab code for simulating Rutherford's experiment

```
R = 14410; % atomic radius
b = 1.423; % impact parameter
n = 1e8;   % number of random points (x10)
inside = 0; % counter for points inside b
outside = 0; % counter for points outside b

% Generate the random points
    % Express the rectangular coordinates of x and y in terms of the
    % point's radius and its angle relative to the positive x-axis. Take
    % the square root of r1 so that the points will be evenly distributed
    % throughout the circle.

for i=1:10
    for i=1:n
        r1 = rand;
        r2 = rand;
        x = R*sqrt(r1)*cos(2*pi*r2);
        y = R*sqrt(r1)*sin(2*pi*r2);

        if (x^2+y^2) < b^2      % if the point is within radius b add 1 to
            inside = inside+1;  % the number of points inside
        else
            outside = outside+1; % otherwise, add 1 to the number of points
        end                      % outside
    end
end         % the second loop allows for more iterations
```

```
disp('Points inside:')
disp(inside)
disp('Points outside:')
disp(outside)
```

```
Points inside:
     8

Points outside:
   999999992
```

Results and analysis

Code that generates table with data from several runs:

```
expected=9.745e-9;                        % from in-class calculations
nshot = [100; 1000; 5000; 10000; 62200];    % number of points 'shot'
pointsin = [0.5; 9.5; 45.3; 93; 597];   % points that fell inside b
fracin = pointsin./(nshot*1e6);             % fraction of points that fell inside b
fracerror = 1-fracin/expected;            % fractional error
data = horzcat(nshot, pointsin, fracin, fracerror); % all data collected for table
hFig1 = figure('Position',[0 0 600 250]);
hTable = uitable(hFig1, 'Data',data,...   % make table with data
       'ColumnName',{'n (in million)' 'Avg. points inside' 'Avg. fraction inside' 'F
       'Position',[10 10 420 150]);
```

	n (in million)	Avg. points inside	Avg. fraction inside	Fractional error
1	100	0.5000	5.0000e-09	0.4869
2	1000	9.5000	9.5000e-09	0.0251
3	5000	45.3000	9.0600e-09	0.0703
4	10000	93	9.3000e-09	0.0457
5	62200	597	9.5981e-09	0.0151

Code for plotting average fraction inside vs. number of particles:

```
close(hFig1);
hFig2 = figure('Position',[1 1 500 300]);
plot(nshot,fracin,'.')
axis([0 70000 4e-9 10e-9])
hold on
plot(nshot,fracin)
title('Fig. 2 Average fraction inside vs. number of particles');
```

```
xlabel('Number of particles (x10^6)')
ylabel('Average fraction inside')
hold off
```

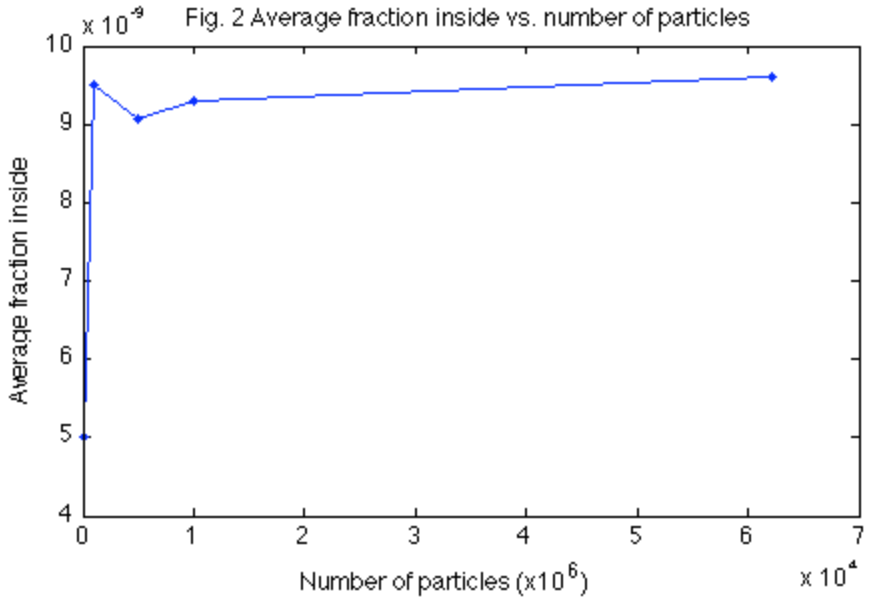
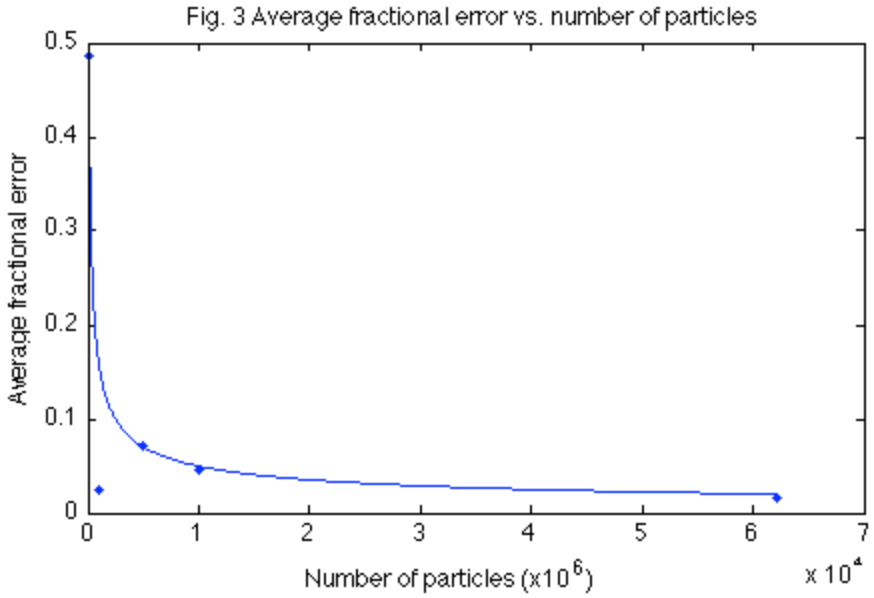


Fig. 2 Average fraction inside vs. number of particles

When n, the number of particles 'shot,' is small the results of the simulation tend to fluctuate because not enough data is being sampled. As n increases, however, the range of results across multiple trials becomes much smaller. Figure 2 shows that the average fraction scattered appears to be converging toward a fraction of approximately $9.0x10^{-9}$, which is close to the expected value of $9.752x10^{-9}$. Additionally, the cumulative total (out of 62.2 billion incident particles) yields a fraction of $9.59x10^{-9}$, which is also very close to the expected value.

Code for plotting average fractional error vs. number of particles:

```
close(hFig2);
hFig3 = figure('Position',[1 1 500 300]);
plot(nshot,fracerror,'.')
axis([0 70000 0 0.5])
hold on
nsqrt = [0:10:62200];
afe = 5./sqrt(nsqrt);
plot(nsqrt,afe)
title('Fig. 3 Average fractional error vs. number of particles');
xlabel('Number of particles (x10^6)')
ylabel('Average fractional error')
clear all
```



As displayed in Figure 3, the average fractional error decreases rapidly with increasing number of particles. Notably, this is very close to an inverse square root relationship. Therefore, I will postulate that AFE ~ $1/\sqrt{(x)}$. According to this model, increasing the number of random points by a factor of x causes the AFE to decrease by a factor of $1/\sqrt{(x)}$. With a significant amount of additional data, it would be possible to test and investigate this relationship.

Conclusion

In this lab, I successfully used the Monte Carlo method in the Matlab computing environment to develop a simulation of the Rutherford experiment. When a sufficient number of points were sampled, the results of the simulation matched expectations very closely. Although a simulation such as this one cannot take the place of real data collection, it can help us validate the inferences that Rutherford made about the structure of the atom based on the data that he collected. Observing the strong correlation between the simulation's data and the expected results, I conclude that the simulation supports Rutherford's deductions. Additionally, I was able to form a reasonable postulate regarding the rate of convergence of the fractional error in the collected data. Further pursuit of this topic could shed light on the nature and limitations of the Monte-Carlo method.

Feedback

As someone who had very little experience with Matlab coming into this course, I think this lab was a great introduction to some of Matlab's key features and its applications in modeling. This lab fulfills the third course objective because it focuses on modeling physics problems on the computer. In addition to the two class periods dedicated to this lab, I spent approximately eight hours collecting additional data, analyzing the data, and writing the lab report. This lab could be improved by adding a more detailed explanation of what is meant by the rate of convergence. From what I've heard, that portion of the lab was somewhat puzzling for myself and others.

Published with MATLAB® 7.11