Physics Project

Haiyan Cheng
Department of Computer Science
Willamette University
Salem, OR 97301
hcheng@willamette.edu

CS-435 Computational Science and Applications

Fall 2016

We adore chaos because we love to produce order.

— M. C. Escher

1 Introduction

In this project you will learn how to do the following things in MATLAB:

1. Generate random numbers to simulate random events.
2. Use an *if else* statement.
3. Write a user defined function.
4. Perform run time test for a numerical simulation.
5. Learn advanced MATLAB plotting techniques and animations in 3-D.

You will learn those MATLAB programming techniques in the context of a physics simulation: simulate Brownian motion of single and multiple particles in one, two and three dimensions using MATLAB.

In the following, I will provide some background with respect to the Brownian motion.

1.1 Introduction to Brownian Motion

The Brownian motion is named after the botanist Robert Brownian. He noticed the pollen suspended in water perform a chaotic dance. He didn't realize that the random fluctuations were caused by the movement of water molecules in the beginning. Later on, he identified that the process is physical and not biological because the same kind of motion was observed among dust particles in a water droplet that had been embedded in quartz for millions of years.

Later that century, J. C. Maxwell, L. Boltzmann and R. J. E Clausius, proposed the kinetic theory of matter and argues that all matter consists of constantly moving atoms or molecules. The movement of the particles are caused by collisions. This constant molecule collision also explains why a drop of dye can completely dissolve in a cup of water without stirring.

Albert Einstein and Marian Smoluchowski both published papers about the Brownian motion and revealed an unexpected connection: the so-called Stokes-Einstein equation for diffusion of spherical particles through liquid with low Reynolds number:

$$D = \frac{k_B T_K}{6\pi\eta r} \tag{1}$$

in which D is the diffusion constant (with unit m^2/sec), k_B is the Boltzmann's constant, T_K is the absolute temperature measured in Kelvin, η is viscosity (with unit $kg/m\,sec$), and r is the radius of the spherical particle (with unit m). The theoretical value of the Boltzmann's constant k_B is:

$$k_B = 1.3806503 \times 10^{-23} \left[\frac{m^2 kg}{sec^2 K}\right] \tag{2}$$

The Brownian motion B_t is a random function of time t, which has a special feature: the displacement from time t to time $t+h$ is normally distributed with mean 0 and variance h.

The physical phenomenon we observe, such as the movement of a particle in liquid or gas, follows a scaled Brownian motion with a scaling factor $\sqrt{2D}$.

2 In-Class Project

2.1 Calculate Theoretical Value of the Diffusion Coefficient

In the first part of the in-class project, we will calculate the theoretical value of the diffusion coefficient using the Stokes-Einstein equation (1) with the following conditions:

- The room temperature is $68°F$.
- The solution is 18% C^{14}-glycerol, which has viscosity 1.66×10^{-3} $[kg/m\,sec]$.
- The C^{14}-glycerol molecule radius is 2.8 Å (Ångstrom), that is, 2.8×10^{-10} $[m]$.

The other fact you should know is the formula to convert °F to °K:

$$T_K = (T_F + 459.67) \times \frac{5}{9} \tag{3}$$

Write a simple function to convert the Fahrenheit to Kelvin. Write another function for calculating the diffusion coefficient, using three input parameters: T_K, η, and r.

2.2 Brownian Motion Simulation

In this part, we will use MATLAB program to simulate a Brownian motion B. We will work on the project with increasing difficulty levels in terms of dimensions.

2.2.1 1-D Brownian Motion Simulation

We will start from the 1-D Brownian motion simulation, in which the particle starts at the origin 0, and can only move in one dimension, with a random displacement. We assume the following

parameters: Diffusion coefficient $D = 1e-10$, time sequence is total one second with equally spaced interval 0.0001 seconds.

Write a MATLAB program to do the following:

1. Generate a time sequence for the planned simulation period.
2. Generate random displacement at each simulation time.
3. Calculate displacement from the starting position.
4. Plot the displacement vs. time with a line plot.
5. Write a for loop to simulate 1-D Brownian motion for 10 particles, each using a different color.

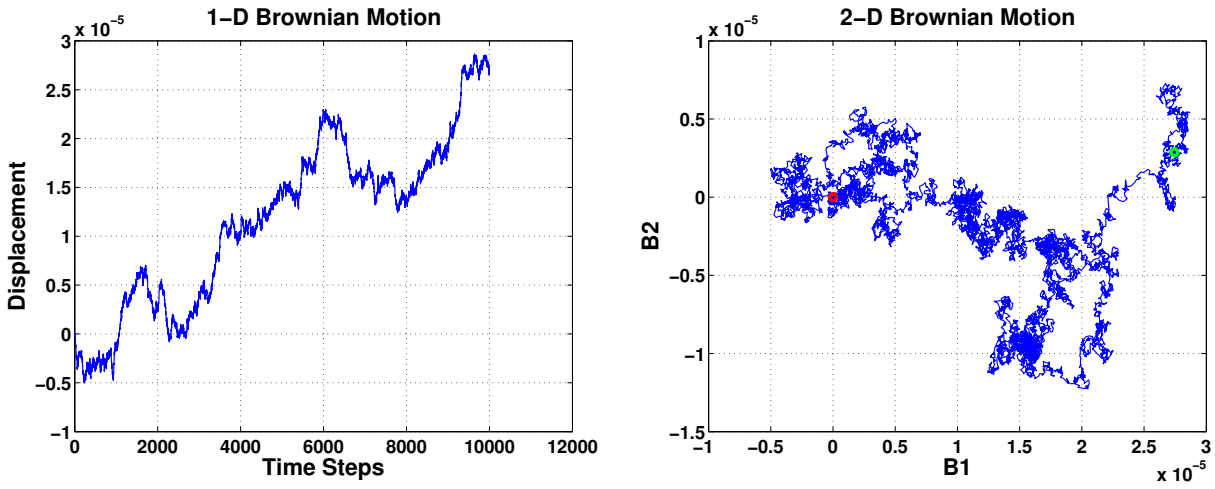

Figure 1. 1-D and 2-D Brownian motion

2.2.2 2-D Brownian Motion Simulation

2-D Brownian motion is the most often observed physical phenomenon in the lab, where you put one drop of solution on a slide and observe under the microscope. In this simulation, we will trace the trajectory of one particle starting at the origin (0, 0), and plot its displacement over time.

Write a MATLAB program to do the following:

1. Generate a time sequence for the planned simulation period.
2. Generate a random displacement at each simulation time along the first dimension for B_1. Generate a random displacement at each simulation time along the second dimension for B_2.
3. For each dimension, calculate displacement from the starting position.
4. Plot the displacement along dimension one vs. the displacement along dimension two with a line plot.
5. Draw a red dot at the starting position and a green dot at the ending position.

2.2.3 3-D Brownian Motion Simulation

Expand the 2-D Brownian motion simulation into 3-D, and repeat the tasks listed in section 2.2.2.

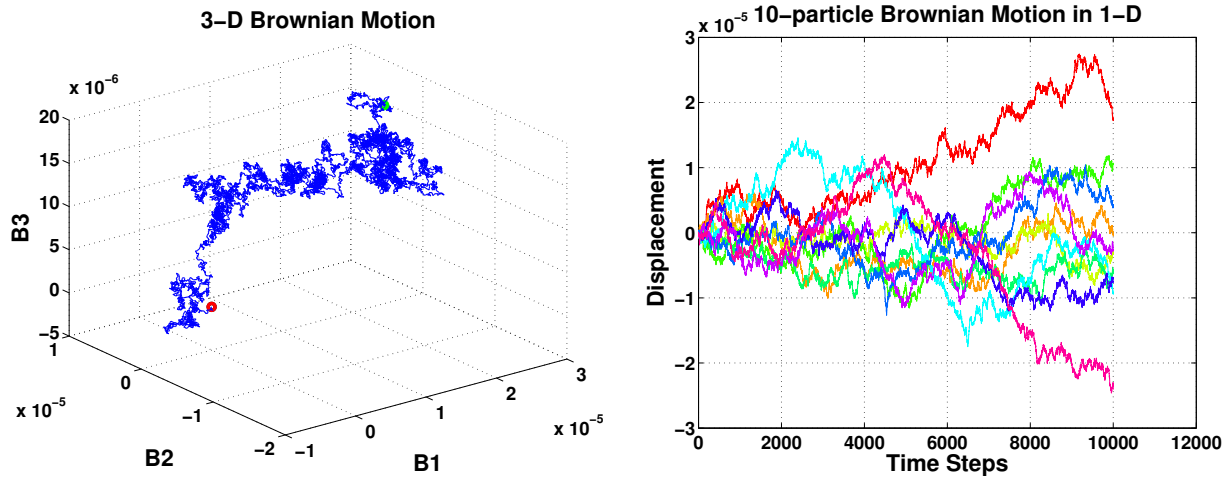

Figure 2. 3-D Brownian motion and 10-particle 1-D Brownian Motion

2.3 Explore MATLAB plotting and simulation functions

In this part, you will simulate particle Brownian motion movements by exploring the MATLAB built-in functions for drawing different shapes: line, rectangle, circle, 3D-sphere, etc. You will generate animated movies and save it in .avi file.

3 Lab Project

3.1 Simulating A 2-D Brownian Motion Hitting A Circular Barrier

Write a MATLAB program to do the following:

1. Based on your 2-D Brownian motion simulation, with N = 10 particles and simulation time T, plot a 2-D traces for each particle with a different color. Mark the end of the trace with a black dot.

2. Predefine a barrier r being the distance of the particle from the origin after a specified time T.

3. Draw a circle centered at (0, 0) with radius r.

4. Plot the Simulated Particles together with the Barrier. To visualize the effect, plot the simulated 2-D Brownian motion for 10 particles together with the circular barrier using "hold on" command. For each particle, use a black dot to indicate its ending position. You can use the following commands to draw a circle centered at origin with radius r:

```
hold on;
t = linspace(0, 2 * pi, 1000);
x = r * cos(t);
```

```
y = r * sin(t);
plot(x, y, 'LineWidth', 2);
axis equal;
```

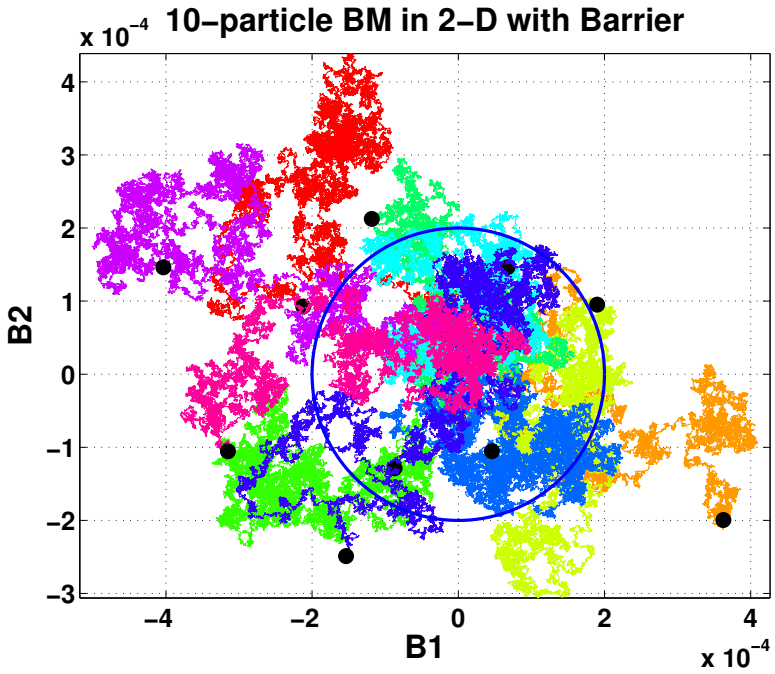

Figure 3. Simulation of a 2-D Brownian Motion Hitting a Circular Barrier.

5. Run the simulation and display on the console how many particles are ended outside of the circle with radius r. To test if a point is outside of the circle, you need to compute the distance of the point from the origin, and compare the distance with the radius of the circle. Given a point with coordinate (x, y), the distance from the origin is $d = \sqrt{x^2 + y^2}$.

6. Verify the result with your plot.

7. Extra: run your simulation for N = 10000, calculate the percentage of the particles that landed outside of the circle.

Your program need to generate the following message in the console:

... out of 10 particles are outside the circle.
... percent out of 10000 particles are outside the circle.

You may use the following to initialize your test:

```
T = 64;      % Temperature
V = 1.32e-3;    % Viscosity
R = 2.8e-10;    % Radius of the particle
h = 0.001;   % Time step size
t = 3;   % Total simulation time
```

```
r = 1e-4;  % Radius of the circular barrier
N = 10;    % Number of particles
```

3.2 MATLAB Animation and Visualization in 3D

Write a MATLAB program to do the following:

1. Generate 4 balls with initial positions at (0, 0, 0). Label them with colors: red, blue, green, and black (you may use MATLAB command `sphere`).
2. For each particle, assign a different variance multiplication factor: (0.1, 0.5, 0.9, 2).
3. Simulate the 4-particle movement and write the output to a .avi file.

Note: For better visual effect, you may simplify the model and adjust parameters as you need.

3.3 Simulate and Plot Distance From the Origin

For 1-D, 2-D, and 3-D Brownian motions, define the distance from the starting point as:

$$\begin{aligned} d_{1D} &= |B_1| \\ d_{2D} &= \sqrt{B_1^2 + B_2^2} \\ d_{3D} &= \sqrt{B_1^2 + B_2^2 + B_3^2} \end{aligned}$$

This is called the Bessel process. Write a MATLAB program to do the following:

1. Use the parameters below to generate your Brownian motion simulation:
   - T = $64°F$.
   - The solution is 64.5% C^{14}-glycerol, which has viscosity 13.2×10^{-3} $[kg/msec]$.
2. Run your 1-D, 2-D, and 3-D Brownian motion simulation for a very long time, compute corresponding distances from the origin using the formula above. Plot time vs. d_{1D}, time vs. d_{2D}, and time vs. d_{3D} in one plot, using different colors for different curves.
3. What did you notice from this plot? Summarize and comment the long-term trend of these three curves and draw conclusions from this comparison.

4 Lab Project Submission

Please create folders for each problem. If there are source data required to run your program, you need to put the data under the same folder, so that as soon as I unzip your code, I can test your code. Please zip all the folders that contain programs you want to submit and name it `YourFirstname_P3.zip`, submit the zipped file to the WISE dropbox before 11:59PM on Oct 12. You need to demonstrate your program and answer questions in order to get credits.