EE1103 - Numerical Methods

QUIZ 2

Simulation of movement of a particle under an optical trap

By:

Vibhhu Sharma(EE19B128)

Sharansh R(EE19B116)

Adityan CG(EE19B003)

Optical tweezers(OT) are scientific instruments used to trap wide range of particles by tightly focusing laser beam.

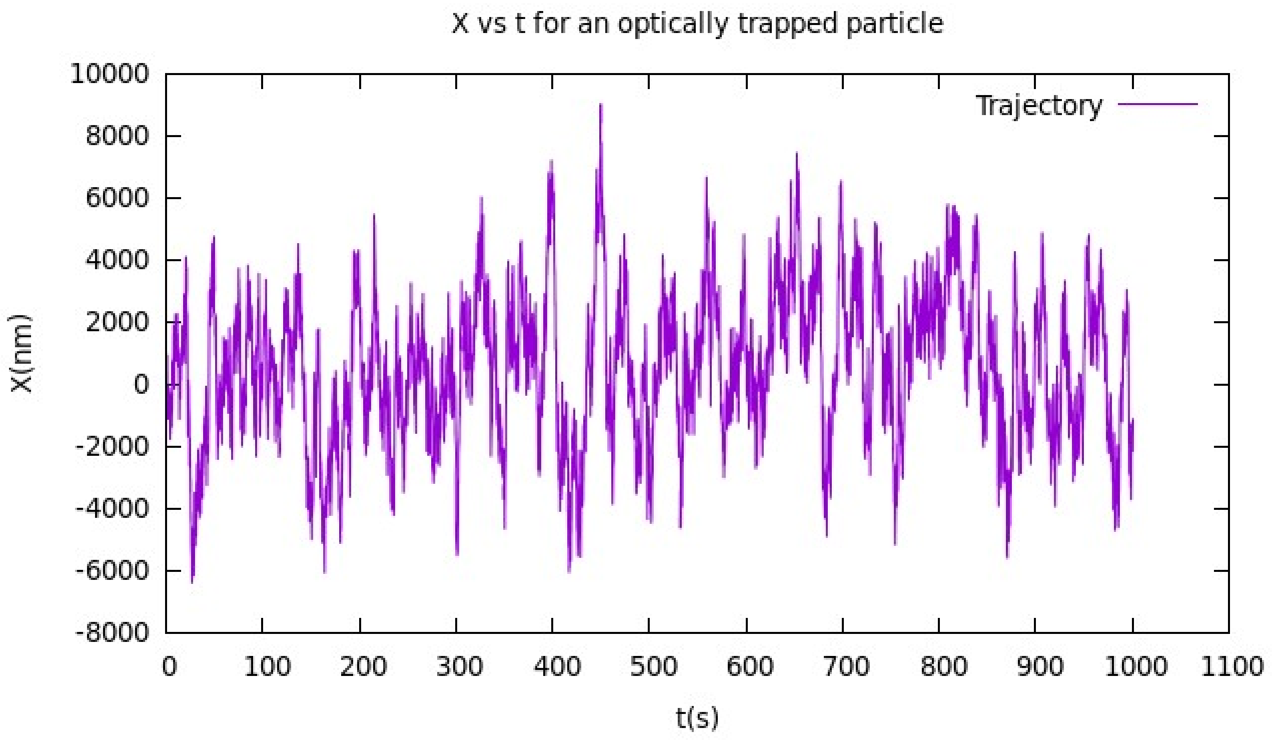
Typically near infrared wavelengths are used to trap to avoid absorption by water and also to avoid damaging biological samples.

The motion of the trapped particle can be described by Langevin equation as shown below:

$$\text{gamma}*\dot{b} + \text{kappa}*b = F(\text{thermal})$$

Where is the drag coefficient (gamma= 3*pi*eta*D), on particle of size D, eta is viscosity of the medium and F random thermal forces with white noise floor given by 2*gamma*kb* T .

1. Motion of the particle of size = 100 microns at a temperature = 300 K with total steps = 10000



This graph shows the trajectory of the paticle under an optical trap in one dimension.

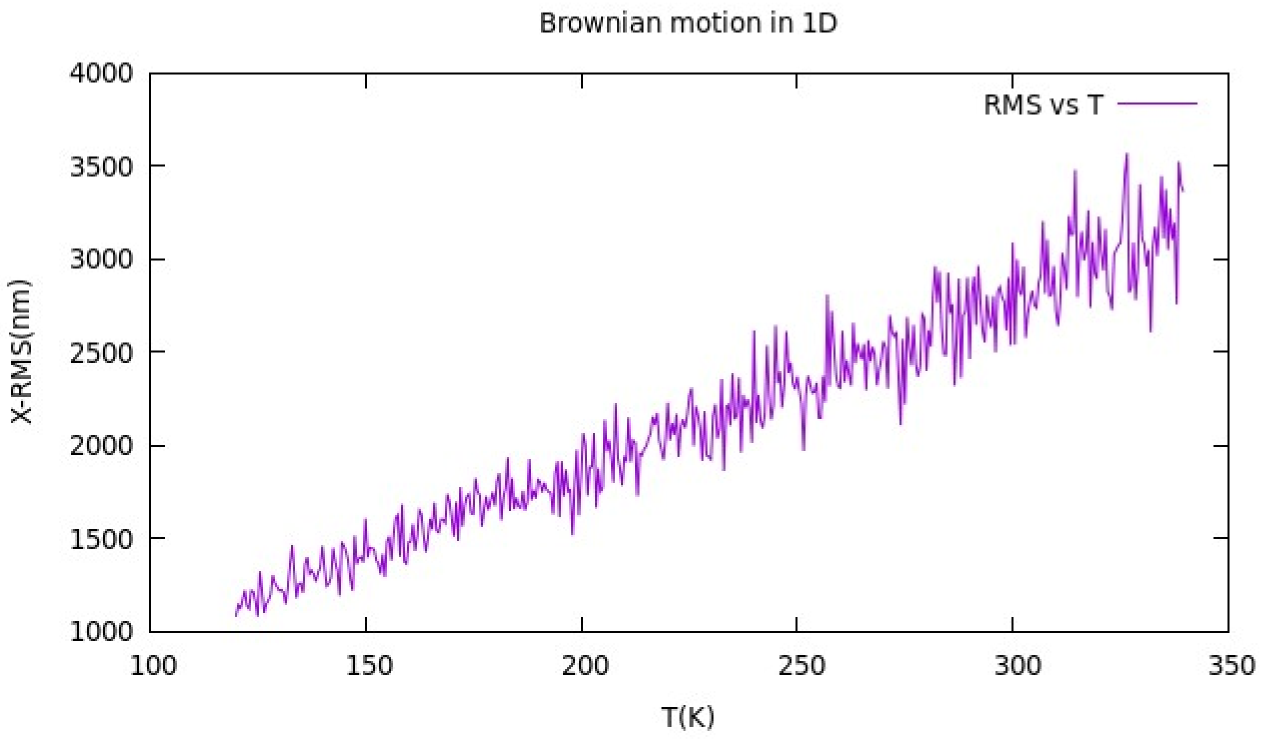
The value of the particles position oscillates about the initial position as depicted by the above plot.

Since the force varies randomly between two extremes(one positive and one negative), therefore the character of this graph is justified.

It is not entirely increasing(as would be the case for positive F) neither entirely decreasing(case for negative F).

There is also maximum bound upto which the particle can oscillate from the mean position.

2. Plot of the RMS value of x over a range of temperature.



Clearly the RMS value of the particle's position increases as a function of temperature. This can be understood by considering the fact that Brownian motion of any particle increases significantly as temperature is increased (particles gain energy and oscillates more rapidly).

Explanation of code:

Units used: Length=nm

Mass=milligram

The code employs a **random normal distribution** in order to generate values between WhiteNoiseMax(2*gamma*kb*T) and WhiteNoiseMin(-2*gamma*kb*T). This is implemented by the uni_rand() and randforce() functions.

The main working of the program involves solving the differential equation in x(b in code) using 4th order Runge_Kutta method. We have used this method keeping in mind its superiority over other methods like euler/heuns.

The *governing differential equation* we need to solve is:

(gamma)b' + (kappa)b = F th ....Langevin Equation

where F th varies between WhiteNoiseMax and WhiteNoiseMin

Function retb()

returns the value of the differential equation whenever it is called.

Function runge()

implements the 4th order runge-kutta algorithm to solve the differential equation. 4 runge-kutta variables(k1,k2,k3,k4) are initialised for this purpose.

5 inputs are taken from the user

Five inputs required: Size of the Particle (in nanometers)
Temperature (in Kelvin)
Temperature_start (in Kelvin)
Temperature_stop (in Kelvin)
Number of steps

An appropriate error is shown to the user in case the number of arguments entered by him do not match.
The units are with length in nanometre mass in milligrams, time in seconds.

We have made use of a **normal distribution** to generate a random floting point number to represnt force between the two extremes of WhiteNoiseMax and WhiteNoiseMin

First part of program

Plotting graph of x vs time

The differential equation in x is solved by **runge-kutta method** and the time is continuously *incremented by 0.1 seconds after each iteration.*
The number of iterations have been taken as input from the user.
At the end of each iteration, the corresponding values of x and time are printed onto the gnuplot file which is opened automatically at the time of program execution.
Standard file handling methods like *fprintf and popen* have been used for this purpose.

Second part of program

Plotting graph of x-rms vs temperature

The user is required to input a starting and ending temperature.
A while loop is used to run iterations from the starting temperature till the ending temperature. The temperature is incremented by 1 degree celsius after each iteration.
The runge() function returns the rms value of x and this is printed alongside the teperature at that instant into another gnuplot file.

An interesting fourth integer argument is passed to the runge function. Its only purpose is to ensure that the gnuplot file for x vs time is plotted only for the first part of the program. Thus, this variable is set to zero for the first part of the program(when that graph is required) and 1 for the second part(when the rms vs T graph is required).