Project 1

Francesc Bagur

October 26, 2025

This is the report on Lab. 1 of the subject of Non-Equilibrium Statistical Mechanics. The goal of this Lab is to implement Brownian motion in code and to test the Fluctuation-Dissipation Theorem.

1 Rewriting the Langevin equation

Given the Langevin equation $\gamma\dot{r}_i(t) = \eta_i(t)$, we are told that $\eta_i(t)$ is a Gaussian white noise that verifies

$$\langle \eta_i^\alpha(t)\rangle = 0 \text{ and } \langle \eta_i^\alpha(t)\eta_j^\beta(t')\rangle = 2\Gamma\delta(t-t')\delta_{\alpha\beta}\delta_{ij}.$$

We begin by defining $\xi_i(t) = \frac{\eta_i(t)}{\sqrt{2\Gamma}}$.

We can proof that this new stochastic variable has zero mean and unit variance,

$$\langle \xi_i^\alpha(t)\rangle = \frac{1}{\sqrt{2\Gamma}}\langle \eta_i^\alpha(t)\rangle = 0,$$

$$\langle \xi_i^\alpha(t)\xi_j^\beta(t')\rangle = \frac{1}{2\Gamma}\langle \eta_i^\alpha(t)\eta_j^\beta(t')\rangle = \frac{1}{2\Gamma}\cdot 2\Gamma\delta(t-t')\delta_{\alpha\beta}\delta_{ij} = \delta(t-t')\delta_{\alpha\beta}\delta_{ij}.$$

If we substitute $\eta_i(t) = \sqrt{2\Gamma}\,\xi_i(t)$ in the Langevin equation, we get

$$\gamma\dot{r}_i(t) = \sqrt{2\Gamma}\,\xi_i(t)$$

where $\xi_i(t)$ is Gaussian white noise with zero mean and unit variance.

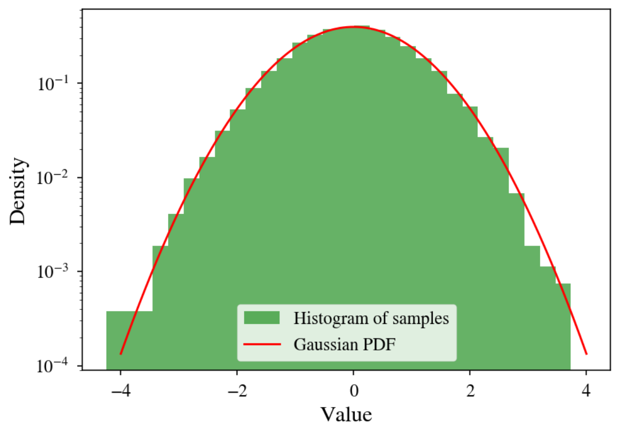

Figure 1: Histogram of $N = 10^4$ random Gaussian numbers generated with the Box-Muller algorithm. The PDF of a $\mathcal{N}(0, 1)$ Gaussian is also plotted for comparison.

2 Generation of Gaussian numbers

We generated $N = 10^4$ Gaussian distributed random numbers using the Box-Muller algorithm. For more information on this method see Appendix B.

The results of this is shown in Fig. 1. Where it can be appreciated that samples are distributed according to a Gaussian distribution with unit variance and zero mean. Moreover, if we calculate numerically the standard deviation and the mean we get the following values

$$\mu = -0.011 \pm 0.010, \qquad \sigma = 0.989583.$$

3 Single particle numerical integration

In this section, we numerically integrate the Langevin equation using what is called the *Euler-Mayurama algorithm*, which can be thought of as the analogous of the Euler method for the initial value problem in the presence of noise.

The final equations we will integrate are

$$x_i(t + \delta t) = x_i(t) + \frac{g_i^x}{\gamma}\sqrt{2\Gamma \delta t}, \tag{1}$$

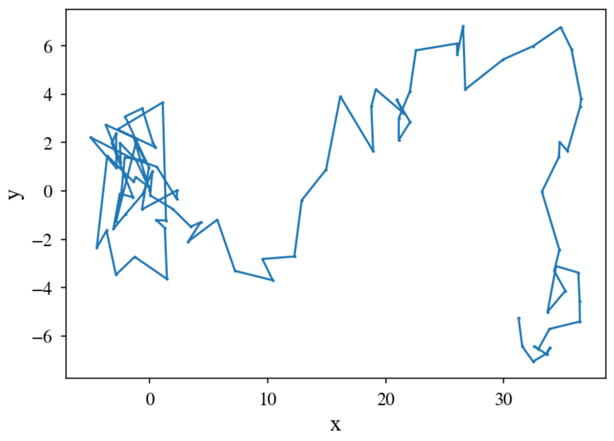

Figure 2: Trajectory of a Brownian particle integrated according to EQs. 1 and 2 for $\tau = 100\delta t$ time steps. With an initial position (0,0), and parameter values $\gamma = 1$ and $\Gamma = 1$.

$$y_i(t + \delta t) = y_i(t) + \frac{g_i^y}{\gamma}\sqrt{2\Gamma\delta t}. \tag{2}$$

Where g_i^α are independent random numbers picked from the normal distribution $\mathcal{N}(0, 1)$ of zero mean and unit variance.

With this equations, we now can simulate the trajectory of a Brownian particle on an open box. We show the result of this simulation in Fig 2, where we have plotted the trajectory of our particle for $\tau = 100\delta t$ time steps.

4 Multiple particles and PBC

In this section we are asked to implement periodic boundary conditions (PBC), so our particles now lay in an $L \times L$ box with $L = 100$. Our implementation of PBC is equivalent to folding our simulation space into a torus, where if a particle escapes through one of the walls, it appears again on the opposite side.

Now, instead of simulating a single particle, we will include $N = 1000$ particles with random initial conditions. In Fig. 3 we plotted the initial and final configurations of these particles after a time $\tau = 100\delta t$.

In this lab, we are asked to discuss which value of δt is more suitable for this simulation. We know that in general when doing numerical integration, we gain precision by decreasing the the time step, but we also lose in performance as we have to perform more operations to integrate the same absolute time. In our case computation efficiency does not matter, as we only simulate up to a $100\delta t$ no matter how big or small those δt are. So, the only differences we observe by varying δt is a *zoom* effect on the trajectories, or how large they appear in comparison with L. For this simulation we chose to use $\delta t = 1$, because we thought that the trajectories were comparable to L, in a way that the trajectory of a single particle can easily be followed if observed frame by frame, but also, some trajectories escaped the box to make use of the PBC.

5 MSD and diffusion coefficient

In the final section, we are asked to compute the Mean-Square displacement (MSD) averaged over N particles given by the equation

$$\Delta^2(t) = N^{-1} \sum_{i=1}^{N} \langle (\mathbf{r}_i(t) - \mathbf{r}(0))^2 \rangle, \tag{3}$$

for $\Gamma = 0.1, 1, 3, 10, 30, ...$ We are also asked to calculate the relationship between Γ and the diffusivity D, defined as

$$D = \lim_{t\to\infty} \frac{\Delta^2(t)}{4t}. \tag{4}$$

The linear relationship between the diffusion coefficient D and the noise amplitude Γ (Fig.4) confirms the theoretical prediction from the Langevin equation. From the lecture notes we know that $D = K_B T/\gamma$, where we can identify $K_B T = \Gamma$ as both terms simply control the amplitude of the noise. Our linear fit yields a slope of 1.015 ± 0.008, which (almost) matches the expected value of $1/\gamma = 1$ within numerical precision ($R^2 = 0.9996$). This validates our implementation of the Euler-Maruyama algorithm and demonstrates that the mean-square displacement grows linearly with time, $\langle \Delta^2(t) \rangle \propto t$, which is the typical diffusive behaviour.

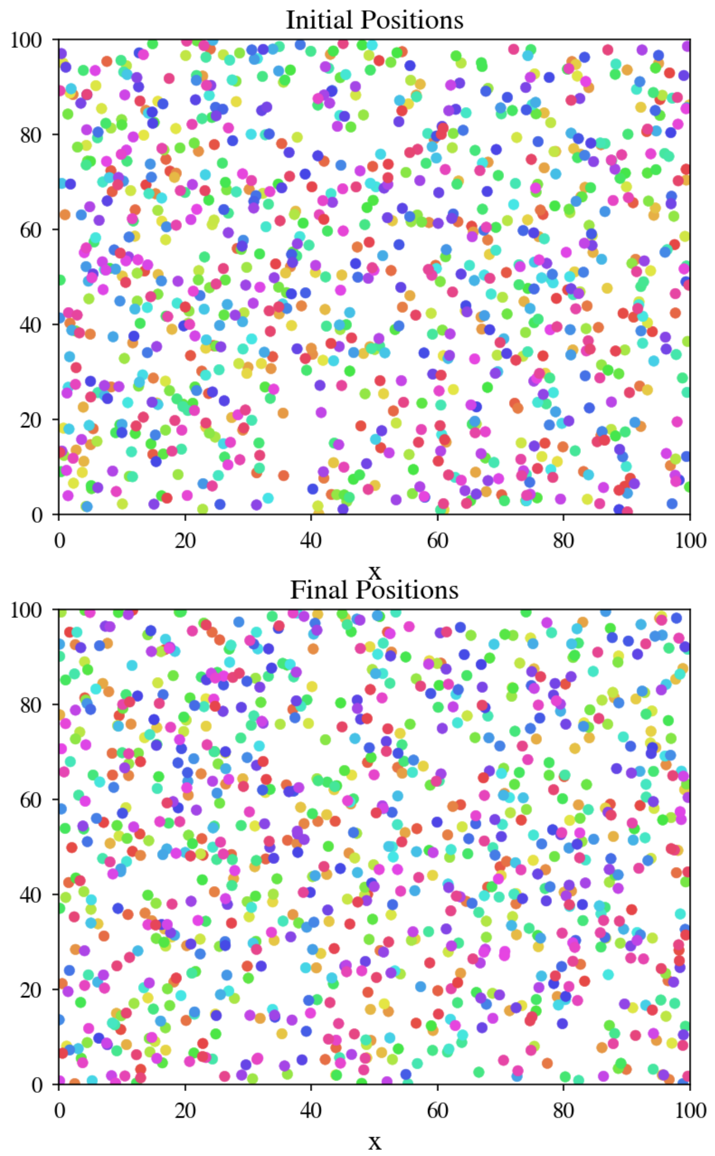

Figure 3: Initial and final configurations of $N = 1000$ non-interacting Brownian particles in an $L \times L$ box, $L = 100$, with PBC, after $\tau = 100\delta t$. With parameters $\gamma = 1$, $\Gamma = 1$ and $\delta t = 1$.

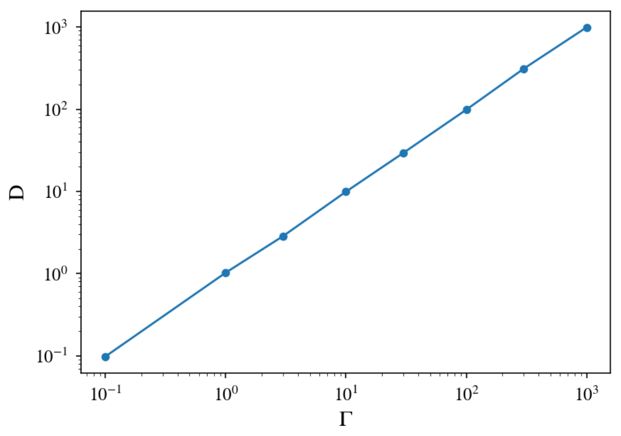

Figure 4: Diffusion coefficients as a function of Γ in log-log scale. Calculated from the MSD averaged over $N = 1000$ particles for $\tau = 100\delta t$. With parameter values of $\gamma = 1$ and $\delta t = 1$

A Error calculation

Only one error calculation was performed in this study. We calculated the error of the mean of our Gaussian distributed numbers in the following way,

$$\sigma_{\bar{x}} = \frac{\sigma_x}{\sqrt{N}}.$$

With $N = 10000$, and σ_x the sample standard deviation.

B Box muller algorithm

The Box-Muller algorithm [1] is a simple way to generate Gaussian distributed numbers from a uniform distribution. To implement it, we generate two random numbers, U_1 and U_2, from the uniform distribution $U[0, 1]$.

With this two numbers, we calculate

$$Z_0 = \sqrt{-2 \ln U_1} \cos(2\pi U_2), \tag{5}$$

$$Z_1 = \sqrt{-2 \ln U_1} \sin(2\pi U_2). \tag{6}$$

Then, Z_0 and Z_1 are independent random variables distributed as (0,1) Gaussian.

C Code availability

All code that was used in this lab is available in the following repository P1_noneq.ipynb.

References

[1] Box, G. E. P., & Muller, M. E. (1958). A note on the generation of random normal deviates. *The Annals of Mathematical Statistics*, 29(2), 610–611.