1 TPS in a simple 2D potential

1.1 Introduction

This exercise lets you explore state and trajectory space for a single particle in a two dimensional potential. For this tutorial it is an educational and computationally tractable problem that exhibits some of the problems encountered in higher dimensional systems.

You are expected to write a working code in C (or any other language) that can perform straightforward stochastic dynamics and perform path sampling. We can provide you with a velocity Verlet routine to integrate the Langevin equation (see appendix), a graphics package to plot trajectories and with some other basic routines. If necessary there is a complete working program you can use as an example.

1.2 Potential surface

Consider the potential energy surface as a function of variables x and y:

$$\begin{aligned}\beta V(x,y) &= -e^{-((x-1)^2+y^2)} - e^{-((x+1)^2+y^2)} + 5e^{-0.32(x^2+y^2+20(x+y)^2)} \\ &\quad + \frac{32}{1875}(x^4+y^4) + \frac{2}{15}e^{-2-4y}, \end{aligned} \tag{1}$$

where $\beta = 1/k_B T$ is the reciprocal temperature, k_B the Boltzmann constant, and T the temperature. A contour plot of the potential is given in figure 1. The potential surface shows two minima at $\beta V(0.96, 0.06) = -0.98$ and $\beta V(-0.98, 0.06) = -0.97$ and one saddle point at $\beta V(-1.96, 196) = 0$. The surface shows the characteristics of the bistable potential mentioned in the first day's lectures. That is, considering x is the only important order parameter and integrating out all the other degrees of freedom (in this case only y) the maximum of the free energy curve is NOT located at the saddle point of the surface. Further, to go over the barrier a particle first has to move in the opposite direction.

1. Calculate the free energy

$$\beta F(x) = -\ln \int dy\ e^{-\beta V(x,y)}$$

and establish the barrier maximum. Conclusion?

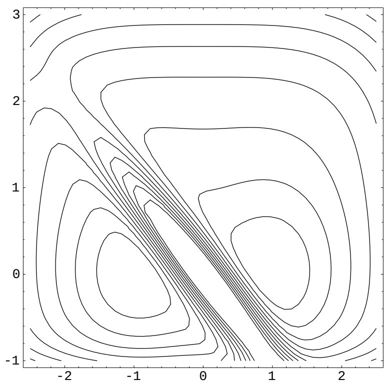

Figure 1: Bistable 2D potential separated by high barrier

1.3 Exploring surface using straightforward dynamics

1. Write a program that performs straightforward Langevin dynamics (see appendix).

2. Run the program and look at trajectories at different temperatures (say $T = 0.1$ to $T = 1$). You can use the following input parameters:

$$\begin{aligned} \Delta t &= 0.25 \\ \gamma &= 2.5 \\ m &= 1 \\ (x_{init}, y_{init}) &= (-1, 0) \end{aligned}$$

   Try also a few other initial coordinates or Δt, β and γ.

3. Write the trajectories to a file and look at them using a plot program, and calculate mean and variance of order parameters (x,y).

4. Write down an expression for the stable region characteristic function $h_A(x, y)$ and $h_B(x, y)$ for $\beta = 8$. This means that most (say 99%) of the equilibrium distribution is captured by these function. (make sure you use both variables in the expression).

1.4 Path sampling

1. Write a program the performs path sampling of full Langevin trajectories.
   Tips: put whole path in memory, not only configuration at time t. Use both shooting and shifting, backward and forward (four different routines).

   The first slice should be confined to the initial state A (choose one) and last slice should be confined to the final region B.

   The initial path can be constructed by

$$\begin{aligned} x &= 2 * \tau / L - 1 \\ y &= 0 \end{aligned}$$

   with τ the time slice index

2. Run the program interactively using the same parameters as before and using the $h_{A,B}$ from the previous exercise The path length (number of time slices) should be about $L = 100$, $\beta = 8$. Look how the trajectories relax to equilibrium using the graphics module. Measure the shooting and shifting acceptance ratios, are they reasonable $(= \pm 0.5)$? Play around with different settings of β, γ etc.

3. Try different definitions of the stable regions e.g.

$$\begin{aligned} h_A(x_0) &= \begin{cases} 1 & \text{if } x < -0.5 \\ 0 & \text{if } x > -0.5 \end{cases} \\ h_B(x_0) &= \begin{cases} 1 & \text{if } x > 0.5 \\ 0 & \text{if } x < 0.5 \end{cases} \end{aligned}$$

   What happens?

4. Change the number of time slices to 50 or even 10 while keeping everything else constant. Any difference?

1.5 Sufficient path length

1. Measure the time correlation function

$$C_{AB}(t) = \frac{< h_A(x_0) h_B(x_t) H_B(x) >}{< h_A(x_0) H_B(x) >} =< h_B(t) >_{AB} \tag{2}$$

where $H(x) = 1$ if the path visits the final region B at least once. (Note that you will have to change the shooting and shifting acceptance criteria in the program so that the last slice is able to move out of B again after it has visited it.

2. Plot c_{AB} and $\dot{c}_{AB} =< \dot{h}_B(t) >_{AB}$. Does it reach a plateau? If not, double the number of time slices and repeat path sampling.

1.6 Rate constant

The rate constant for the transition can be calculated from the linear part of time correlation function

$$C(t) \equiv \frac{\langle h_A(x_0) h_B(x_t) \rangle}{\langle h_A \rangle} \approx k_{A \to B} t \quad \text{for} \quad \tau_{\text{mol}} < t \ll \tau_{\text{rxn}}. \tag{3}$$

$C(t)$ can be obtained from path sampling by

$$C(t) = \int d\lambda \, P(\lambda, t), \tag{4}$$

where $P(\lambda, t)$ is the probability to find slice x_t at an order parameter λ

$$P(\lambda, t) \equiv \frac{\int dx_0 \, \rho(x_0) h_A(x_0) \delta[\lambda - \lambda(x_t)]}{\int dx_0 \, \rho(x_0) h_A(x_0)}, \tag{5}$$

where λ is an order parameter that defines the final region B.
To determine $P(\lambda, t)$ by umbrella sampling, we first define a sequence of $N+1$ overlapping regions $B[i]$ (windows) such that $B[0] = B$ and the union $\bigcup_{i=0}^{N} B[i]$ of all regions comprises the whole phase space. The regions $B[i]$ are defined through

$$x \in B[i] \quad \Leftrightarrow \quad \lambda_{\min}[i] \leq \lambda(x) \leq \lambda_{\max}[i]. \tag{6}$$

For $0 < i < N$ we require region $B[i]$ to overlap with the neighboring regions $B[i-1]$ and $B[i+1]$. Next, one calculates the distribution of the order parameter λ in each of the windows $B[i]$ separately:

$$P(\lambda, t; i) \equiv \frac{\int dx_0 \, \rho(x_0) h_A(x_0) h_{B[i]}(x_t) \delta[\lambda - \lambda(x_t)]}{\int dx_0 \, \rho(x_0) h_A(x_0) h_{B[i]}(x_t)}. \tag{7}$$

In words, $P(\lambda, t; i)$ is obtained by path sampling with the final region defined by $B[i]$ and making a histogram of the $\lambda(x_t)$. In practice, one just keeps track of the endpoint x_L.
The order parameter λ has to be chosen carefully. For example, λ can be the x coordinate (although it turns out that this is not a good choice), other possibilities are $\lambda = x + y$, or $\lambda = \arctan(y/x)$. The latter is the best choice.

1. Calculate C(t) by straightforward simulation for a low barrier (eg. $\beta = 3$), extract the rate constant.

2. Compute $P(\lambda, L)$ using 4-8 windows, and $L = 50$, $\beta = 8$ (other parameters same as before). Be sure to confine the endpoint to region B, i.e. use h_B instead of H_B as characteristic function.

3. Match the histograms of all windows by multiplying by a constant (or shifting the logarithm of the histograms by a constant) to obtain $P(\lambda, L)$.

4. Compute $C(L)$ by integrating Eqn(3).

5. Now

$$C(t) = \frac{\langle h_B(t) \rangle_{AB}}{\langle h_B(L) \rangle_{AB}} \times C(L). \tag{8}$$

So it is only necessary to do the umbrella sampling once, as we have $\langle h_B(t) \rangle_{AB}$ from exercise 1.5. In the linear regime $\tau_{\rm mol} < t \ll \tau_{\rm rxn}$

$$\dot{C}(t) = \frac{\langle \dot{h}_B(t) \rangle_{AB}}{\langle h_B(L) \rangle_{AB}} \times C(L). \approx k_{A \to B}. \tag{9}$$

Calculate the rate constant.

Appendix

Langevin integration algorithm

In this paper, we consider classical many-body systems evolving according to the Langevin equation of motion

$$\begin{aligned} \dot{r} &= v, \\ \dot{v} &= \frac{F(r)}{m} - \gamma v + \frac{\mathcal{R}}{m}. \end{aligned} \tag{10}$$

The positions and velocities of all particles are specified by r and v, respectively. F is the intermolecular force derived from the potential $V(r)$, and γ is a friction constant. The random force, $\mathcal{R}$, is responsible for the stochastic character of the time evolution. It is a Gaussian random variable with $\langle \mathcal{R}(t)\mathcal{R}(0) \rangle = 2m\gamma k_{\rm B}T\delta(t)$. For a short time increment, Δt, the assumption of force which depends linearly on r leads to the integration algorithm

$$\begin{aligned} r_{\tau+1} &= r_\tau + c_1 \Delta t \, v_\tau + c_2 \Delta t^2 \, a_\tau + \delta r_R, \\ v_{\tau+1} &= c_0 v_\tau + (c_1 - c_2) \, \Delta t \, a_\tau + c_2 \Delta t \, a_{\tau+1} + \delta v_R, \end{aligned} \tag{11}$$

where $a_\tau = F(r_\tau)/m$ and the coefficients c_0, c_1 and c_2 are given by

$$c_0 = e^{-\gamma \Delta t}; \quad c_1 = \frac{1 - c_0}{\gamma \Delta t}; \quad c_2 = \frac{1 - c_1}{\gamma \Delta t}. \tag{12}$$

δr_R and δv_R are small random displacements caused by the random force. As derived by Chandrasekhar the random displacements are distributed according to

$$\begin{aligned} w\left(\delta x_R\right) &= \left[2\pi\sigma_r\sigma_v\sqrt{1-c_{rv}^2}\right]^{-D} \\ &\times \prod_{\alpha=1}^{D} \exp\left\{-\frac{1}{2\left(1-c_{rv}^2\right)}\left[\left(\frac{\delta r_R^\alpha}{\sigma_r}\right)^2+\left(\frac{\delta v_R^\alpha}{\sigma_v}\right)^2\right.\right. \\ &- \left.\left. 2c_{rv}\left(\frac{\delta r_R^\alpha}{\sigma_r}\right)\left(\frac{\delta v_R^\alpha}{\sigma_v}\right)\right]\right\}, \end{aligned} \tag{13}$$

where $\delta x_R = \{\delta r_R, \delta v_R\}$ and α denotes the different components of the displacement vectors δr_R and δv_R. The variances σ_r and σ_v and the correlation coefficient c_{rv} are given by

$$\begin{aligned} \sigma_r^2 &= \Delta t \frac{k_{\mathrm{B}}T}{m\gamma}\left[2-\left(3-4e^{-\gamma\Delta t}+e^{-2\gamma\Delta t}\right)/\gamma\Delta t\right], \\ \sigma_v^2 &= \frac{k_{\mathrm{B}}T}{m}\left(1-e^{-2\gamma\Delta t}\right), \\ c_{rv}\sigma_r\sigma_v &= \frac{k_{\mathrm{B}}T}{m\gamma}\left(1-e^{-\gamma\Delta t}\right)^2. \end{aligned} \tag{14}$$

Applying Eq. (11) iteratively with random numbers drawn from the bivariate distribution (13) leads to a stochastic trajectory of arbitrary length. This procedure is usually called Brownian dynamics. Each trajectory generated with this algorithm has an associated probability that depends on the sequence of random displacements used to obtain the trajectory.