LAB IX
Simulation of a quantum particle in harmonic potential

Jakub Tworzydlo

Chair of Condensed Matter
Institute of Theoretical Physcis
Jakub.Tworzydlo@fuw.edu.pl

14/05/2024 Pasteura 5, Warszawa

Pseudo-code

procedure `naive-harmonic-path`
input $\{x_0, \ldots, x_{N-1}\}$
$\Delta_\tau \leftarrow \beta/N$
$k \leftarrow \texttt{nran}\,(0, N-1)$
$k_\pm \leftarrow k \pm 1$ **modulo N**

$x'_k \leftarrow x_k + \texttt{ran}\,(-\delta, \delta)$
$\pi_a \leftarrow \rho^{\text{free}}\left(x_{k_-}, x_k, \Delta_\tau\right) \rho^{\text{free}}\left(x_k, x_{k_+}, \Delta_\tau\right) \exp\left(-\frac{1}{2}\Delta_\tau x_k^2\right)$
$\pi_b \leftarrow \rho^{\text{free}}\left(x_{k_-}, x'_k, \Delta_\tau\right) \rho^{\text{free}}\left(x'_k, x_{k_+}, \Delta_\tau\right) \exp\left(-\frac{1}{2}\Delta_\tau x'^2_k\right)$
$\Upsilon \leftarrow \pi_b/\pi_a$
if $(\texttt{ran}\,(0,1) < \Upsilon) x_k \leftarrow x'_k$
output $\{x_0, \ldots, x_{N-1}\}$

———

where $\rho^{\text{free}}(x, y, \Delta\tau) = e^{-(x-y)^2/2\Delta\tau}$ and $\Delta\tau = \beta/N$.

Task I

Simulate a string (quantum path) for the harmonic potential $V(x) = \frac{1}{2}x^2$ by implementing our pseudo-code. Consider 1MCS corresponding to N update trials.

Plot average string position $< x >= \sum_k x_k / N$ and variance $< (x- < x >)^2 >$ versus simulation time (preferably panel plot). Take $N = 8$, $\beta = 4$, $\delta = 1$ and `MCS_max = 10 000`, start with a random configuration $x_k \in [-\delta, \delta]$.

Add horizontal lines to the plot, with mean (over MCS) position and mean (over MCS) variance. What are the values?

Task II

Speed up the simulation by implementing the update routine (Task I) under `numba` package. On a few trial runs estimate the acceptance ratio of Metropolis step. Update (by hand) $\delta_{\rm new} = (r_{\rm acc}/0.75)\delta$ untill you find a value for which $r_{\rm acc} \approx 0.75$.

As before, plot average string position $< x >= \sum_k x_k/N$ and variance $< (x- < x >)^2 >$ versus simulation time (preferebly panel plot). Take $N = 40$, $\beta = 10$, and `MCS_max = 100 000`. Store positions: x_0 and $x_{N//2}$ throughout the simulation.

Make normalized histogram of the stored values of positions. Compare with the analytic solution for the harmonic oscillator:

$$\pi(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-x^2/2\sigma^2}, \quad \sigma^2 = \frac{1}{2\tanh\beta/2}.$$

Numba package

You can speed up your simulations considerably by using the `numba` package.

Consider the example https://numba.pydata.org/numba-doc/dev/user/5minguide.html.

Write a function `update(x,delta,dtau,N)` that performs 1MCS on a `numpy` array `x` of positions. Accelerate it with the decorator `@jit(nopython=True)`. The function can contain all arithmetic operations, mathematical functions from `numpy` and operations on `numpy` arrays. Accepted random number generators are `random.uniform` and `random.randint`.

Installation of the package is easiest with `pip3 install numba`. In the computer lab (if all fails), you need to downgrade:
`pip3 install llvmlite==0.31 --user` and then
`pip3 install numba==0.46 --user`.

Extra Task (rather for home)

Read about Levy's algorithm from the Krauth book. Try to understand first `levy-free` and then `levy-harmonic` procedures. Write, test and run the code for `levy-harmonic`. Illustrate the agreement with the theoretical $\pi(x)$. How many MCS steps already give a reasonable data?

Hint. After 1MCS (which is done by getting $x_1, \ldots, x_{N-1}$ from Levy's algorithm) "roll" the result such, that $x_{N//2}$ becomes x_0.