Theory Computing Project Notebook

Tingyu Chen

School of Physics and Astronomy

University of Manchester

February 2025

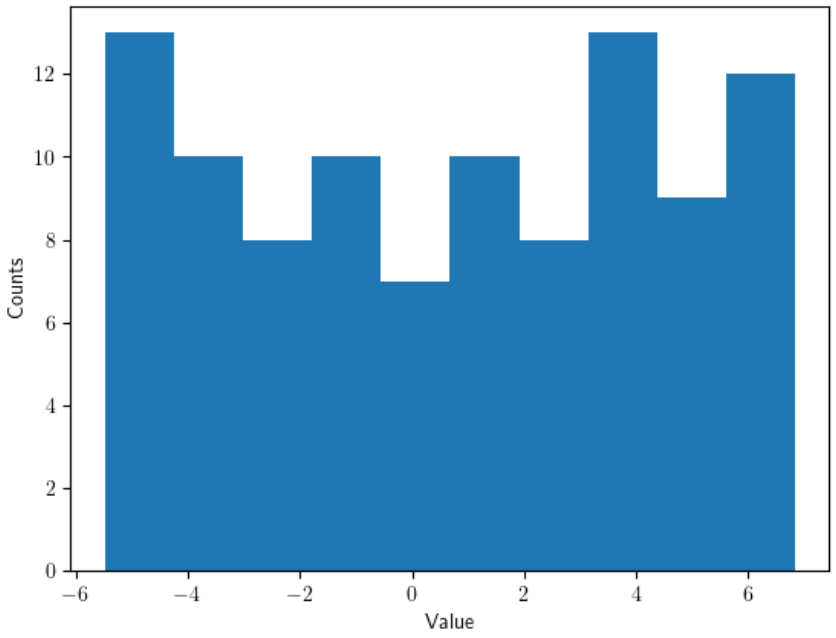
Week 2 - Monte Carlo methods:

Task 1: Uniform distributions:

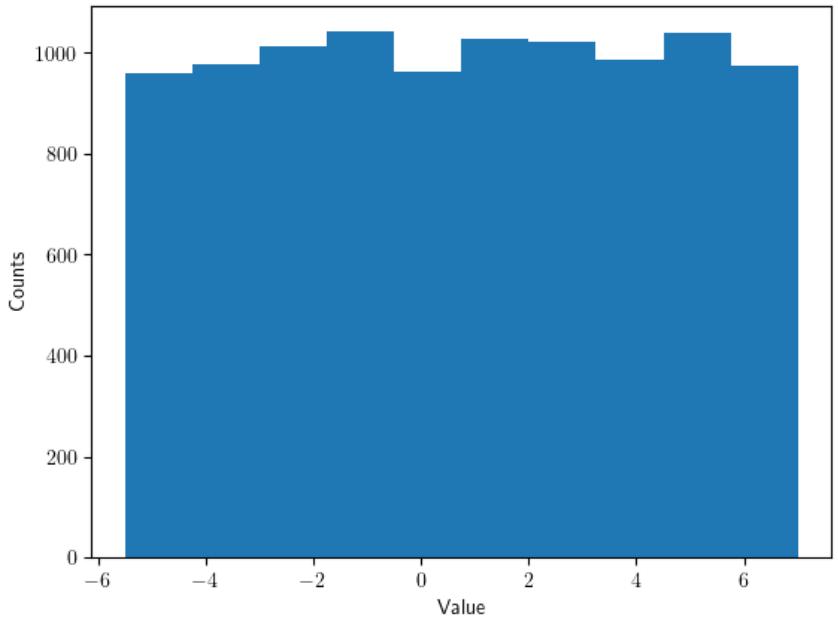
numpy.random.uniform() allows us to generate a random number between 0 and 1, three optional arguments: low, high, size, are provided.

Example 1:

Change the number of points in the random sample and inspect the fluctuation in the histogram. Can you explain what you see?



For less number of points, the histogram appears noisy and irregular. The randomness of the sample dominates, leading to large fluctuations. Whereas for more number of points, the histogram becomes smoother and more representative of the true uniform distribution. The fluctuations reduce as more data points contribute to each bin.



Example 2:

As explained in the notes we can estimate an integral using random variables:

$$\int_{x_1}^{x_2} f(x)\mathrm{d}x \approx (x_2 - x_1)\frac{1}{N}\sum_{i=1}^{N} f(x_i).$$

Let's first integrate a simple $y = x$ line between 0 and 10. This is given by

$$I = \int_0^{10} x\mathrm{d}x = \left[\frac{x^2}{2}\right]_0^{10} = \frac{10^2}{2} = 50$$

Change the number of points and inspect the accuracy of the expectation value.

size=1000, expectation=48.9002

size=10000000, expectation=49.9926

Use the same sample to estimate the standard deviation of the random sample. Use the above result to find the error on the integral.

'Integral Estimate': 49.96029890697102, 'Standard Deviation of f': 2.8902692349470627, 'Monte Carlo Error': 0.09139833833545048

Bonus question: Test the speed of the numpy.random.uniform function. How much faster is it to use the size argument compared to repeatedly calling the function (e.g. appending values to an array in a for loop)?

'Time using size argument (s)': 0.00849294662475586, 'Time using for loop (s)': 0.8912241458892822, 'How much faster': 104.93697714895289

Task 2: Rejection method:

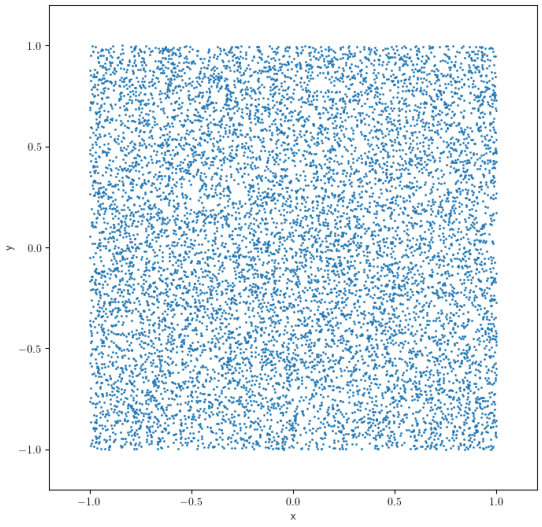
Example 1:

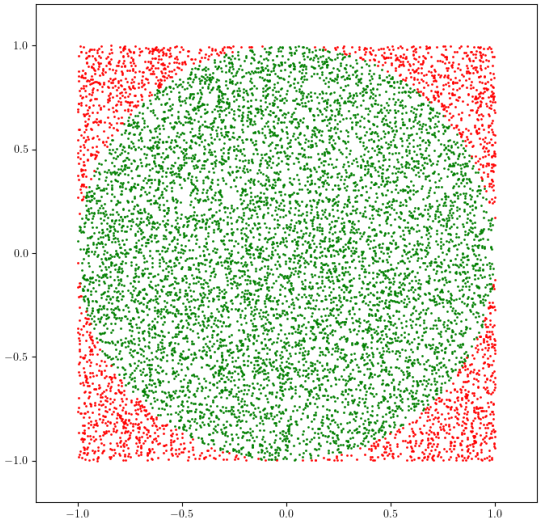
A classic example of a Monte Carlo integral with rejection method is to calculate a value of π.

The idea is to use what we know about the area of the circle and a square that encases it. A box must be of length $L = 2R$ to encase a circle of radius R.

$$\frac{A_{\text{circle}}}{A_{\text{square}}} = \frac{\pi R^2}{L^2} = \frac{\pi R^2}{4R^2} = \frac{\pi}{4}$$

All that is missing is to estimate the ratio of areas using MC techniques! Let's first populate a square spanning $[-1, 1]$ in both x and y.





· Use the samples above to estimate the value of π with the formula given at the start of the notebook.

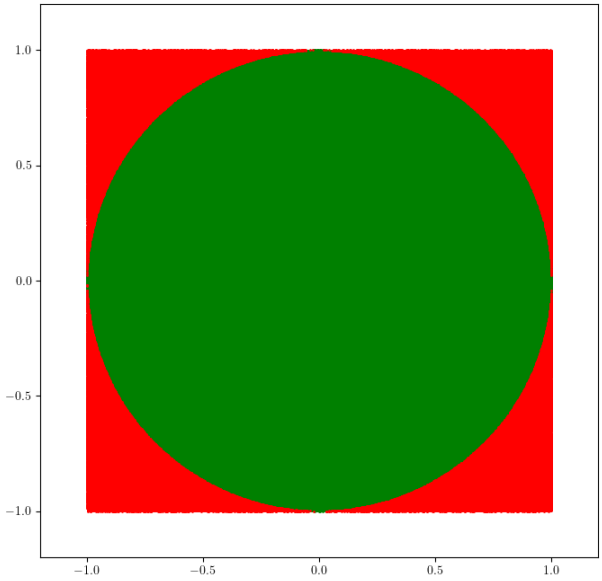
pi estimate $= 4 * (insidecircle/10000)$, pi estimate $= 3.1244$

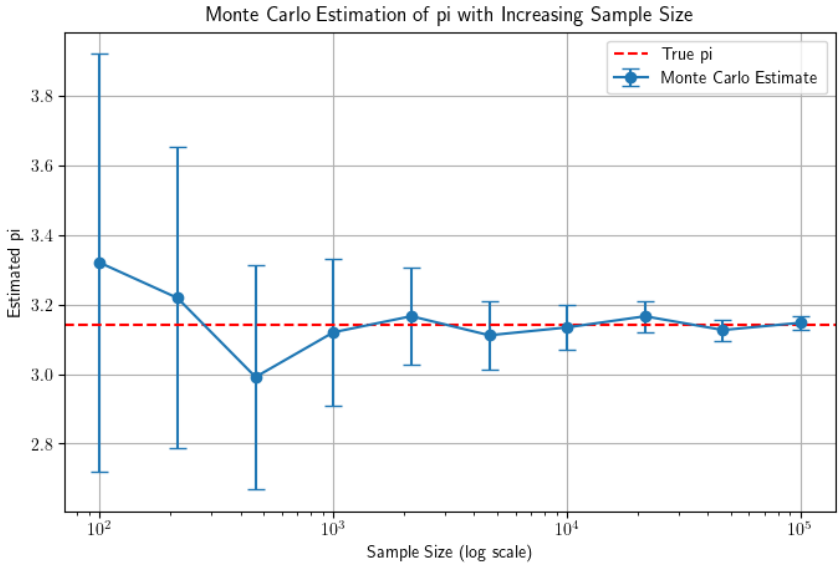
· Can you estimate the error on the value of π?

0.06616010447391993

· Calculate π with growing sample size, and demonstrate the method converges towards the expected value.

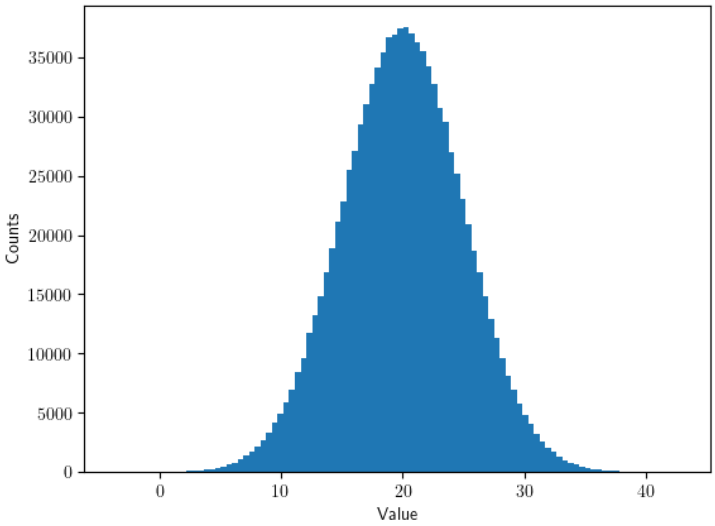
N=10000000, pi=3.1415704





Task 3: Non-uniform distributions:

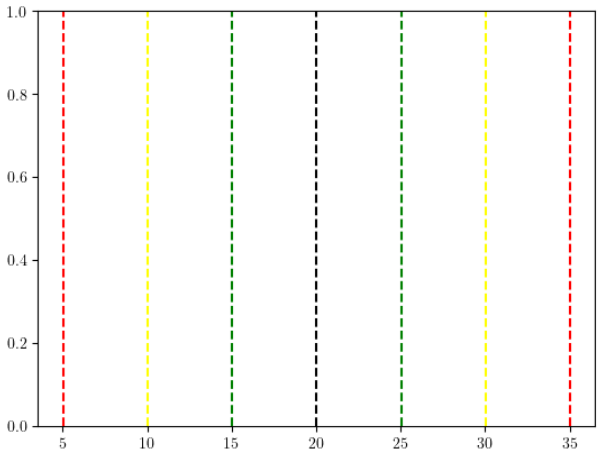
Example 1:



· Use MC integration to estimate the mean and standard deviation of the sample (pretend you don't know the values from the above cell!).

20.0278659111271, 4.996728002036475

· Plot them below using the following code snippet (replace with variable names you prefer)



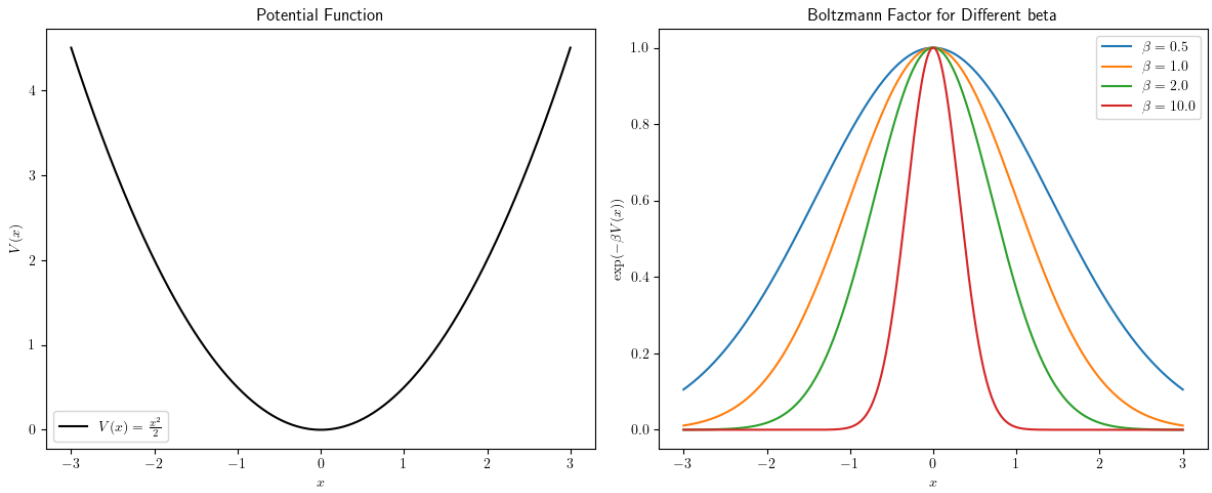
· Once confident your estimate of the standard deviation is sensible (does it match what you plugged in?) use that to estimate the probability of a point lying between the $\pm 1\sigma$ bounds. Does the value match your expectation? What about $\pm 2\sigma$ and $\pm 3\sigma$.

Yes, it does.

Task 4: Metropolis Algorithm:

Example 1:

· Using the potential $V(x) = x^2/2$ produce a figure showing both the potential and Boltzmann factor in the same figure.



· Show the plot for various values of β. What do you expect to happen to the physical system at low temperature (high β) and high temperature (low β)

At a higher β (i.e. $\beta = 10$, corresponding to low temperature), the Boltzmann factor is very narrowly peaked around $x = 0$ (the minimum of $V(x)$), indicating that the particle is most likely to be found near the minimum.

At a lower β (e.g. $\beta = 0.5$, corresponding to high temperature), the distribution is more spread out, meaning the particle can more easily explore regions further from the minimum.

The Metropolis algortithm – a method for choosing a new value of x based on the current position. This creates a chain of x values that samples the probability distribution.

Metropolis algrothim:

1) Start by picking a random point x_0 from a ¡i¿sensible¡/i¿ range.

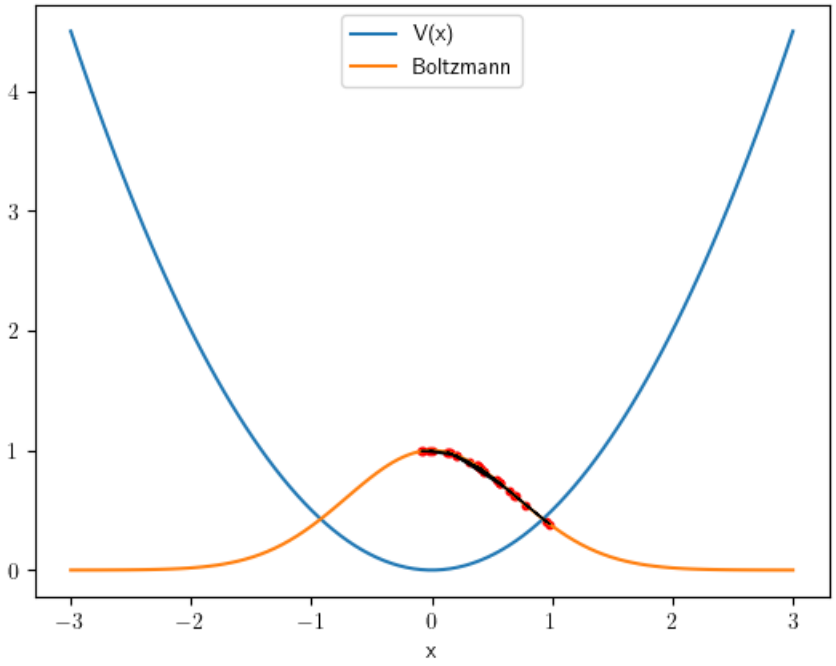
2) Record your position in your data structure of choice.

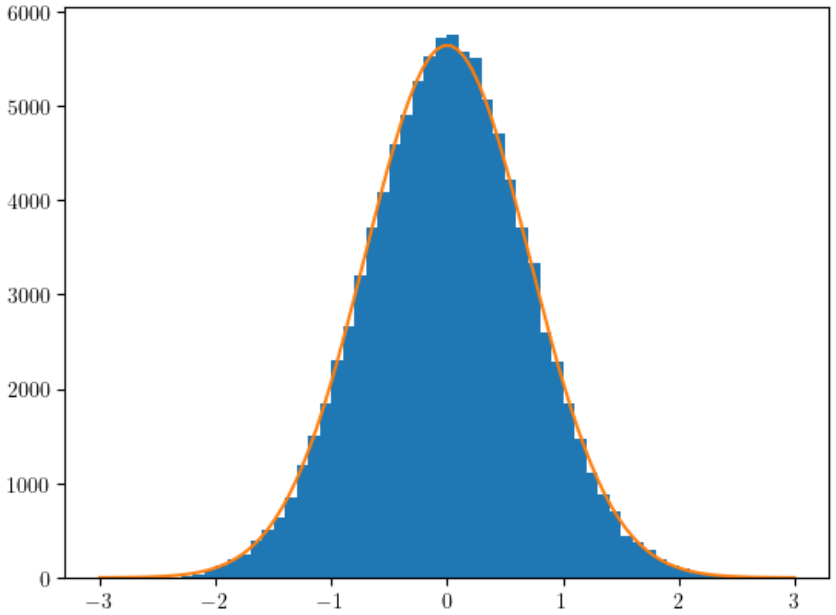
3) Propose a step by choosing a δx, again from a ¡i¿sensible¡/i¿ distribution.

4) Evaluate the ratio of probabilities $\alpha = P(x + \delta x)/P(x)$. If this ratio is favourable ($\alpha \geq 1$) then take the step. If it is not ($\alpha < 1$) then take the step with probability α.

5) Go back to step 2 till enough samples have been taken.

· Use the function provided above with your probability function to achieve a sample of x values. Visualise this using the code snippets below.





· Use these trajectories to give expectation values $\langle x \rangle$ and $\langle V \rangle$. You could also calculate $\langle x^2 \rangle$ and $\langle V^2 \rangle$ and use these to calculate the uncertainties on $\langle x \rangle$ and $\langle V \rangle$. Give these for a range of β values and discuss the results. For our simple quadratic potential, it is possible to calculate both $\langle x \rangle$ and $\langle V \rangle$ analytically, but you might be able to anticipate their values (hints: 1. the potential is symmetric, 2. the equipartition theorem applies).

For beta = 0.50:

$$< x >= 0.07633 \pm 0.00440 < x^2 >= 1.94444 < V >= 0.97222 \pm 0.00437 < V^2 >= 2.85269$$

For beta = 1.00:

$$< x >= 0.02578 \pm 0.00310 < x^2 >= 0.96091 < V >= 0.48046 \pm 0.00213 < V^2 >= 0.68592$$

For beta = 2.00:

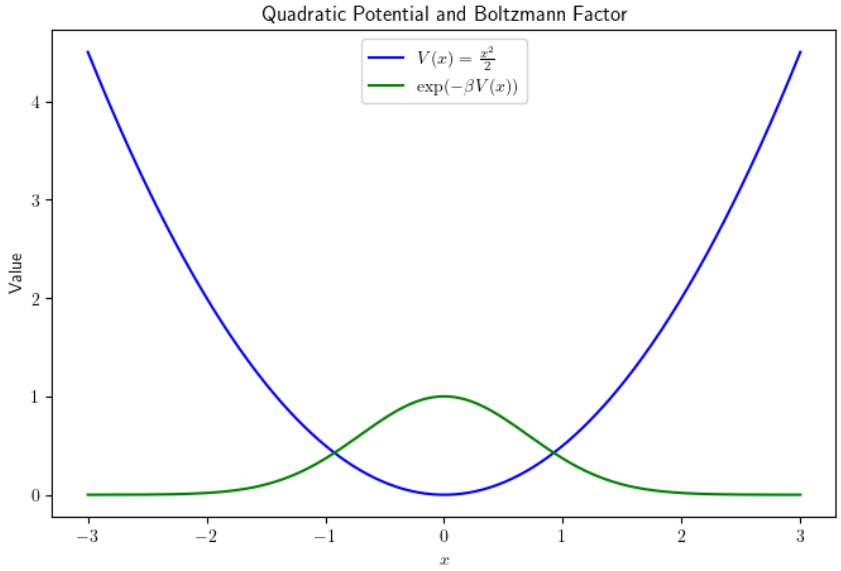
$$< x >= 0.00565 \pm 0.00223 < x^2 >= 0.49782 < V >= 0.24891 \pm 0.00112 < V^2 >= 0.18849$$

For beta = 5.00:

$$< x >= -0.00433 \pm 0.00142 < x^2 >= 0.20104 < V >= 0.10052 \pm 0.00046 < V^2 >= 0.03098$$

For beta = 10.00:

$$< x >= 0.00446 \pm 0.00100 < x^2 >= 0.10057 < V >= 0.05029 \pm 0.00022 < V^2 >= 0.00749$$



· Modify the potential to $V(x) = x^2/2 + \exp(-8x^2)$. Plot $V(x)$ and $p(x)$. Discuss what changes you expect in the results and test your intuition with MC integration.

For the modified potential, the added term $exp(-8x^2)$ creates a non-zero value at x=0. In fact, a short calculation shows that x=0 becomes a local maximum, with minima emerging symmetrically at approximately $x = \sqrt{(\ln 16)/(8)}$. As a result, although the overall distribution remains symmetric (so ¡x¿=0), the distribution p(x) becomes bimodal, with the majority of the weight away from x=0. Consequently, it is expected that $< x^2 >$ to be larger and ¡V¿ to be shifted relative to the simple quadratic case. Running the MC integration above for a range of beta values confirms this behaviour. Also, for the modified potential,

the expectation values and the uncertainties differ from the quadratic case. For instance, ¡V¿ is higher for low beta (high temperature) since the extra bump contributes more, while at high beta (low temperature) the system concentrates near the minima of the modified potential.

For beta = 0.50:

$$< x >= 0.09669 \pm 0.00466 < x^2 >= 2.18335 < V >= 1.22678 \pm 0.00450 < V^2 >= 3.52695$$

For beta = 1.00:

$$< x >= -0.01375 \pm 0.00345 < x^2 >= 1.19166 < V >= 0.74397 \pm 0.00212 < V^2 >= 1.00466$$

For beta = 2.00:

$$< x >= 0.06578 \pm 0.00272 < x^2 >= 0.74158 < V >= 0.51606 \pm 0.00110 < V^2 >= 0.38677$$

For beta = 5.00:

$$< x >= 0.01581 \pm 0.00223 < x^2 >= 0.49686 < V >= 0.35399 \pm 0.00050 < V^2 >= 0.15018$$

For beta = 10.00:

$$< x >= -0.15695 \pm 0.00199 < x^2 >= 0.41941 < V >= 0.29127 \pm 0.00024 < V^2 >= 0.09063$$