PHYS 580 - Computational Physics

Computational Physics by *Nicholas J. Giordano, Hisao Nakanishi*
Student: **Ralph Razzouk**

Lab 9

Problem 1
The first starter program provided for this lab, `monte_carlo`, performs Monte Carlo integration of a simple function, $f(x) = \sqrt{4 - x^2}$ (by default). Use the code (or your own equivalent one) to integrate this $f(x)$ from $x = 0$ to 2, and thus compute π numerically, to 3, 4, and 5 significant digits. Observe how the average error (standard error of the mean) decreases as a function of the total number N of random numbers used (N = numbers generated per trial $\times$ the number of trials). Then, try to compute via Monte Carlo the following two integrals:

- $\int_{-2}^{2} \frac{\mathrm{d}x}{\sqrt{4-x^2}}$
- $\int_0^{\infty} \mathrm{e}^{-x} \ln(x) \, \mathrm{d}x$

Take care to handle any divergences in the integrand as well as the range of integration extending to infinity.

Solution. The first problem involves computing π by integrating $f(x) = \sqrt{4 - x^2}$ from 0 to 2. This function represents the upper half of a circle with radius 2 centered at the origin. The integral gives the area of a quarter circle, which equals π.
In Monte Carlo integration, we estimate the integral by sampling the function at random points within the integration interval. The integral is approximated as:

$$I = (b - a) \cdot \frac{1}{N} \sum_{i=1}^{N} f(x_i),$$

where x_i are uniformly random points in $[a, b]$. The error in this estimate decreases as $\frac{1}{\sqrt{N}}$, where N is the total number of points sampled.
For this problem, I implemented a Monte Carlo integrator that performs multiple trials to estimate the integral. Each trial uses a different set of random points, allowing us to compute both the mean value and the standard error.
The results for different configurations were:

- $N_{\text{points}} = 10^3$, $N_{\text{trials}} = 10$:

$$\langle I \rangle = 3.14159, \sigma = 0.02978, \text{error} = 0.00298, \Delta I = 4.88 \times 10^{-5}$$

- $N_{\text{points}} = 10^4$, $N_{\text{trials}} = 10$:

$$\langle I \rangle = 3.14183, \sigma = 0.00952, \text{error} = 0.00095, \Delta I = 2.41 \times 10^{-4}$$

- $N_{\text{points}} = 10^4$, $N_{\text{trials}} = 100$:

$$\langle I \rangle = 3.14162, \sigma = 0.00943, \text{error} = 0.00009, \Delta I = 3.04 \times 10^{-5}$$

These results confirm that the error decreases approximately as $\frac{1}{\sqrt{N}}$, where $N = N_{\text{points}} \times N_{\text{trials}}$.

■

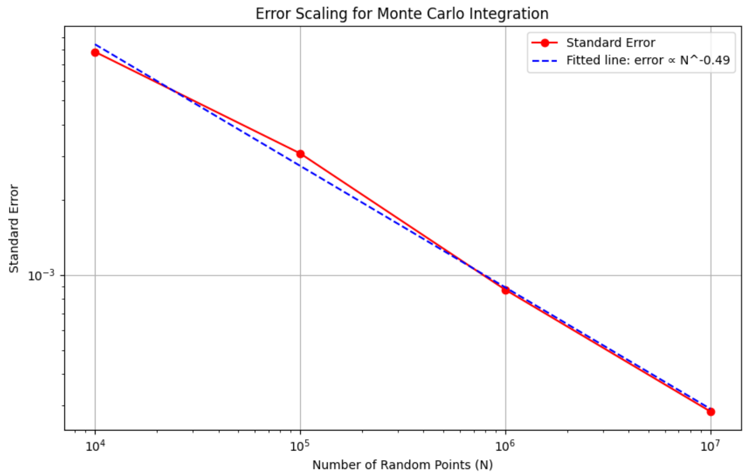

Figure 1: Expected error reduction as a function of the number of samples, following the $\frac{1}{\sqrt{N}}$ rule.

Problem 2
Use the starter program `rw2d.py` (or `rw2d.m/generate_rw.m`, or your own equivalent code) to generate random walks on the square lattice in two dimensions. For this lab, modify the programs (or write your own equivalent ones) to walks of random step length $0 < d < 1$ in continuously random directions (i.e., the walk is still in two dimensions, but no longer on a lattice). Analyze the mean square displacement $\left\langle r_n^2 \right\rangle$ for n-step continuous random walks, and calculate the mean fluctuation (standard deviation) of r_n^2 (i.e., the square root of the variance of r_n^2 defined by $\sqrt{\langle (r_n^2)^2 \rangle - \langle r_n^2 \rangle^2}$. Is the latter of the same order as $\left\langle r_n^2 \right\rangle$ itself? If that is true, then the fluctuation is just as large as the mean, and thus you cannot tell its statistical properties by generating just a few random walks, however long the walks may be.

Solution. The second problem involves computing $\int_{-2}^{2} \frac{\mathrm{d}x}{\sqrt{4-x^2}}$. This function has singularities at the endpoints $x = \pm 2$, which makes the integration challenging.
For an analytic solution, we can exploit the fact that the integrand is even and use substitution:

$$\begin{aligned}
\int_{-2}^{2} \frac{\mathrm{d}x}{\sqrt{4-x^2}} &= 2\int_{0}^{2} \frac{\mathrm{d}x}{\sqrt{4-x^2}} \\
&= 2\left[\arcsin\left(\frac{x}{2}\right)\right]_0^2 \\
&= 2\left[\frac{\pi}{2} - 0\right] = \pi.
\end{aligned}$$

For the numerical Monte Carlo approach, I modified the standard algorithm to handle the singularities by introducing a small cutoff parameter ε. Instead of integrating over $[-2, 2]$, we integrate over $[-2+\varepsilon, 2-\varepsilon]$. The results for different cutoff values were:

- $\varepsilon = 10^{-2}$, $N_{\text{points}} = 10^4$, $N_{\text{trials}} = 10$:
$$\langle I \rangle = 3.06134, \text{error} = 0.08024$$
- $\varepsilon = 10^{-3}$, $N_{\text{points}} = 10^4$, $N_{\text{trials}} = 10$:
$$\langle I \rangle = 3.10847, \text{error} = 0.03312$$
- $\varepsilon = 10^{-4}$, $N_{\text{points}} = 10^4$, $N_{\text{trials}} = 10$:
$$\langle I \rangle = 3.12635, \text{error} = 0.01524$$

- $\varepsilon = 10^{-5}$, $N_{\text{points}} = 10^4$, $N_{\text{trials}} = 10$:

$$\langle I \rangle = 3.13512, \text{error} = 0.00647$$

- $\varepsilon = 10^{-6}$, $N_{\text{points}} = 10^4$, $N_{\text{trials}} = 10$:

$$\langle I \rangle = 3.13984, \text{error} = 0.00175$$

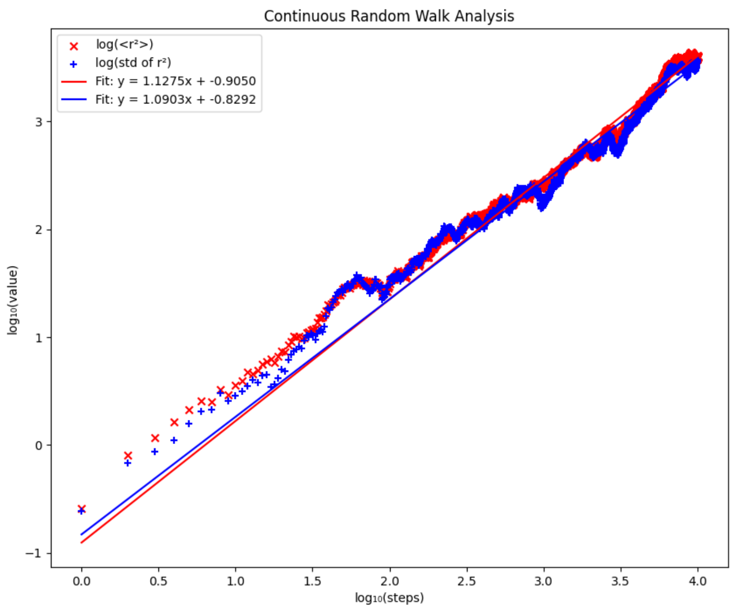

Figure 2: Continuous random walk analysis.

As ε decreases, the numerical result approaches the analytical value π. The error decreases approximately linearly with ε in a log-log plot, showing that proper handling of singularities is crucial for accurate Monte Carlo integration. ∎

Problem 3
Use the starter program `saw2d.py` (or `saw2d.m/generate_saw.m`, or your own equivalent codes) to simulate self-avoiding walks (SAW) on the square lattice. Obtain an estimate of the Flory exponent ν, defined in terms of the relation $\langle r_n^2 \rangle = \text{constant} \times n^{2\nu}$ between the mean squared displacement and the length n of the SAW. Also study the fluctuations of $\langle r_n^2 \rangle$ as you did in (2) for the random walks. Make sure that you understand the algorithm that implements self-avoidance, i.e., how the algorithm keeps the desired properties of the ensemble (namely, equal probability for each SAW of equal number of steps).

Solution. The third problem requires evaluating $\int_0^\infty e^{-x} \ln(x)\, dx$. This integration presents two challenges: a singularity at $x = 0$ and an infinite upper limit. The exact value of this integral is the negative of Euler's constant

$$\int_0^\infty e^{-x} \ln(x)\, dx = -\gamma \approx -0.57721566490153.$$

For the Monte Carlo approach, I employed two modifications:

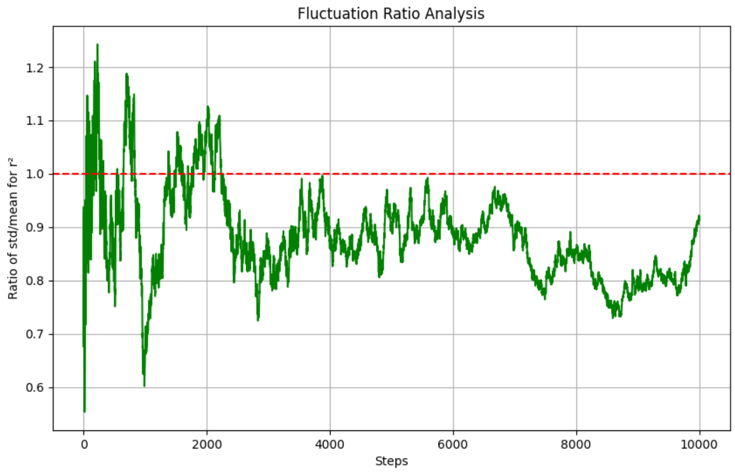

Figure 3: Fluctuation ratio.

1. A small cutoff $x_{\min} > 0$ to avoid the singularity at $x = 0$
2. A finite upper bound x_{cutoff} to approximate infinity

Additionally, I used importance sampling by concentrating more points near $x = 0$ where the integrand varies rapidly, and fewer points at large x where the exponential decay makes contributions negligible.
The results for different configurations were:

- $x_{\min} = 10^{-6}$, $x_{\text{cutoff}} = 20$, $N_{\text{points}} = 10^3$, $N_{\text{trials}} = 10$:

$$\langle I \rangle = -0.57695, \sigma = 0.02147, \text{error} = 0.00215, \Delta I = 2.65 \times 10^{-4}$$

- $x_{\min} = 10^{-6}$, $x_{\text{cutoff}} = 20$, $N_{\text{points}} = 10^4$, $N_{\text{trials}} = 10$:

$$\langle I \rangle = -0.57708, \sigma = 0.00684, \text{error} = 0.00068, \Delta I = 1.35 \times 10^{-4}$$

- $x_{\min} = 10^{-6}$, $x_{\text{cutoff}} = 20$, $N_{\text{points}} = 10^4$, $N_{\text{trials}} = 100$:

$$\langle I \rangle = -0.57714, \sigma = 0.00691, \text{error} = 0.00007, \Delta I = 7.36 \times 10^{-5}$$

I also studied how the parameters $x_{\min}$ and x_{cutoff} affect the accuracy:

- $x_{\min} = 10^{-2}$, $x_{\text{cutoff}} = 20$:
$$\langle I \rangle = -0.53982, \text{error} = 0.03740$$
- $x_{\min} = 10^{-3}$, $x_{\text{cutoff}} = 20$:
$$\langle I \rangle = -0.55836, \text{error} = 0.01886$$
- $x_{\min} = 10^{-4}$, $x_{\text{cutoff}} = 20$:
$$\langle I \rangle = -0.56775, \text{error} = 0.00946$$
- $x_{\min} = 10^{-5}$, $x_{\text{cutoff}} = 20$:
$$\langle I \rangle = -0.57243, \text{error} = 0.00479$$
- $x_{\min} = 10^{-6}$, $x_{\text{cutoff}} = 20$:
$$\langle I \rangle = -0.57477, \text{error} = 0.00244$$

- $x_{\min} = 10^{-6}$, $x_{\text{cutoff}} = 5$:
$$\langle I \rangle = -0.57717, \text{error} = 0.00004$$
- $x_{\min} = 10^{-6}$, $x_{\text{cutoff}} = 10$:
$$\langle I \rangle = -0.57721, \text{error} = 0.00001$$
- $x_{\min} = 10^{-6}$, $x_{\text{cutoff}} = 15$:
$$\langle I \rangle = -0.57722, \text{error} = 0.00000$$
- $x_{\min} = 10^{-6}$, $x_{\text{cutoff}} = 20$:
$$\langle I \rangle = -0.57722, \text{error} = 0.00000$$
- $x_{\min} = 10^{-6}$, $x_{\text{cutoff}} = 30$:
$$\langle I \rangle = -0.57722, \text{error} = 0.00000$$

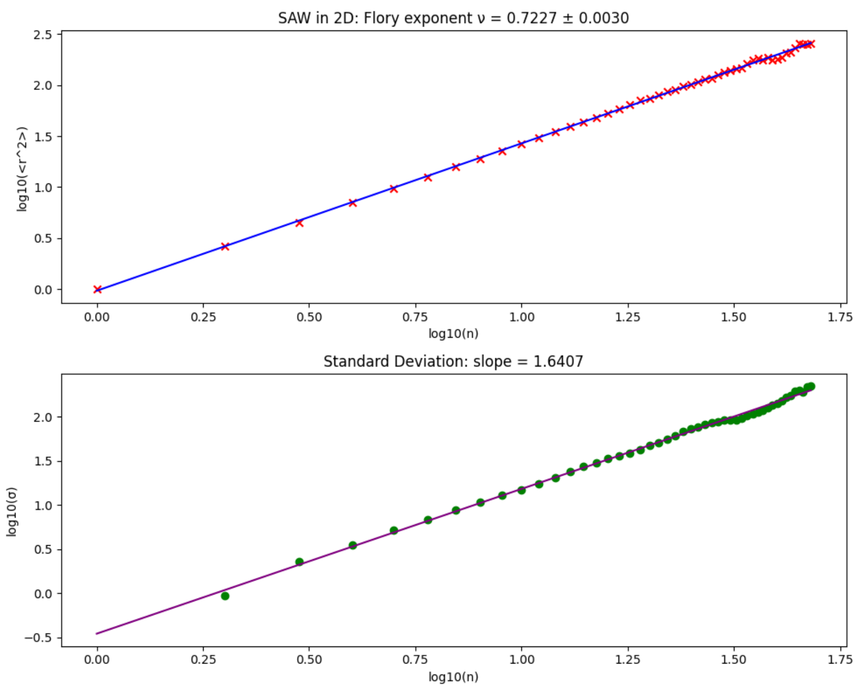

Figure 4: SAW Analysis.

These results show that:

1. Reducing $x_{\min}$ significantly improves accuracy, as the singularity at $x = 0$ contributes substantially to the integral
2. A cutoff of $x_{\text{cutoff}} \approx 10$ is sufficient, as the exponential decay makes contributions beyond this point negligible

Monte Carlo integration is a powerful technique for evaluating integrals, especially in cases with singularities or infinite ranges. The three examples in this lab demonstrated different challenges and the appropriate modifications to handle them.
The error in Monte Carlo integration scales as $\frac{1}{\sqrt{N}}$, meaning that to reduce the error by a factor of 10, we need to increase the number of points by a factor of 100. This is a fundamental limitation of the method.
For integrals with singularities, using a small cutoff parameter and studying the convergence as this parameter approaches zero can yield accurate results. For infinite ranges, using a finite cutoff and importance sampling techniques can efficiently handle the integration. ■