Computational Exploration of Random Processes: Diffusion-Limited Aggregation, Simulated Annealing, and Importance Sampling

Umi Yamaguchi
December 2, 2024

Background

This lab investigates computational methods to model and analyze processes influenced by randomness and optimization. Using simulations, we explore phenomena such as Diffusion-Limited Aggregation (DLA), simulated annealing, and importance sampling. DLA serves as a model for the formation of complex structures by simulating particles undergoing random motion and adhering to surfaces or other particles. By simulating this process on a discrete grid, we can study how seemingly random paths accumulate into fractal-like structures.

Simulated annealing is an optimization method inspired by thermodynamic cooling processes, where a system transitions to its lowest-energy state. The algorithm uses stochastic sampling and a controlled cooling schedule to locate global minima in complex functions, avoiding local minima traps. Importance sampling complements these techniques by improving the efficiency of numerical integration, particularly for functions with singularities or sharp variations. By biasing sample points toward regions of higher contribution, importance sampling reduces computational overhead while maintaining accuracy. These methods demonstrate the power of computational approaches in addressing challenges across physical, mathematical, and engineering domains. Through simulation and analysis, this lab highlights the interplay between randomness, optimization, and numerical precision.

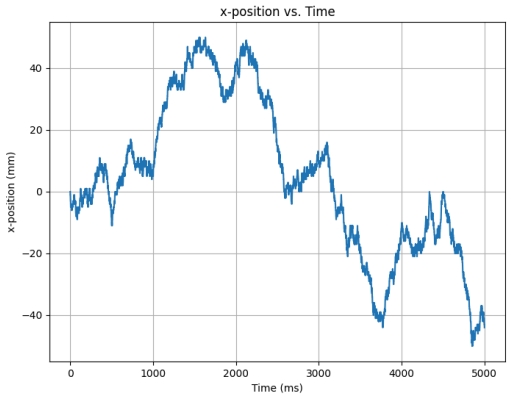
1. Simulating the DLA Process

In this experiment, we explore the phenomenon of DLA by simulating the Brownian Motion of particles in a two-dimensional grid. Brownian Motion describes the random movement of particles suspended in a fluid. In the context of DLA particles undergoing Brownian Motion aggregate to form complex, fractal-like structures upon contact with an anchored particle.

We discretized a two-dimensional space by creating a square grid of size $L \times L$, where $L = 101$ to ensure a central grid point at $(i, j) = (50, 50)$. The grid spacing is set to $\Delta x = \Delta y = 1\,\mathrm{mm}$, and

each time-step corresponds to $\Delta t = 1\,\mathrm{ms}$. The particle begins at the center of the grid, representing $x = y = 0$ at $t = 0$. At each time step, the particle moves one grid space in a randomly selected direction: up, down, left, or right. This movement is constrained by the grid boundaries; if a proposed move would take the particle outside the grid, another direction is randomly chosen. This process is repeated for $N = 5000$ time-steps, simulating a total duration of $5\,\mathrm{s}$.

The simulation was implemented using a computer program that records the particle's position at each time step. The random directions were generated using a uniform random number generator to ensure equal probability for all four possible movements.



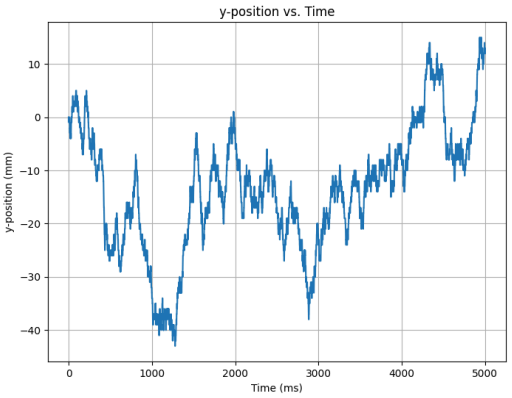


Figure 1 (Left): This plot shows the x-position of a single particle undergoing Brownian motion over 5000 time steps. The random fluctuations highlight the unpredictable nature of the particle's movement in a discretized $L \times L$ grid.

Figure 2 (Right): This plot depicts the y-position of the particle as it performs a random walk. Like the x-position, it showcases the erratic trajectory characteristic of Brownian motion.

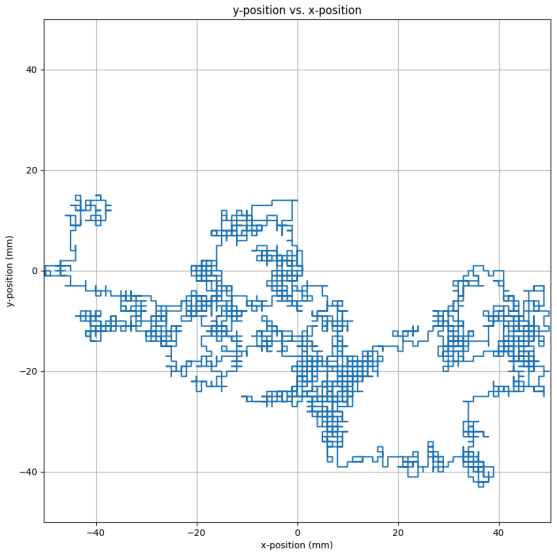


Figure 3: The two-dimensional trajectory of the particle shows how it explores the grid randomly. The center grid site ($x = y = 0$) marks the starting point, and the path traces the particle's movements across the $L \times L$ space.

The particle's x- and y-positions were plotted against time to illustrate its movement along each axis. Figure 1 shows the x-position as a function of time, while Figure 2 depicts the y-position over the same period. Both plots exhibit the characteristic random fluctuations of Brownian Motion.
Figure 3 presents the trajectory of the particle in the xy-plane over the entire simulation. The path demonstrates the random walk nature of the movement, covering various regions of the grid without a predictable pattern. The particle's displacement from the origin varies over time, sometimes returning close to the center and other times venturing towards the grid's edges.

For the next experiment, we simulate the DLA process in a two-dimensional grid. Starting with a single anchored particle at the boundary. Each particle performs a random walk until it attaches to the existing cluster or the grid boundary. The simulation continues until an anchored particle occupies the center grid site. The objective is to visualize the aggregation pattern formed by the particles and understand the dynamics of the DLA process. Each particle undergoes a random walk similar to the Brownian Motion described in the previous experiment. At each time step, we also introduced mechanisms to handle particle aggregation such as anchoring conditions, storing anchored positions, and checking the neighbouring sites. This is required for the simulation to continue to introduce and move particles until an anchored particle occupies the center grid site. Upon termination, the positions of all anchored particles are recorded for visualization.

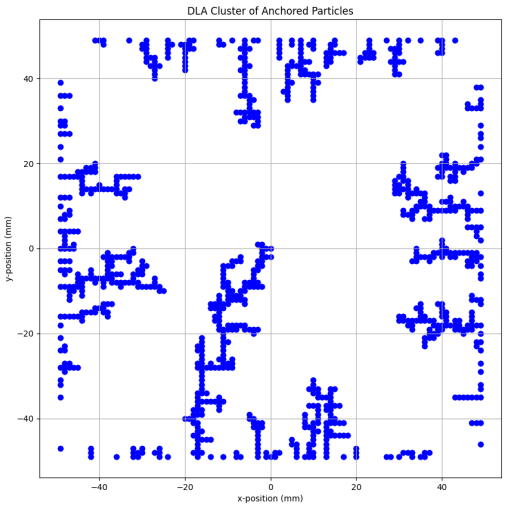


Figure 4: This figure displays the anchored particles' positions during the DLA process. The clustering structure illustrates how particles adhere to the sides or existing anchored particles, forming fractal-like growth patterns.

The simulation resulted in a cluster of anchored particles forming a fractal-like pattern emanating from the boundaries towards the center. As particles performed random walks and anchored upon contact with existing anchored particles or the boundaries, a dendritic structure emerged. DLA simulation successfully demonstrated the formation of complex patterns through simple stochastic rules. The resulting structure is a visual representation of how randomness at the microscopic level can lead to intricate patterns at the macroscopic level. This pattern formation influenced several factors such as random walk mechanics, anchoring rules, and storage of anchored positions. Each run of the simulation can produce a different pattern due to the stochastic nature of particle movement and anchoring. This highlights the sensitivity of DLA to initial conditions and random fluctuations.

2. Efficiency of Simulated Annealing

Simulated annealing is a probabilistic technique used for finding the global minimum of a function that may have numerous local minima. Inspired by the annealing process, where controlled cooling of a material allows it to reach a state of minimum energy, simulated annealing employs a temperature parameter to probabilistically accept or reject moves to new states. By gradually lowering the temperature, the algorithm balances exploration of the solution space with convergence to a minimum. In this experiment, we apply simulated annealing to optimize two functions:

- $f(x, y) = x^2 - \cos(4\pi x) + (y-1)^2$ (Eq.1)
- $f(x, y) = \cos x + \cos(\sqrt{2}x) + \cos(\sqrt{3}x) + (y-1)^2$ (Eq. 2)

Our goal is to find the global minimum of these functions using simulated annealing and to analyze the efficiency of the method by adjusting the cooling schedule parameters.

For the simulated annealing algorithm, we start with an initial point $(x_0\, y_0)$ and an initial temperature T_0. For each time step, propose a move to a new point $(x'\, y') = (x + \delta x\, y + \delta y)$, where δx and δy are random numbers drawn from a Gaussian distribution with mean 0 and standard deviation 1. Then, calculate the change in function value $\Delta f = f(x', y') - f(x, y)$. Accept the new point with probability:

$$P = \begin{cases} 1, & \Delta f \leq 0 \\ \exp\left(-\dfrac{\Delta f}{k_B T}\right), & \Delta f > 0 \end{cases}$$

where T is the current temperature, and k_B is the Boltzmann constant (set to 1 for simplicity).

In the end, we update the temperature according to the exponential cooling schedule and repeat the proposal and acceptance steps until the temperature reaches a final value T_f or a maximum number of iterations is reached.

$$T = T_0 \exp\left(-\frac{t}{\tau}\right)$$

Where t is the current time-step, and τ is the time constant controlling the cooling rate.

We begin with Eq.1 and initialize the point as $(x_0, y_0) = (2, 2)$. We ran the simulated annealing algorithm multiple times, adjusting T_0, T_f, and τ to find a set of parameters that reliably converge to the global minimum at $(0, 1)$ in a reasonable time.

After experimentation, we found that the $(T_0, T_f, \tau) = (5.0, 1.0 \times 10^{-5}, 3.0 \times 10^4)$ yielded consistent convergence to the global minimum. The algorithm was run for a total of $N \approx 400,000$ time steps. The plots of x and y versus time step are shown in Figure 5 and Figure 6, respectively.

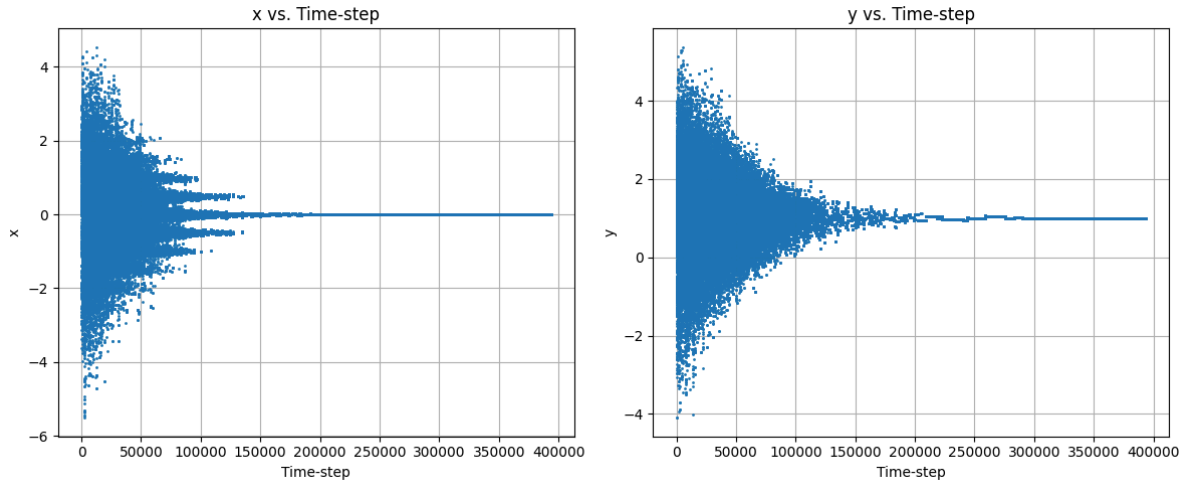


Figure 5 (Left): This plot shows the progression of x values during simulated annealing to minimize $f(x, y)$. The cooling schedule drives the system towards the global minimum at $(x, y) = (0, 1)$.
Figure 6 (Right): This plot shows the progression of y values during simulated annealing to minimize $f(x, y)$. The cooling schedule drives the system towards the global minimum at $(x, y) = (0, 1)$.

At the end of the simulation, the final values obtained were $(x_{\text{final}}, y_{\text{final}}) = (-0.0007, 1.0047) = -0.9999$. These values are very close to the known global

minimum at $(0, 1)$. The plots show that x and y values fluctuate significantly at higher temperatures, allowing the algorithm to explore the solution space widely. As the temperature decreases, the fluctuations reduce, and the values settle near the global minimum. The choice of T_0 needs to be high enough to allow escaping local minima, while T_f must be low enough for convergence. The time constant τ determines the rate of cooling. By tuning these parameters, we achieved reliable convergence to the global minimum within a reasonable computational time.

We adapted the program from the previous algorithm to optimize Eq.2. The same simulated annealing algorithm was used, with modifications to enforce the constraints $(0 < x < 50)$ and $(-20 < y < 20)$. We initialize the point as $(x_0, y_0) = (10, 1)$. After experimentation, the parameters the provided convergence to a minimum near $x \approx 16$ were when $(T_0, T_f, \tau) = (1000.0, 1.0 \times 10^{-10}, 1.0 \times 10^4)$. The algorithm was run for a total of $N \approx 280,000$ time steps. The plots of x and y versus time step are shown in Figure 7 and Figure 8, respectively.

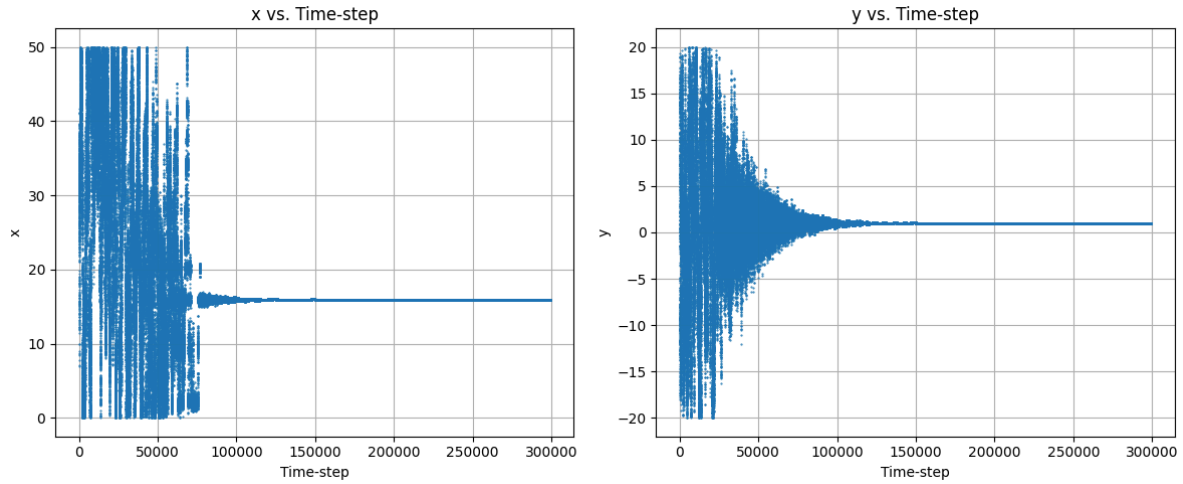


Figure 6 (Left): This plot shows the progression of x values during simulated annealing to minimize $f(x, y)$. The cooling schedule drives the system towards the global minimum at $(x, y) = (16, 1)$.
Figure 7 (Right): This plot shows the progression of y values during simulated annealing to minimize $f(x, y)$. The cooling schedule drives the system towards the global minimum at $(x, y) = (16, 1)$.

At the end of the simulation, the final values obtained were $(x_{\text{final}}, y_{\text{final}}) = (15.9580, 0.9946) = -2.6126$. These values are very close to the known global minimum at $(16, 1)$. The complex function has multiple local minima due to the combination of

cosine terms. The simulated annealing algorithm, with appropriately chosen parameters, was able to navigate the solution space and converge to a global minimum near $x \approx 16$. The plots show that x values initially explore a wide range, including other local minima near $x \approx 2$ and $x \approx 42$. As the temperature decreases, the algorithm settles into the basin of the global minimum.

The convergence rate and the algorithm's ability to avoid local minima in simulated annealing were significantly influenced by the cooling schedule parameters T_0, T_f, and τ. Through trial and error, we discovered that a higher initial temperature T_0 allowed the algorithm to accept more uphill moves early on, facilitating exploration of the solution space and preventing entrapment in local minima. Adjusting τ to a larger value slowed the cooling process, granting more time at higher temperatures for exploration, though it increased computational time. A lower final temperature T_f ensured that the algorithm became more selective over time, reducing the acceptance of uphill moves and allowing convergence to the global minimum.

Our trial-and-error process involved systematically varying one parameter at a time while observing the impact on convergence and computational efficiency. For Eq.2 with multiple local minima, setting $T_0 = 1000.0$ and $\tau = 1.0 \times 10^4$ provided sufficient thermal energy and time for the algorithm to overcome energy barriers between minima, but did not guarantee convergence to the global minimum at $x \approx 16$. This difficulty highlights the limitations of simulated annealing when dealing with complex functions with multiple deep local minima. This emphasizes the importance of multiple runs or incorporating additional strategies, such as restarting the algorithm from different initial points, to improve the chances of finding the global minimum.

3. Importance Sampling

In numerical integration, the Monte Carlo method provides a powerful approach to estimating integrals, particularly in higher dimensions or when the integrand is challenging to evaluate analytically. However, standard Monte Carlo integration becomes inefficient for integrands with singularities or sharp variations. Importance sampling addresses this inefficiency by sampling points more frequently in regions where the integrand has higher values, reducing variance and improving convergence. In this experiment, we evaluate two integrals using the mean value method and importance sampling to compare their accuracy and efficiency.

The mean value method was applied to evaluate the integral $\int_0^1 \frac{x^{-1/2}}{1+e^x} dx$. Using $10,000$ uniformly sampled points across $[0,1]$, the result from a single simulation was $$0.838256$$. This value was obtained by averaging the integrand's values over the sampled points and multiplying by the interval width.

The accuracy of this method is influenced by the inefficiency of uniform sampling near the singularity at $x = 0$. Although the method is unbiased, it requires a large number of samples to achieve precise results due to the disproportionate contribution of values near the singularity. Repeating the process $1,000$ times would yield a distribution of estimates, but the broad spread observed in the histogram (Figure 8) highlights the high variance inherent in this method.

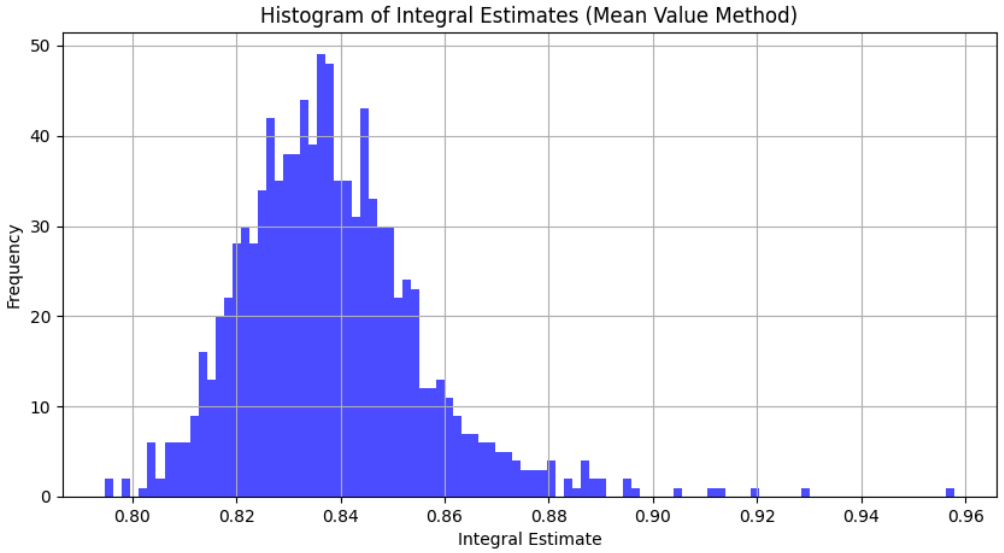


Figure 8: Histogram of 1000 results for the integral using the mean value method. The wide distribution reflects greater variability in results, underscoring the challenges posed by the integrand's divergence. The approximate variance is 0.04

The importance sampling approach, designed to address the singularity of the integrand, yielded a result of $$0.838919$$. This method used the weighting function $w(x) = x^{-1/2}$, which allowed efficient sampling near the singularity where the integrand is most significant. The transformation $x = z^2$, with z uniformly distributed over $[0,1]$, ensured that the sampled points followed the desired probability distribution $p(x) \propto x^{-1/2}$.

This approach significantly reduced the variance of the estimates, as evidenced by the sharp peak in the histogram of $1,000$ repetitions (Figure 9). The closeness of the result ($$0.838919$$) to the mean value method's result ($$0.838256$$) demonstrates the correctness of both methods, but the reduced variability and efficiency of importance sampling make it the superior choice for this integral.

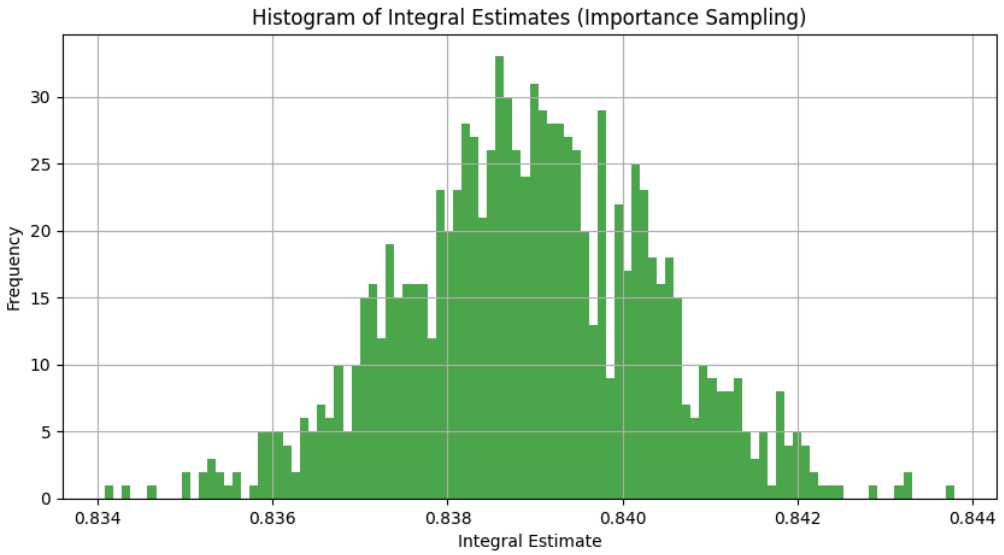


Figure 9: Histogram of 1000 results for the integral using importance sampling with $w(x) = x^{-1/2}$. The tighter distribution demonstrates improved accuracy and efficiency in handling the singularity. The approximate variance is 0.004

The slight difference between the results of the two methods is due to the higher precision of importance sampling in capturing the contributions of the integrand near its singularity. While accurate in principle, the mean value method requires substantially more samples to achieve comparable precision, making it less practical for challenging integrals like this one. Importance sampling's ability to focus computational resources on the critical regions of the integrand highlights its utility in numerical integration.

The histograms of the results from the mean value method and importance sampling highlight the latter's advantages. The mean value method (Figure 8) produced a broad distribution of estimates, variance is approximately 0.04, reflecting the inefficiency of uniform sampling for an integrand with a singularity. In contrast, the histogram for importance sampling (Figure 9) showed a sharp peak around the mean value, demonstrating its ability to reduce variability and achieve accurate results with fewer samples. These findings underscore the importance of selecting an appropriate sampling strategy for integrals with singularities or other challenging features.

We extended the importance sampling approach to evaluate the integral $\int_0^{10} \exp(-2|x-5|)dx$, where the integrand is sharply peaked near $x = 5$. Sampling uniformly over $[0, 10]$ would waste computational resources on regions contributing little to the integral. Instead, we used the weighting function $w(x) = \frac{1}{\sqrt{2\pi}} e^{-(x-5)^2/2}$, which closely matches the integrand's shape. Sampling from $w(x)$ was achieved using the normal distribution centred at $x = 5$ with a standard deviation of 1.

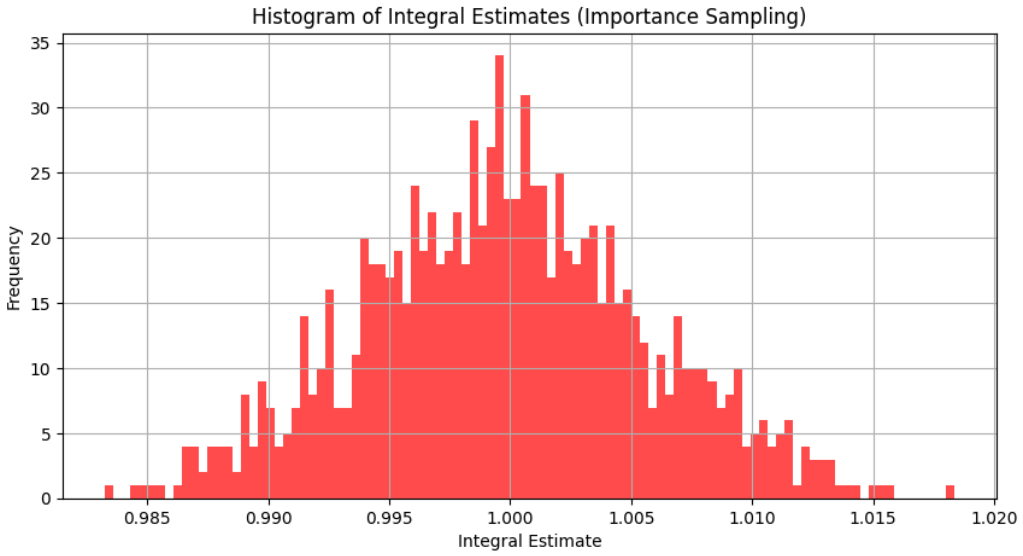


Figure 10: This histogram shows the results of importance sampling for a sharply peaked integrand. The weighting function $w(x) = \frac{1}{\sqrt{2\pi}} e^{-(x-5)^2/2}$ ensures efficient sampling near $x = 5$, significantly reducing variability in the results. The approximate variance is 0.015

The integral was estimated using $10,000$ samples, and the calculation was repeated $1,000$ times. The histogram of the results confirmed the efficiency of importance sampling, showing a narrow distribution around the mean value. This method allowed precise integral evaluation with minimal computational effort compared to uniform sampling.

The experiments demonstrate the power of importance sampling in handling integrals with singularities or sharply peaked regions. By adapting the sampling strategy to the structure of the integrand, we significantly reduced variance and improved accuracy compared to the mean value method. These results underscore the value of carefully chosen weighting functions and probability distributions in Monte Carlo integration.