Monte Carlo Dissertation

Candidate Number: 265113

May 2025

Q1

Introduction

The goal of this part is to generate random numbers following the given probability density function (PDF). Then compare the generated distribution to the theoretical distribution and estimate $E(|X|)$ using importance sampling.

Description of PDF

We are given a probability distribution function defined as:

$$f(x) = \begin{cases} \frac{\alpha}{k}\left(\frac{k-\mu+x}{k}\right)^{-\alpha-1}, & x \geq \mu \\ 0, & \text{otherwise} \end{cases}. \tag{1}$$

The distribution is characterised by 3 variables:

- α (shape parameter): The shape parameter controls the heaviness of the tail with lower values giving a heavier tail.
- k (scale parameter): The scale parameter stretches the distribution.
- μ (location parameter): The location parameter shifts the distribution along the x-axis and gives the distribution its lower bound.

The distribution is defined for $x \in [\mu, \infty]$. It is a monotonically decreasing function decreasing smoothly for $x \to \infty$ and it behaves like a power law $f(x) \propto x^{-\alpha-1}$. The distribution is therefore right skewed, meaning it has a long tail to the right.

To ensure that the expectation value $E(|X|)$ and Variance $\mathrm{Var}(X)$ of the random variable X exist and is finite, we need to impose a constraint on α. According to the lecture notes (Section 2.6), the expectation value exists if the integral $\int xf(x)dx$ converges. For the probability distribution function given the expectation value is defined and finite when $\alpha > 1$. This is shown from,

$$E[X] = \int_{\mu}^{\infty} xf(x)dx = \mu + \frac{k}{\alpha - 1} \quad \text{for } \alpha > 1. \tag{2}$$

From the lecture note the variance of X is defined as,

$$\mathrm{Var}(X) = E[(X - E(X))^2] = E(X^2) - (E(X))^2, \tag{3}$$

the convergence exist if the integral $\int x^2 f(x)dx$ converges and thus:

$$\mathrm{Var}(X) = \frac{k^2}{(\alpha-1)^2(\alpha-2)} \quad \text{for } \alpha > 2. \tag{4}$$

Therefore, we selected our variables $a = 3$, $k = 2$, and $\mu = 1$, which ensured that the expectation value and variance were finite.

I chose the inverse transform method to generate random numbers distributed according to the given distribution because the cumulative distribution function (CDF) of the target distribution was mathematically invertible allowing me to directly sample from the target distribution. This removed the need to use a proposal distribution or apply weights. In addition to this, the inverse transform method is efficient because the generated samples follow the exact shape of the target distribution.

The inverse transform method is a fundamental method for generating random samples for a probability distribution function using uniform random numbers. As explained in the lecture notes (Section 3.3), the method consists of generating a random variable $U \sim \text{Uniform}(0, 1)$. This variable is then passed into the inverse of the CDF to produce a sample from the desired distribution.

For a continuous random variable X with PDF $f(x)$, the CDF $F(x)$ is given, as seen in the lecture notes (Section 2.3) by:

$$F(x) = P(X \leq x) = \int_{-\infty}^{x} f(x')\,dx'. \tag{5}$$

and thus the inverse CDF is given by:

$$F(x) = P(X \leq x) = \int_{-\infty}^{x} f(x')\,dx'. \tag{6}$$

If the PDF has a lower bound a, the formula simplifies to:

$$F(x) = \int_{a}^{x} f(x')\,dx', \quad \text{for } x \geq a. \tag{7}$$

With the definition of the CDF given in Equation 7 we can define the CDF:

$$F(x) = \int_{1}^{x} \frac{\alpha}{k} \left(\frac{k - \mu + x'}{k} \right)^{\alpha - 1} dx' = 1 - \left(\frac{k - \mu + x}{k} \right)^{-\alpha}, \quad \text{for } x \geq \mu. \tag{8}$$

Therefore, the derived CDF and the inverse CDF is given by:

$$F(x) = \begin{cases} 0, & x < \mu \\ 1 - \left(\frac{k - \mu + x}{k} \right)^{-\alpha}, & x \geq \mu \end{cases}, \tag{9}$$

$$F^{-1}(u) = k(1 - u)^{-1/\alpha} - k + \mu. \tag{10}$$

Histogram and Theoretical Curve Comparison

The inverse CDF was defined in R and used to generate random samples of sizes 100, 500, 1000, 5000, 7500 and $10,000$ for our analysis. A histogram for each sample was plotted alongside the theoretical density function. The histogram and the theoretical density function showed a strong agreement as the sample size increased with both the shape and the tail behaviour matching closely, demonstrating the convergence property of the empirical distribution to the theoretical distribution.

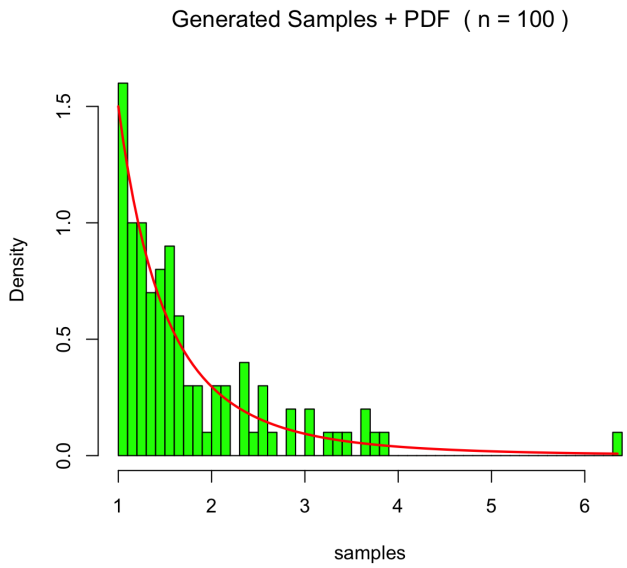

Figure 1: Histogram of 100 samples generated using inverse transform sampling overlaid with the theoretical probability density function (PDF) in red. The accuracy of the sampling method is demonstrated by the close alignment of the histogram and the PDF.

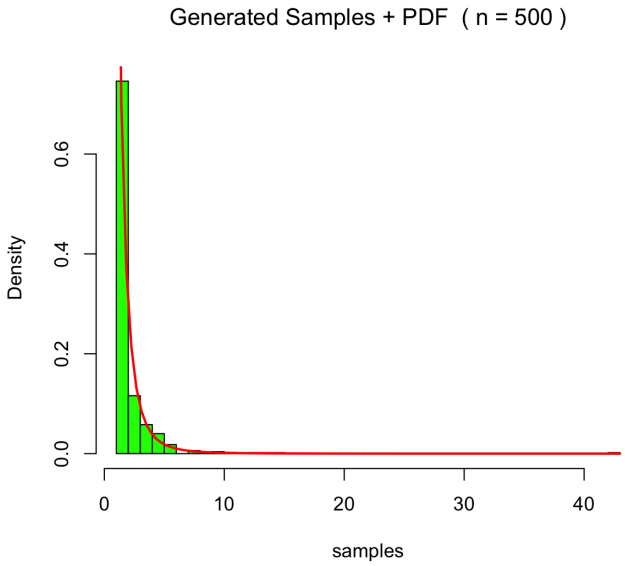

Figure 2: Histogram of 500 samples generated using inverse transform sampling overlaid with the theoretical probability density function (PDF) in red. The accuracy of the sampling method is demonstrated by the close alignment of the histogram and the PDF.

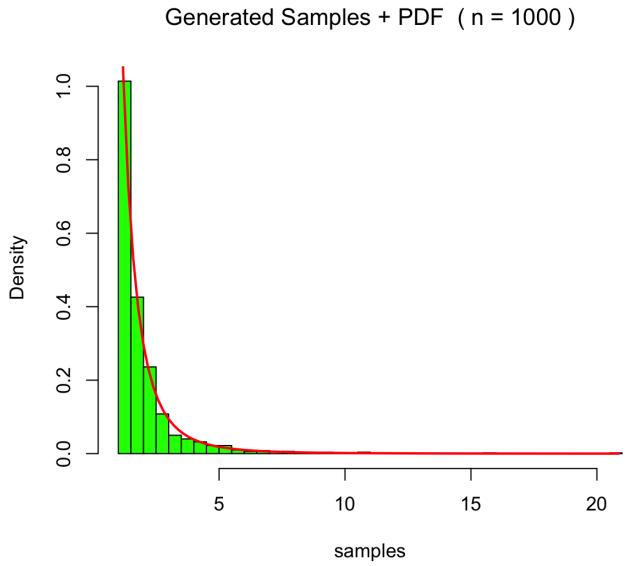

Figure 3: Histogram of 1,000 samples generated using inverse transform sampling overlaid with the theoretical probability density function (PDF) in red. The accuracy of the sampling method is demonstrated by the close alignment of the histogram and the PDF.

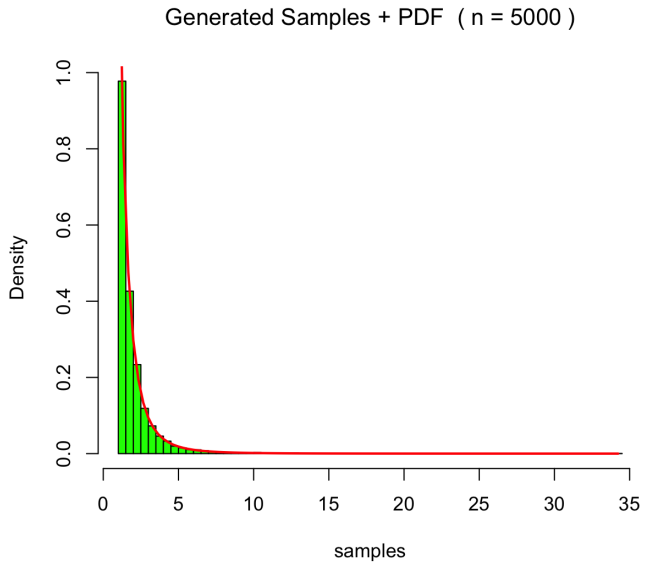

Figure 4: Histogram of 5,000 samples generated using inverse transform sampling overlaid with the theoretical probability density function (PDF) in red. The accuracy of the sampling method is demonstrated by the close alignment of the histogram and the PDF.

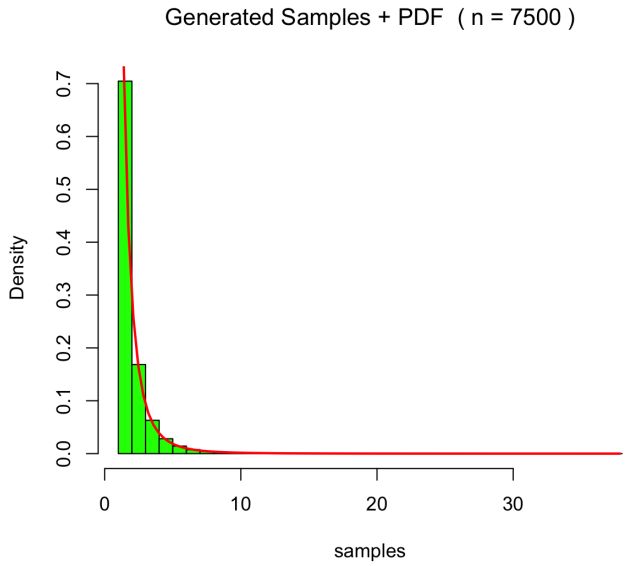

Figure 5: Histogram of 7,500 samples generated using inverse transform sampling overlaid with the theoretical probability density function (PDF) in red. The accuracy of the sampling method is demonstrated by the close alignment of the histogram and the PDF.

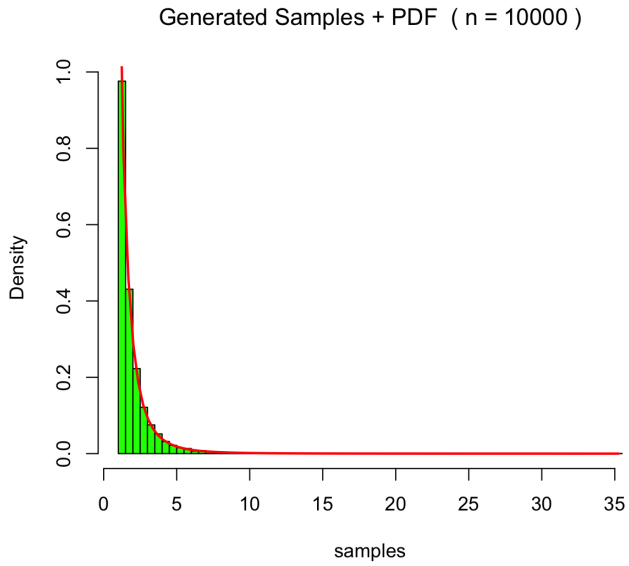

Figure 6: Histogram of 10,000 samples generated using inverse transform sampling overlaid with the theoretical probability density function (PDF) in red. The accuracy of the sampling method is demonstrated by the close alignment of the histogram and the PDF.

Goodness of Fit

To assess the agreement of the histogram and theoretical density function quantitatively we used the Kolmogorov-Smirnov (KS) test, comparing the empirical cumulative distribution function (ECDF) of the sample and the theoretical CDF, for each of our sample sizes. The KS test returns two results: D-statistic, which is a measure of the maximum vertical distance from the two distributions and the p-value.

As expected, the D-statistic decreases with increasing sample size indicating improved agreements with the with the theoretical distribution. The calculated p-value's for low sample sizes, such as sample sizes: 100 and 500, were low suggesting that the ECDF did not agree with the theoretical CDF. However, as the sample increased the p-value approached 1 suggesting that the ECDF compared well to the theoretical CDF. This is also consistent with expectation, as seen in the lecture notes (Section 8), stating that increasing the sample size improves the alignment of empirical results with theoretical expectations.

Importance Sampling

Using importance sampling, I estimated the expected value using the definition form the lecture notes,

$$E(|X|) \approx \frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i)}{g(x_i)} |x_i|, \tag{11}$$

found in Section 2.6. The expectation value was obtained for each sample size and the weights were computed as:

$$w(x_i) = \frac{f(x_i)}{g(x_i)}. \tag{12}$$

Here, for the proposed suitable function $g(x_i)$, where $g(x_i) > 0$ for almost all x when $f(x_i) > 0$, I picked the shifted exponential function given by:

$$g(x) = \begin{cases} \lambda \cdot e^{-\lambda(x-\mu)}, & x \geq \mu \\ 0, & \text{otherwise} \end{cases}, \tag{13}$$

where I defined $\mu = 1$ and $\lambda = 0.5$. The expectation value converged for increasing sample sizes to $E(|X|) \approx 3.9$. Which visually looks consistent with our right skewed distribution. The estimate stabilised as I increased the sample size, shown in Figure 7, indicating that the chosen proposal distribution $g(x_i)$ was accurate.

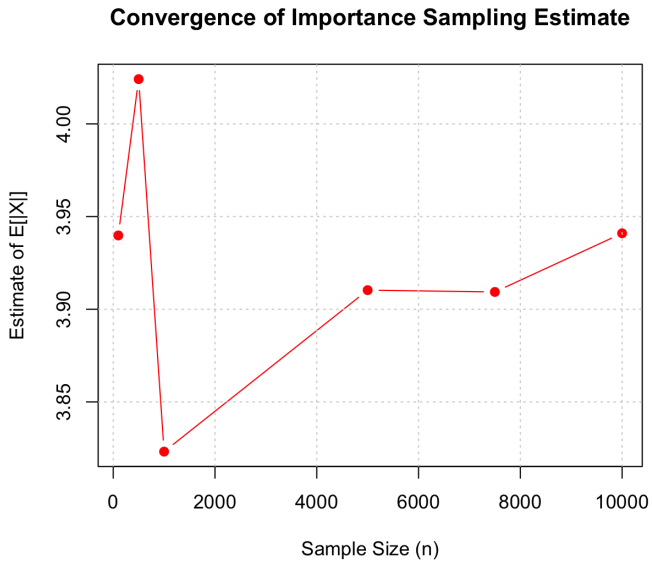

Figure 7: Convergence of Importance sampling showing for increasing sample sizes the expectation value converges to $E(|X|) \approx 3.9$.

Q2

Why Markov Chains Monte Carlo Algorithms Work

Markov Chain Monte Carlo (MCMC) algorithms work by constructing a Markov chain that, if run long enough, the proportion of time spent in each state matches the target distribution. A Markov chain is a discrete time stochastic process in which the probability to transition to next state of the system X_{t+1} depends only on the current state X_t not the previous states. This property of Markov chains, as seen in Section 5.1 of the lecture notes, is known as the Markov property. The evolution of a Markov chain is defined by a transition matrix. A transition matrix is a square matrix that encodes the probability of moving between states in a Markov chain. Each element of the transition matrix, denoted as $p_{i,j}$, represent the probability of moving from state i to state j, where each row sums to one so that it represent a valid probability distribution. The key theorem of Markov chains are that an irreducible and aperiodic chain converges to a single unique stationary distribution, regardless of the starting state, given a long enough run time. This allows us to estimate quantities using the empirical average of the samples. This shows why MCMC methods are effective for sampling from complex distribution that would otherwise be too complex work with.

Irreducibility and Aperiodicity

A Markov chain is irreducible if every state communicates which means that every state is accessible from any other state. Formally, stated in the lecture notes (Definition 5.1.3), for any pair $i, j \in S$, where S is a finite or countably infinite state space, there exist an integer $k \geq 1$, the number of steps, such that $p_{i,j}^k > 0$. To test if the Markov chain was irreducible, I defined $reach\_matrix$, to represent the cumulative probabilities of reaching any state from state 1 in one step. For an irreducible Markov chain it is expect that this matrix is non zero everywhere. Therefore, by repeated calculation of the probabilities of reaching any other state from the current state and cumulatively adding them to the $reach\_matrix$, after 100 steps we checked if every element of $reach\_matrix$ was non-zero concluding if the Markov chain was irreducible.

A irreducible Markov chain is aperiodic if the period of every state is 1, meaning the chain doesn't get trapped in a fixed cycle of transitions. To test that a Markov chain is aperiodic, I simulated a Markov chain starting from each state i for 100 steps. In an array $return\_steps$, I recorded the step k for each step there was a non-zero probability of returning to the initial state. Then I calculated the period of state i, as seen in the lecture notes (Section 5.1.4), by computing the greatest common devisor of the $return\_steps$ array.

With the transition matrix given, it was found that the Markov chain was irreducible and aperiodic as every element in the defined $reach\_matrix$ was non zero and the period was 1 for every state.

Calculation of the Invariant Distribution (Stationary Distribution)

An invariant distribution or stationary distribution given by π is a probability distribution that doesn't change as the Markov chain evolves. The stationary distribution is important for understanding the long-term behaviour of a Markov chain as for increase number of steps the Markov chain converges to the invariant distribution. The stationary distribution can be defined with the equation in matrix notation found in the lecture notes (Section 5.4.3):

$$\pi P = \pi \quad \Rightarrow \quad P^T \pi^T = \pi^T \tag{14}$$

.

This equation defines the invariant distribution as the left-eigenvector of the transition matrix P with eigenvalue 1. In R , I implemented the equation $\pi P = \pi$ and obtained the invariant distribution for the probability matrix provided. The steps for this proccess are detailed as:

- Transpose the transition matrix P
- Compute the eigenvectors π
- Get the eigenvector with an eigenvalue of 1

- Normalise the eigenvector

The obtained invariant distribution can be found in Table 1.

State	1	2	3	4	5
Invariant Distribution	0.1908	0.2014	0.2397	0.1117	0.2564

Table 1: Invariant distribution of the Markov chain across states 1 to 5

Empirical Distribution Calculation

To simulate a random walk of a Markov chain, I defined a starting state and repeatedly sampled, for n steps, the next state based on the transition probability given by the transition matrix. At each step, I recorded the state that was visited, allowing me to count how often each state was visited during the random walk. By normalising the number of times each state was visited I calculated the empirical distribution of the chain. As stated in the ergodic theorem given in the lecture notes (Section 5.4.4), if the Markov chain is irreducible, the empirical distribution should converge to the invariant distribution as n is increased. I simulated the Markov chain for $n = 1000$ allowing me to compare the empirical distribution with the theoretical invariant distribution found earlier.

Comparison of Theoretical and Empirical Distribution

The bar chart compares the theoretical distribution with the obtained empirical distribution from simulation of the Markov chain for 1000 steps. The red bars represent the probabilities of each state for the invariant distribution calculated using the eigenvector method, whereas the blue bar represent the empirical distribution recovered from simulating a Markov chain for $n = 1000$ steps. Overall, the empirical distribution matches the theoretical invariant distribution closely, with only minor deviations present. The small deviations from the theoretical distribution is expected due to the randomness present with a finite number of samples. As n is increase the empirical distribution is expected to converge even more closely with the invariant distribution, as predicted by the ergodic theorem, shown in the lecture notes (Section 5.4.4). This bar chart demonstrates that the correctness of my the simulation and that the Markov chain is approaching its long-run behaviour.

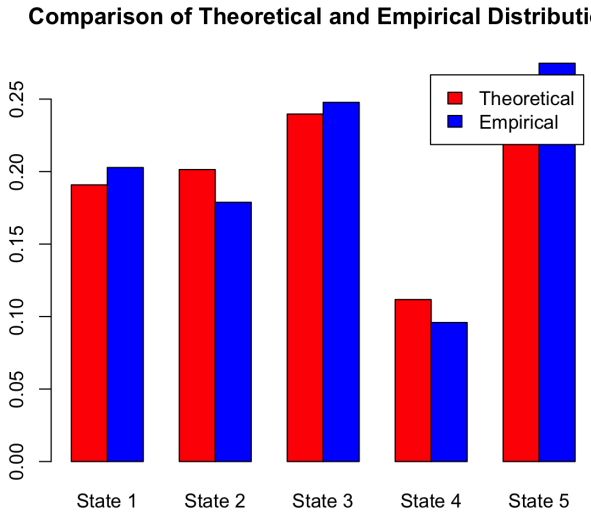

Figure 8: Comparison of the empirical distribution (blue) obtained from simulation the Markov chain with the theoretical invariant distribution (red) obtained from the eigenvector method.

Q3 – Monte Carlo Method for Solving Leplace and Poisson Equations

Introduction

For this part of the project, I selected the **1D parallel-plate capacitor simulation** from the research paper *"Monte Carlo Methods for Solving the Poisson and Laplace Equations"* by Ankur Sonawane. The paper investigates how random walk-based Monte Carlo methods can be applied to solve classical PDEs such as Laplace's and Poisson's equations in 1D and 2D electrostatic systems. Among the several applications presented, I focused on the 1D parallel plate capacitor governed by the Laplace equation because of it's simplicity, clarity, and direct relevance to the methods studied in this module.

Monte Carlo Method Solution to the Poisson and Leplace Equations

This is done by simulating a random walk, w, on a lattice grid that discretises the domain Ω. Each random walk models a discrete approximation to the diffusion process governed by the Laplace operator.

This method relies on the interpretation that the solutions of $\phi(A)$, where $A \in \Omega$, correspond to,

$$\phi(A) = \sum_{a \in all\ boundary\ points} g(a) \cdot P_A(a) - \sum_{\text{for all } w} F(w). \tag{15}$$

Here I have defined: $g(a)$ the value of the boundary condition at boundary point a; $P_A(a)$ which is the probability of the random walk starting at A to end at a, given by,

$$P_A(a) = \frac{Number\ of\ times\ random\ walk\ ends\ at\ a}{N}. \tag{16}$$

and $F(w)$, the sum of the contribution from the source function $f(i)$ along the path scaled by h^2 as,

$$F(w) = \sum_{i \in interior\ points\ in\ random\ walk} f(i) \cdot h^2. \tag{17}$$

However, I cannot compute,

$$\sum_{for\ all\ w} F(w), \tag{18}$$

because there are infinitely many paths. Therefore, I approximate it by take the sample average given by:

$$\frac{1}{N} \sum_{i=1}^{N} F(w^{(i)}). \tag{19}$$

Substituting this into Equation 15, gives the approximation for the potential $\phi(A)$,

$$\phi(A) = \sum_{a \in boundary\ points} g(a)\, P_A(a) - \frac{1}{N} \sum_{i=1}^{N} F(w^{(i)}), \tag{20}$$

Implementation in R

To evaluate the effectiveness of the Monte Carlo Method in solving the Laplace equation for a one-dimensional parallel plate capacitor, I simulated the electric potential between two plates separated by 0.1m, with the boundary conditions set to 0V and 5V, respectively. The analytical solution for this setup is known to be linear, given by:

$$u_{\text{true}}(x) = \frac{5}{0.1} x, \tag{21}$$

where u_{true} is the electric potential. To approximate this numerically, a standard fixed-step random walk Monte Carlo method is used to estimate the potential at each interior point A on the lattice. The domain Ω was discretised into a grid of N_x lattice points between $x = 0$ and $x = 0.1$. Because there is no source of charge between the plates our source function $f(i) = 0$ for all lattice points i.

The key idea is that the potential at any interior point is the expected value of the boundary condition reached by a random walk starting from that point. The algorithm proceeds as follows:

For each interior point on the grid:

- A fixed number of random walks (N) are initiated.
- At each step, the walk moves left or right with equal probability.
- When the walk reaches either boundary, it stops, and the boundary voltage (0 or 5 V) is recorded.
- The average of these voltages over all walks gives the estimated potential at that point.

This method was implemented in a function called *solve_poisson_mc_1d*(), which accepts:

- N_x: number of lattice points (grid resolution)
- N: number of random walks per point
- $f(i)$: source function (set to 0 for Laplace)

The implementation supports flexible resolution and sampling parameters, allowing for experimentation with error analysis. The analytical solution was also computed for comparison using the known linear formula, and used as a reference to calculate errors (absolute, mean, and total).

To evaluate convergence and accuracy, I performed simulations across a range of values for:

- P: number of random walks per point
- N_x: number of lattice points

The results were summarised using plots and error metrics, total absolute error vs. random walks (10), total error vs. Number of Lattice points (Figure 11), and used to analyse the efficiency and limitations of the method.

Simulation Results and Analysis

a) Potential vs Distance from Negative Plate

The first experiment aimed to verify whether the Monte Carlo method could accurately reproduce the expected linear potential profile between two plates held at $0V$ and $5V$. A total of $N_x = 15$ lattice points were used, with 400 random walks per interior point.

Figure 9 shows the estimated potential (blue) using the Monte Carlo method compared to the exact linear solution (dashed red). To quantify the approximation accuracy, I computed the absolute error between the Monte Carlo solution $u_{mc}(x)$ and the analytical solution $u_{true}(x)$. The error at each point is defined as:

$$\varepsilon(x) = |u_{mc}(x) - u_{\text{true}}(x)| \,. \tag{22}$$

The mean absolute error (MAE) and root mean squared error (RMSE) were computed to summarise the performance. They showed that the approximation of the linear potential profile between two plate using the Monte Carlo method fit the true linear potential expected well. My results matched closely with the result achieved in the paper demonstrating the correctness of my R implementation with any variation plausibly due to random fluctuation inherent from the random nature of Monte Carlo methods.

b) Total Absolute Error vs Number of Random Walks

I then investigated the MAE dependence on the number of random walks, varying the number of random walks N from 10 to 1000 in increments of 20 keeping the number of lattice points constant, $N_x = 15$. For each value of N, the total absolute error was calculated as the sum of the absolute differences between the Monte Carlo estimate and the analytical solution across all interior points.

The resulting curve, shown in Figure 10, shows a clear inverse relationship between error and the number of walks. Initially, the error drops rapidly with increasing N, indicating fast convergence as more samples are averaged. Demonstrating the stochastic convergence property of Monte Carlo Method, as expected from Monte Carlo theory, the error decreases at a rate proportional

to $\frac{1}{\sqrt{N}}$. However, after $N = 400$ the curve flattens, reflecting diminishing returns. Further increasing N results in smaller incremental improvements. This matches the results found in the paper closely showing correct implementation of the Monte Carlo method.

c) Total Absolute Error vs Number of Lattice Points

The final plot (Figure 11) shows the effect of grid refinement by varying the number of lattice points N_x while keeping the number of random walks per point constant. At first glance, one might expect the error to decrease with a finer discretization. However, the results show that the total absolute error increases as N_x increases.

This apparent contradiction is explained, as discussed in the lectures, increasing the number of evaluation points in Monte Carlo methods improves spatial resolution but introduces more stochastic noise unless the number of samples per point is also increased. This result aligns with the findings in Sonawane's paper and emphasizes the importance of scaling the number of random walks with resolution.

Plots

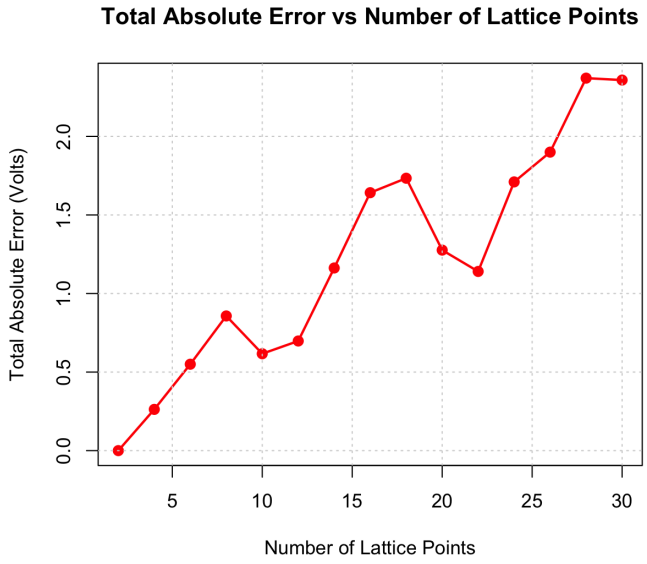

Figure 9: Potential in volts vs distance from negative plate. The figure shows the estimated potential (blue) using the Monte Carlo method with 15 lattice points compared to the exact linear solution (dashed red) between 2 parallel plate with a separation of 0.1m

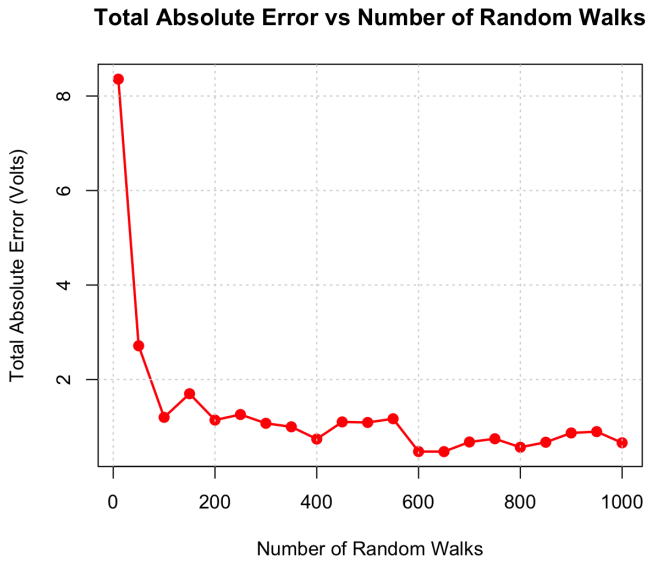

Figure 10: Total absolute error in volts vs the number of random walks for our simulation with 15 lattice points. Initially, the error drops rapidly with increasing N, indicating fast convergence as more samples are averaged. Demonstrating the stochastic convergence property of Monte Carlo Method.

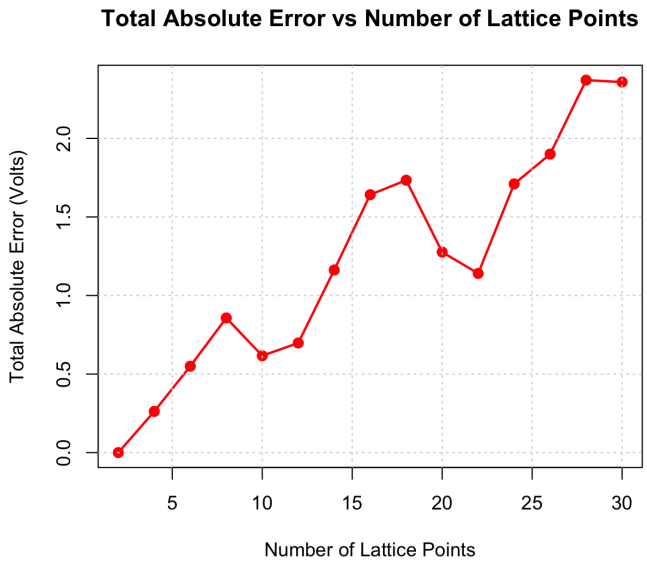

Figure 11: Total absolute error in volts vs the number of lattice points for our simulation with 400 steps per walk. The error increases for higher lattice resolution.