**Markov Chain Monte Carlo (MCMC)**

Markov Chain Monte Carlo (MCMC) methods are a class of algorithms used to sample from probability distributions when direct sampling is difficult. They are particularly useful for estimating the distribution of complex, high-dimensional spaces.

**Metropolis-Hastings Algorithm**

The Metropolis-Hastings algorithm is a specific type of MCMC method. It generates a sequence of sample values from a target probability distribution \( f(x) \), even if the distribution is known only up to a normalizing constant.

**Key Concepts:**

1. **Target Distribution**: The probability distribution we want to sample from.

2. **Proposal Distribution**: A distribution used to generate new sample candidates. In your problem, the proposal distribution is a normal distribution with mean equal to the current sample value and standard deviation .

3. **Acceptance Ratio**: The ratio that determines whether to accept or reject the proposed sample. For the Metropolis-Hastings algorithm:

If , the proposed sample is always accepted. If , the proposed sample is accepted with probability .

**Steps in the Algorithm**

1. **Initialization**: Start with an initial value .

2. **Iteration**

* Propose a new sample from the proposal distribution.
* Compute the acceptance ratio .
* Generate a uniform random number from .
* If , accept the new sample and set . Otherwise, reject the new sample and set .

The given target distribution is:

This is the probability density function of the Laplace distribution (also known as the double-exponential distribution), centred at 0 with a scale parameter of 1.

**Why Use the Logarithm?**

To avoid numerical errors, particularly underflow or overflow issues when dealing with very small or very large numbers, it is common to work with the logarithm of the acceptance ratio:

This ensures stability in computations.

**Outputs**

The algorithm returns a sequence of samples used to generate;

1. **Histogram and Kernel Density Plot**: Estimate the target distribution visually.

2. **Monte Carlo Estimates**: Compute the sample mean and standard deviation, which are estimates of the true mean and standard deviation of the target distribution.

**Summary of the Statistical Background**

- **MCMC Methods**: Useful for sampling from complex distributions.

- **Metropolis-Hastings Algorithm**: Generates a Markov chain that has the target distribution as its equilibrium distribution.

- **Laplace Distribution**: The given is a Laplace distribution with mean 0 and scale 1.

- **Logarithms for Stability**: Using logarithms in the acceptance ratio computation helps prevent numerical errors.

*In the coursework problem, the target distribution is known. This allows us to directly compare the generated samples to the true distribution. In practice, the Metropolis-Hastings algorithm is particularly useful when the target distribution is complex or unknown in full form. Often, we may know the distribution up to a normalizing constant or only have a proportional form of it. The algorithm allows us to generate samples that approximate this distribution, which is crucial for tasks like* ***Bayesian Inference*** *(a statistical method that updates the probability of a hypothesis as more evidence or information becomes available) where direct sampling is infeasible. By using a simpler proposal distribution and computing acceptance probabilities, we can iteratively build a sample that mirrors the target distribution. This method is essential for estimating parameters and understanding distributions in complex models, providing a way to visualize and compute statistics such as mean and standard deviation, even when the exact distribution is difficult to handle analytically.*

**Data Expo 2009: Airline On Time Data**

This is a large dataset; there are nearly 120 million records in total, and it takes up 1.6 gigabytes of space when compressed and 12 gigabytes when uncompressed.

To avoid filtering out columns, we must optimise system resources.

<https://stackoverflow.com/questions/70924810/handling-150-million-rows-with-r-and-sql>

<https://www.r-bloggers.com/2024/03/r-dtplyr-how-to-efficiently-process-huge-datasets-with-a-data-table-backend/>