MCMC

Azmain Biswas

April 2023

Contents

List of Figures

Chapter 1

Introduction

In the ever-evolving landscape of mathematical and statistical research and application, the integration of simulation techniques has emerged as a powerful tool to unravel complex phenomena, validate theoretical frameworks, and facilitate a deeper understanding of intricate mathematical structures. Simulation is a computer-based exploratory exercise that aids in understanding how the behavior of a random or even a deterministic process changes in response to changes in input or the environment. It is essentially the only option left when exact mathematical calculations are impossible, or require an amount of effort that the user is not willing to invest. Even when the mathematical calculations are quite doable, a preliminary simulation can be very helpful in guiding the researcher to theorems that were not a priori obvious or conjectured, and also to identify the more productive corners of a particular problem. Although simulation in itself is a machine-based exercise, credible simulation must be based on appropriate theory. A simulation algorithm must be theoretically justified before we use it.

The classic theory of simulation includes such time-tested methods as the original Monte Carlo, Inverse Transform method, Accept-Reject method, Bivariate techniques from standard distributions in common use. They involve a varied degree of sophistication. Markov chain Monte Carlo is the name for a collection of simulation algorithms for simulating from the distribution of very general types of random variables taking values in quite general spaces. MCMC methods have truly revolutionized simulation because of an inherent simplicity in applying them, the generality of their scopes, and the diversity of applied problems in which some suitable form of MCMC has helped in making useful practical advances. MCMC methods are the most useful when conventional Monte Carlo is difficult or impossible to use.

Simulation depend on various theoretical aspect such as The weak law of Large Number, The Central limit theory, The sample mean and sample variance etcetera.

1.1 Mathematical Preliminaries

Theorem 1.1.1 (The Weak Law of Large Numbers)**.** *Let $X_1, X_2, \ldots$ be a sequence of in dependent and identically distributed random variables having mean μ, Then, for any $\epsilon > 0$,*

$$P\left(\left|\frac{X_1 + X_2 + \ldots + X_n}{n} - \mu\right| > \epsilon\right) \to 0$$

Theorem 1.1.2 (The Central Limit Theorem)**.** *Suppose $X_1, X_2, \ldots$ is a sequence of i.i.d random variables with $E[X_i] = \mu$ and $Var[X_i] = \sigma^2 < \infty$. Then,*

$$\lim_{n\to\infty} P\left(\frac{X_1 + \ldots + X_n - n\mu}{\sigma\sqrt{n}} < n\right) = \Phi(Z)$$

Were, $\Phi(Z)$ *denote the distribution function of a standard normal distribution.*

Chapter 2

Generating Random Variables

2.1 Generating Discrete Random Variables

Main component of a simulation study is the ability to generate random number, where a random number represents the value of random variable uniform distribution on $(0, 1)$.

2.1.1 Pseudorandom Number Generation

Random numbers were originally either manually or mechanically generated, by using spinning wheels or dice rolling or card shuffling but the modern approach is to use a computer to successively generate pseudorandom numbers.

One of the common approaches to generate pseudorandom numbers starts with an initial value x_0, called seed, and then recursively computes successive values $x_n, n \geq 1$, by letting

$$x_n = ax_{n-1} \text{ modulo } m \tag{2.1}$$

where a and m are given positive integers, and where the equation (2.1) means that ax_{n-1} is divided by m and remainder is taken as the value of x_n. Thus, each value of x_n is either $0, 1, \ldots, m-1$ and the quantity x_n/m is Pseudorandom number and follows an approximation to the value of a uniform $(0, 1)$ random variable.

The approach specified by equation (2.1) to generate random numbers is called the Multiplicative Congruential Method.

Another method is

$$x_n = (ax_{n-1} + c) \text{ modulo } m$$

this method is known as *Mixed Congruential Generators* or *Linear congruential Generations (LCGs)* where c is a non-negative integer.

2.1.2 The Inverse Transform Method

Suppose we want to generate the value of a discrete random variable X having probability mass function

$$P(X = x_i) = p_i, \ i = 0, 1, \ldots, \ \sum_i p_i = 1$$

To do this, we generate a random number from a uniform distribution $(0,1)$ U, and set

$$X = \begin{cases} x_0 \text{ if } U < p_0 \\ x_1 \text{ if } p_0 \leq U \leq p_0 + p_1 \\ \vdots \\ x_j \text{ if } \sum_{i=0}^{j-1} p_i \leq U \leq \sum_{i=0}^{j} p_i \\ \vdots \end{cases}$$

Since, for $0 < a < b < 1, P(a \leq U < b) = b - a$, we have,

$$P(X = x_j) = P\left(\sum_{i=0}^{j-1} p_i \leq U < \sum_{i=0}^{j} p_i\right) = p_j.$$

So, X has the desired distribution.

Example 2.1 (Bernoulli Distribution). Let, $X \sim Ber(p)$ where p is success probability i.e. $P(X = 0) = 1 - p$ and $P(X = 1) = p$ and $0 \leq p \leq 1$. Then, to generate X we first generate $U \sim U[0,1]$ then, we set

$$X = \begin{cases} 1, \text{ if } U \leq p \\ 0, \text{ if } U > p \end{cases}$$

Hence, X follows Bernoulli Distribution with the parameter p.

Algorithm for Inverse Transform Algorithm for Generating Bernoulli Distribution:

STEP 1: Generate a random variable $U \sim U[0,1]$.

STEP 2: If $U \leq p$ set $X = 1$ or set $X = 0$.

STEP 3: Go to STEP 1.

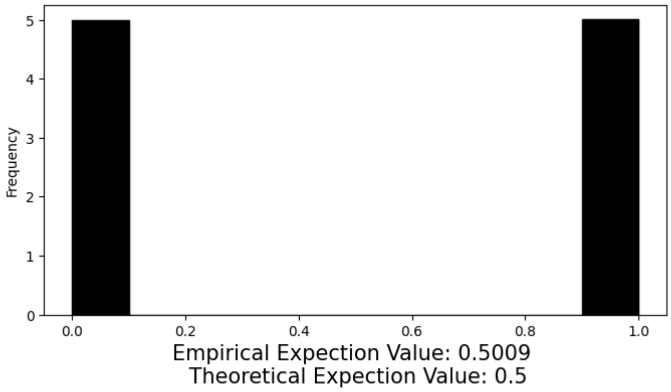

Figure 2.1: Inverse Transform method for generating Bernoulli random numbers with $p = 0.5$

Example 2.2 (Binomial Distribution). Let, $X \sim Bin(n,p)$ then, X has probability mass function

$$f(r) = P(X = r) = \binom{n}{r} p^r (1-p)^{n-r}, \; i = 1, 2, \ldots$$

The generation of $X \sim Bin(n,p)$ by Inverse Transform Algorithm can be tedious. We can use the relation between Binomial and Bernoulli distribution. If $x_i \sim Ber(p), \forall i = 1, 2, \ldots, n$ then, $\sum_{i=1}^{n} x_i \sim Bin(n,p)$.

Hence, by generating x_i n independent random variable from Bernoulli distribution and summing them we get binomial distribution

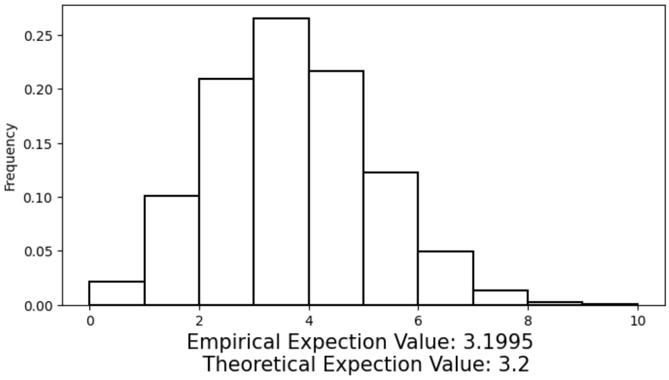

Figure 2.2: Generating binomial random numbers with $n = 10$ and $p = 0.32$

2.2 Generating Continuous Random Variables

2.2.1 The Inverse Transform Algorithm

To generate Continuous random variables The Inverse Transform Algorithm is very important method. It is based on a following theorem.

Theorem 2.2.1. *Let U be a uniform $(0,1)$ random variable. For any continuous distribution function F the random variable X defined by*

$$X = F^{-1}(U)$$

has distribution F.

Proof. Let, F_X denote the distribution function of $X = F^{-1}(U)$. Then,

$$\begin{aligned} F_X(x) &= P(X \leq x) \\ &= P(F^{-1}(U) \leq x) \end{aligned}$$

Since, F is a cumulative distribution function it follows that $F(x)$ is monotonic increasing function of x and range of $F(x)$ is $(0,1)$. Then,

$$\begin{aligned} F_X(x) &= P\left(F\left(F^{-1}(U)\right) \leq F(x)\right) \\ &= P(U \leq F(x)) \\ &= F(x) \text{ since } U \sim U(0,1) \end{aligned}$$

□

The above theory tells us we can generate a random variable X from the continuous distribution function F by generating a random number $U \sim U(0,1)$ and setting $X = F^{-1}(U)$.

Example 2.3 (Exponentian Distribution). Suppose we want to generate a random variable $x \sim Exp(\lambda)$, then its probability density function is

$$f(x) = \lambda e^{-\lambda x}.$$

Hence, The cumulative distribution function is,

$$F(x) = 1 - e^{\lambda x}$$

if we let $x = F^{-1}(u)$, then,

$$\begin{aligned} u &= F(x) = 1 - e^{-\lambda x} \\ 1 - u &= e^{-\lambda x} \\ x &= -\frac{\ln(1-u)}{\lambda} \end{aligned}$$

Hence, we can generate an exponential random variable with parameter 1 by generating a uniform $(0,1)$ random number U and then setting

$$X = F^{-1}(U) = -\frac{\ln(1-U)}{\lambda}.$$

We see that if $U \sim U(0,1)$ then also $1 - U \sim U(0,1)$ thus $\ln(1-U)$ has the same distribution as $\ln(U)$ so,

$$X = F^{-1}(U) = -\frac{\ln(U)}{\lambda}.$$

will also work. If we use second expression then the algorithm will take less computing power hence less time.

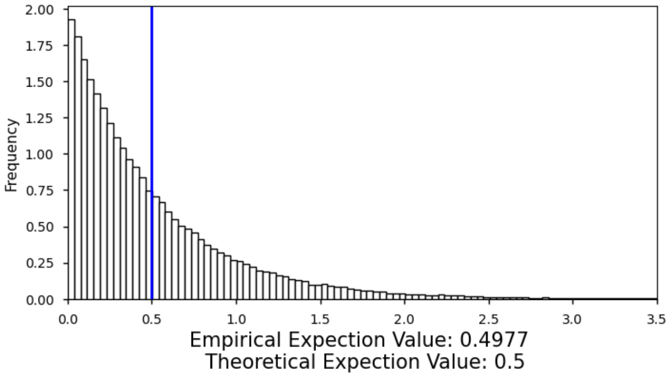

Figure 2.3: Inverse Transform method for generating $Exp(2)$

Example 2.4 (Gamma Distribution). Let $X \sim G(n, \lambda)$ Then, its probability mass function is given by,

$$f(x) = \frac{1}{\Gamma(n)} \lambda^n x^{n-1} e^{-\lambda x}$$

We know if $X_i \sim Exp(\lambda) \forall i = 1, 2, \ldots, n$ then $Y = \sum_i X_i \sim G(n, \lambda)$. As,

$$\begin{aligned} M_Y(t) = E\left[e^{tY}\right] = E\left[e^{\sum_{i=1}^n X_i t}\right] &= E\left[\prod_{i=1}^n e^{X_i t}\right] \\ &= \prod_{i=1}^n E\left[e^{X_i t}\right] \text{ As all } X_i \text{ are independent} \\ &= \prod_{i=1}^n \frac{\lambda}{\lambda - t} = \left(\frac{\lambda}{\lambda - t}\right)^n \end{aligned}$$

Then, Generating n number of $X_i \sim Exp(\lambda)$ and summing them we can easily generate a random variable which follows gamma distribution

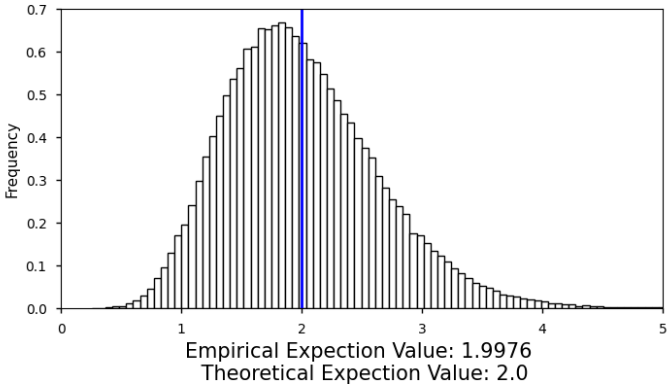

Figure 2.4: $G(10,5)$ generated by summing of $Exp(5)$

2.2.2 Accept - Reject Method

The accept–reject method is useful when it is difficult to directly simulate $f(x)$ but we can generate another density $g(x)$ such that $f(x)/g(x)$ is uniformly bounded and it is much easier to simulate $g(x)$. We simulate X from g, and retain it or toss it according to a probability proportional to $f(x)/g(x)$. Because an X value is either retained or discarded, depending on whether it passes the admission rule, the method is called the accept–reject method. The density $g(x)$ is called the envelope density.

The method proceeds as follows,
STEP 1: Find a density g and a finite constant c such that $\frac{f(x)}{g(x)} \leq c \ \forall x$.
STEP 2: Generate $X \sim g$.
STEP 3: Generate $U \sim U(0,1)$, independent of X.
STEP 4: Retain this generated value X if $U \leq \frac{f(x)}{cg(x)}$.
STEP 5: Repeat the same until the required number of n values of X has been obtained.

The following theorem supports the method.

Theorem 2.2.2. *Let $X \sim g$, and U, independent of, be a distributed as $U[0,1]$. Then the conditional density of X given that $U \leq \frac{f(X)}{cg(X)}$ is f.*

Proof. Denote the CDF of f by F. Then,

$$\begin{aligned} P\left(X \leq x | U \leq \frac{f(X)}{cg(X)}\right) &= \frac{P\left(X \leq x, U \leq \frac{f(X)}{cg(X)}\right)}{P\left(U \leq \frac{f(x)}{cg(x)}\right)} \\ &= \frac{\int_{-\infty}^{x} \int_0^{\frac{f(t)}{cg(t)}} g(t) du dt}{\int_{-\infty}^{\infty} \int_0^{\frac{f(t)}{cg(t)}} g(t) du dt} \\ &= \frac{\int_{-\infty}^{x} f(t) dt}{\int_{-\infty}^{\infty} f(t) dt} = \frac{F(x)}{1} = F(x). \end{aligned}$$

□

Example 2.5 (Generating a Normal Random Variable). To generate a standard normal variable Z i.e. $Z \sim N(0,1)$, note first that the absolute value of Z has probability density function

$$f(x) = \frac{2}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} \ 0 \leq x \leq \infty. \tag{2.2}$$

Then, we can choose g as the exponential density function with mean 1 i.e.

$$g(x) = e^{-x} \; 0 \le x \le \infty$$

Now,

$$\frac{f(x)}{g(x)} = \sqrt{\frac{2}{\pi}} e^{x - \frac{x^2}{2}}$$

and so the maximum value of $f(x)/g(x)$ occurs at the value of x that maximize $x - x^2/2$ hence $x = 1$ so we take

$$c = \max_x \frac{f(x)}{g(x)} = \frac{f(1)}{g(1)} = \sqrt{\frac{2e}{\pi}}.$$

Now,

$$\frac{f(x)}{cg(x)} = \exp\left(x - \frac{x^2}{2} - \frac{1}{2}\right) = \exp\left(\frac{-(x-1)^2}{2}\right)$$

Then, its follows that we can generate the absolute value of a standard normal random variable as follows:
STEP 1: Generate $X \sim Exp(1)$.
STEP 2: Generate $U \sim U(0,1)$, independent of X.
STEP 3: If $U \le \exp\left(-(X-1)^2/2\right)$, retain X, Otherwise, return to Step 1.

Once, we have simulated a random variable X having density function as in Equation (2.2) we can obtain a standard normal Z by letting Z be equally likely to be either X or $-X$. In Step 3, the value X is accepted if $U \le \exp\left(-(X-1)^2/2\right)$, which is equivalent to $-\ln U \ge (X-1)^2/2$. However, in Example (2.3) we have seen that $-\ln U \sim Exp(1)$ When $U \sim U(0,1)$.

So, summing up, we can generate the standard normal random variable Z as follows:
STEP 1: Generate independent $X_1, X_2 \sim Exp(1)$
STEP 2: If $X_2 \ge (X_1 - 1)^2/2$ retain X_1. Otherwise, return to Step 1.
STEP 3: Generate $U \sim U(0,1)$ and set,

$$Z = \begin{cases} X_1 \text{ if } U \le \frac{1}{2}, \\ X_1 \text{ if } U > \frac{1}{2}. \end{cases}$$

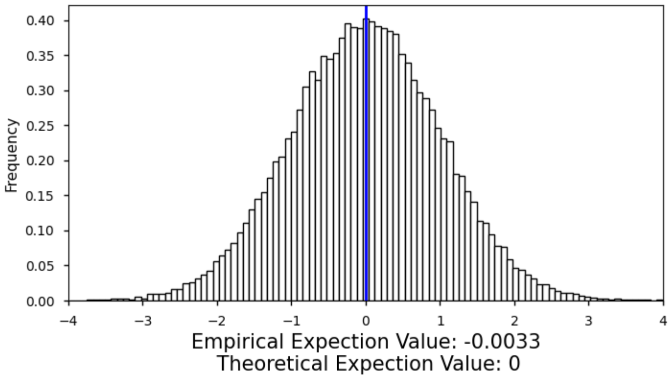

Figure 2.5: Generating $N(0,1)$ with Accept - Reject method

If we want to generate normal random variable to have mean μ and variance σ^2, just take $\mu + \sigma Z$.

Example 2.6 (Generating Beta Distribution). If α and β are both getter then 1, then Beta density is uniformly bounded and its maximum attain at $\frac{\alpha-1}{\alpha+\beta-2}$. As a result the $U[0,1]$ density can be served as an envelope density for generating such Beta distribution by using accept-reject method. Precisely, generate U, $X \sim U[0,1]$ (independently), and retain the value if $U \leq \frac{f(X)}{\sup_x f(X)}$, where,

$$f(X) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1}, 0 < x < 1.$$

Because

$$\sup_x f(X) = f\left(\frac{\alpha-1}{\alpha+\beta-2}\right) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \frac{(\alpha-1)^{\alpha-1}(\beta-1)^{\beta-1}}{(\alpha+\beta-2)^{\alpha+\beta-2}}$$

The algorithm finally works out as follows:
STEP 1: Generate independent $U, X \sim U[0,1]$.
STEP 2: Retain the value X if,

$$U \leq \frac{X^{\alpha-1}(1-X)^{\beta-1}(\alpha+\beta-2)^{\alpha+\beta-2}}{(\alpha-1)^{\alpha-1}(\beta-1)^{\beta-1}}.$$

Otherwise, return to STEP 1.

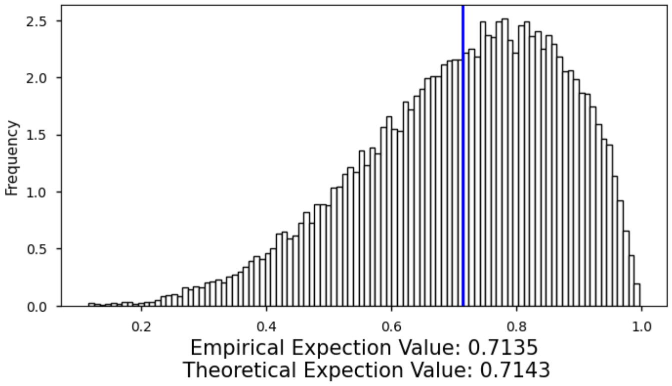

Figure 2.6: Generating Beta(5,2) with accept-reject method

An issue about an accept-reject method is the acceptance rate. Our goal make it as large as possible to increase the efficiency of the method. This can be achieved by choosing c to be smallest possible number, described in the result bellow.

Theorem 2.2.3 (Acceptance Rate). *For an accept-reject scheme, the probability that an $X \sim g$ is acceded is $\frac{1}{c}$, and is maximized when c is chosen to be $c = \sup_x \frac{f(x)}{g(x)}$.*

Proof.

$$\begin{aligned} P\left(U \leq \frac{f(x)}{cg(x)}\right) &= \int_{-\infty}^{\infty} \int_0^{\frac{f(x)}{cg(x)}} g(t) du dt \\ &= \int_{-\infty}^{\infty} \frac{f(t)}{cg(t)} g(t) dt = \int_{-\infty}^{\infty} \frac{f(t)}{c} dt = \frac{1}{c}. \end{aligned}$$

Because any c that can be chosen must be at least as large as $\sup_x \frac{f(x)}{g(x)}$, obviously $1/c$ is maximized by choosing $c = \sup_x \frac{f(x)}{g(x)}$. □

In the example (2.5) for $N(0,1)$ the acceptance rate is $\sqrt{\frac{\pi}{2e}} = 0.7601$. And for example (2.6) for Beta(5,2) the acceptance rate is 0.4069

2.2.3 The Polar Method for Generating Normal Random Variables

Let X and Y be independent slandered normal random variable and let R and θ denote the polar coordinates of vector (X, Y). That is,

$$R^2 = X^2 + Y^2$$
$$\tan\theta = \frac{Y}{X}$$

Since X and Y are independent, their joint density is the product of their individual densities and thus given by

$$\begin{aligned} f(x,y) &= \frac{1}{\sqrt{2\pi}} e^{-x^2/2} \frac{1}{\sqrt{2\pi}} e^{-y^2/2} \\ &= \frac{1}{\sqrt{2\pi}} e^{-(x^2+y^2)/2} \end{aligned}$$

To determine the joint density of R^2 and Θ - call it $f(d,\theta)$ we make the change of variables

$$d = x^2 + y^2, \quad \theta = \tan^{-1}\left(\frac{y}{x}\right)$$

Then the joint density function of R^2 and Θ is,

$$f(d,\theta) = \frac{1}{2}\frac{1}{2\pi} e^{-d/2}, \quad 0 < d < \infty, 0 < \theta < 2\pi. \tag{2.3}$$

As $f(d,\theta)$ is equal to product of the product of $Exp(1/2)$ density and $U(0,2\pi)$, it follows that, R^2 and Θ are independent, with $R^2 \sim Exp(1/2)$ and $\Theta \sim U(0,2\pi)$

Hence to generate a pair of independent slandered normal random variables X and Y by generating R^2 and Θ in polar coordinates and then transform back to rectangular coordinates. Hence the algorithm is:

STEP 1: Generate random number $U_1, U_2 \sim U(0,1)$.

STEP 2: $R^2 = -2\ln U_1$ and $\Theta = 2\pi U_2$.

STEP 3: Now let,

$$\begin{aligned} X &= R\cos\Theta = \sqrt{-2\ln U_1}\cos(2\pi U_2) \\ Y &= R\sin\Theta = \sqrt{-2\ln U_1}\sin(2\pi U_2). \end{aligned} \tag{2.4}$$

The transformation given by equitations (2.4) are known as Box-Muller transformation.

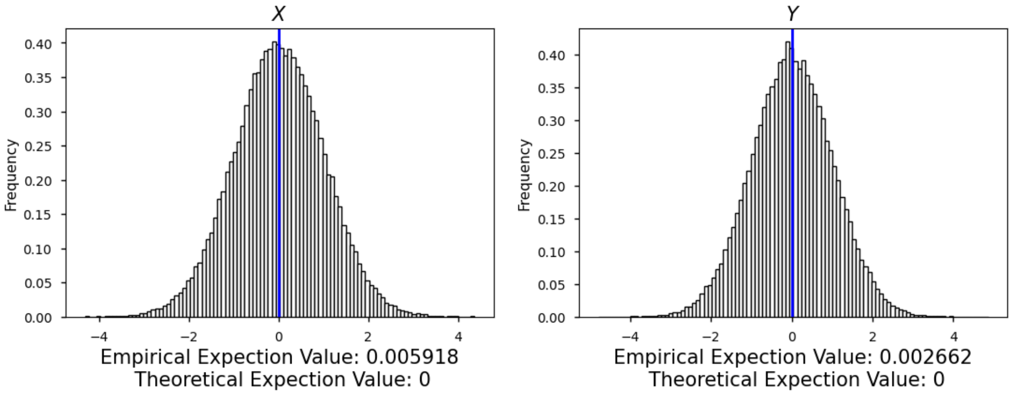

Figure 2.7: Generating independent $X, Y \sim N(0,1)$ with polar method

Chapter 3

Ordinary Monte Carlo Simulation

Polish-American mathematician Stanislaw Ulam, recovering from an illness, was playing a lot of solitary card game. He wanted to calculate the probability of winning and quickly it is impossible to calculate analytically. Then he thought about playing lots of hands counting number of wins, but decided it will take years. After falling several times he asked Von Neumann to build a program to simulate solitary card game in ENIAC. Then Around 1940 they used Monte Carlo simulation in Manhattan Project in which physicists wanted to understand how the physical properties of neutrons would be affected by various possible scenarios following a collision with a nucleus.

The basis for Monte Carlo is Law of Large Number. If we simulate large number $X_1, X_2, \ldots$ iid copes of random variable X Then we can approximate the true value $E(f(X))$ by simple mean $\frac{1}{n}\sum_{i=1}^{n} f(X_i)$. Here in Monte Carlo Random Sampling play the key factor for good estimation of $E(f(X))$.

3.1 Evaluate Integrals using Monte Carlo simulation

The application of Monte Carlo is to computation of integrals. Let $g(x)$ be a function and suppose we wanted to compute I where

$$I = \int_0^1 g(x)dx$$

To compute the value of I, note that if $U \sim U[0,1]$ then we can express I as

$$I = E[g(U)]$$

If $U_1, \ldots, U_n$ are independent uniform $(0,1)$ random variables, it thus follows that the random variables $g(U_1), \ldots, g(U_n)$ are independent and identically distributed random variable having mean I. Therefore, by law of large numbers, its follows that, with probability,

$$\sum_{i=0}^{n} \frac{g(U_i)}{n} \to E(g(U)) = I \text{ as } k \to \infty$$

Hence we can approximate I by generating a large number of random numbers U_i and taking as our approximation the average value of $g(U_i)$.

If we wanted to compute

$$I = \int_a^b g(x)dx$$

then , by taking the substitute $y = (x-a)/(b-a)$, $dy = dx/(b-a)$, we see that

$$\begin{aligned} I &= \int_0^1 g(a+[b-a]y)(b-a)dy \\ &= \int_0^1 h(y)dy \end{aligned}$$

Where $h(y) = (b-a)g(a+[b-a]y)$. Thus we can approximate I by continually generating random numbers and then taking the average value of h evaluated at these random numbers.

Similarly, if we wanted

$$I = \int_0^\infty g(x)dx$$

we could apply the substitution $y = 1/(x+1)$, $dy = -dx/(x+1)^2 = -y^2dx$, to obtain the identity

$$I = \int_0^1 h(y)dy$$

where,

$$h(y) = \frac{g(\frac{1}{y}-1)}{y^2}$$

Using this technique we can also evaluate multidimensional integrals. Suppose that g is a function with n-dimention argument and we are interested in computing

$$I = \int_0^1 \int_0^1 \dots \int_0^1 g(x_1, x_2, \dots, x_n)dx_1dx_2 \dots dx_2.$$

Then, we can express I as

$$I = E(g(U_1, U_2, \dots U_n))$$

where $U_1, U_2, \dots U_n \sim U[0,1]$ Hence if we generate k independent sets, each consisting of n independent $U[0,1]$ random variable

$$\begin{gathered} U_1^1 \dots U_n^1 \\ U_1^2 \dots U_n^2 \\ \vdots \\ U_1^k, \dots, U_n^k \end{gathered}$$

then, since the random variables $g(U_1^i, \dots, U_n^i), i = 1, 2, \dots, k$ are all independent and identically distributed random variable with mean I, we can estimate I by $\sum_{i=1}^k g(U_1^i, \dots, U_n^i)/k$.

Example 3.1. Suppose we want to integrate,

$$I = \int_0^1 e^{-\frac{x^2}{2}}dx.$$

Then we can say that,

$$I = E(e^{-\frac{U^2}{2}})$$

where, $U \sim U[0,1]$. Then simulating a large number of $U_1, U_2, \dots, U_n \sim U[0,1]$ and calculating,

$$\sum_{i=1}^n \frac{e^{-\frac{U_i^2}{2}}}{n}$$

we can evaluate I.

Hence the algorithm is:

STEP 1: Generate $U \sim U[0,1]$.

STEP 2: Calculate $e^{-\frac{U^2}{2}}$ and retain it. goto STEP 1.

STEP 3: After large number of iteration evaluate the average.

Monte Carlo sample size	Monte Carlo Estimate of I = 0.8556
50	0.8555
100	0.8558
1000	0.8555
10000	0.8556
100000	0.8556

Table 3.1: Monte Carlo Integration of $e^{-x^2/2}$.

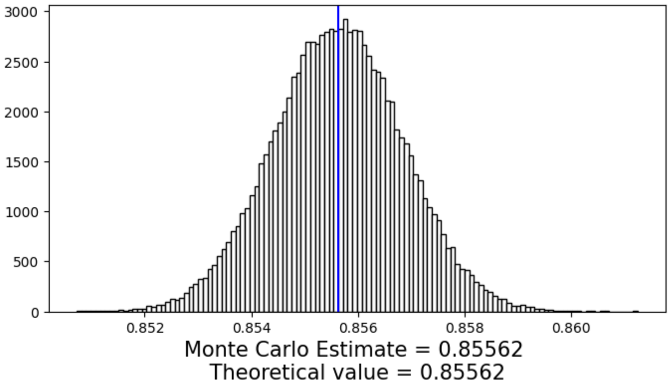

Figure 3.1: Monte Carlo Integration of $e^{-x^2/2}$

Here, we see by the time the Monte Carlo sample size is 100000, we get fairly accurate estimates for the value I.

3.2 The Estimation of π

Suppose that the random vector (X, Y) is uniformly distribution in the square of area 4 centered at the origin. That is, it is a random point in the region specified in Figure 3.2. Let us consider now the probability that this random point in the square in contained within the inscribed circle of radius 1 like the Figure 3.3.

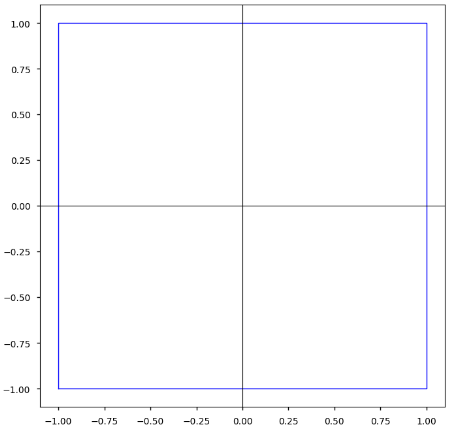

Figure 3.2: Square

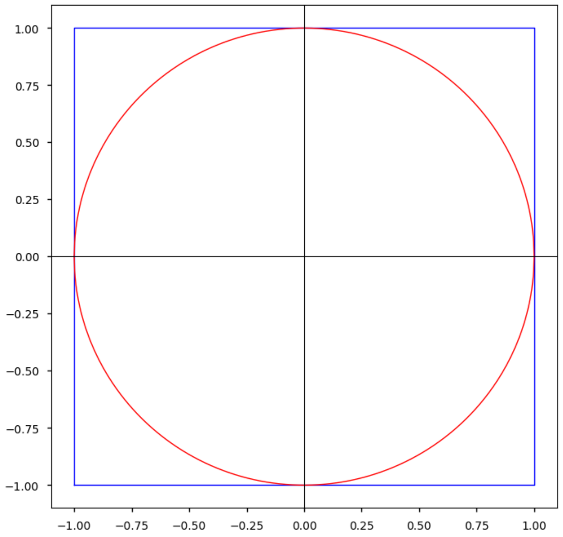

Figure 3.3: Circle within Square

Note that since (X, Y) is uniformly distributed in the square it follows that

$$\begin{aligned} P((X,Y) \text{ is in the circle }) &= P(X^2 + Y^2 \leq 1) \\ &= \frac{\text{Area of the circle}}{\text{Area of the square}} = \frac{\pi}{4} \end{aligned}$$

Hence we generate a large number of points in the square, the proportion of points that fall within the circle will be approximately $\pi/4$. Now, if X and Y were independent and both were uniformly distributed over $(-1, 1)$, their joint density would be

$$\begin{aligned} f(x,y) &= f(x)f(y) \\ &= \frac{1}{2} \times \frac{1}{2} \\ &= \frac{1}{4}, \ -1 \leq x \leq 1, \ -1 \leq y \leq 1 \end{aligned}$$

Since, the density function of (X, Y) is constant in the square, it thus follows that (X, Y) is uniformly distributed in the square. Now, if $U \sim U[0, 1]$ then $2U \sim U[0, 2]$ and so $2U - 1 \sim U[-1, 1]$. Therefore, if we generate ranom numbers U_1 and U_2 and set $X = 2U_1 - 1$ and $Y = 2U_2 - 1$, and define,

$$I = \begin{cases} 1 \text{ if } X^2 + y^2 \leq 1 \\ 0 \text{ otherwise} \end{cases}$$

Then,

$$E(I) = P(X^2 + y^2 \leq 1) = \frac{\pi}{4}.$$

Hence the Algorithm for estimating π is:
STEP 1: Set Circle = 1.
STEP 2: Generate $U_1, U_2 \sim U[0, 1]$
STEP 3: If $(2U_1 - 1)^2 + (2U_2 - 1)^2 \leq 1$ Set Circle = Circle + 1, Otherwise return to STEP 2.
STEP 4: After simulating N time, set Area of Circle = Circle/N

Monte Carlo sample	Monte Carlo Estimate of π
50	2.9600
100	3.0800
1000	3.1200
10000	3.1428
100000	3.1397
1000000	3.1394
10000000	3.1414

Table 3.2: Monte Carlo Estimates of π

Here, we see by the time the Monte Carlo sample size is 10000000, we get fairly accurate estimates for π

Chapter 4

Importance Sampling

There are two different ways to think about importance sampling. The more traditional one is to go back to the primary problem that Monte Carlo wants to solve, namely to approximate the value of an expectation $\mu = \int \phi_0(x)dF_0(x)$ for some function ϕ_0 and some CDF F_0. However, (ϕ_0, F_0) is not the only pair (ϕ, F) for which $\int \phi(x)dF(x)$ equals the specific number μ. Indeed, given any other CDF F_1,

$$\begin{aligned}\mu &= \int \phi_0(x)dF_0(x) \\ &= \int \phi_0(x)\frac{dF_0}{dF_1}(x)dF_1(x) \\ &= \int \lambda(x)\phi_0(x)dF_1(x).\end{aligned}$$

where $\lambda(x) = \frac{dF_0}{dF_1}(x)$. If F_0, F_1 have densities f_0, f_1, then $\lambda(x) = \frac{f_0(x)}{f_1(x)}$; if F_0, F_1 have respective pmfs f_0, f_1, then also $\lambda(x) = \frac{f_0(x)}{f_1(x)}$ This raises the interesting possibility that we can sample from a general F_1, and subsequently use the usual Monte Carlo estimate

$$\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} \lambda(X_i)\phi_0(X_i) = E_{F_1}[\lambda(X_i)\phi_0(X_i)].$$

where $X_1, X_2, \ldots, X_n$ is Monte Carlo sample from F_1. Importance sampling poses the problem of finding an optimal choice of F_1 for which to sample, so that $\hat{\mu}$ has the smallest possible variance. The distribution F_1 hat ultimately gets chosen is called the *importance sampling distribution.*

We can visualize this method by an example.

Example 4.1. Suppose we want to evaluate

$$I = \int_0^{10} e^{-2|x-5|}dx.$$

doing it analytically we get $I = 0.9999$.

Now, suppose $h(x) = e^{-2|x-5|}$ then we want to evaluate

$$I = \int_0^{10} h(x)dx$$

Now,

$$\begin{aligned} I &= \int_0^{10} h(x)dx \\ &= \int_0^{10} h(x)\frac{10}{10}dx \\ &= \int_0^{10} 10 \times h(x)\frac{1}{10}dx = \int_0^{10} 10h(x)p(x)dx \text{ where } p(x) \text{ pdf of } U(0,1) \\ &= E_U[10 \times h(U)] \text{ where, } U \sim U(0,10). \end{aligned} \tag{4.1}$$

By Ordinary Monte Carlo technique we can estimate I by $\frac{1}{N}\sum_{i=1}^{N} 10 \times h(U_i)$ where $U_i \sim U(0,10)$ for $i = 1, 2, \ldots, N$ for the large number of N.

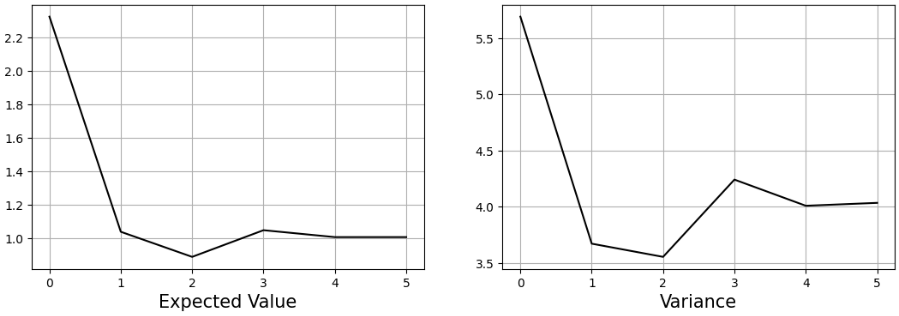

Figure 4.1: Monte Carlo integration of $\int_0^{10} e^{-2|x-5|}dx$.

Sample Size	Estimated Value I=0.9999	Variance
10	2.3243	5.6912
100	1.0372	3.6717
1000	0.8871	3.5543
10000	1.0467	4.2416
100000	1.0053	4.0089
1000000	1.0054	4.0346

Table 4.1: Monte Carlo integration of $\int_0^{10} e^{-2|x-5|}dx$.

Here we can see that for sample size 1M, we get a pretty good estimation of I with less then 0.6% error. But the variance is very high. If we see at left of Figure (4.2) we can see we are taking unnecessary value from low frequency part of $h(x)$ that is, from extreme left and right. If we choose an importance sampling distribution that has similar curve as $h(x)$, then we can estimate I with low variance. If we choose importance sampling distribution as $N(5,1)$ then we see from right of Figure (4.2) it has similar pattern as $h(x)$.

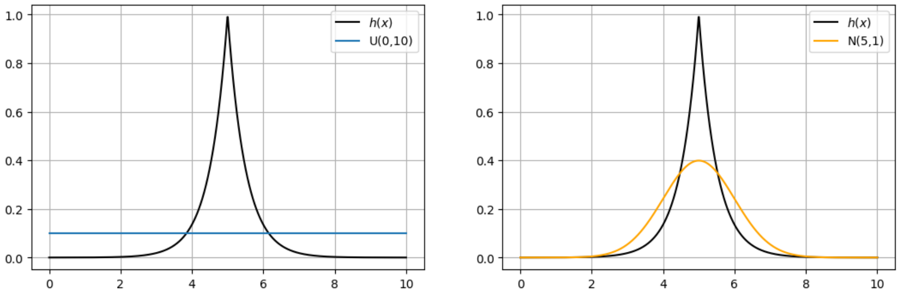

Figure 4.2: $h(x)$ with $U(0,10)$ and $N(5,1)$

Let, $q(x)$ is pdf of $N(0,1)$ then, from (4.1)

$$\begin{aligned}
I &= \int_0^{10} 10h(x)p(x)dx \\
&= \int_0^{10} 10h(x)\frac{p(x)}{q(x)}q(x)dx \\
&= E_X\left[10h(X)\frac{p(X)}{q(X)}\right] \text{ where } X \sim N(5,1) \\
&= E_X\left[10h(X)\lambda(X)\right] \text{ where } X \sim N(5,1)
\end{aligned}$$

Where, $\lambda(x) = \frac{p(X)}{q(X)}$ here $p(x)$ is pdf of $U(0,1)$ and $q(x)$ is pdf of $N(5,1)$. Now using usual Monte Carlo estimate

$$I = \frac{1}{N}\sum_{i=1}^{N} 10h(X_i)\lambda(X_i)$$

where $X_1, X_2, \dots, X_N$ is Monte Carlo sample from $N(5,1)$.

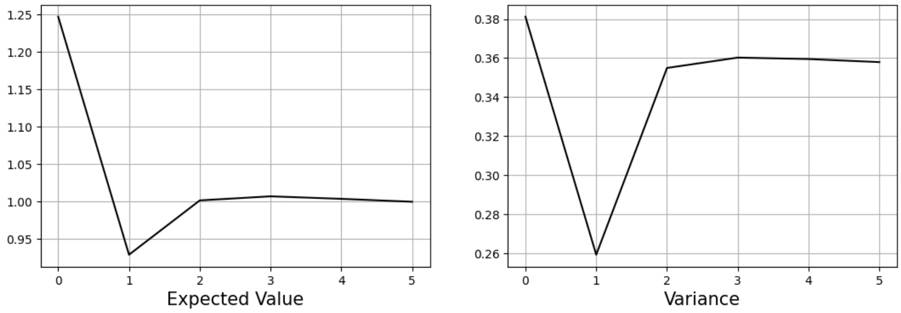

Figure 4.3: Evaluating $\int_0^{10} e^{-2|x-5|}dx$ using Importance Sampling.

Sample Size	Estimated Value (I=0.9999)	Variance
10	1.2473	0.3812
100	0.9292	0.2593
1000	1.0018	0.3550
10000	1.0072	0.3603
100000	1.0039	0.3595
1000000	0.9999	0.3580

Table 4.2: Evaluating $\int_0^{10} e^{-2|x-5|}dx$ using Importance Sampling.

Here we can see that the estimation of I is pretty close and variance is also lower the Original Monte Carlo method.

A more contemporary view of importance sampling is that we do not approach importance sampling as an optimization problem, but because the circumstances force us to consider different sampling distributions F.

Now, we also assume that F_0, F_1 both have densities, say f_0, f_1. If F_0, F_1 are both discrete then the notation only change but the argument is same. Suppose then $f_i(x) = \frac{h_i(x)}{c_i}$, $i = 0, 1$, where the assumption is that h_0, h_1 are completely known and also computable, but c_0, c_1 are unknown and are not even computable. Then, as we showed above, for any function ϕ for which the expectation $E_{F_0}[\phi(X)]$ exist,

$$\begin{aligned}\mu = E_{F_0}[\phi(X)] &:= \int \frac{f_0(x)}{f_1(x)}\phi(x) f_1(x)dx \\ &= \frac{c_1}{c_0}\int \frac{\phi(x)h_0(x)}{h_1(x)} f_1(x)dx \\ &= \frac{c_1}{c_0} E_{F_1}\left(\frac{\phi(X)h_0(X)}{h_1(X)}\right).\end{aligned}$$

This is a useful reduction, but we have to deal with the fact that ratio $\frac{c_1}{c_0}$ is not known to us. Now, if we use the special function $\phi(x) \equiv 1$, the same representation above gives us

$$\begin{aligned}1 &= \frac{c_1}{c_0} E_{F_1}\left(\frac{h_0(X)}{h_1(X)}\right) \\ \implies \frac{c_1}{c_0} &= \frac{1}{E_{F_1}\left(\frac{h_0(X)}{h_1(X)}\right)}\end{aligned}$$

and because h_0, h_1 are explicitly known to us, we have a way to get rid of the quotient $\frac{c_1}{c_0}$ and write the final *importance sampling identity*

$$E_{F_0}[\phi(x)] = \frac{E_{F_1}\left(\frac{\phi(X)h_0(X)}{h_1(X)}\right)}{E_{F_1}\left(\frac{h_0(X)}{h_1(X)}\right)}$$

We can now use an available Monte Carlo sample $X_1, X_2, \ldots, X_n$ from F_1 to find Monte Carlo estimates for $\mu = E_{F_0}[\phi(x)]$

The basic plug-in estimate for μ is the so-called ratio estimate

$$\hat{\mu} = \frac{\sum_{i=1}^{n} \frac{\phi(X_i)h_0(X_i)}{h_1(X_i)}}{\sum_{i=1}^{n} \frac{h_0(X_i)}{h_1(X_i)}}.$$

Example 4.2 (Binomial Bayes problem with an Atypical Prior). Suppose $X \sim Bin(m, p)$ for some fixed m and p has the prior density $c\sin^2(\pi p)$, where c is a normalizing constant. Throughout the example, c denotes a generic constant, and is not intended to mean the same constant at every use.

The posterior density of p given $X = x$ is

$$\pi(p|X = x) = cp^x(1-p)^{m-x}\sin^2(\pi p), \quad 0 < p < 1.$$

The problem is to find the posterior mean

$$\mu = c\int_0^1 p[cp^x(1-p)^{m-x}\sin^2(\pi p)]dp.$$

We use importance sampling to approximate the value of μ. Towards this, choose

$$\phi(p) = p,\ h_0(p) = p^x(1-p)^{m-x}\sin^2(\pi p),\ h_1(p) = p^x(1-p)^{m-x},$$

so that if $p_1, p_2, \ldots, p_n$ are samples from F_1, (i.e. $p_i \sim Beta(x+1, m-x+1)$), then the importance sampling estimate of the posterior mean μ is

$$\begin{aligned}\hat{\mu} &= \frac{\sum_{i=1}^{n} \frac{\phi(p_i)h_0(p_i)}{h_1(p_i)}}{\sum_{i=1}^{n} \frac{h_0(p_i)}{h_1(p_i)}} \\ &= \frac{\sum_{i=1}^{n} p_i \sin^2(\pi p_i)}{\sum_{i=1}^{n} \sin^2(\pi p_i)}.\end{aligned}$$

Note that we did not need to calculate the normalizing constant in the posterior density. We take $m = 100$, $x = 45$ for specificity.

Sample Size	Importance Sampling Estimate of μ	Variance
20	0.4821	0.5465
50	0.4629	0.5162
100	0.4597	0.4929
250	0.4635	0.4902
500	0.4585	0.4962

Table 4.3: Importance Sampling Estimates of μ for Different Sample Sizes
Here, even with importance sampling size $n = 50$, the estimation error is pretty low.

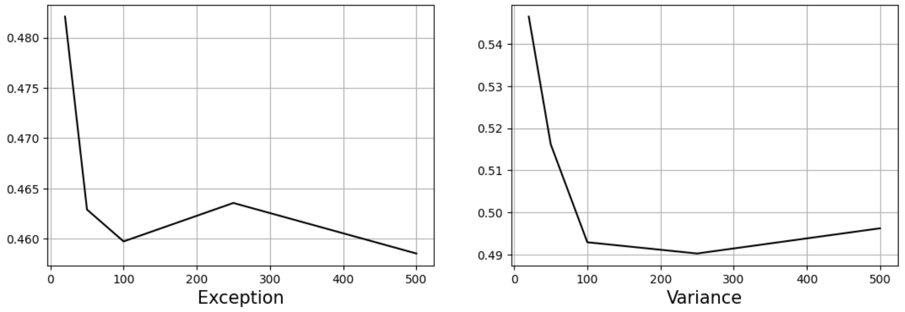

Figure 4.4: Exception and variance graph of Binomial Bayes problem with an Atypical Prior

4.1 Optimal Importance Sampling Distribution

We now address the question of the optimal choice of the importance sampling distribution. There is no unique way to define what an optimal choice means. We formulate one definition of optimality and provide an optimal importance sampling distribution. The optimal choice would not be practically usable, as we shown. However, the solution still gives useful insight.

Theorem 4.1.1. *Consider the importance sampling estimator* $\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} \lambda(X_i)\phi(X_i)$ *for* $\mu = \int \phi(x)f_0(x)dx$*, where* $\lambda(x) = \frac{f_0(x)}{f_1(x)}$*, and* $X_1, \ldots, X_n$ *are iid observations from* F_1 *Assume that* $\phi(x) \geq 0$*, and* $\mu > 0$*. Then,* $Var_{F_1}(\hat{\mu})$ *is minimized when* $f_1(x) = \frac{\phi(x)f_0(x)}{\mu}$*.*

Proof. Because $X_1, \ldots, X_n$ is iid, so are $\lambda(X_1)\phi(X_1), \ldots, \lambda(X_n)\phi(X_n)$, and hence,

$$Var_{F_1}(\hat{\mu}) = \frac{1}{n} Var_{F_1}(\lambda(X_1)\phi(X_1)).$$

Clearly, this is minimized when with probability one under F_1, $\lambda(X_1)\phi(X_1)$ is constant, say k.

The constant k must be equal to the mean of $\lambda(X_1)\phi(X_1)$, that is,

$$\begin{aligned} k &= \int \lambda(x)\phi(x)f_1(x)dx \\ &= \int \frac{\phi(x)f_0(x)}{f_1(x)}f_1(x)dx \\ &= \int \phi(x)f_0(x)dx = \mu. \end{aligned}$$

Therefore, the optimal importance sampling density satisfies $\lambda(x)\phi(x) = \mu$ hence,

$$f_1(x) = \frac{\phi(x)f_0(x)}{\mu}.$$

□

This is not usable in practice, because it involves μ,which is precisely the unknown number we want to approximate. However, the theoretically optimal solution suggests that the importance sampling density should follow key properties of the unnormalized function $\phi(x)f_0(x)$. For example, f_1 should have the same shape and tail behavior as $\phi(x)f_0(x)$.

We have seen this phenomena in the Example (4.2). Because Graph of $\phi(x)h_0(x)$ and $h_1(x)$ in the Figure (4.5), they both have the same key properties.

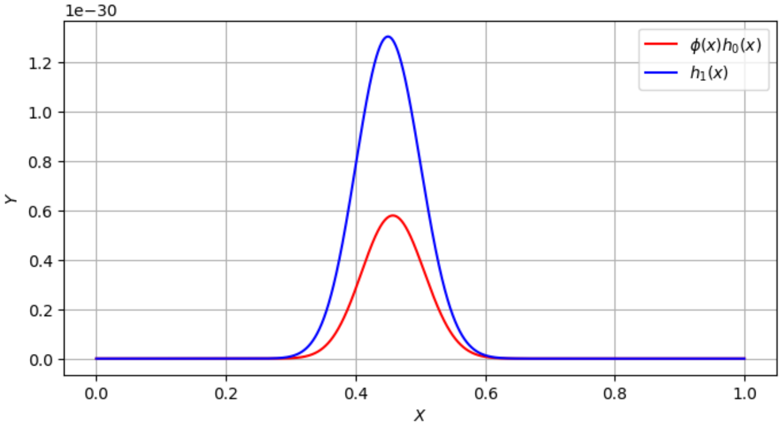

Figure 4.5: Graph of $\phi(x)h_0(x)$ and $h_1(x)$