Topic 10

The Law of Large Numbers

Monte Carlo Integration

Outline

Monte Carlo Integration

Monte Carlo methods is a collection of computational algorithms that use stochastic simulations to approximate solutions to questions that are very difficult to solve analytically.

This approach has seen widespread use in fields as diverse as statistical physics, astronomy, population genetics, protein chemistry, and finance.

Monte Carlo Integration

Let $X_1, X_2, \ldots$ be independent random variables uniformly distributed on the interval $[a, b]$ and write f_X for their common density.

Then, by the law of large numbers, for n large we have that

$$\overline{g(X)}_n = \frac{1}{n}\sum_{i=1}^{n} g(X_i) \approx Eg(X_1) = \int_a^b g(x) f_X(x)\ dx = \frac{1}{b-a}\int_a^b g(x)\ dx.$$

$$\text{Thus,} \quad \int_a^b g(x)\ dx. \approx (b-a)\overline{g(X)}_n.$$

Recall that in calculus, we defined the average of g to be

$$\frac{1}{b-a}\int_a^b g(x)\ dx.$$

We can now interpret this integral as an expected value.

Monte Carlo Integration

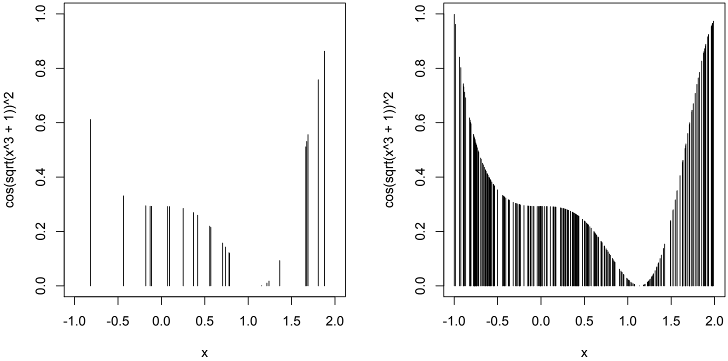
Thus, Monte Carlo integration leads to a procedure for estimating integrals.

- Simulate uniform random variables $X_1, X_2, \ldots, X_n$ on the interval $[a, b]$,
- Evaluate $g(X_1), g(X_2), \ldots, g(X_n)$.
- Average this values and multiply by $b - a$ to estimate the integral.

Let $\cos^2(\sqrt{x^3 + 1})$ for $x \in [-1, 2]$, to find $\int_{-2}^{1} g(x)\ dx$. The three steps above become the following R code.

```
> x<-runif(250,-1,2)
> g<-cos(sqrt(x^3+1))^2
> 3*mean(g)
[1] 1.074919
```

Monte Carlo Integration



Monte Carlo integration of $g(x) = \cos^2(\sqrt{x^3+1})$ on $[-1, 2]$, we simulate n uniform random variables using `runif(n,-1,2)` and then use R to compute `3*mean(cos(sqrt(x^3+1))^2)`. $n = 25$ (left) and $n = 250$ (right) are shown

Monte Carlo Integration

With only a small change in the algorithm, we can also use this to evaluate multivariate integrals. For example, in three dimensions, the integral

$$\int_{a_1}^{b_1}\int_{a_2}^{b_2}\int_{a_3}^{b_3} g(x,y,z)\,dz\,dy\,dx \approx (b_1-a_1)(b_2-a_2)(b_3-a_3)\frac{1}{n}\sum_{i=1}^{n} g(X_i,Y_i,Z_i).$$

where $X_i \sim U(a_1,b_1)$, $Y_i \sim U(a_2,b_2)$, and $Z_i \sim U(a_3,b_3)$

Example. To estimate

$$\int_{-2}^{2}\int_{1/2}^{1}\int_{0}^{1} \frac{e^{-x^2/2y}}{x^2z+1}\,dz\,dy\,dx$$

```
> x<-runif(250,-2,2);y<-runif(250,1/2,1);z<-runif(250)
> g<-exp(-x^2/(2*y))/(x^2*z+2)
> 4*0.5*1*mean(g)
[1] 0.4550264
```

Monte Carlo Integration

Monte Carlo integration uses the averages of a simulated random sample and consequently, its value is itself random. To obtain a sense of the distribution of the approximations to the integral

$$\int_0^8 \frac{1+e^{-x/2}}{\sqrt[3]{x}}\,dx,$$

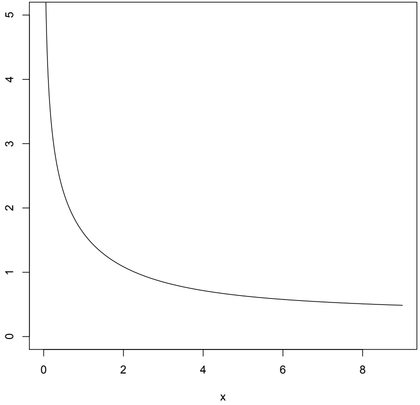
we perform 1000 simulations using 250 uniform random variables. The command `Ig<-rep(0,1000)` creates a vector of 1000 zeros. This is added so that R has a place ahead of the simulations to store the results.

```
> Ig<-rep(0,1000)
> for (i in 1:1000){x<-runif(250,0,8);Ig[i]<-8*mean((1+exp(-x/2))/x^(1/3))}
> mean(Ig)
[1] 8.120468
> sd(Ig)
[1] 0.4715746
```

Monte Carlo Integration

To reduce the standard deviation, we can

- Increase the size of the simulation.
  - An increase from 250 to 1000 decreases the variance by a factor of 4 and thus the standard deviation by a factor of 2.
- Concentrate the values of x where the function g changes rapidly.
  - Such a strategy is called importance sampling.



The graph of $g(x)$

Importance Sampling

Goal. Reduce the standard deviation in the approximation of the integral

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

Write $g(x) = w(x)f_X(x)$ where

- $f_X(x)$ is a density function that captures the change in $g(x)$ *and* has an easy to determine distribution function $F_X(x)$.
  - f_X is called the importance sampling function or the proposal density.
  - w is called the importance sampling weight.

Now, simulate $X_1, X_2, \ldots, X_n$ independent random variables with common density f_X. Then by the law of large numbers.

$$\frac{1}{n}\sum_{i=1}^{n} w(X_i) \approx Ew(X_1) = \int_a^b w(x)f_X(x)\ dx = \int_a^b g(x)\ dx.$$

Importance Sampling

$$\int_0^8 \frac{1+e^{-x/2}}{\sqrt[3]{x}}\,dx = \int_0^8 (1+e^{-x/2})\frac{1}{\sqrt[3]{x}}\,dx$$

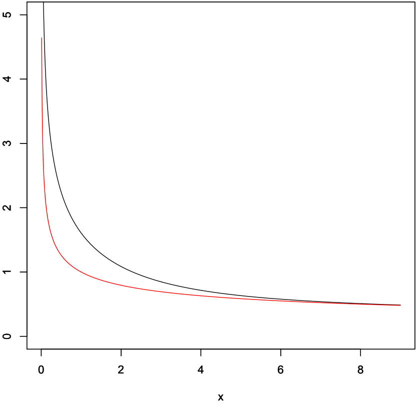
The distribution function

$$F_x(x) = c\int_0^x \frac{1}{\sqrt[3]{t}}\,dt = \frac{3c}{2}t^{2/3}\Big|_0^x = \frac{3c}{2}x^{2/3}$$

Now, $1 = F_X(8) = \frac{3c}{2}8^{2/3} = \frac{3c}{2}4 = 6c.$

So, $c = 1/6$ and $f_X(x) = \frac{1}{6\sqrt[3]{x}}$ is a density

and distribution function $F_X(x) = \frac{1}{4}x^{2/3}$ on $[0,8]$.



The graph of $g(x)$ (black) and the proposal density $f_X(x)$ (red)

Importance Sampling

$$\int_a^b g(x)\ dx = \int_a^b w(x) f_X(x)\ dx$$

$$\int_0^8 \frac{1+e^{-x/2}}{\sqrt[3]{x}}\ dx = \int_0^8 6(1+e^{-x/2}) \frac{1}{6} \frac{1}{\sqrt[3]{x}}\ dx$$

So,

the density $f_X(x) = \dfrac{1}{6\sqrt[3]{x}}$ and the weight function $w(x) = 6(1+e^{x/2})$.

To simulate the X_i we use the probability transform.

$$u = F_x(x) = \frac{1}{4} x^{2/3}. \quad \text{Thus, } x = (4u)^{3/2}.$$

Importance Sampling

For the probability transform in R we enter `u<-runif(250);x<-(4*u)^3/2`. Thus, for 1000 importance sampling approximations, we find

```
> ISg<-rep(0,1000)
> for (i in 1:1000){u<-runif(250);x<-(4*u)^(3/2);ISg[i]<-mean(6*(1+exp(-x/2)))}
> mean(ISg)
[1] 8.132918
> sd(ISg)
[1] 0.1164385
```

Compare this with simple Monte Carlo.

```
> sd(Ig)
[1] 0.4715746
```

The standard deviation is reduced by a factor of ~ 4 and thus, we would need to increase the number of simulations by a factor of ~ 16 for simple Monte Carlo to meet the same standard deviation.