Monte Carlo Integration

Su2016 - Dr. Junvie Pailden

July 18, 2016

Contents

Strong Law of Large Numbers

If $X_1, X_2, \ldots, X_n, \ldots$ are are independent and identically distributed random variables with $E(X_k) = \mu$, $k = 1, 2, 3, \ldots$, then

$$P\left(\lim_{n\to\infty} \frac{\sum_{i=1}^{n} X_k}{n} = \mu\right) = 1$$

Consider an event A whose probabilty is p. Define $X_1, \ldots, X_n$ IID $Bernoulli(p)$ random variables.

A natural estimator (moment and maximum likelihood) of p is based on $\hat{p} = \frac{\sum_{i=1}^{n} X_i}{n}$.

$\hat{p}$ is unbiased, $E(\hat{p}) = p$, and by Strong Law of Large Numbers (SLLN),

$$P\left(\lim_{n\to\infty} \frac{\sum_{i=1}^{n} X_k}{n} = p\right) = 1 \text{ or } E(\hat{p}) = p \text{ with probability 1.}$$

Example 1: Estimation of π

> Suppose you have a disk with radius 1 inscribed in a square with length 2. The experiment simply consists of throwing darts on this figure (square) completely at random. The question is: what is the probability that the dart lies within the circle? How can we use this experiment to estimate π?

To estimate π, let U_1 and U_2 be independent random variables uniformly distributed on the interval $(-1, 1)$, and they may be treated as two sides of a square. Define r.v. X as

$$X = \begin{cases} 1, & \text{if} U_1^2 + U_2^2 \leq 1 \\ 0, & otherwise \end{cases}$$

Thus, $E(X) = P(U_1^2 + U_2^2 \leq 1) = \frac{\pi}{4}$ (Why?)

For a sequence of IID varibles $X_1, X_2, \ldots, X_n \sim Unif(-1, 1)$,

$$\frac{\pi}{4} = E(X) \approx \lim_{n \to \infty} \frac{\sum_{i=1}^{n} X_k}{n}.$$

For a large n, a natural estimator of π is then $4 * \hat{p}$.

```
my_pi <- function(n) {
 u1 <- runif(n,-1,1) # create a sequence of uniformly distributed #'s between -1 to 1
 u2 <- runif(n,-1,1) # same as above
 pi4 <- mean(u1^2 + u2^2 <=1)
 return(4*pi4)
}
n <- c(10,10^3,10^5,10^8)
sapply(n,my_pi)
```

```
## [1] 3.600000 3.244000 3.137560 3.141572
```

Monte Carlo Integration

Let $g(x)$ be a function and suppose that we want to compute

$$\theta = \int_a^b g(x) dx.$$

Recall that if X is a random variable with density $f(x)$, then the mathematical expectation of the random variable $Y = g(X)$ is

$$E[g(X)] = \int_{-\infty}^{\infty} g(x) f(x) dx.$$

If a random sample is available from the distribution of X, an unbiased estimator of $E[g(X)]$ is the sample mean.

Simple Monte Carlo Estimator

Consider estimating

$$\theta = \int_0^1 g(x) dx = \int_0^1 g(x) f(x) dx,$$

where $f(x) = 1$ is the $Unif(0, 1)$ pdf.

If $X_1, \ldots, X_n$ is a random uniform $Unif(0, 1)$ sample then

$$\hat{\theta} = \bar{g}_m(X) = \frac{1}{m} \sum_{i=1}^{m} g(X_i)$$

converges to $E[g(X)] = \theta$ with proby 1, by Strong Law of Large Numbers. The simple Monte Carlo estimator of θ is $\hat{\theta}$.

Example 2: Monte Carlo Estimate of a Definite Integral

Compute a Monte Carlo Estimate of

$$\theta = \int_0^1 e^{-x} dx$$

and compare the results with the exact value.

```
m <- 10000
x <- runif(m)
theta.hat <- mean(exp(-x))
data.frame(theta.hat = theta.hat, theta =  1 - exp(-1))
```

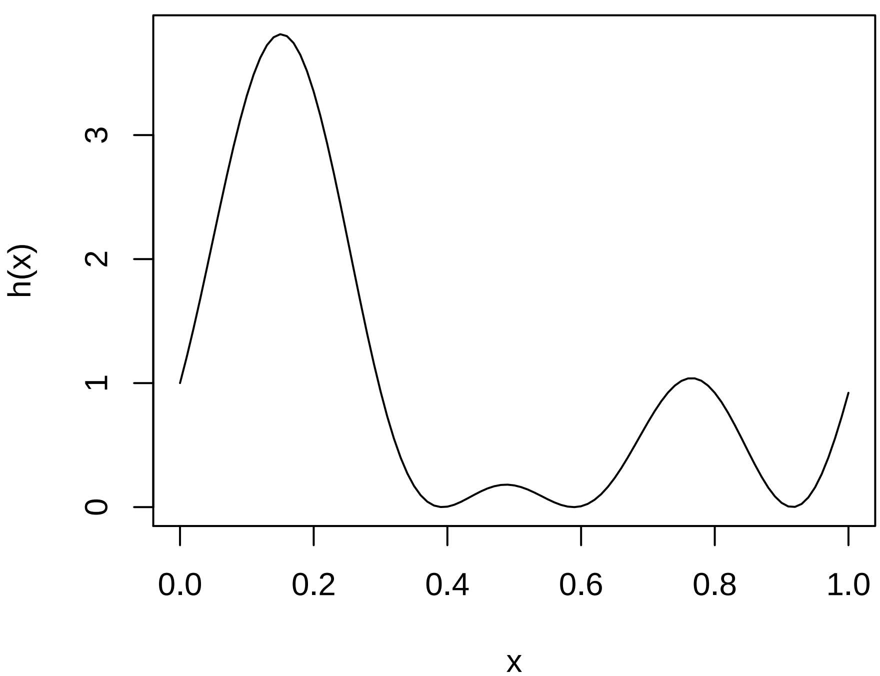
```
##   theta.hat     theta
## 1 0.6340264 0.6321206
```

Example 3: Monte Carlo Integration

Compute a Monte Carlo Estimate of

$$\theta = \int_0^1 (\cos(2x) + \sin(10x))^2 dx.$$

```
h <- function(x){(cos(2*x)+sin(10*x))^2}
curve(h(x))
```




```
m <- 10000
theta.hat = mean(h(runif(m)))
theta = integrate(h,0,1)
data.frame(theta.hat, theta = theta$value)
```

```
##   theta.hat    theta
## 1  1.043532 1.038776
```

What if the limits of integration is $\int_a^b$ and not $\int_0^1$?

To compute

$$\int_a^b g(t)dt,$$

we can replace the $Unif(0,1)$ density with any other density supported on the interval between the limits of intergration. For example,

$$\theta = \int_a^b g(t)dt = (b-a)\int_a^b g(t)\frac{1}{b-a}dt$$

is (b-a) times the expected value of $g(Y)$, where Y has the uniform density on (a,b). Thus,

$$\hat{\theta} = (b-a)\frac{1}{m}\sum_{i=1}^{m} g(X_i).$$

Example 4: Compute a Monte Carlo estimate of

$$\theta = \int_2^4 e^{-x}dx.$$

```r
set.seed(17)
m <- 1000
x <- runif(m,min=2,max=4)
theta.hat <- mean(exp(-x))*2
data.frame(theta.hat = theta.hat, theta=exp(-2)-exp(-4))
```

```
##   theta.hat     theta
## 1  0.119904 0.1170196
```

Algorithm for Simple Monte Carlo Integration (SMCI)

The simple Monte Carlo estimator of the integral $\theta = \int_a^b g(x)dx$ is computed as follows:

1. Generate $X_1, \ldots, X_m$, IID from $Unif(a,b)$.
2. Compute $\hat{g}(X) = \frac{1}{m}\sum_{i=1}^{m} g(X_i)$.
3. Compute $\hat{\theta} = (b-a)\hat{g}(X)$.
4. An estimate for the error

$$Error \approx (b-a)\sqrt{\frac{1}{m}\sum_{i=1}^{m}[\hat{g}^2(X_i) - [\hat{g}(X_i)]^2]},$$

where

$$\hat{g}^2(X) = \frac{1}{m}\sum_{i=1}^{m}[g(X_i)]^2.$$

Every time a Monte Carlo simulation is made using the same sample size it will come up with a slightly different value. Larger values of m will produce more accurate approximations.

Generic Function for SMCI

```
mc_integral = function(fn, n.iter = 10^5, interval){
  # take a sample using 'runif'
  x <- runif(n.iter, interval[1], interval[2])
  # apply the user-defined function
  y <- fn(x)
  # calculate
  theta.hat <- mean(y)*(interval[2] - interval[1])
  error <- sqrt((mean(y^2)-mean(y)^2)/n.iter)*
        (interval[2] - interval[1])
  data.frame(theta.hat,error)
}

fn <- function(x) exp(-x)
set.seed(17)
mc_integral(fn, interval = c(2,4))
```

```
##   theta.hat      error
## 1 0.1168877 0.00020699
```

Example 3 (cont): Another example of SMCI

Compute a Monte Carlo Estimate of

$$h(x) = \int_0^{\pi/2} (\cos(2x) + \sin(10x))^2 dx.$$

```
h <- function(x){(cos(2*x)+sin(10*x))^2}
# curve(h(x),0,pi/2)
set.seed(17)
mc_integral(h, interval = c(0,pi/2))
```

```
##   theta.hat       error
## 1  1.575246 0.005561775
```

```
integrate(h,0,pi/2) # true value
```

```
## 1.570796 with absolute error < 2.4e-13
```

Multiple Integrals

Sometimes we are given integrals which cannot be done analytically, especially in higher dimensions where the standard methods of discretization (trapezoidal, simpsons rule, etc) can become computationally expensive.

For a function of d variables, the multiple integral (assuming it exists) is

$$\theta = \int \cdots \int_V g(x_1, \ldots, x_d) dx_1 \cdots dx_d,$$

with volume

$$V = [a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_d, b_d].$$

Monte Carlo Approximation for Multiple Integrals

The Monte Carlo can be used to approximate the value of multiple integrals θ.

1. Let $\mathbf{X} = (x_1, \ldots x_d)$ be a point in volume V. Pick n randomly distributed points $\mathbf{X_1}, \ldots \mathbf{X_n}$ in volume V.
2. Determine the average value of the function

$$\hat{g} = \frac{1}{n} \sum_{i=1}^{n} g(\mathbf{X_i}).$$

3. Compute the approximation to the integral

$$\int \cdots \int_V g(x_1, \ldots, x_d) dx_1 \cdots dx_d \approx (b_1 - a_1) \cdot (b_2 - a_2) \cdots (b_d - a_d) \times \hat{g}.$$

4. An estimate for the error

$$\text{Error} \approx (b_1 - a_1) \cdot (b_2 - a_2) \cdots (b_d - a_d) \times \sqrt{\frac{1}{m} \sum_{i=1}^{m} [\hat{g}^2(\mathbf{X_i}) - [\hat{g}(\mathbf{X_i})]^2]}.$$

Generic function for Multiple Monte Carlo Integration:

```
mc_mult_integral = function(fn,d,n.iter=10^5,interval){
  x <- matrix(NA,nrow = n.iter, ncol=d)
  # each row is a sample from 'runif(a1,b1',...
  # `runif(a2,b2)`, ... `runif(ad,bd)`
  for(k in 1:d)
    x[,k] <- runif(n.iter,interval[k,1],interval[k,2])
  y <- apply(x,1,fn)
  theta.hat <- prod(interval[,2]-interval[,1])*mean(y)
  # `prod` returns the product of all entries in the vector
  error <- sqrt((mean(y^2)-mean(y)^2)/n.iter)*
        prod(interval[,2]-interval[,1])
  data.frame(theta.hat,error)
}
```

Example 5: Monte Carlo Approximation for Multiple Integrals

Let $f(x,y,z)=\sqrt{4-x^2-y^2-z^2}$. Use the Monte Carlo method to calculate approximations to the integral

$$\int_0^{9/10}\int_0^1\int_0^{11/10}\sqrt{4-x^2-y^2-z^2}dzdydx.$$

```
fn <- function(x) sqrt(4-x[1]^2-x[2]^2-x[3]^2)
set.seed(17)
inter <-  matrix(c(0,0,0,9/10,1,11/10),ncol=2)
mc_mult_integral(fn, d = 3, interval=inter)
```

```
##   theta.hat        error
## 1  1.705861 0.0004912637
```

Example 6: MC Integration, Unbounded Interval

Use the Monte Carlo approach to estimate the standard normal cumulative distribution function (cdf) $\Phi(x)$ given by

$$\Phi(x)=\int_{-\infty}^{x}\frac{1}{\sqrt{2\pi}}e^{-t^2/2}dt=\begin{cases}\int_{-\infty}^{0}\frac{1}{\sqrt{2\pi}}e^{-t^2/2}dt+\int_0^x\frac{1}{\sqrt{2\pi}}e^{-t^2/2}dt, & x\geq 0\\ 1-\left(\int_{-\infty}^{0}\frac{1}{\sqrt{2\pi}}e^{-t^2/2}dt+\int_0^{-x}\frac{1}{\sqrt{2\pi}}e^{-t^2/2}dt\right), & x<0\end{cases}$$
$$=\begin{cases}0.5+\frac{1}{\sqrt{2\pi}}\int_0^x e^{-t^2/2}dt, & x\geq 0\\ 0.5-\frac{1}{\sqrt{2\pi}}\int_0^{-x}e^{-t^2/2}dt, & x<0\end{cases}$$

In estimating $\Phi(x)$, we can then just estimate $\theta(x)=\int_0^x e^{-t^2/2}dt$ We generate IID $Unif(0,x)$ random numbers $u_1,\ldots,u_m$. Thus, for any $x\geq 0$,

$$\Phi(x)\approx 0.5+\hat{\theta}(x)/\sqrt{2\pi}=0.5+\frac{1}{\sqrt{2\pi}}\frac{1}{m}\sum_{i=1}^{m}e^{-u_i^2/2}$$

First we compute,

$$\hat{\theta}(x)=\frac{1}{m}\sum_{i=1}^{m}e^{-u_i^2/2}$$

and deliver $0.5+\frac{1}{\sqrt{2\pi}}\hat{\theta}(x)$.

```
hn <- function(t){exp(-t^2/2)}
Phi.est <- function(x) {
  if (x >= 0) {
      tmp <- as.numeric(mc_integral(hn, interval = c(0,x))[1]) # need only the first cell
      tmp <- 0.5 + tmp/sqrt(2*pi)
    } else {
      tmp <- as.numeric(mc_integral(hn, interval = c(0,-x))[1])
      tmp <- 0.5 - tmp/sqrt(2*pi)
    }
  return(tmp)
```

```r
  }

x <- seq(-3,3, length = 7)
# Using MC Integration to estimate Phi(x) with different values of x
Phi.hat <- sapply(x,Phi.est)
# True Normal CDF using pnorm() in R
Phi.true <- pnorm(x)
data.frame(x,Phi.hat,Phi.true)
```

```
##    x       Phi.hat    Phi.true
## 1 -3 -0.0007424815 0.001349898
## 2 -2  0.0225568972 0.022750132
## 3 -1  0.1585947731 0.158655254
## 4  0  0.5000000000 0.500000000
## 5  1  0.8414074547 0.841344746
## 6  2  0.9774569737 0.977249868
## 7  3  1.0014530405 0.998650102
```

Efficiency and Variance Reduction

In the Monte Carlo approach, we estimate

$$\theta = \int_a^b g(x)dx = (b-a)\int_a^b g(x)\frac{1}{b-a}dx = (b-a)E[g(X)]$$

by

$$\hat{\theta} = \frac{(b-a)}{m}\sum_{i=1}^{m} g(X_i).$$

Thus,

$$E[\hat{\theta}] = \theta$$

and

$$Var[\hat{\theta}] = \frac{(b-a)^2}{m}Var(g(X)).$$

By the Central Limit Theorem, for large m,

$$\hat{\theta} \text{ is approx distrbuted } N\left(\theta, \frac{(b-a)^2}{m}Var(g(X))\right).$$

If $\hat{\theta}_1$ and $\hat{\theta}_2$ are two estimators for θ, then $\hat{\theta}_1$ is more efficient (in statistical sense) than $\hat{\theta}_2$ if

$$\frac{Var(\hat{\theta}_1)}{Var(\hat{\theta}_2)} < 1.$$

If $\hat{\theta}_1$ is more efficient than $\hat{\theta}_2$, then the reduction in variance achieved by using $\hat{\theta}_1$ is

$$100\left(\frac{Var(\hat{\theta}_2) - Var(\hat{\theta}_1)}{Var(\hat{\theta}_2)}\right)$$

Increasing the number of replicates m clearly reduces the variance of the Monte Carlo estimator. However, a large increase in m is needed to get even a small improvement in standard error. See page 127 in our text for more details.

Antithetic Variables

Consider the mean of two identically distributed random variables U_1 and U_2. We have

$$Var\left(\frac{U_1+U_2}{2}\right) = \frac{1}{4}(Var(U_1)+Var(U_2)) + 2Cov(U_1,U_2)$$

$$= \begin{cases} < \frac{1}{4}(Var(U_1)+Var(U_2)) & \text{if } Cov(U_1,U_2) < 0 \\ = \frac{1}{4}(Var(U_1)+Var(U_2)) & \text{if } Cov(U_1,U_2) = 0 \\ > \frac{1}{4}(Var(U_1)+Var(U_2)) & \text{if } CovU_1,U_2) > 0 \end{cases}$$

so variance is reduced if U_1 and U_2 have $Cov(U_1,U_2) \le 0$.

In many situations, a θ estimator is $X_1 = g(U_1,\dots,U_n)$ for some function g; so consider the antithetic estimator $X_2 = g(1-U_1,\dots,1-U_n)$. Combine estimator is $\frac{X_1+X_2}{2}$.

Simplest example:

If $X_1 = U$, then $X_2 = 1-U$, $\frac{X_1+X_2}{2} = \frac{1}{2}$, and $Var\left(\frac{X_1+X_2}{2}\right) = 0$ (perfect negative correlation).

Theorem: If $g(X_1,\dots,X_n)$ is monotone for each variable and $U_1,\dots,U_n$ are IID $Unif(0,1)$ then,

$$Cov(g(U_1,\dots,U_n), g(1-U_1,\dots,1-U_n)) \le 0.$$

Corollary: If $g(X_1,\dots,X_n)$ is monotone for each variable, and X_i's are not IID uniform, but

$$X_i = F_i^{-1}(U_i),\ \ i = 1,\dots,n,$$

then $X(\mathbf{U}) = g(F_1^{-1}(U_1),\dots,F_n^{-1}(U_n))$ is monotone in U_i's, so use

$$Y = \frac{X(\mathbf{U}) + X(1-\mathbf{U})}{2}.$$

Generic R Function: Antithetic Variable

```
AV.theta <- function(FUN,n=10^4,antithetic=TRUE){
    u <- runif(n/2)
    if (!antithetic) {
        v <- runif(n/2)
        x <- FUN(c(u,v))
    } else {
        v <- 1-u
        x <- (FUN(u) + FUN(v))/2
    }
 data.frame(theta.hat = mean(x), theta.var = var(x))
}
```

Example 7: Antithetic Variable

Estimate the integral

$$\theta = \int_0^\infty \ln(1+x^2)e^{-x}dx.$$

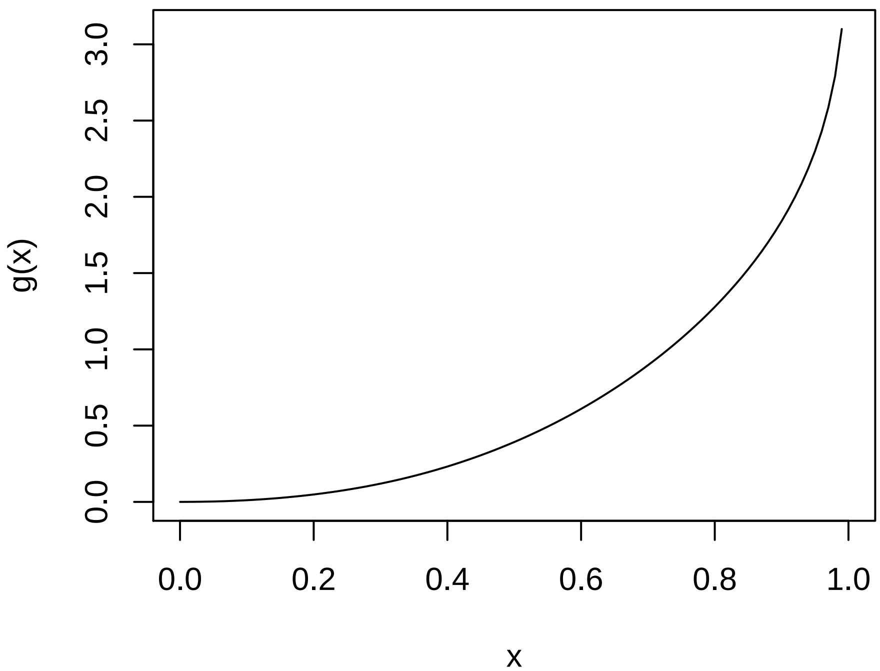
Note the infinite limit of integration so we need to do a change of variables. Let $t = 1 - e^{-x}$, $x = -\ln(1-t)$, $dt = e^{-x}dx$. So,

$$\theta = \int_0^1 \ln[1 + (\ln(1-t))^2]dt.$$

The function g is monotone on (0,1),

$$g(u) = \ln[1 + (\ln(1-u))^2], \; 0 < u < 1.$$

```
g <- function(u) log(1+log(1-u)^2)
curve(g)
```




```
MC1 <- AV.theta(FUN=g,antithetic=FALSE)
MC2 <- AV.theta(FUN=g,antithetic=TRUE)
rbind(MC1,MC2)
```

```
##   theta.hat  theta.var
## 1 0.6903032 0.59835142
## 2 0.6882239 0.09855574
```

```
reduc <- 100*(MC1[2] - MC2 [2])/MC1[2]
# percent reduction in variance when using antithetic var
as.numeric(reduc)
```

```
## [1] 83.52879
```

Control Variates

Another approach to reduce the variance in a Monte Carlo estimator of $\theta = E[g(X)]$ is the use of control variates. Suppose that there is a function f, such that $\mu = E[f(X)]$ is known, and $f(X)$ is correlated with $g(X)$.

Then for any constant c, $\hat{\theta}_c = g(X) + c(f(X) - \mu)$ is an unbiased estimator of θ. The variance

$$Var(\hat{\theta}_c) = Var(g(X)) + c^2 Var(f(X)) + 2cCov(g(X), f(X))$$

is a quadratic function of c.

Thus, $Var(\hat{\theta}_c)$ is minimized at $c = c^*$, where

$$c^* = -\frac{Cov(g(X), f(X))}{Var(f(X))}$$

and

$$Var(\hat{\theta}_c) \geq Var(g(X)) - \frac{[Cov(g(X), f(X))]^2}{Var(f(X))}.$$

The random variable $f(X)$ is called a control variate for the estimator $g(X)$. The percent reduction in variance is

$$100\frac{[Cov(g(X), f(X))]^2}{Var(g(X))Var(f(X))} = 100[Cor(g(X), f(X))]^2$$

In selecting the function g, it is advantageous if $f(X)$ and $g(X)$ are strongly correlated.

Generic R Function: Control Variate

```
CV.theta <- function(f,g, mu.y, n=10^4){
    u <- runif(n)
    x <- g(u)
    y <- f(u)
    c.star <- - cov(x,y)^2/var(x)
    z <- x + c.star*(y-mu.y)
    data.frame(theta.hat = mean(z),
        var.reduc = 100*cor(x,y)^2)
}
```

Example 8: Using Control Variates

Estimate

$$\theta = \int_0^2 e^{-t^2} dt = \int_0^1 2e^{-(2u)^2} du = \int_0^1 g(u) du,$$

where $X = g(U) = 2e^{-(2U)^2}$ and $U \sim Unif(0,1) 0 < u < 1$.

Let's try the control variate $Y = f(U) = 2e^{-2U}$. Note that

$$\mu_Y = E[Y] = E[f(U)] = \int_0^1 2e^{-2u} du = 1 - e^{-2}.$$

Let $\mathbf{U} = (U_1, \ldots, U_n), U_i \sim Unif(0,1)$. The control variate estimator is then

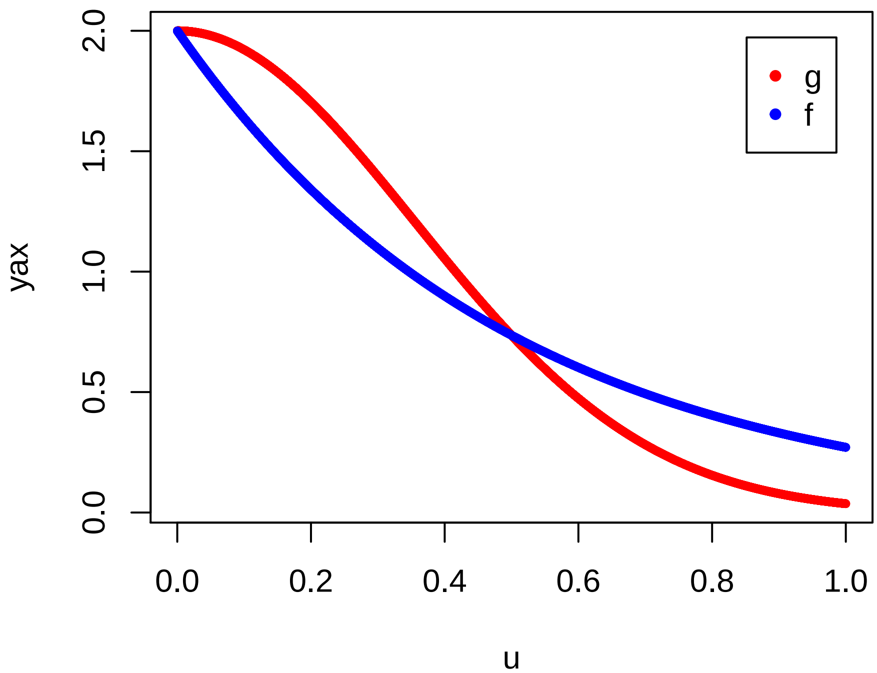
$$\hat{\theta}_c = \bar{g}(\mathbf{U}) + c^*(\bar{f}(\mathbf{U}) - \mu_Y),$$

where

$$c^* = -\frac{Cov(g(\mathbf{U}), f(\mathbf{U}))}{Var(f(\mathbf{U}))}.$$

The percent reduction in variance is $100[Cor(g(\mathbf{U}), f(\mathbf{U}))]^2$.

```r
f <- function(u) 2*exp(-2*u)
g <- function(u) 2*exp(-(2*u)^2) # control variate fn
u <- runif(10^4)
yax <- cbind(g(u),f(u))
matplot(u,yax,pch=20,col=c(2,4),cex=0.7)
legend("topright", inset=0.05,
   legend=c("g","f"), pch=20, col=c(2,4))
```




```r
f <- function(u) 2*exp(-2*u)
g <- function(u) 2*exp(-(2*u)^2) # control variate
mu.y <- 1-exp(-2)  # mean of control variate
set.seed(17)
CV.ex <- CV.theta(f,g,mu.y)
CV.ex
```

```
##   theta.hat var.reduc
## 1 0.8799938  96.49555
```

```r
as.numeric(sqrt(CV.ex[2]/100)) # correlation
```

```
## [1] 0.9823215
```

Antithetic Variate as Control Variate

The control varate estimator is just a linear combination of unbiased estimators of θ. If $\hat{\theta}_1 and \hat{\theta}_2$ are any two unbiased estimators of θ, then for every constant c,

$$\hat{\theta}_c = c\hat{\theta}_1 + (1-c)\hat{\theta}_2 = \hat{\theta}_2 + c(\hat{\theta}_1 - \hat{\theta}_2),$$
$$Var(\hat{\theta}_c) = Var(\hat{\theta}_2) + c^2 Var(\hat{\theta}_1 - \hat{\theta}_2) + 2c \cdot Cov(\hat{\theta}_2, \hat{\theta}_1 - \hat{\theta}_2).$$

and $\hat{\theta}_c$ is also unbiased for θ.

In the antithetic variables, $\hat{\theta}_1 and \hat{\theta}_2$ are identically distributed with $Cor(\hat{\theta}_1, \hat{\theta}_2) = -1$. So, $Cov(\hat{\theta}_1, \hat{\theta}_2) = -Var(\hat{\theta}_1)$,

$$Var(\hat{\theta}_c) = (4c^2 - 4c + 1)Var(\hat{\theta}_1).$$

and the optimal estimator is $c^* = 1/2$. Thus the control variate estimator is

$$\hat{\theta}_{c^*} = \frac{\hat{\theta}_1 + \hat{\theta}_2}{2},$$

which is the same as the antithetic estimator of θ.

Importance Sampling

Suppose X is a random variable wth density function $f(x)$, such that $f(x) > 0$ on the set $\{x : g(x) > 0\}$. Let Y be the random variable $g(X)/f(X)$. Then

$$\int g(x)dx = \int \frac{g(x)}{f(x)} f(x)dx = E[Y].$$

We can estimate $E[Y]$ by simple Monte Carlo integration. That is, compute

$$\frac{1}{m}\sum_{i=1}^{m} Y_i = \frac{1}{m}\sum_{i=1}^{m} \frac{g(X_i)}{f(X_i)},$$

where the r.v.'s $X_1, X_2, \ldots, X_m$ are generated from distirbution with density $f(X)$.

The density $f(X)$ is called the *importance function*.

1. The variance of the estimator based on $Y = g(X)/f(X)$ is $Var(Y)/m$, so the variance of Y should be small.
2. Also, the variance of Y is small if Y is nearly constant, so the density $f(\cdot)$ should be close to $g(x)$.
3. In Naive Monte Carlo approach, estimates in the tails of the distribution are less precise.
4. We expect a more precise estimate for a given sample size if the simulated distribution is not uniform.
5. The average must be a weighted average to correct for this bias.
6. This method is called importance sampling.
7. The advantage of importance sampling is that the importance sampling distribution can be chosen so that variance of the MC estimator is reduced.

Importance Sampling Function

Suppose that $f(x)$ is a density supported on a set $\mathcal{A}$. If $\phi(x) > 0$ on $\mathcal{A}$, then we can write

$$\theta = \int_{\mathcal{A}} g(x)f(x)dx = \int_{\mathcal{A}} g(x)\frac{f(x)}{\phi(x)}\phi(x)dx.$$

If $\phi(x)$ is a density on $\mathcal{A}$, then an estimator of $\theta = E_\phi[g(X)f(X)/\phi(X)]$ is

$$\hat{\theta} = \frac{1}{n}\sum_{i=1}^{n} g(X_i)\frac{f(X_i)}{\phi(X_i)},$$

where $X_1, \ldots, X_n$ is a random sample from density $\phi(x)$.

$\phi(x)$ is called the *importance (envelope) sampling function*.

Typically, one should choose $\phi(x)$ so that

$$\phi(x) \simeq |g(x)|f(x)$$

on $\mathcal{A}$ and has $\phi(x)$ finite variance.

Example 9: Importance Sampling

Consider the function $g(x) = 10\exp(-2|x-5|)$. Suppose we want to calculate $\theta = E(g(X))$, where $X \sim Unif(0,10)$, i.e,

$$\begin{aligned}\theta = E[g(X)] &= \int_0^{10} g(x)f(x)dx \\ &= \int_0^{10} 10\exp(-2|x-5|)\frac{1}{10}dx \\ &= \int_0^{10} \exp(-2|x-5|)dx,\end{aligned}$$

where X has density $f(x) = 1/10$ on (0,10).

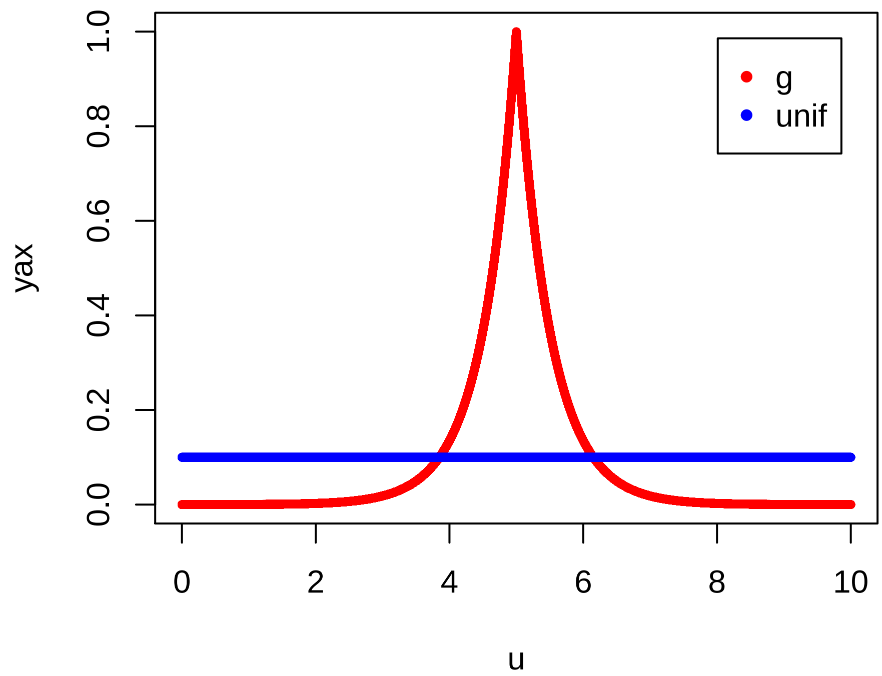
Consider the function $g(x) = \exp(-2|x-5|)$. Suppose we want to calculate $\theta = E(g(U))$, where $U \sim Unif(0,10)$, i.e,

$$\theta = \int_0^{10} \exp(-2|x-5|)dx$$

```
g <- function(u) exp(-2*abs(u-5))
integrate(g,0,10)
```

```
## 0.9999546 with absolute error < 2.8e-14
```

```
unf <- function(u) 1/10
u <- runif(10^5,0,10)
yax <- cbind(g(u),unf(u))
matplot(u,yax,pch=20,col=c(2,4),cex=0.7)
legend("topright", inset=0.05,
   legend=c("g","unif"), pch=20, col=c(2,4))
```



We can estimate θ using Naive MC approach,

```
y <- 10* exp(-2*abs(u-5))
MC1 <- c(mean(y),var(y))
MC1
```

```
## [1] 1.003497 4.029237
```

Note that g peaked at 5, and decays elsewhere, therefore under uniform distribution, many of the points are contributing very little to this expectation.

Now, let's consider the normal distribution with a peak of 5 small variane of 1. We can rewrite

$$\begin{aligned}
\theta = & \quad E[g(X)] = \int_0^{10} g(x)\frac{f(x)}{\phi(x)}\phi(x)dx, \\
&= \int_0^{10} 10\exp(-2|x-5|)\frac{1/10}{\frac{1}{\sqrt{2\pi}}e^{-(x-5)^2/2}}\frac{1}{\sqrt{2\pi}}e^{-(x-5)^2/2}dx \\
&= \int_0^{10} \exp(-2|x-5|)\sqrt{2\pi}e^{-(x-5)^2/2}\frac{1}{\sqrt{2\pi}}e^{-(x-5)^2/2}dx
\end{aligned}$$

where $\phi(x)$ (importance sampling function) is the density of $N(5,1)$. Let $w(x) = \frac{f(x)}{\phi(x)} = \frac{\sqrt{2\pi}e^{-(x-5)^2/2}}{10}$.

```
w <- function(x) dunif(x, 0, 10)/dnorm(x, mean=5, sd=1)
g <- function(x) 10*exp(-2*abs(x-5))
x <- rnorm(2^5,mean=5,sd=1)
y <- w(x)*g(x)
MC2 <- c( mean(y), var(y))
MC2
```

```
## [1] 0.9196219 0.4069826
```

```r
100*(MC1[2]-MC2[2])/MC1[2]
```

```
## [1] 89.89926
```

In this case, there is a 90% reduction in variance when using importance sampling with the given normal pdf as importance sampling function.

```r
u <- runif(10^5,0,10)
g <- function(u) exp(-2*abs(u-5))
phi <- function(u) dnorm(u, mean=5, sd=1)
ynx <- cbind(g(u),phi(u))
matplot(u,ynx,pch=20,col=c(2,4),cex=0.7)
legend("topright", inset=0.05,
   legend=c("g","normal"), pch=20, col=c(2,4))
```