Controlling and Accelerating Convergence

Jonathan Navarrete

May 29, 2017

Topics to be covered:

1. Monitoring Convergence
2. Antithetic Variables

Monitoring Convergence

Monitoring convergence of Monte Carlo samples is important to assessing the quality of estimators. For some MC estimate θ_{MC}, it is possible to run many parallel processes and graphically monitor how they converge, and from those samples obtain a confidence band. However, this may be computationally costly, and resource (e.g. hardware + time) intensive.

An approximate but cheaper version of this basic Monte Carlo estimate of the variability is to bootstrap the originally obtained samples and from there estimate a 95% confidence band.

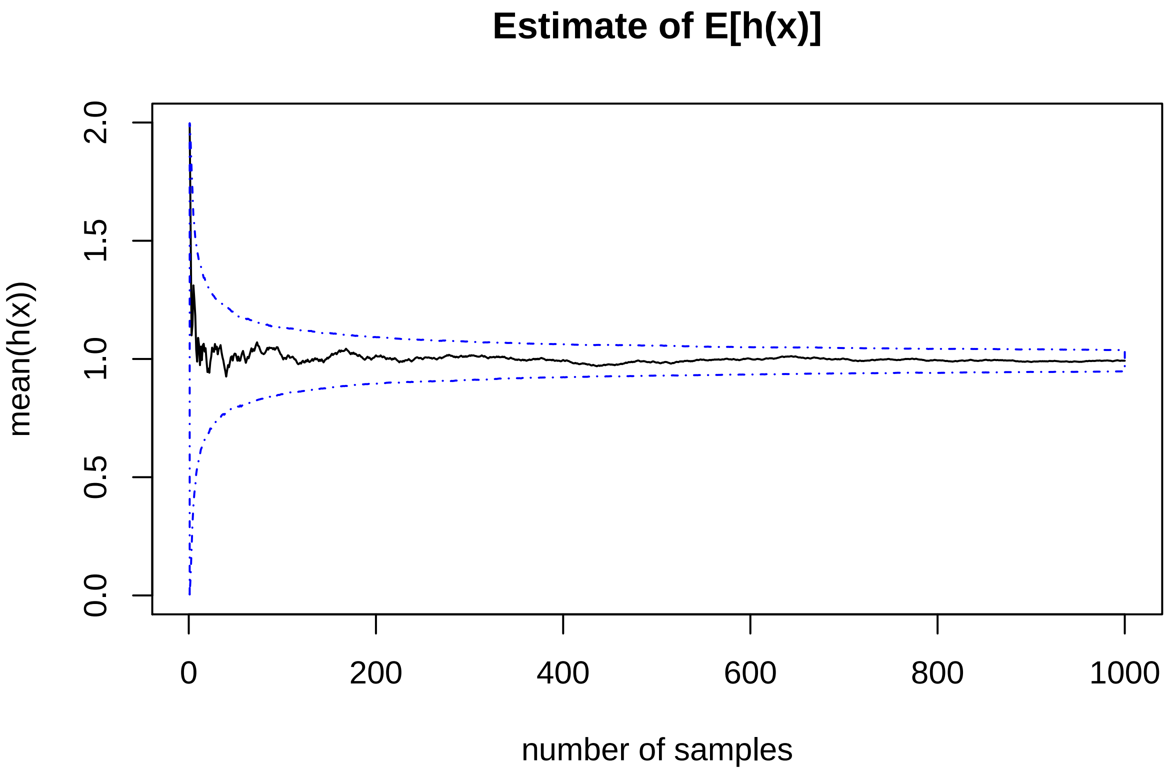
As a toy example, consider the simple function $h(x) = [cos(50x) + sine(50x)]^2$. Using a simple MC algorithm, we can estimate $\theta = \int_0^1 h(x)$. Let us generate n samples $x_1, ..., x_n \sim Unif(0,1)$, such that $\theta = E[h(x)] \approx \frac{1}{n}\sum^n h(x_i)$.

```
set.seed(5692)
n = 1000L
x = runif(n)

h = function(x){
  v = (cos(50*x) + sin(50*x))^2
  return(v)
}


theta_est = mean(h(x))

theta_est = cumsum(h(x))/1:n


## bootstrap
M = 5000L
boot_samples = h(x[sample(x = 1:n, size = n*M, replace = TRUE)])
boot_samples = matrix(data = boot_samples, nrow = n, ncol = M)

boot_est = apply(X = boot_samples, MARGIN = 2, FUN = cumsum) / 1/1:n

CI = t(apply(X = boot_est, MARGIN = 1, FUN = quantile, c(.025,.975)))

summary(CI)
```

```
##       2.5%              97.5%
##  Min.   :0.003814   Min.   :1.038
```

```
##  1st Qu.:0.904812   1st Qu.:1.045
##  Median :0.929640   Median :1.056
##  Mean   :0.906270   Mean   :1.079
##  3rd Qu.:0.940633   3rd Qu.:1.081
##  Max.   :0.947503   Max.   :1.997
```

```r
plot(x = 1:n, y = theta_est, type = "l", ylim  = c(0, 2),
     main = "Estimate of E[h(x)]", ylab = "mean(h(x))", xlab = "number of samples")
polygon(x = c(1:n, rev(1:n)), y = c(CI[,1], rev(CI[,2])), lty = 4, border = "blue")
```



Antithetic Variables

In previous experiments, when we've worked to generate pseudo-random samples from distributions, we've worked with *iid* (independent and identically distributed) peuso-random samples from an instrumental distribution. Generally, *iid* samples are always preferable, but not always cost efficient. As problems become more complicated, generating random samples from a target distribution will become more cumbersome and time/resource consuming. Therefore, in this section we will present methods in which we can double down on our generated samples to speed up convergence and utilize more of our available resources.

The method of antithetic variables is based on the idea that higher efficiency can be obtained through correlation. Given who samples $X = (x_1, ..., X_n)^T$ and $Y = (y_1, ..., y_n)^T$ from the distribution f used in monte carlo integration.

The monte carlo integration estimator

$$\theta = \int_{-\infty}^{\infty} h(x) f(x) dx$$

If X and Y are negatively correlated, then the estimator $\hat{\theta}$ of θ

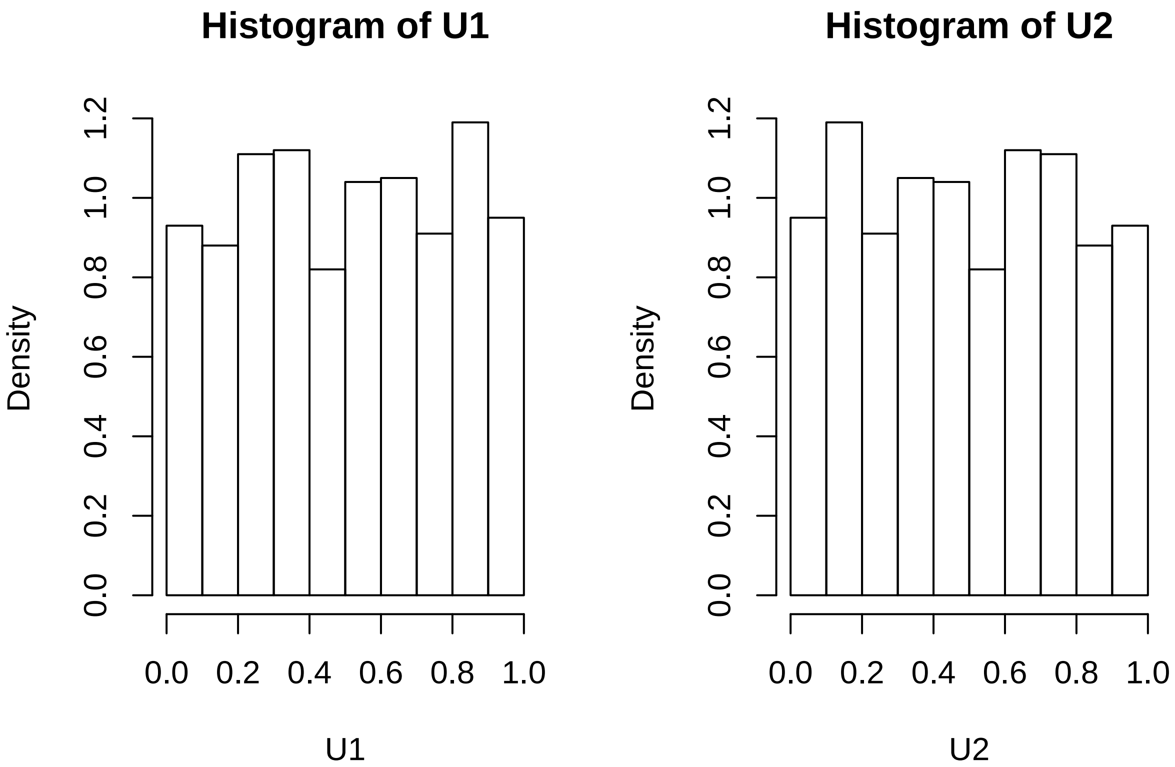
$$\hat{\theta} = \frac{1}{2n} \sum_{i=1}^{n} [h(x_i) + h(y_i)]$$

is more efficient than the estimator $\hat{\theta} = \frac{1}{2n} \sum_{i=1}^{2n} h(x_i)$. The random variables X and Y are then called *antithetic variables*.

Albeit useful, this method is not always possible. For arbitrary transformations $h(.)$, it is not always possible to generate negatively correlations X and Y.

As covered in the introduction, we can generate negatively correlated samples from a uniform distribution.

```
U1 = runif(1000)
U2 = (1 - U1)

par(mfrow = c(1,2))
hist(U1, probability = TRUE)
hist(U2, probability = TRUE)
```




```r
par(mfrow = (c(1,1)))

print(cor(U1, U2))
```

```
## [1] -1
```

Exercise 5.6

Compute $Cov(e^U, e^{1-U})$ and $Var(e^U, e^{1-U})$ where $U \sim Unif(0,1)$. What is the percent reduction in variance of $\hat{\theta}$ that can be acheived using antithetic variates?

Covariance:

$$
\begin{aligned}
Cov(e^U, e^{1-U}) &= E[e^U e^{1-U}] - E[e^U]E[e^{1-U}] \\
&= E[e^1] - (e-1)E[e^{1-U}] \\
&= E[e^1] - (e-1)E[e^1 e^{-U}] \\
&= e - (e-1)e^1[\frac{e-1}{e}] \\
&= e - (e-1)^2 = -0.23421
\end{aligned}
$$

And variance:

$$
\begin{aligned}
Var(e^U + e^{1-U}) &= var(e^u) + var(e^{1-U}) + 2Cov(e^U, e^{1-U}) \\
&= E[e^{2U}] - E[e^U]^2 + E[e^{2-2U}] - E[e^{1-U}]^2 + 2Cov(e^U, e^{1-U}) \\
&= \frac{e^2-1}{2} - (e-1)^2 + E[e^2 e^{-2U}] - E[e^1 e^{-U}]^2 + 2Cov(e^U, e^{1-U}) \\
&= \frac{e^2-1}{2} - (e-1)^2 + \frac{e^2-1}{2} - (e-1)^2 + 2Cov(e^U, e^{1-U}) \\
&= -1 - 2(e-1)^2 + e^2 + 2Cov(e^U, e^{1-U}) \\
&= -1 - 2(e-1)^2 + e^2 + 2(-0.23421) \\
&= 0.0156512
\end{aligned}
$$

Therefore, the variance reduction is approximately 96%

Exercise 5.7

Refer to Exercise 5.6. Use a Monte Carlo simulation to estimate θ by the antithetic variate approach and by the simple Monte Carlo method. Compute an empirical estimate of the percent reduction in variance using the antithetic variate. Compare the result with the theoretical value from Exercise 5.6.

For this example, $g(U) = e^U$. Simple Monte Carlo Method:

```
set.seed(6)
m = 10000
U = runif(m)
g = exp(U)

theta = mean(g) ## theta for simple MC
theta
```

```
## [1] 1.721562
```

```
var_theta1 = var(g)/m
```

```
set.seed(6)
m = 5000
U = runif(m)
T1 = exp(U)
T2 = exp(1-U)

cov(T1, T2)
```

```
## [1] -0.2317587
```

```
c = (1/2)

anti_thetic = c*mean(T1) + (1-c)*mean(T2) ## Antithetic Control Variate
anti_thetic
```

```
## [1] 1.717577
```

```
##variance of theta2
var_theta2 = var(T2)/m + c**2 * var(T1 - T2)/m + 2*c*cov(T2, T1 - T2)/m

(var_theta1 - var_theta2) / var_theta1
```

```
## [1] 0.9678411
```

The true value of θ is 1.718282. The antithetic control variate estimator came closest to the true value, and has a extremely low variance, a 96.78% reduction.

Exercise 5.8

Let $U \sim Uniform(0,1)$, $X = aU$, and $X' = a(1-U)$, where a is a constant. Show that $\rho(X, X')$ = -1.

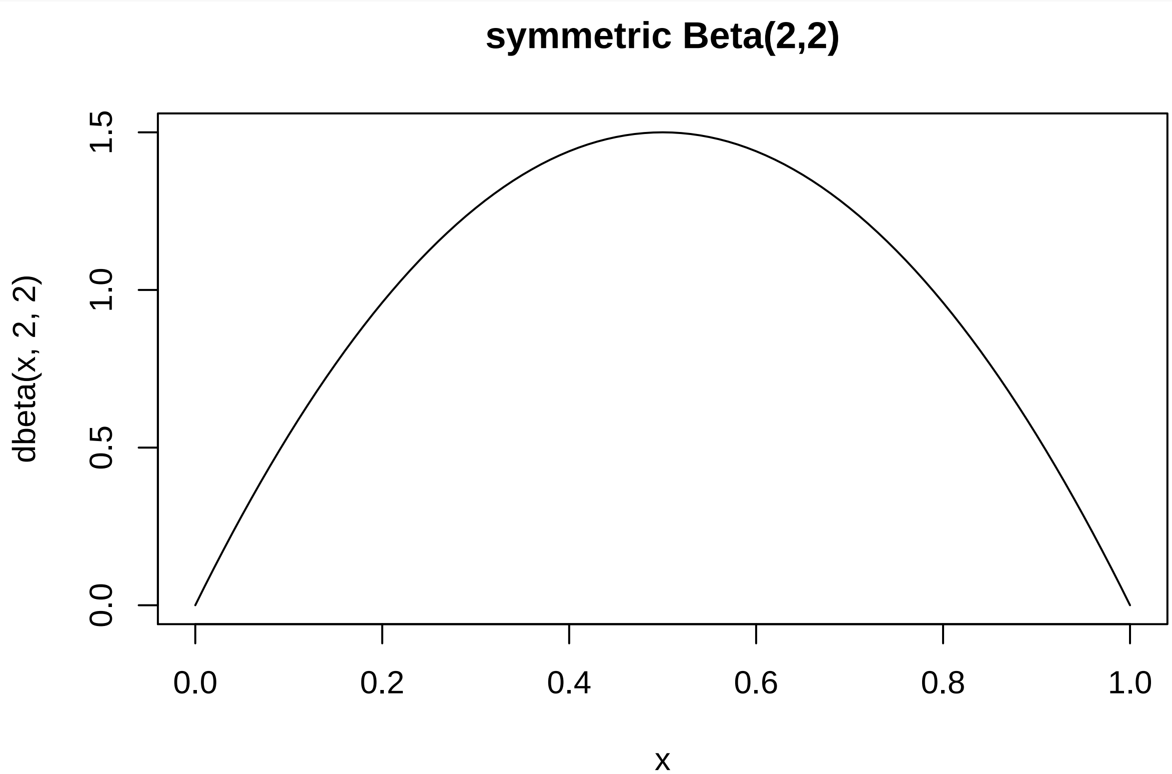
Note that, since $U \sim Unif(0,1)$ then $(1-U)$ is also $Unif(0,1)$ distributed. $E[U] = 1/2$ and $Var(U) = 1/12$

$$\begin{aligned}
Cov(X, X') &= a^2 Cov(U, 1-U) \rightarrow a^2 E[U(1-U)] - E[U]E[1-U] \\
&= a^2 E[U - U^2] - (1/2)(1/2) \rightarrow a^2(E[U] - E[U^2] - 1/4) \\
&= a^2(1/2 - E[U^2] - 1/4) \rightarrow a^2(1/4 - E[U^2]) \\
&= a^2(1/4 - 4/12) \\
&= a^2(-1/12)
\end{aligned}$$

$$\begin{aligned}
\rho &= \frac{Cov(X, X')}{\sqrt{Var(X)}\sqrt{Var(X')}} \\
&= \frac{a^2(-1/12)}{\sqrt{a^2(1/12)}\sqrt{a^2(1/12)}} \\
&\rightarrow \frac{a^2(-1/12)}{a^2(1/12)} = -1
\end{aligned}$$

Is $\rho(X, X')$ = -1 if U is a symmetric beta random variable? Yes, we can show this computationally.

```
curve(expr = dbeta(x, 2, 2), from = 0, to = 1, main = "symmetric Beta(2,2)")
```



```r
U = rbeta(1000, 2, 2)
cor(U, 1-U)
```

```
## [1] -1
```

Example 6.10

Suppose $X \sim N(0,1)$ and we wish to estimate $\theta = E[h(X)]$ where $h(X) = \frac{x}{(2^x - 1)}$. By regular Monte Carlo estimation, we can estimate θ with $n = 10^6$ samples from $N(0,1)$. By antithetic variable estimation we can estimate θ by m = n/2 = 50,000. The antithetic estimator can be constructed using the $X = (x_1, ..., x_m)$, and `c(X, -X)` as our antithetic sample, where X and -X are negatively correlated. $\hat{\theta}_A S = \frac{1}{n} \sum_{i=1}^{m} h(x_i) + h(-x_i)$

```r
n = 10^6
m = n/2

h <- function(x){
    out = x/(2^x - 1)
    return(out)
}

y = rnorm(n)
theta_MC = mean(h(y))

w = rnorm(m)
theta_AS = sum(h(w) + h(-1 * w)) / n

print(theta_MC)
```

```
## [1] 1.499453
```

```r
print(theta_AS)
```

```
## [1] 1.499308
```

```r
## standard errors
rho = cor(h(w),h(-w))
se.a = (1+rho)*var(h(w))/n
```