Suggested answer to exercise 8, chapter 3

from Ross S.: "Simulation" - 5th edition

Author: Nicklas S. Andersen

Date: April 21, 2016

No analytical solution is readily available for the following definite integral:

$$\int_0^1 \int_0^1 \exp\left[(x+y)^2\right] \, dydx$$

In this case we can estimate it by applying Monte Carlo integration. This is easily done, since our target region in $\mathbb{R}^2$ is a square $0 \leq x \leq 1$, $0 \leq y \leq 1$. It is thus sufficent (without any manipolation of the integrand and the integral, to fit the Monte Carlo integration scheme), to generate $(U_{1(1)}, U_{2(1)}), (U_{1(2)}, U_{2(2)}), ..., (U_{1(n)}, U_{2(n)})$ pairs of uniform random numbers $U_1, U_2 \sim U(0,1)$, to estimate the volume of the integral.

We now proceed to write a simple function in R, to compute the above-mentioned definite integral:

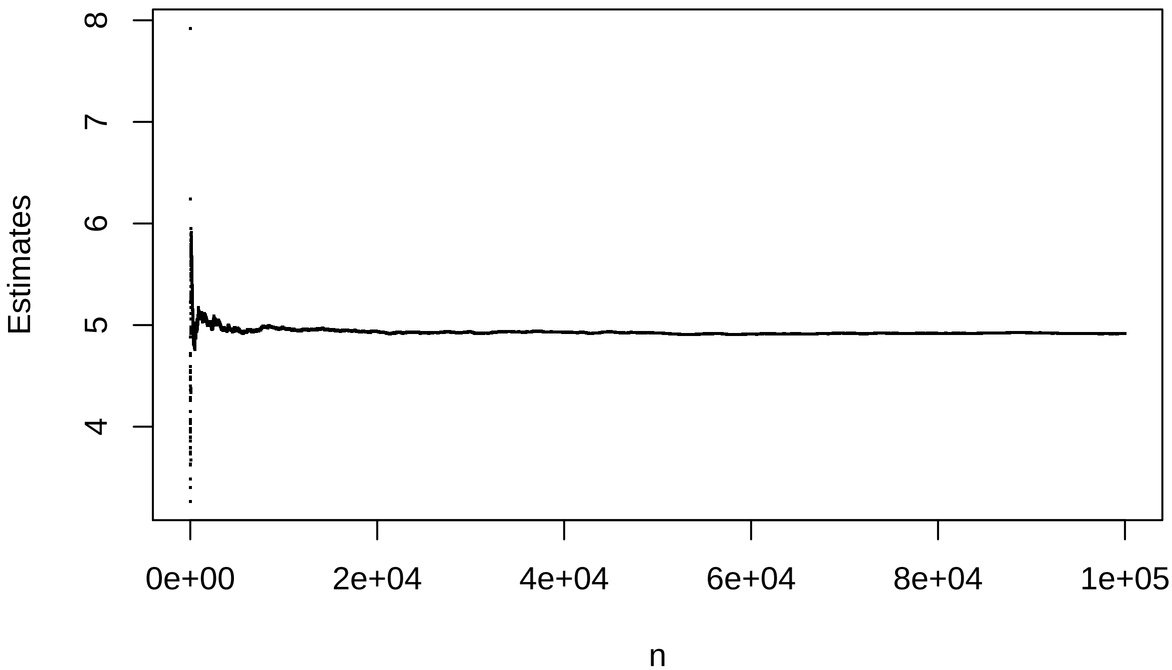
```
#------------------------------------------------------------
# Function: estInt1(n)
#           - (Function to estimate given definite integral)
# Input   : n
#           - (The nr. of paris of (U_1, U_2) to generate)
# Output  : est
#           - (The estimate of the definite integal)
#------------------------------------------------------------
estInt1 <- function(n) {
  storeVal1 <- c()
  for(i in 1:n){
    U1 <- runif(1)
    U2 <- runif(1)
    evalFun <- exp((U1 + U2)^2)         #Evaluate given function.
    storeVal1 <- c(evalFun, storeVal1) #Store function evaluation.
    est <- sum(storeVal1)/n            #Calculate estimate.
  }
  return(est)
}
estInt1(100000)
```

```
## [1] 4.911276
```

Plot of the estimates (law of large numbers)

Additonally, it is also possible to plot the generated data, if we slightly change the above function. By plotting the generated data we can get a good idea of how the law of large numbers works and that it actually applies. As we can see in the plot below and by the estimate above, the estimates converge to some value around 5.

```r
#------------------------------------------------------------
# Function: estInt2(n)
#           - (Function to estimate given definite integral)
# Input   : n
#           - (The nr. of paris of (U_1, U_2) to generate)
# Output  : mat
#           - (A matrix of corresponding pairs of estimates
#              and estimate nr.)
#------------------------------------------------------------
estInt2 <- function(n) {
  storeVal1 <- c()
  storeVal2 <- c()
  storeVal3 <- c()
  for(i in 1:n){
    U1 <- runif(1)
    U2 <- runif(1)
    evalFun <- exp((U1 + U2)^2)        #Evaluate given function.
    storeVal1 <- c(evalFun, storeVal1) #Store function evaluation.
    est <- sum(storeVal1)/i            #Calculate estimate for each i.
    storeVal3 <- c(i, storeVal3)       #Store each i for plotting purposes.
    storeVal2 <- c(est, storeVal2)     #Store each estimate for plotting purposes.
  }
  mat <- matrix(c(storeVal3, storeVal2), nrow=length(storeVal1))
  return(mat)
}
maxSim <- 100000
tmp <- estInt2(maxSim)
plot(tmp, xlim=c(0, maxSim), ylim=c(min(tmp[,2]), max(tmp[,2])),
     xlab="n", ylab="Estimates", pch=".")
```


Estimates
8
7
6
5
4
0e+00
2e+04
4e+04
6e+04
8e+04
1e+05
n