# statComp_hw2

凌浩东

10/12/2021



# 1

# 5.6

$$\begin{aligned}
Cov(e^U, e^{1-U}) &= E(e^U e^{1-U}) - Ee^U Ee^{1-U} = e - (e-1)^2 = -0.2342106 \\
Var(e^U + e^{1-U}) &= Var(e^U) + Var(e^{1-U}) + 2Cov(e^U, e^{1-U}) \\
&= 2Var(e^U) + 2Cov(e^U, e^{1-U}) \\
&= 2(\frac{e^2-1}{2} - (e-1)^2) + 2(e-(e-1)^2) \\
&= 0.01564999
\end{aligned}$$

using Antithetic variate,

$$\hat{\theta} = \frac{2}{m}\left(\sum_{i=1}^{\frac{m}{2}}\left(\frac{1}{2}e^{U_i} + \frac{1}{2}e^{1-U_i}\right)\right) = \frac{1}{m}\sum_{i=1}^{\frac{m}{2}}\left(e^{U_i} + e^{1-U_i}\right)$$

Hence

$$Var(\hat{\theta}) = \frac{1}{m^2}\sum_{i=1}^{\frac{m}{2}} Var(e^{U_i} + e^{1-U_i}) = \frac{1}{2m}Var(e^U + e^{1-U})$$

while

$$Var(\theta) = \frac{1}{m}Var(e^U)$$

Hence, the percent of reduction is

$$\frac{Var(\theta) - Var(\hat{\theta})}{Var(\theta)} = \frac{Var(e^U) - \frac{1}{2}Var(e^U + e^{1-U})}{Var(e^u)} = \frac{0.2420351 - 0.0078250}{0.2420351} \times 100\% = 96.77\%$$

# 5.6

```r
set.seed(42)

MC.myfun <- function(R = 1000, antithetic = T) {
  u <- runif(R/2)
  if (!antithetic) v <- runif(R/2) else
    v <- 1-u
  u <- c(u, v)
  g <- exp(u)
  mean(g)
}

m <- 1000
T1 <- T2 <- numeric(m)
for (i in 1:m) {
  T1[i] <- MC.myfun(antithetic = F)
  T2[i] <- MC.myfun()
}

c(mean(T1), mean(T2))
```

```
## [1] 1.719236 1.718247
```

```r
(var(T1) - var(T2)) / var(T1)
```

```
## [1] 0.9673021
```

The result is consistent with the theoretical value computed in 5.6.

# 5.8

$$
\begin{aligned}
Cov(U, 1-U) &= E[U(1-U)] - EUE(1-U) = \frac{1}{6} - \frac{1}{2} \times \frac{1}{2} = -\frac{1}{12} \\
\rho(aU, a(1-U)) &= \frac{Cov(aU, a(1-U))}{\sqrt{Var(aU)Var(a(1-U))}} \\
&= \frac{a^2 Cov(U, 1-U)}{\sqrt{a^2 Var(U) \times a^2 Var(1-U)}} = \frac{Cov(U, 1-U)}{Var(U)} = -1
\end{aligned}
$$

Suppose $U \sim Beta(\alpha, \alpha)$, thus

$$
\begin{aligned}
EU &= \frac{\alpha}{2\alpha} = \frac{1}{2} \\
EU^2 &= \frac{\alpha+1}{2\alpha+1} EU = \frac{\alpha+1}{2(2\alpha+1)} \\
Var(U) &= \frac{\alpha^2}{(\alpha+\alpha)^2(\alpha+\alpha+1)} = \frac{1}{4(2\alpha+1)} \\
Cov(U, 1-U) &= E[U(1-U)] - EUE(1-U) = \frac{1}{2} - \frac{\alpha+1}{2(2\alpha+1)} - \left(\frac{1}{2}\right)^2 \\
&= \frac{1}{4} - \frac{\alpha+1}{2(2\alpha+1)} \\
&= -\frac{1}{4(2\alpha+1)} \\
\rho(aU, a(1-U)) &= \frac{a^2 Cov(U, 1-U)}{\sqrt{a^2 Var(U) \times a^2 Var(1-U)}} = \frac{Cov(U, 1-U)}{Var(U)} = -1
\end{aligned}
$$

# 5.10

```
set.seed(42)

MC.Myfun <- function(R = 10000, antithetic = T) {
  u <- runif(R/2)
  if (!antithetic) v <- runif(R/2) else
    v <- 1-u
  u <- c(u, v)
  g <- exp(-u) / (1+u^2)
  mean(g)
}

m <- 1000
T1 <- T2 <- numeric(m)
for (i in 1:m) {
  T1[i] <- MC.Myfun(antithetic = F)
  T2[i] <- MC.Myfun()
}

c(mean(T1), mean(T2))
```

```
## [1] 0.5247198 0.5248056
```

```
c(var(T1), var(T2))
```

```
## [1] 6.349395e-06 2.347622e-07
```

```
print((var(T1) - var(T2)) / var(T1))
```

```
## [1] 0.9630261
```

The estimator is 0.5248056 using Monte Carlo integration with antithetic variables. The approximate reduction in variance is 96.30%.

# 2

> Monte Carlo method can be used to approximate the fraction of a d-dimensional hypersphere which lies in the inscribed d-dimensional hypercube. Simulate with different dimensions $d = 2, 3, 4, \cdots, 10$. (Hint: use `apply` function.)

> 1. Derive the formula for the EXACT values for the above problem for each d-dimension.

> 2. Using the above formula, approximate the value of $\pi$. Find the number of points needed to approximate $\pi$ to its 4-th digit for each dimension d. Set the random seed with `set.seed(123)` at the beginning of your R code.

# 1)

$$\begin{aligned} V_n(R) &= \int_{x_1^2+\cdots+x_n^2\le R^2} dx_1 \cdots dx_n \\ &= \int_{x_1^2+\cdots+x_{n-1}^2\le R^2-x_n^2} dx_1 \cdots dx_n \\ &= \int_{-R}^{R} V_{n-1}(\sqrt{R^2 - x_n^2}) dx_n \end{aligned}$$

Denote $x_n$ as $Rsin\varphi$, $\varphi \in [-\frac{\pi}{2}, \frac{\pi}{2}]$. Then

$$V_n(R) = R\int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} V_{n-1}(Rcos\varphi_1)cos\varphi_1 d\varphi_1$$

Likewise, denote $\varphi_2 = Rcos\varphi_1 sin\varphi_2$, we have

$$V_{n-1}(R) = R\int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} V_{n-2}(Rcos\varphi_1 cos\varphi_2)cos^2\varphi_1 cos\varphi_2 d\varphi_1 d\varphi_2$$

$$
\begin{aligned}
V_n(R) &= R\int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} V_{n-1}(Rcos\varphi_1)cos\varphi_1 d\varphi_1 \\
&= R^2\int_{-\frac{\pi}{2}}^{\frac{\pi}{2}}\int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} V_{n-2}(Rcos\varphi_1 cos\varphi_2)cos^2\varphi_1 cos\varphi_2 d\varphi_1 d\varphi_2 \\
&= R^{n-2}\int\cdots\int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} V_2(Rcos\varphi_1\cdots cos\varphi_{n-2})cos^{n-2}\varphi_1\cdots cos\varphi_{n-2}d\varphi_1\cdots d\varphi_{n-2} \\
&= \pi R^n\int\cdots\int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} cos^n\varphi_1\cdots cos^3\varphi_{n-2}d\varphi_1\cdots d\varphi_{n-2}
\end{aligned}
$$

For any constant $\alpha$, denote $cos\varphi$ as $t$

$$
\int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} cos\varphi d\varphi = 2\int_0^1 t\,darccos(t) = \int_0^1 (t^2)^{\frac{\alpha-1}{2}}(1-t^2)^{-\frac{1}{2}}dt^2 = Beta\left(\frac{\alpha+1}{2},\frac{1}{2}\right)
$$

Hence,

$$
\begin{aligned}
V_n(R) &= \pi R^n Beta\left(\frac{n+1}{2},\frac{1}{2}\right)\cdots Beta\left(\frac{3+1}{2},\frac{1}{2}\right) \\
&= \pi R^n\frac{\Gamma(\frac{n+1}{2})\Gamma(\frac{1}{2})}{\Gamma(\frac{n+2}{2})}\cdots\frac{\Gamma(\frac{4}{2})\Gamma(\frac{1}{2})}{\Gamma(\frac{5}{2})} \\
&= \frac{\pi R^n\left(\Gamma\left(\frac{1}{2}\right)\right)^{n-2}\Gamma(2)}{\Gamma(\frac{n}{2}+1)} \\
&= \frac{\pi^{\frac{n}{2}}R^n}{\Gamma(\frac{n}{2}+1)}
\end{aligned}
$$

# 2)

```r
set.seed(123)

MC.ballVolumn <- function(d, m) {
  # generate matrix of uniform random variables
  # each column represent a point in the hyper space
  data <- matrix(runif(d*m), d, m)

  # for each point, check if the distance to the origin
  # if the distance smaller than 1
  # the point is in the hyper sphere
  p <- apply(data, 2, function(x) sum(x^2)<=1)

  # calculate how many points are in the hyper sphere
  fraction <- apply(matrix(p, 1), 1, function(x) {sum(x)/m} )

  fraction
}

for (d in 2:10) {
  mat <- c(1:1000)
  predict <- unlist(lapply(mat, function(x)(2^d * gamma(d/2+1) * MC.ballVolumn(d, x))^(2
        /d)))
  for (j in 1:1000) {
    if (abs(predict[j] - pi) <= 1e-3) {
      print(c(d, j))
      break
    }
  }
}
```

```
## [1]   2 247
## [1]   3 149
## [1]   4 282
## [1]   5 231
## [1]   6 508
## [1]   7 352
## [1]   8 315
## [1]   9 465
## [1]  10 802
```