Monte Carlo Integration

5361 Homework 9

Qinxiao Shi *

15 November 2018

1 Importance Sampling

Suppose X has the following probability density function

$$f(x) = \frac{1}{5\sqrt{2\pi}} x^2 e^{-\frac{(x-2)^2}{2}}, -\infty < x < \infty$$

Consider using the importance sampling method to estimate$\mathbb{E}(X^2)$.

1.1 Result of Estimating

```
estimation <- function(n){
  est.mu <- matrix(NA, nrow = length(n), ncol = 2)
  for (i in 1:length(n)) {
    x <- rnorm(n[i], 0, 1)
    fx <- 1/5*x^2*dnorm(x, mean = 2, sd = 1)
    gx <- dnorm(x, mean = 0, sd = 1)
    hx <- x^2
    weight <- fx/gx
    est.mu[i,1] <- mean(hx*weight)
    est.mu[i,2] <- var(hx*weight)
}
  return(est.mu)
}

n <- c(1000,10000,50000)
est <- estimation(n)

for (i in 1:length(n)) {
  cat <- cat("When n = ", n[i], ", estimated mu is ", est[i,1], " and variance of the estimate
}
```

```
## When n =  1000 , estimated mu is  6.069808  and variance of the estimate is 4674.131
## When n =  10000 , estimated mu is  6.016493  and variance of the estimate is 10760.75
## When n =  50000 , estimated mu is  7.046056  and variance of the estimate is 29766.12
```

*qinxiao.shi@uconn.edu

1.2 Design A Better Importance Sampling Method

If estimating, the importent point is determing a $g(x) \propto f(x)h(x)$, so $g(x) = Ch(x)f(x) = C\frac{1}{5\sqrt{2\pi}}x^4e^{-\frac{(x-2)^2}{2}}$.

Before estimating, it is possible to do a calculation, for $\mathbb{E}(X^2)$.

$$
\begin{aligned}
&\int_{-\infty}^{\infty} h(x)f(x)\ \mathrm{d}x = \int_{-\infty}^{\infty} \frac{1}{5\sqrt{2\pi}}x^4e^{-\frac{(x-2)^2}{2}}\ \mathrm{d}x \\
\overset{t=x-2}{\Longrightarrow} &\int_{-\infty}^{\infty} h(x)f(x)\ \mathrm{d}x = \int_{-\infty}^{\infty} \frac{1}{5\sqrt{2\pi}}(t+2)^4e^{-\frac{t^2}{2}}\ \mathrm{d}t \\
&= \frac{1}{5\sqrt{2\pi}}\Big[\int_{-\infty}^{\infty} t^4e^{-\frac{t^2}{2}}\ \mathrm{d}t + 8\int_{-\infty}^{\infty} t^3e^{-\frac{t^2}{2}}\ \mathrm{d}t \\
&\quad + 32\int_{-\infty}^{\infty} te^{-\frac{t^2}{2}}\ \mathrm{d}t + 16\int_{-\infty}^{\infty} e^{-\frac{t^2}{2}}\ \mathrm{d}t\Big] \\
&= \frac{1}{5\sqrt{2\pi}}\Big[2\int_{0}^{\infty} 2\sqrt{2}u^{\frac{3}{2}}e^{-u}\ \mathrm{d}u + 0 + 48\int_{0}^{\infty} \sqrt{2}u^{\frac{1}{2}}e^{-u}\ \mathrm{d}u + 0 + 32\int_{0}^{\infty} \frac{\sqrt{2}}{2}u^{-\frac{1}{2}}e^{-u}\ \mathrm{d}u\Big] \\
&= \frac{1}{5\sqrt{2\pi}}\Big[4\sqrt{2}\ \Gamma(\frac{5}{2}) + 48\sqrt{2}\ \Gamma(\frac{3}{2}) + 16\sqrt{2}\ \Gamma(\frac{1}{2})\Big] \\
&= 8.6
\end{aligned}
$$

So, $\mathbb{E}(X^2) = 8.6$.

Now, let's implement new method and estimate $\mathbb{E}(X^2)$ to see how does new $g(x)$ do.

```r
new.gx <- function(x){
  gx <- 1/(5*sqrt(2*pi))*x^4*exp(-(x-2)^2/2)
}

C <- 5/43

estimation2 <- function(n){
  est.mu <- matrix(NA, nrow = length(n), ncol = 2)
  for (i in 1:length(n)) {
  xseq <- matrix(0, 1, n[i])
  x <- rnorm(n[i], 2, 1)
  w <- x^4
  xseq[1, ] <- sample(x, size = 1000, replace = TRUE, prob = w)

  w <- 1/xseq^2/mean(1/xseq^2)
  est.mu[i,1] <- mean(xseq^2*w)
  est.mu[i,2] <- sd(xseq^2*w)^2
}
return(est.mu)
}

n <- c(1000,10000,50000)
est2 <- estimation2(n)
```

```
for (i in 1:length(n)) {
  cat <- cat("When n = ", n[i], ", estimated mu is ", est2[i,1], " and variance of the estimate
}
```

```
## When n =  1000 , estimated mu is  8.675525  and variance of the estimate is 4.516801e-31
## When n =  10000 , estimated mu is  8.565667  and variance of the estimate is 4.9861e-31
## When n =  50000 , estimated mu is  8.573985  and variance of the estimate is 5.869243e-31
```

1.3 Comment

Since the new $g(x)$ reduces variance to near 0 and $\hat{\mu}$ is very close to 8.6, which is the real expactation, new $g(x)$ is a better $g(x)$ for importance sampling method.

2 Geometric Brownian motion

2.1 Sample Path of S(t)

Under the Black-Sholes model, $S(t_{j+1}) = S(t_j)(1 + r + \sigma\sqrt{h}Z_{j+1})$, where $t_j = \frac{jT}{m}$, $j = 0, 1, ..., m$, $h = \frac{T}{m}$, $Z \sim N(0, 1)$.

```
s0 <- 1
r <- 0.05
n <- 12

s <- function(bigT, sigma, N){

## Create empty matrix for st path
  s.path <- list()
  dim <- as.numeric(length(bigT)*length(sigma))
  for (i in 1:dim) {
    s.path[[i]] <- matrix(NA, nrow = N, ncol = n+1)
    s.path[[i]][,1] <- s0
  }

## Initial value table
  init <- matrix(NA, nrow = dim, ncol = 2)
  it <- 1
  for (i in 1:length(bigT)) {
    for (j in 1:length(sigma)) {
      init[it,] <- c(bigT[i], sigma[j])
      it <- it + 1
    }
  }

## Fill st path table
```

```
  for (i in 1:dim) {
    for (k in 1:n+1){
      z <- rnorm(N,0,1)
      s.path[[i]][,k] <- s.path[[i]][,k-1]*exp((r - 0.5*init[j,2]^2)*init[j,1]/n +  init[j,2]*s
    }
  }
return(s.path)
}
```

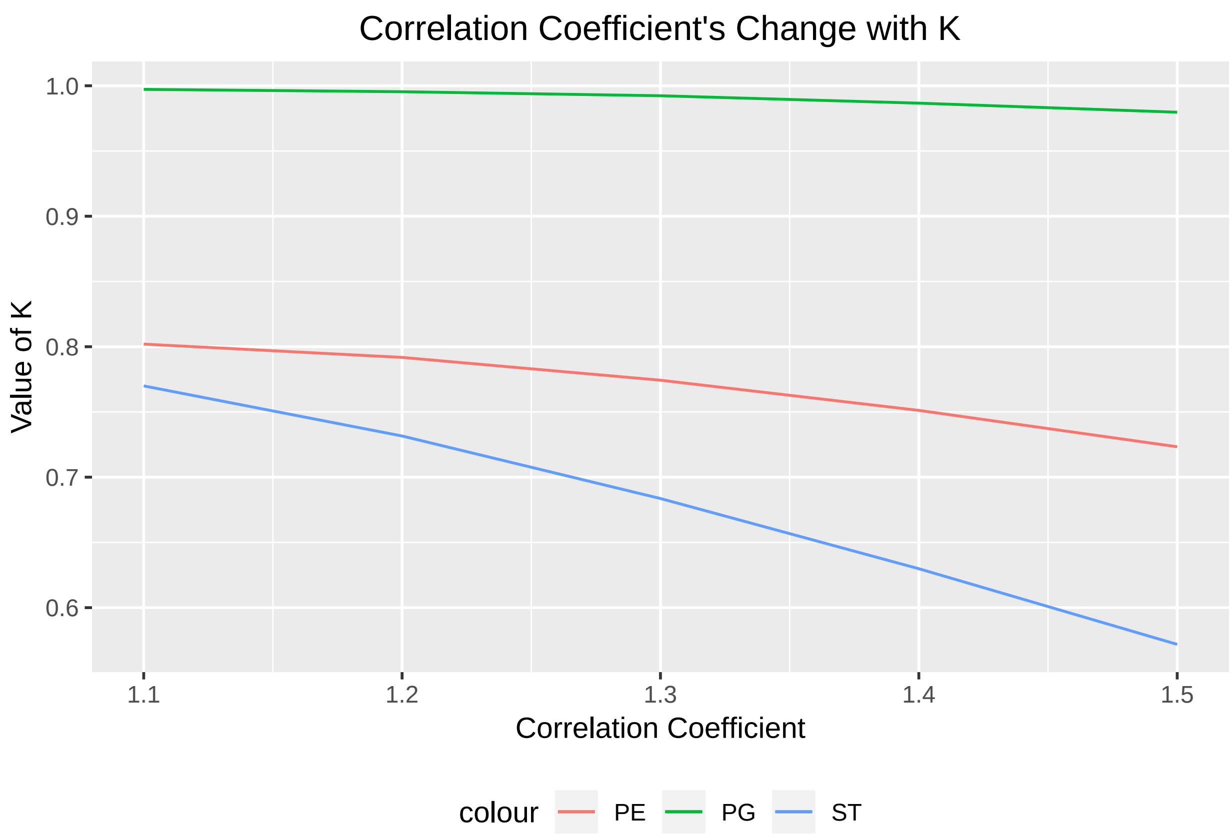
2.2 Correlation Coefficients Changing with K

```
library(ggplot2)
bigT <- 1
sigma <- 0.5
N <- 5000

s.path <- s(bigT, sigma, N)

correlation <- function(K) {
dim <- as.numeric(length(bigT)*length(sigma))
init <- matrix(NA, nrow = dim, ncol = 2)
it <- 1
for (i in 1:length(bigT)) {
  for (j in 1:length(sigma)) {
    init[it,] <- c(bigT[i], sigma[j])
    it <- it + 1
  }
}

SA <- ST <- SG <- matrix(NA, nrow = N, ncol = length(s.path))
for (i in 1:N) {
  for (j in 1:length(s.path)){
    SA[i,j] <- mean(s.path[[j]][i,2:n])
  }
}

for (i in 1:N) {
  for (j in 1:length(s.path)){
    ST[i,j] <- s.path[[j]][i,n+1]
    SA[i,j] <- mean(s.path[[j]][i,2:n])
    SG[i,j] <- prod(s.path[[j]][i,2:n])^(1/n)
  }
}

PA <- PE <- PG <- list()
```

```r
for (i in 1:length(K)) {
  PA[[i]] <- PE[[i]] <-PG[[i]] <- matrix(NA, nrow = N, ncol = length(s.path))
  for (j in 1:N) {
    for (k in 1:length(s.path)){
      PA[[i]][j,k] <- exp(-r*init[k,1])*max(SA[j,k]-K[i], 0)
      PE[[i]][j,k] <- exp(-r*init[k,1])*max(ST[j,k]-K[i], 0)
      PG[[i]][j,k] <- exp(-r*init[k,1])*max(SG[j,k]-K[i], 0)
    }
  }
}

cor.ST <- cor.PE <- cor.PG <- matrix(NA, nrow = length(s.path), ncol = length(K))
for (i in 1:length(K)) {
  for (j in 1:length(s.path)) {
    cor.ST[j,i] <- cor(ST[,j],PA[[i]][,j])
    cor.PE[j,i] <- cor(PE[[i]][,j],PA[[i]][,j])
    cor.PG[j,i] <- cor(PG[[i]][,j],PA[[i]][,j])
  }
}
return(list(cor.ST,cor.PE,cor.PG))
}

K <- c(1.1,1.2,1.3,1.4,1.5)
cor <- correlation(K)

table <- matrix(NA, nrow = 4, ncol = length(cor[[1]]))
table[1,] <- K
for (i in 2:4) {
  table[i,] <- cor[[i-1]]
}
table <- as.data.frame(t(table))

print(table)
```

```
##    V1        V2        V3        V4
## 1 1.1 0.7699762 0.8019960 0.9972404
## 2 1.2 0.7315701 0.7917999 0.9954431
## 3 1.3 0.6836422 0.7743104 0.9923469
## 4 1.4 0.6298483 0.7511871 0.9866625
## 5 1.5 0.5718567 0.7233245 0.9797249
```

```r
ggplot(table) + geom_line(aes(x=table[,1], y=table[,2], color = "ST")) +
  geom_line(aes(x=table[,1], y=table[,3], color = "PE")) +
  geom_line(aes(x=table[,1], y=table[,4], color = "PG")) +
  theme(plot.title = element_text(hjust = 0.5), legend.position = "bottom") +
  labs(x="Correlation Coefficient",y="Value of K", title="Correlation Coefficient's Change wit
```



Hence, when K increasing, the correlation coeffient betweem P_A and $S(T)$, P_A and P_E, P_A and P_G are decreasing.

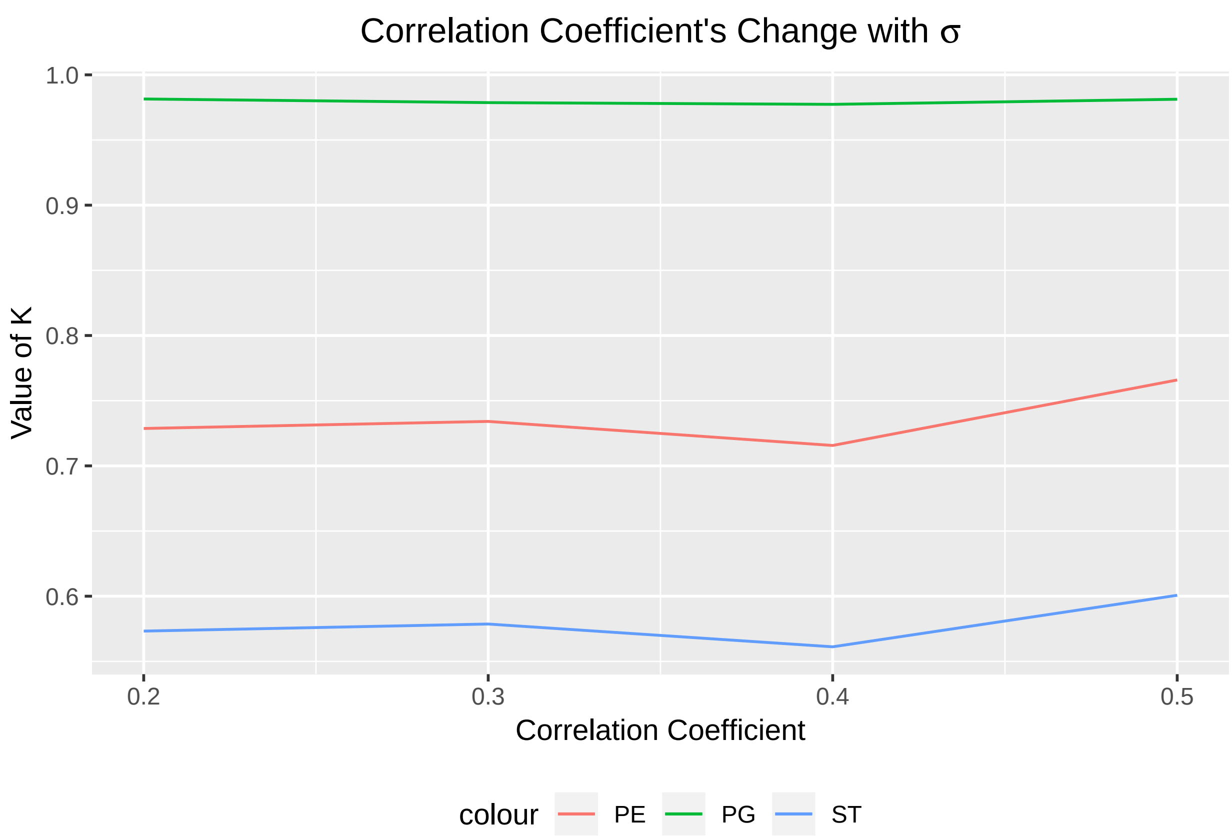
2.3 Correlation Coefficients Changing with σ

```
bigT <- 1
K <- 1.5
sigma <- c(0.2, 0.3, 0.4, 0.5)

s.path <- s(bigT, sigma, N)
cor <- correlation(K)

table <- matrix(NA, nrow = 4, ncol = length(cor[[1]]))
table[1,] <- sigma
for (i in 2:4) {
  table[i,] <- cor[[i-1]]
}

table <- as.data.frame(t(table))
print(table)
```

```
##     V1         V2         V3         V4
```

```
## 1 0.2 0.5732230 0.7286748 0.9814560
## 2 0.3 0.5786488 0.7340958 0.9786735
## 3 0.4 0.5611817 0.7156395 0.9773517
## 4 0.5 0.6007134 0.7658950 0.9812468
```

```
ggplot(table) + geom_line(aes(x=table[,1], y=table[,2], color = "ST")) +
  geom_line(aes(x=table[,1], y=table[,3], color = "PE")) +
  geom_line(aes(x=table[,1], y=table[,4], color = "PG")) +
  theme(plot.title = element_text(hjust = 0.5), legend.position = "bottom") +
  labs(x="Correlation Coefficient",y="Value of K", title=expression(paste("Correlation Coeffic
```



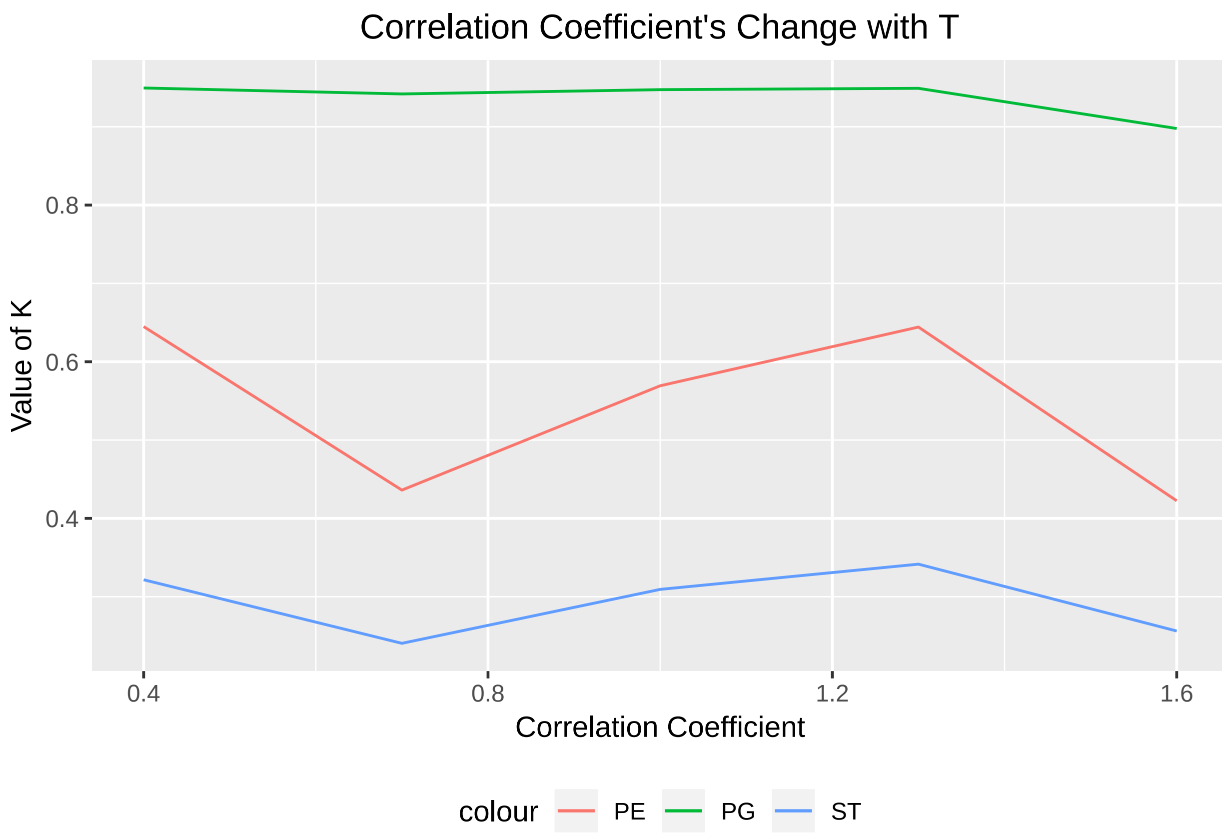
Hence, correlation coeffients do not change with σ increasing.

```
bigT <- c(0.4, 0.7, 1, 1.3, 1.6)
K <- 1.5
sigma <- 0.5

s.path <- s(bigT, sigma, N)
cor <- correlation(K)

table <- matrix(NA, nrow = 4, ncol = length(cor[[1]]))
table[1,] <- bigT
for (i in 2:4) {
```

```r
  table[i,] <- cor[[i-1]]
}

table <- as.data.frame(t(table))
print(table)
```

```
##    V1        V2        V3        V4
## 1 0.4 0.3216157 0.6448733 0.9495278
## 2 0.7 0.2404465 0.4360718 0.9419181
## 3 1.0 0.3093365 0.5692983 0.9473959
## 4 1.3 0.3415210 0.6442613 0.9490912
## 5 1.6 0.2561656 0.4224850 0.8977770
```

```r
ggplot(table) + geom_line(aes(x=table[,1], y=table[,2], color = "ST")) +
  geom_line(aes(x=table[,1], y=table[,3], color = "PE")) +
  geom_line(aes(x=table[,1], y=table[,4], color = "PG")) +
  theme(plot.title = element_text(hjust = 0.5), legend.position = "bottom") +
  labs(x="Correlation Coefficient",y="Value of K", title="Correlation Coefficient's Change wit
```



Hence, correlation coeffients do not change with σ increasing.

2.4 Control Variate

$$\widehat{\mu_{CV}} = \widehat{\mu_{MC}} - b(\widehat{\theta_{MC}} - \theta)$$

where $\widehat{\mathbb{E}(P_A)} = \widehat{\mu_{MC}}$, and $\widehat{\mathbb{E}(P_G)} = \widehat{\theta_{MC}}$. Hence, the goal is find $b^* = \frac{\text{Cov}(\hat{\mu}_{MC}, \hat{\theta}_{MC})}{\text{Var}(\hat{\theta}_{MC}))}$ to minimize $\hat{\theta}_{CV}$. As we known, $\frac{\bar{S}_G}{S(0)} = \left\{ \prod_{i=1}^{n} exp\Big[(r - \frac{1}{2}\sigma^2)t_i + \sigma W(t_i)\Big] \right\}^{\frac{1}{n}} \sim LN((r - \frac{1}{2}\sigma^2)\bar{t}, \bar{\sigma}^2\bar{t})$, where $\bar{t} = \frac{1}{n}\sum_{i=1}^{n} t_i$, $\bar{\sigma}^2 = \frac{\sigma^2}{n^2\bar{t}}\sum_{i=1}^{n}(2i-1)t_{n+1-i}$. So, $\mathbb{E}[(\bar{S}_G - K)^+] = S(0)e^{\mu}\phi(d) - K\phi(d-\sigma)$, where ϕ is distribution function of $N(0,1)$, $d = \frac{1}{\sigma}(\ln\frac{S(0)}{K} + \mu + \sigma^2)$, $\mu = (r - \frac{1}{2}\sigma^2)\bar{t}$, and $\sigma^2 = \bar{\sigma}^2\bar{t}$. Hence, $\mathbb{E}[e^{-rT}(\bar{S}_G - K)^+] = e^{-rT}S(0)e^{\mu}\phi(d) - e^{-rT}K\phi(d-\sigma)$

```
bigT <- 1
K <- 1.5
sigma <- 0.4

s.path <- s(bigT, sigma, N)

dim <- as.numeric(length(bigT)*length(sigma))
init <- matrix(NA, nrow = dim, ncol = 2)
it <- 1
for (i in 1:length(bigT)) {
   for (j in 1:length(sigma)) {
    init[it,] <- c(bigT[i], sigma[j])
    it <- it + 1
  }
}

SA <- ST <- SG <- matrix(NA, nrow = N, ncol = length(s.path))
for (i in 1:N) {
   for (j in 1:length(s.path)){
     SA[i,j] <- mean(s.path[[j]][i,2:n])
  }
}

for (i in 1:N) {
  for (j in 1:length(s.path)){
    ST[i,j] <- s.path[[j]][i,n+1]
    SA[i,j] <- mean(s.path[[j]][i,2:n])
    SG[i,j] <- prod(s.path[[j]][i,2:n])^(1/n)
 }
}

PA <- PE <- PG <- list()
for (i in 1:length(K)) {
  PA[[i]] <- PE[[i]] <-PG[[i]] <- matrix(NA, nrow = N, ncol = length(s.path))
  for (j in 1:N) {
    for (k in 1:length(s.path)){
```

```
      PA[[i]][j,k] <- exp(-r*init[k,1])*max(SA[j,k]-K[i], 0)
      PE[[i]][j,k] <- exp(-r*init[k,1])*max(ST[j,k]-K[i], 0)
      PG[[i]][j,k] <- exp(-r*init[k,1])*max(SG[j,k]-K[i], 0)
    }
  }
}


i <- c(1:n)
t <- i*bigT/n
bar.t <- mean(t)

temp <- array(NA, dim = n)
for (i in 1:n) {
  temp[i] <- (2*i-1)*t[n+1-i]
}

sigmasqr.t <- sigma^2/n^2/bar.t*sum(temp)

mu <- (r-0.5*sigma^2)*bar.t
sigmasqr <- sigmasqr.t*bar.t
d <- 1/sqrt(sigmasqr)*(log(s0/K) + mu + sigmasqr)

theta <- exp(-r*bigT)*(s0*exp(mu + 0.5*sigmasqr)*pnorm(d, 0, 1) - K*pnorm(d-sqrt(sigmasqr), 0,

b.star <- as.numeric(cov(PA[[1]], PG[[1]])/var(PG[[1]]))

mu.CV <- mean(PA[[1]]) - b.star*(mean(PG[[1]]) - theta)

SD.MC <- as.numeric(sqrt(var(PA[[1]])))
SD.CV <- as.numeric(sqrt(var(PA[[1]]) + b.star^2*var(PG[[1]]) - 2*b.star*cov(PA[[1]], PG[[1]])
```

The control variate MC estimator for $\mathbb{E}(P_A)$ using P_G as a control variate is

```
print(mu.CV)
```

```
## [1] 0.01261601
```

The variance of the MC estimator for $\mathbb{E}(P_A)$ that has no control variate is

```
print(SD.MC)
```

```
## [1] 0.05014949
```

The variance of the MC estimator for $\mathbb{E}(P_A)$ that has control variate is

```r
print(SD.CV)
```

```
## [1] 0.01296818
```

P_G, as a control variate, has decreased variance of $\mathbb{E}(P_A)$.