MGMTMFE405 - Project 3

Yanxiang Zhao

January 31, 2019

Question 1

```
# run simulations
x2 <- q1_x(seed1, 1, c(0,2), 1000, 0.001)
y2 <- q1_y(seed1, 3/4, c(0,2), 1000, 0.001)
y3 <- q1_y(seed1, 3/4, c(0,3), 1000, 0.001)
# outputs
prob <- length(y2[y2>5])/length(y2)
E1 <- mean(cbrt(x2))
E2 <- mean(y3)
E3 <- mean((x2*y2)*(x2>1))
# print
cat("prob = ", prob, "\n E1 = ", E1, "\n E2 = ", E2, "\n E3 = ", E3, "\n")
```

```
## prob =  0.976
##  E1 =  0.6415846
##  E2 =  26.03174
##  E3 =  4.196048
```

Question 2

```
# outputs
E1 <- mean((1+q2_x(seed1, seed2, 1, c(0,3), 1000, 0.01))^(1/3))
E2 <- mean((1+exp(-0.08*3+1/3*sqrt(3)*rnorm_pmm(seed1, 10^3) + 3/4*sqrt(3)*rnorm_pmm(seed2, 10^5)))^(1/3
cat("Question 2: \n The expected values are E1 = ",E1, ", E2 = ", E2)
```

```
## Question 2:
##  The expected values are E1 =  1.341573 , E2 =  1.344235
```

Question 3

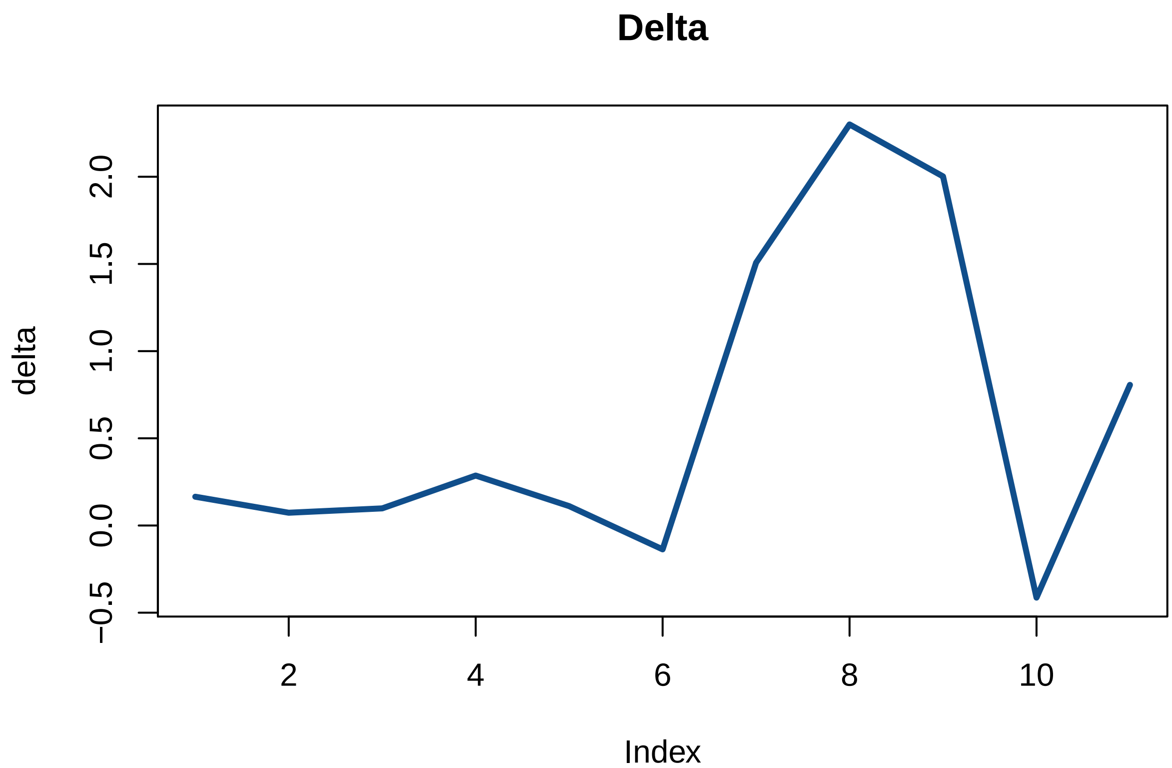
```
# (a): compute call option price with monte carlo simulation
# set parameters
s0 <- 20
t <- 0.5
x <- 20
r <- 0.04
sigma <- 0.25
# output: monte carlo price
C1 <- callprice(seed1,s0,t,x,r,sigma)
cat("Given the variables:\n", "S0 = 20\n T = 0.5\n X = 20\n r = 0.04\n sigma = 0.25\n The call option pr
```

```
## Given the variables:
##  S0 = 20
##  T = 0.5
##  X = 20
##  r = 0.04
##  sigma = 0.25
##  The call option price computed via Monte Carlo simulation =  1.602721
```

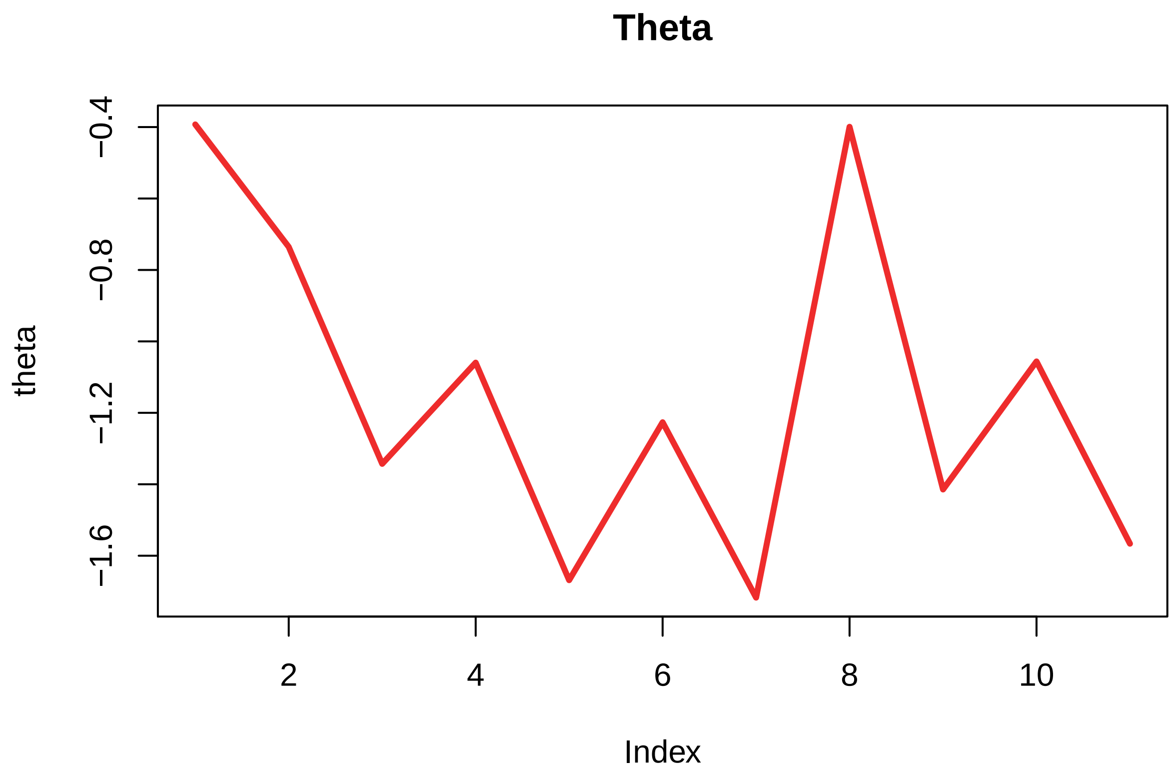
```r
# (b): compute call option price with black-scholes
C2 <- callprice_bs(s0,t,x,r,sigma)
cat("With the same condition, the call option price computed via Black-Scholes Model = ", C1)
```

```
## With the same condition, the call option price computed via Black-Scholes Model =  1.602721
```

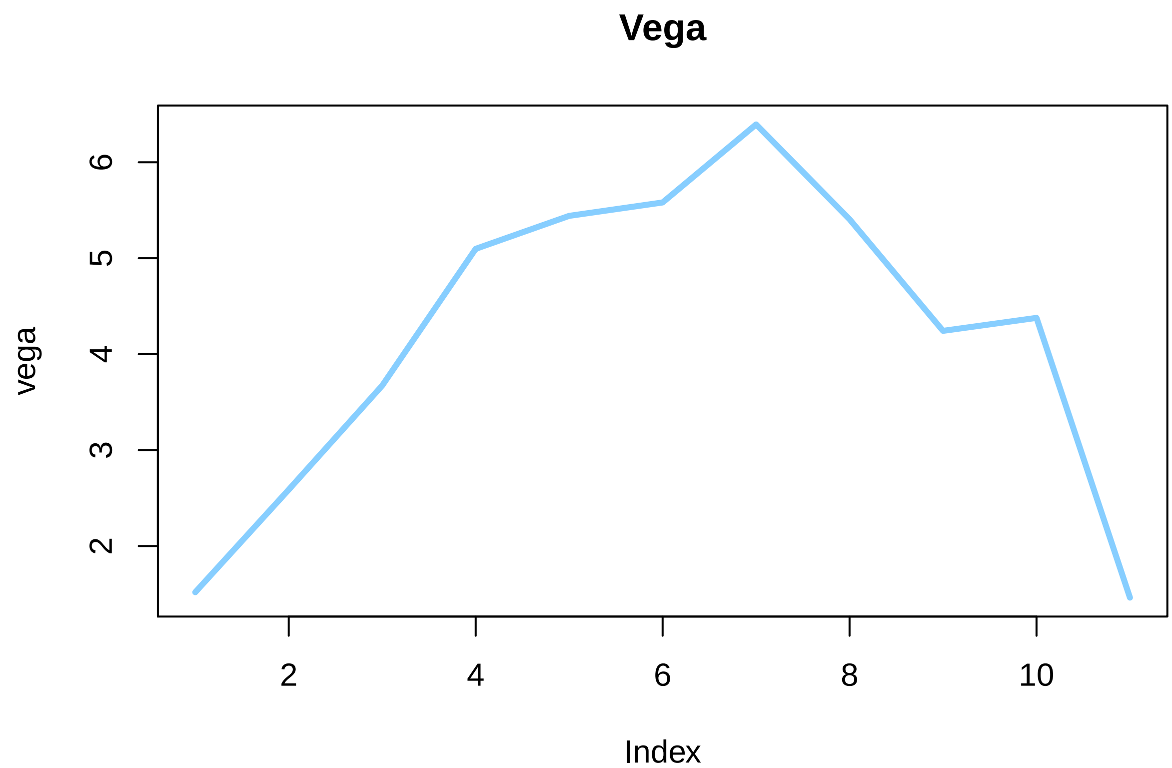
```r
# (c) greeks
# set initial prices
s0 <- 15:25
# outputs
delta <- rep(0,11)
theta <- rep(0,11)
vega  <- rep(0,11)
rho   <- rep(0,11)
gamma <- rep(0,11)
# plots from simulation
for (i in 1:11){
  del <- 0.001
  cprice <- callprice(seed3, s0[i], t, x, r, sigma)
  delta[i] <- (callprice(seed3, s0[i]+del, t, x, r, sigma)-cprice)/del
  theta[i] <- (callprice(seed3, s0[i], t-del, x, r, sigma)-cprice)/del
  vega[i]  <- (callprice(seed3, s0[i], t, x, r, sigma+del)-cprice)/del
  rho[i]   <- (callprice(seed3, s0[i], t, x, r+del, sigma)-cprice)/del
  gamma[i] <- (callprice(seed3, s0[i]+del, t, x, r, sigma)-2*cprice
                +callprice(seed3, s0[i]-del, t, x, r, sigma))/(del)^2
}
# plots
plot(delta, type = "l", lwd = 3, col = "dodgerblue4", main = "Delta")
```



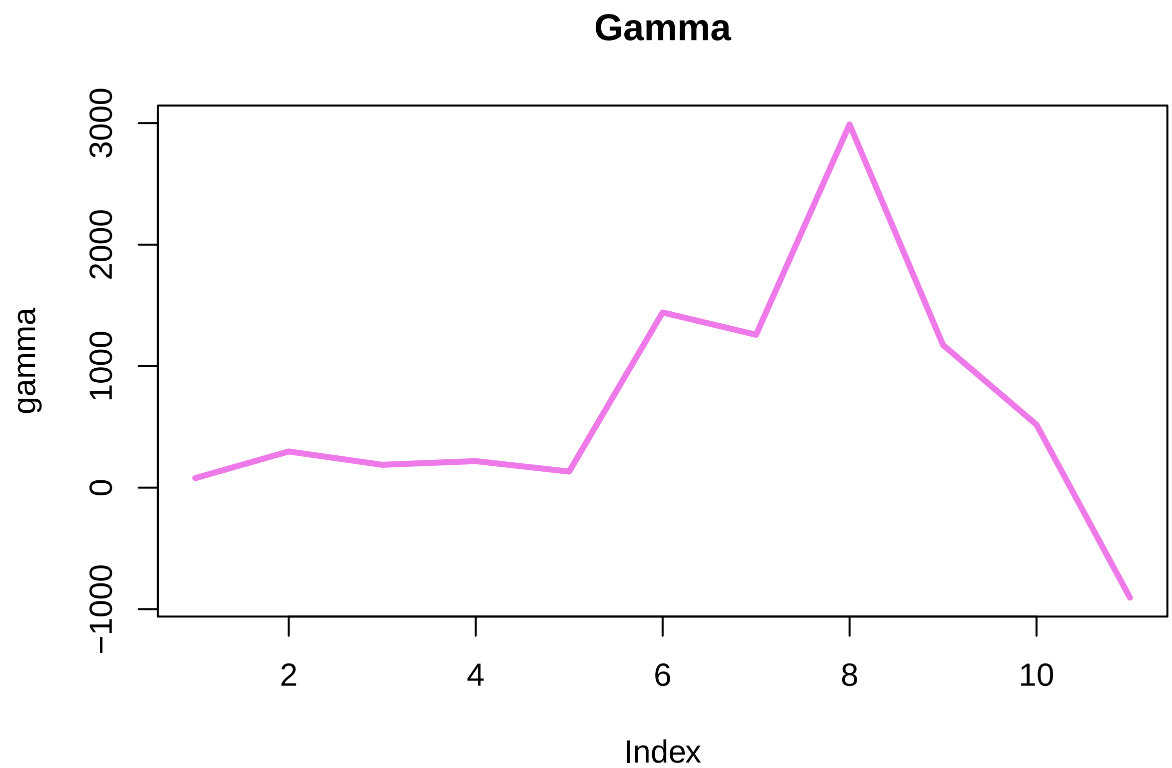

```
plot(theta, type = "l", lwd = 3, col = "firebrick2", main = "Theta")
```



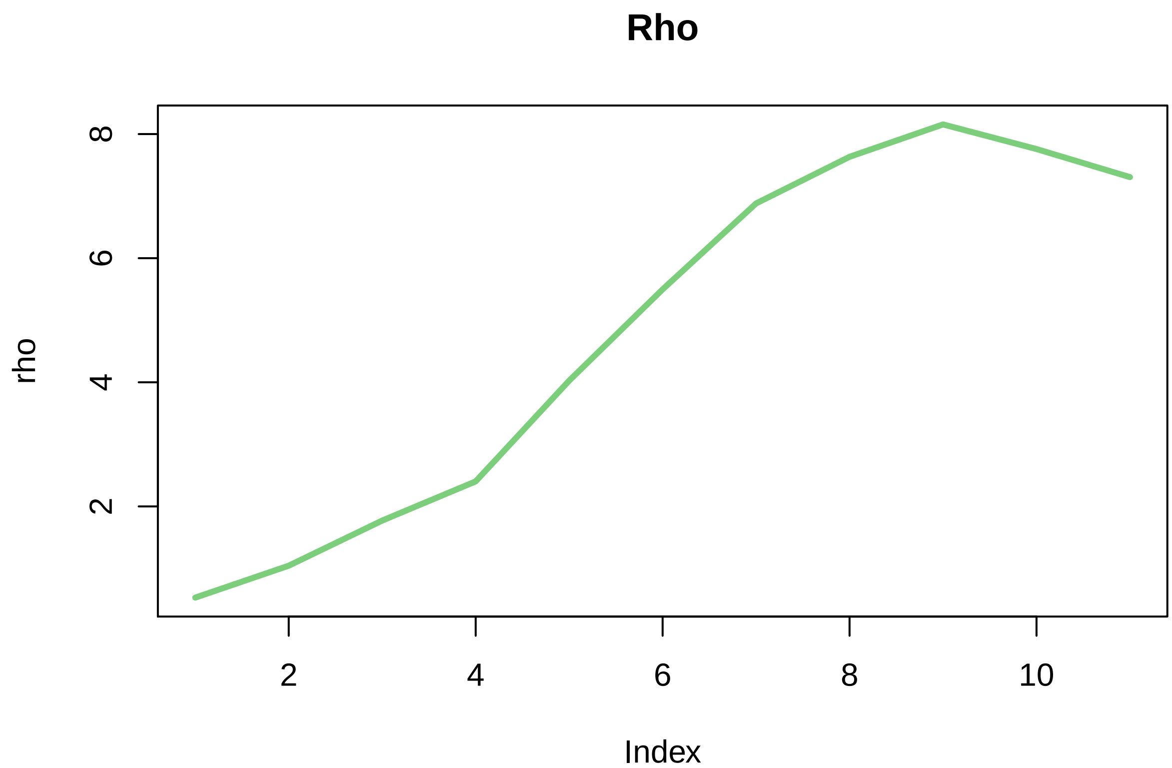

```r
plot(vega, type = "l", lwd = 3, col = "skyblue1", main = "Vega")
```



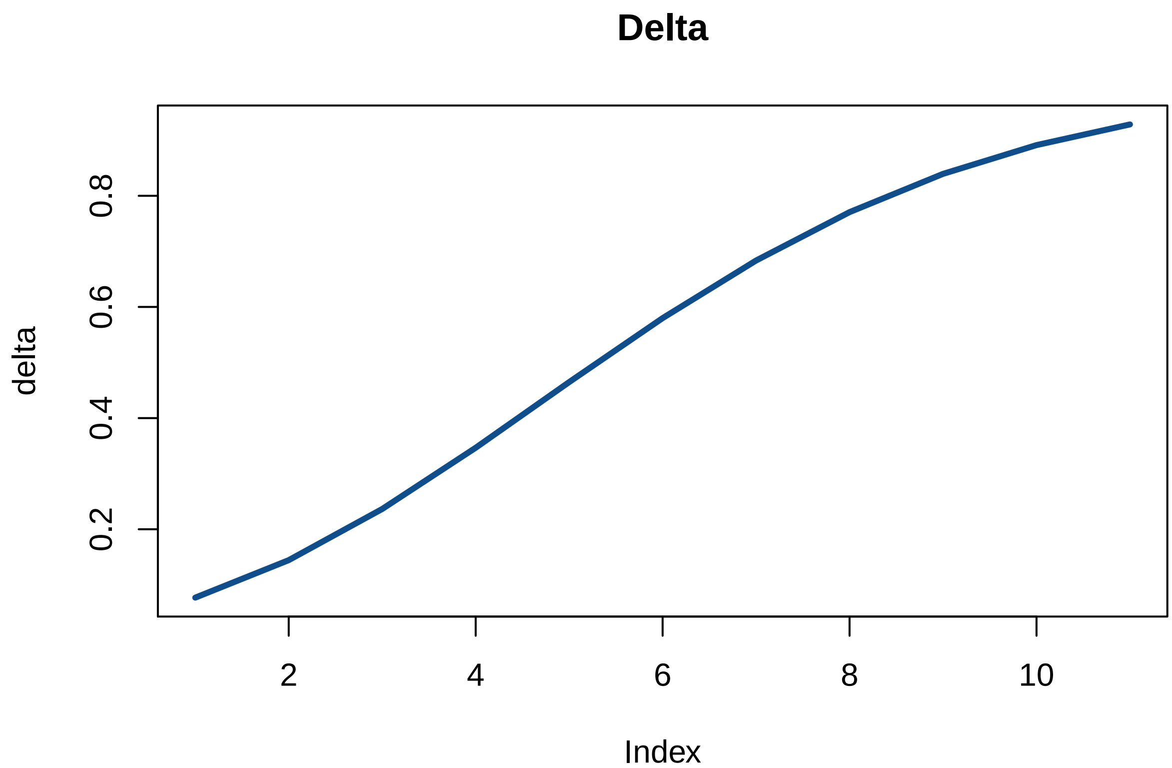

```r
plot(gamma, type = "l", lwd = 3, col = "orchid2", main = "Gamma")
```



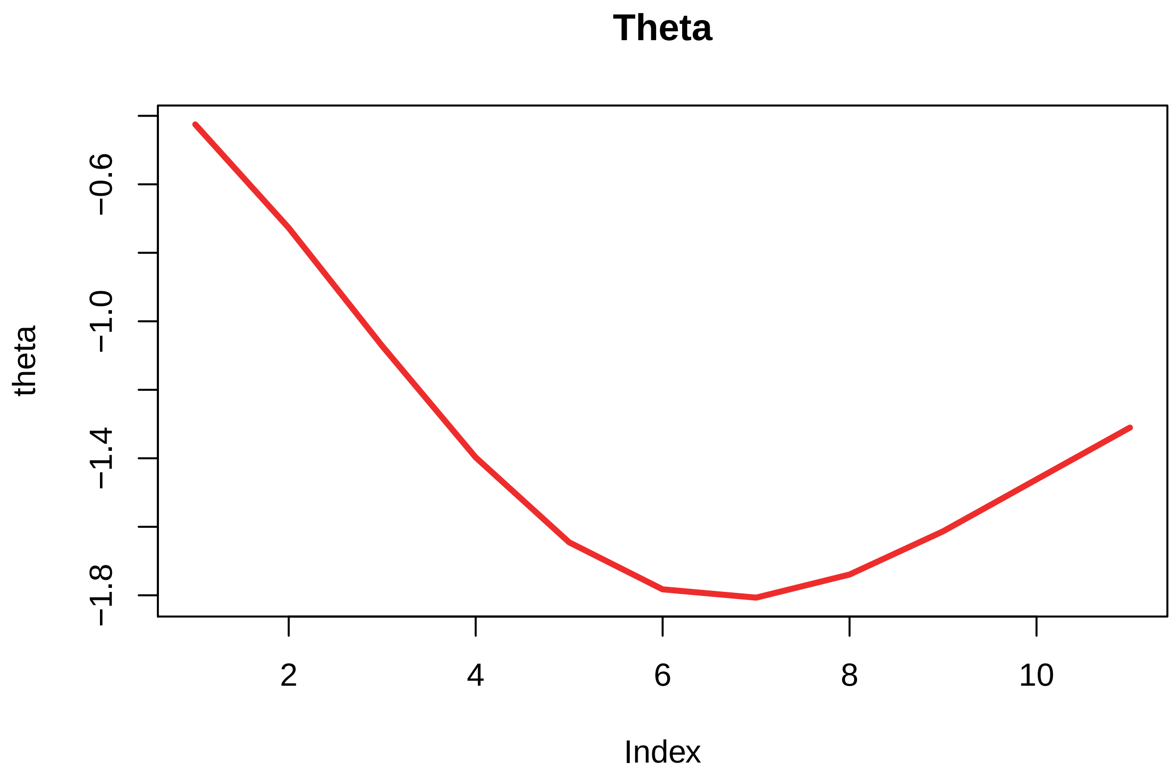

```
plot(rho, type = "l", lwd = 3, col = "palegreen3", main = "Rho")
```



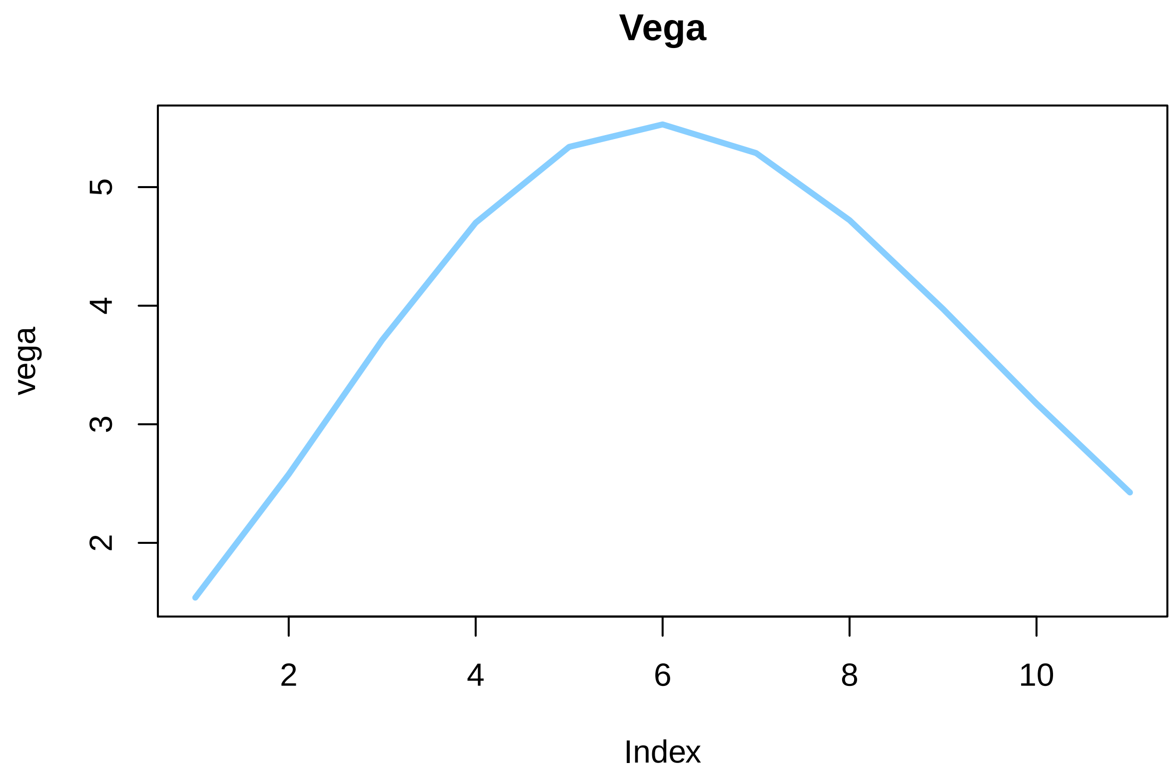

```r
# plots from bs model
for (i in 1:11){
  del <- 0.001
  cprice <- callprice_bs(s0[i], t, x, r, sigma)
  delta[i] <- (callprice_bs(s0[i]+del, t, x, r, sigma)-cprice)/del
  theta[i] <- (callprice_bs(s0[i], t-del, x, r, sigma)-cprice)/del
  vega[i]  <- (callprice_bs(s0[i], t, x, r, sigma+del)-cprice)/del
  rho[i]   <- (callprice_bs(s0[i], t, x, r+del, sigma)-cprice)/del
  gamma[i] <- (callprice_bs(s0[i]+del, t, x, r, sigma)-2*cprice
                +callprice_bs(s0[i]-del, t, x, r, sigma))/(del)^2
}
# plots
plot(delta, type = "l", lwd = 3, col = "dodgerblue4", main = "Delta")
```



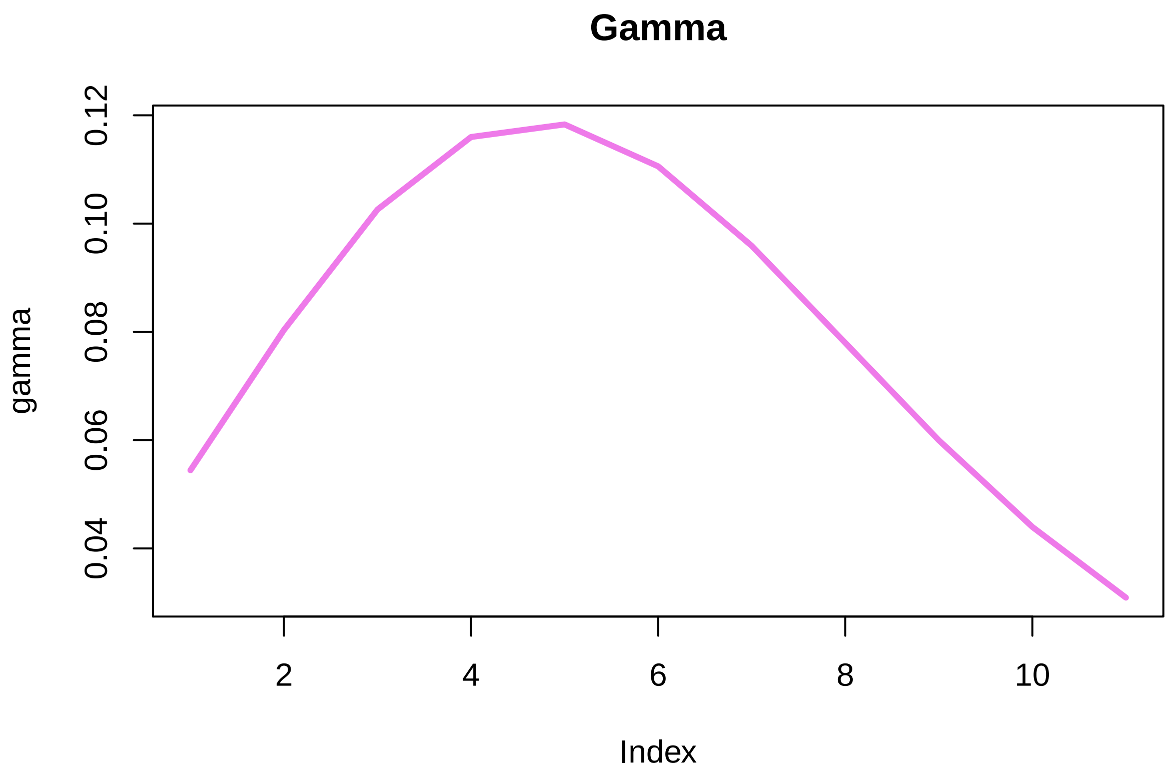

```r
plot(theta, type = "l", lwd = 3, col = "firebrick2", main = "Theta")
```



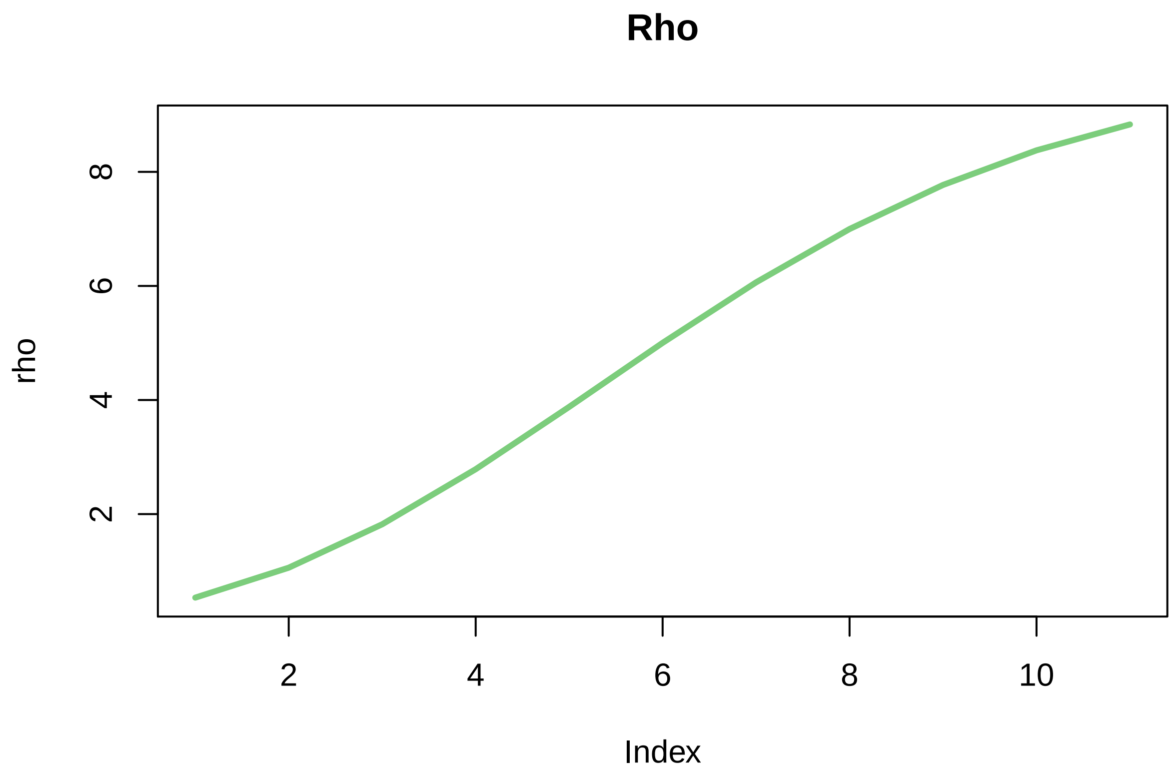

```r
plot(vega, type = "l", lwd = 3, col = "skyblue1", main = "Vega")
```




```r
plot(gamma, type = "l", lwd = 3, col = "orchid2", main = "Gamma")
```




```r
plot(rho, type = "l", lwd = 3, col = "palegreen3", main = "Rho")
```



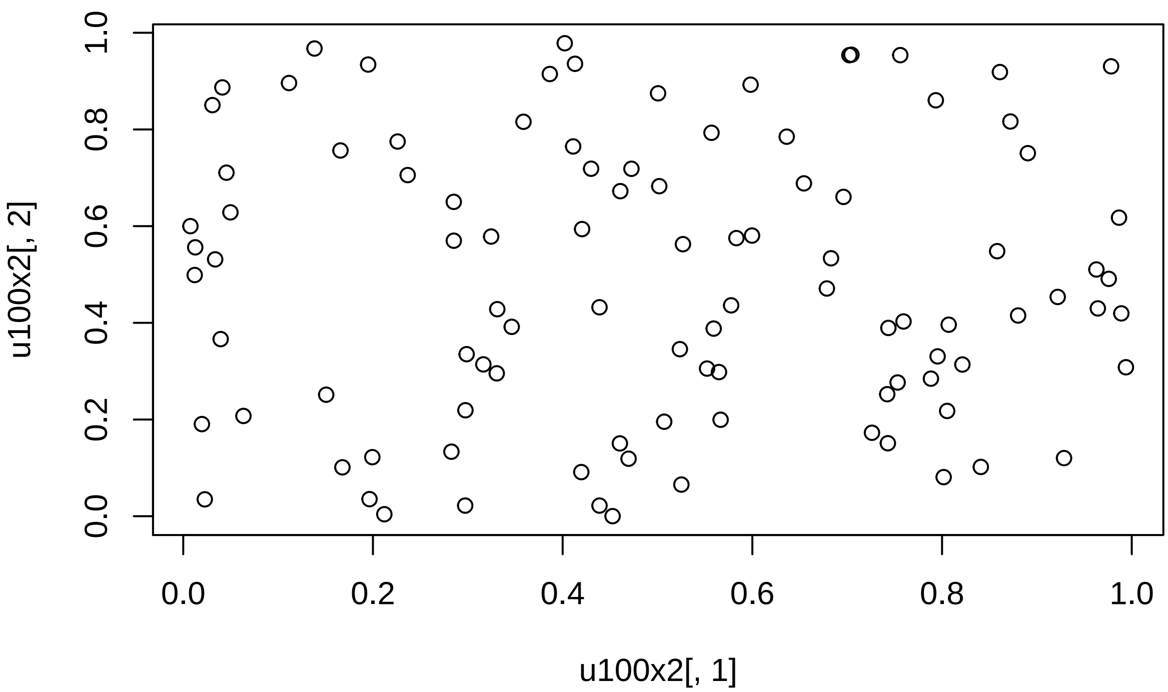
Question 4

```
# set parameters
rho   <- -0.6
r     <- 0.03
s0    <- 48
v0    <- 0.05
sigma <- 0.42
alpha <- 5.8
beta  <- 0.0625
t     <- 0.5
x     <- 50
# outputs
outQ4 <- q4(seed1, seed2)
C1 <- mean(ifelse(outQ4$ft-x>0, outQ4$ft-x, 0))
C2 <- mean(ifelse(outQ4$pt-x>0, outQ4$pt-x, 0))
C3 <- mean(ifelse(outQ4$re-x>0, outQ4$re-x, 0))
cbind(full_trunc = C1, partial_trunc = C2, reflection = C3)
```

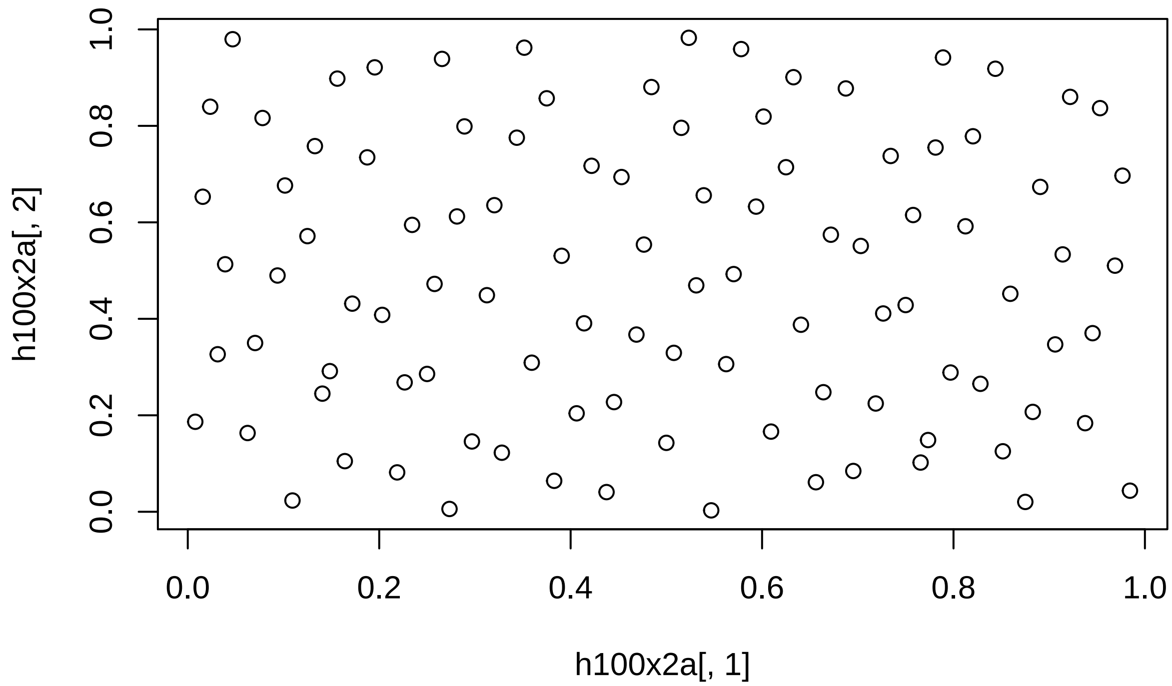
```
##      full_trunc partial_trunc reflection
## [1,]   2.573353      2.573353   2.573353
```

Question 5

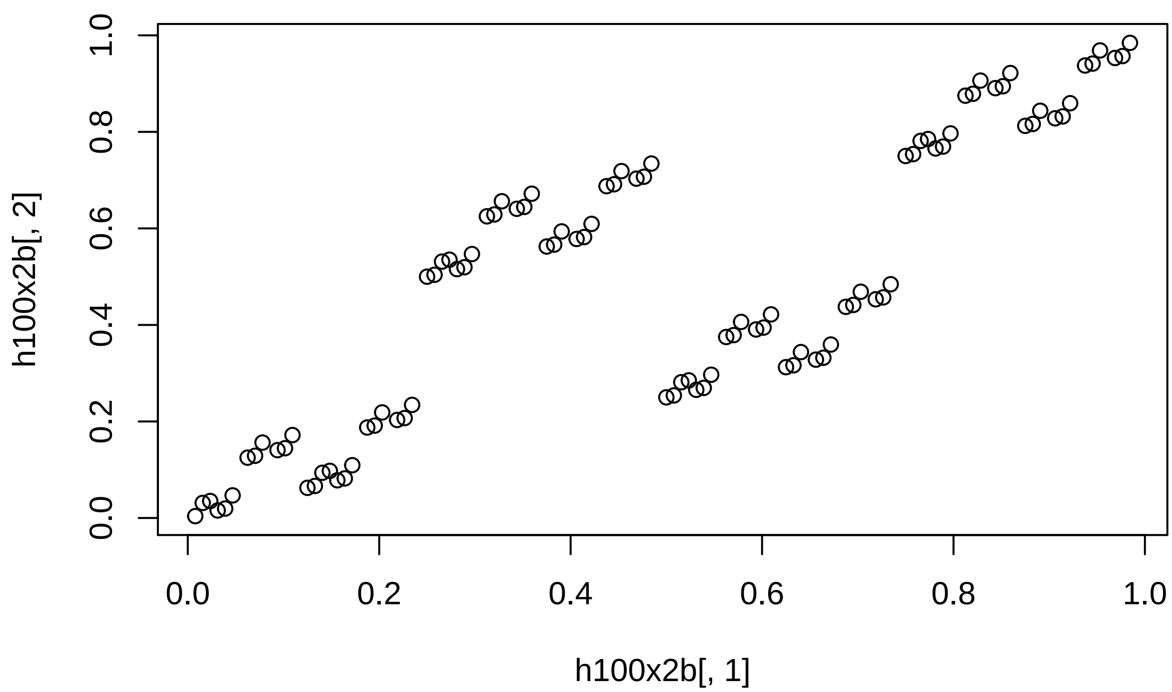
```r
# (a) generate 100 2-dimension vector of U[0,1]x[0,1]
# uniform number generation
u100x2 <- cbind(runif(seed1, 100), runif(seed2*5, 100))
# (b) halton sequences base(2,7)
h100x2a <- cbind(haltonSeq(100, 2), haltonSeq(100, 7))
# (c) halton sequences base(2,4)
h100x2b <- cbind(haltonSeq(100, 2), haltonSeq(100,4))
# (d) plot the sequences
plot(u100x2[,1],u100x2[,2])
```




```r
plot(h100x2a[,1],h100x2a[,2])
```




```
plot(h100x2b[,1],h100x2b[,2])
```



The first plot shows the 2-dimensional distribution of the two uniform variates. From the plot, we could see that, even though both variates are independently uniformly distributed, their 2-dimensional distribution is not as evenly spread as we would like to have. By chance there exist areas with clusters of observations, and area with low density. The second plot shows the distribution of 100 2-dimensional Halton sequences with base 2 and 7. The plot has a more evenly spread distribution than the first plot. The Halton's low-discrepancy sequences with prime number bases filled evenly across the [0,1]x[0,1] domain. The two prime numbers has not common divisor, therefore, the 100 2-dimonsional vectors distributed evenly and never overlap. The third plot shows the distribution of 100 2-dimensional Halton sequences with base 2 and 4. We see many clusting and overlaping of the observations because 4 is divisable by 2. This makes the uniform variates no longer evenly distributed in the 2-dimensional space.

```
# (e)
q5e(10000, c(2,4))
```

```
## [1] -0.004883898
```

```
q5e(10000, c(2,7))
```

```
## [1] 0.0261144
```

```
q5e(10000, c(5,7))
```

```
## [1] 0.02616366
```