Class Project

Hukai Luo

04 December 2018

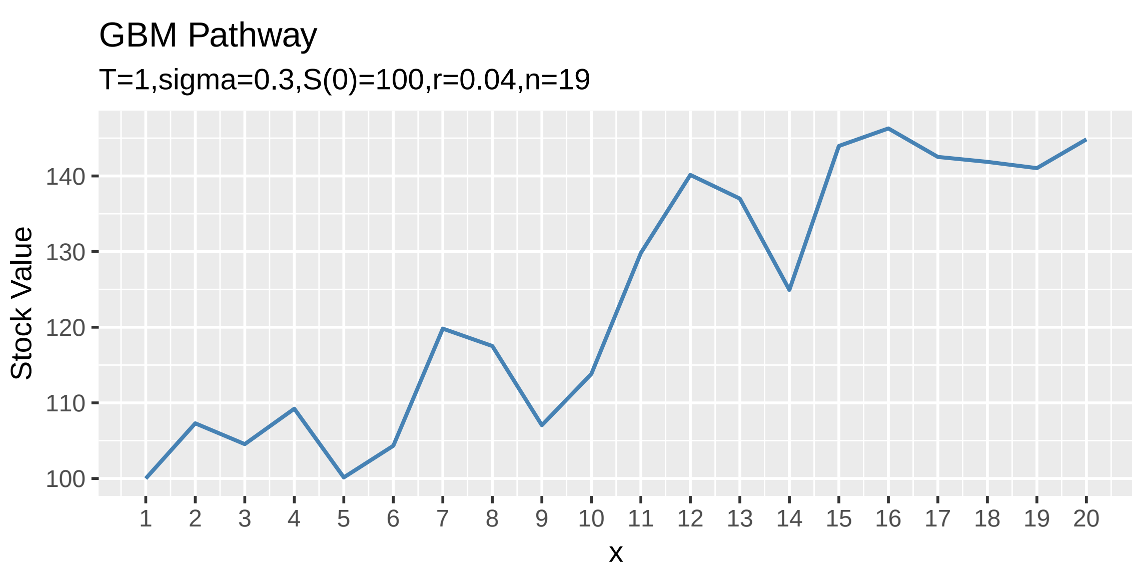
1 Simulate Geometric Brownian Motion

Geometric Brownian Motion solution

$$S(T) = S(t)e^{(\mu - \frac{1}{2}\sigma^2)(T-t) + \sigma\sqrt{T-t}z}$$

Now, given some initial data $S(0) = 100$, $r = 0.04$, $\sigma = 0.3$, $T = 1$, let's simulate the Geometric Brownian Motion pathways.

```
s0 <- 100
r <- 0.04
sigma <- 0.3                                   # input initial data
T <- 1
S_T <- function(s0,r,sigma,T,n=19){            # build the GBM function below
  data <- double(0)
  data[1] <- s0
  for(i in 1:19){
    s0 <- s0*exp((r-0.5*sigma^2)*(T/n)+sigma*sqrt(T/n)*rnorm(1,0,1))
    data[i+1] <- s0
  }
  return(data)
}
data <- S_T(s0,r,sigma,T)                      # use the function the get each path's stock value S_T
library(ggplot2)                               # plot the pathway
plot1 <- ggplot(data.frame(x=seq(1:20),S_t = data),aes(x=x,y=S_t))+
  geom_line(col="steelblue", size=0.7)+scale_x_continuous(breaks=seq(1, 20, 1))+
  labs(title="GBM Pathway", subtitle="T=1,sigma=0.3,S(0)=100,r=0.04,n=19",y="Stock Value")
plot1
```

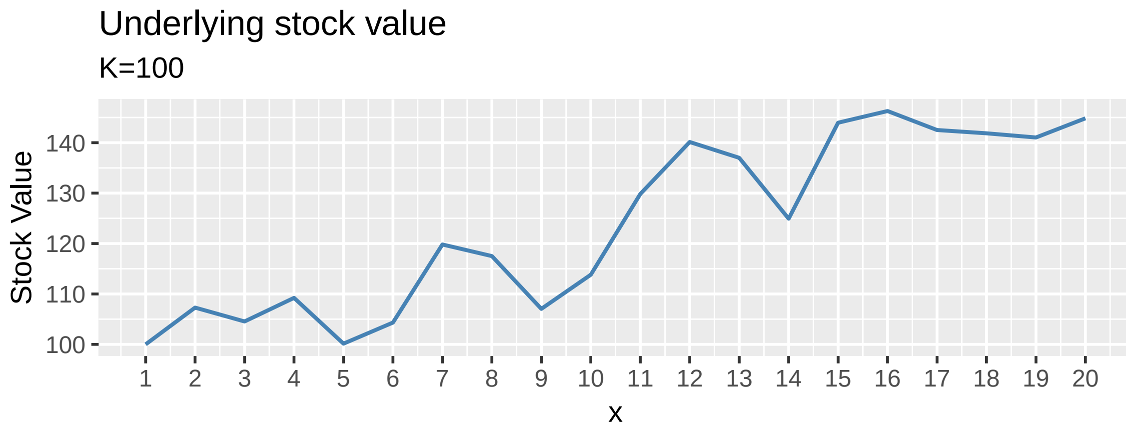


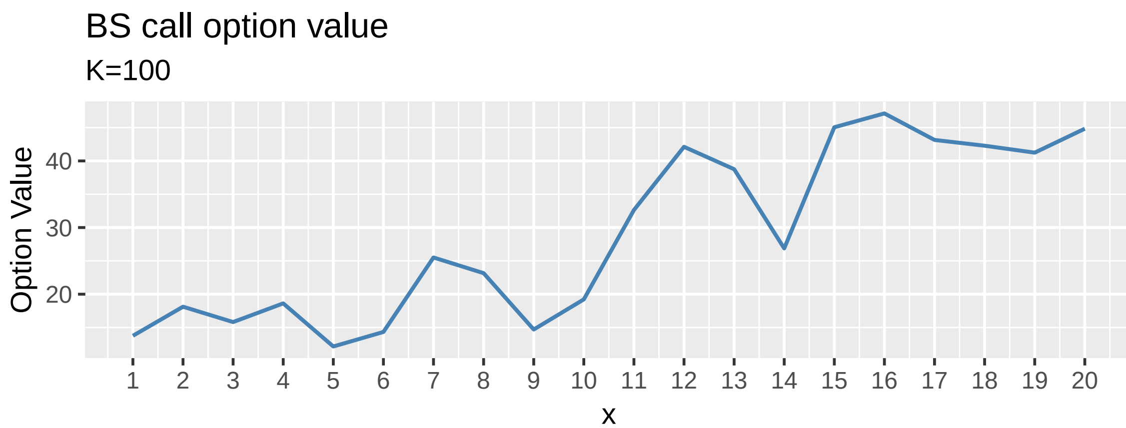
2 Vanilla black-scholes european call option

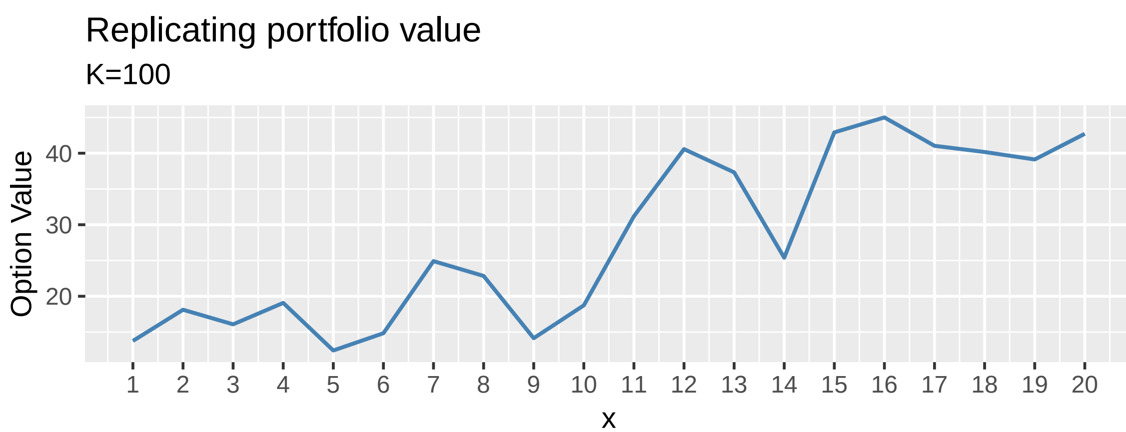
Set the $K = 100$

```
K=100                                # Excercise price K=100
n <- 19
B_S <- function(data,K){             # Build the Bs call option function
  call_value <- double(0)            # calculate d1,d2,call option value
  d1 <- double(0)
  d2 <- double(0)
  for(i in 1:20){
    d1[i] <- (log(data[i]/K)+(r+0.5*sigma^2)*(20-i)*T/n)/(sigma*sqrt((20-i)*T/n))
    d2[i] <- (log(data[i]/K)+(r-0.5*sigma^2)*(20-i)*T/n)/(sigma*sqrt((20-i)*T/n))
    call_value[i] <- data[i]*pnorm(d1[i])-K*exp(-r*(20-i)*T/n)*pnorm(d2[i])
    }
  cashflow <- double(0)              # calculate the cash flow, bt and replicating value
  bt <- double(0)
  replicate <- double(0)
  bt[1] <- K*exp(-r*T)*pnorm(d2[1])
  cashflow[1] <- 0
  replicate[1] <- -bt[1]+data[1]*pnorm(d1[1])
  for(i in 2:20){
    cashflow[i] <- data[i]*(pnorm(d1[i])-pnorm(d1[i-1]))
    bt[i] <- bt[i-1]*exp(r*T/n)+cashflow[i]
    replicate[i] <- data[i]*pnorm(d1[i])- bt[i]
  }
  return(data.frame(stock=data,d1=d1,delta=pnorm(d1),Bt=bt,replicate=replicate,call_option=call_value))
}
call <- B_S(data,100)
print(call)
```

```
##       stock        d1     delta        Bt replicate call_option
## 1  100.0000 0.2833333 0.6115393  47.40067  13.75326    13.75326
## 2  107.2955 0.5169295 0.6973973  56.71274  18.11484    18.11941
## 3  104.5432 0.4245756 0.6644270  53.38543  16.07586    15.81720
## 4  109.2257 0.5805530 0.7192291  59.48375  19.07459    18.62032
## 5  100.1430 0.2571089 0.6014526  47.81462  12.41665    12.15075
## 6  104.3274 0.4077184 0.6582598  53.84193  14.83258    14.33041
## 7  119.8154 0.9628808 0.8321963  74.79567  24.91426    25.50758
## 8  117.5138 0.9020781 0.8164923  73.10787  22.84124    23.15682
## 9  107.0416 0.5136895 0.6962654  60.39267  14.13669    14.69709
## 10 113.8172 0.8002104 0.7882055  70.98431  18.72702    19.22031
## 11 129.8152 1.4588023 0.9276902  89.24114  31.18717    32.63023
## 12 140.1362 1.9173078 0.9724006  95.69475  40.57377    42.11535
## 13 137.0039 1.9009775 0.9713475  95.75215  37.32623    38.76039
## 14 124.9284 1.4794446 0.9304892  90.84959  25.39496    26.87584
## 15 143.9603 2.5129558 0.9940138 100.18606  42.91245    45.05210
## 16 146.2810 2.8932470 0.9980936 100.99400  45.00811    47.13115
## 17 142.5150 3.0845133 0.9989806 101.33325  41.03650    43.14958
## 18 141.8620 3.6845893 0.9998855 101.67518  40.17052    42.28251
## 19 141.0330 5.0606415 0.9999998 101.90558  39.12736    41.24328
## 20 144.8364       Inf 1.0000000 102.12038  42.71600    44.83638
```







The value of the replicating portfolio at time T

$$C = \triangle S + B$$

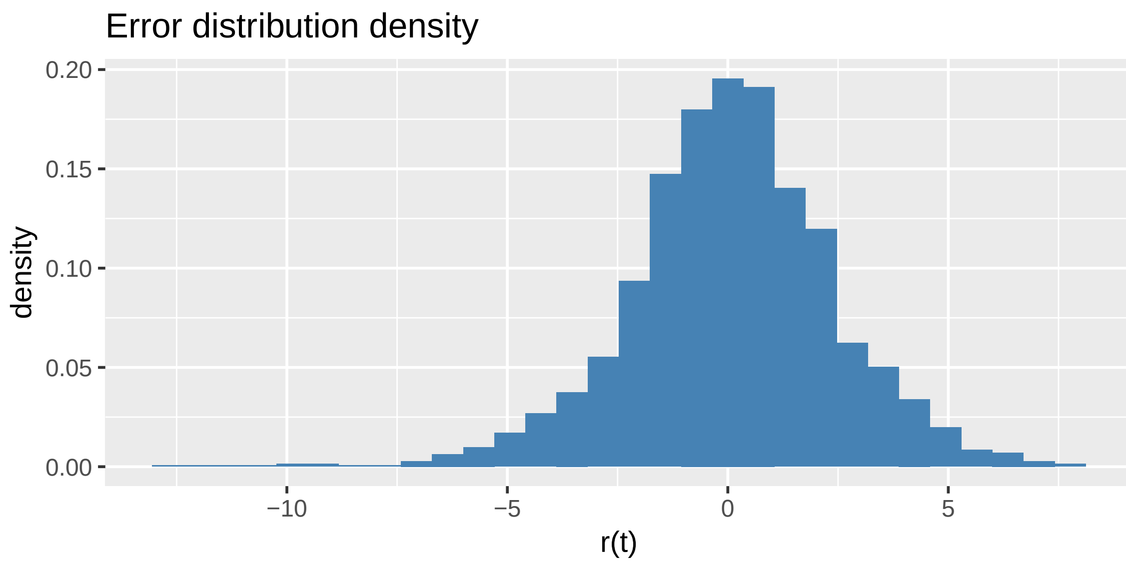
The terminal value of the call at time T

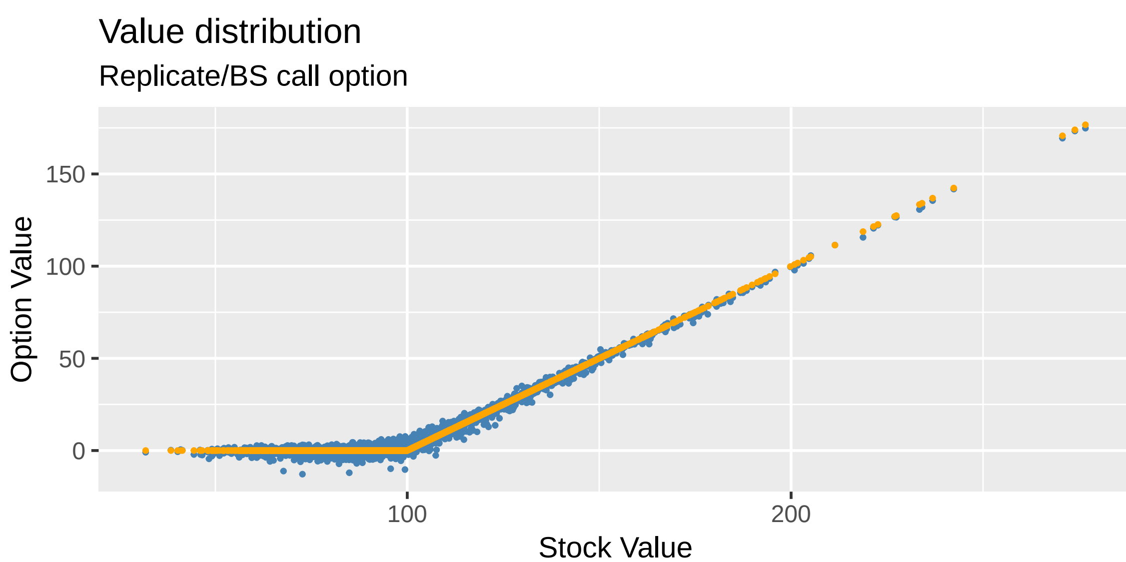
$$C = max(S - K, 0)$$

Now repeat above process for 2000 times to generate the distribution of replicating error.

```
stock <- call <- replicate <- error <- double(0)
for(i in 1:2000){
```

```
  t <- B_S(S_T(s0,r,sigma,T),100)
  error[i] <- t$replicate[20]-t$call_option[20]
  replicate[i] <- t$replicate[20]
  call[i] <- t$call_option[20]
  stock[i] <- t$stock[20]
}
```





3 Cash-or-nothing call option

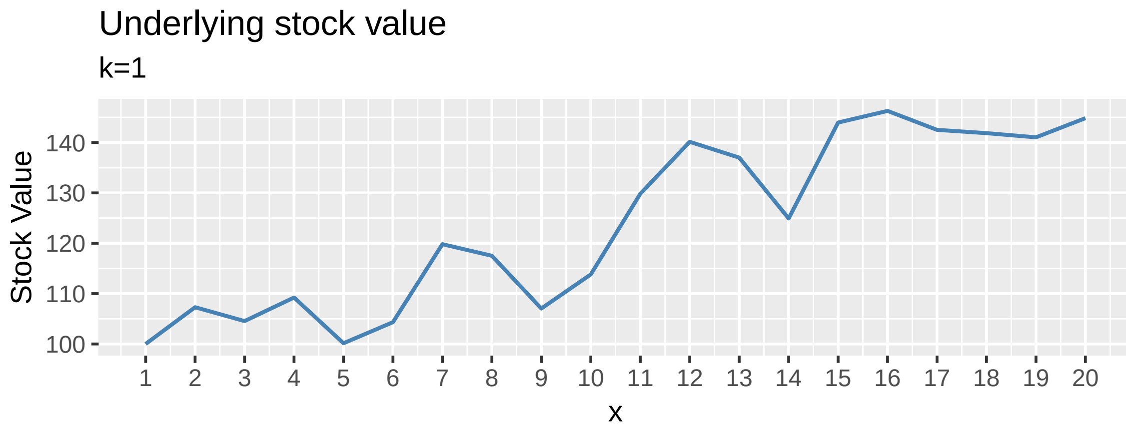
Cash-or-nothing call option either pays you a fixed amount of money or nothing at all.

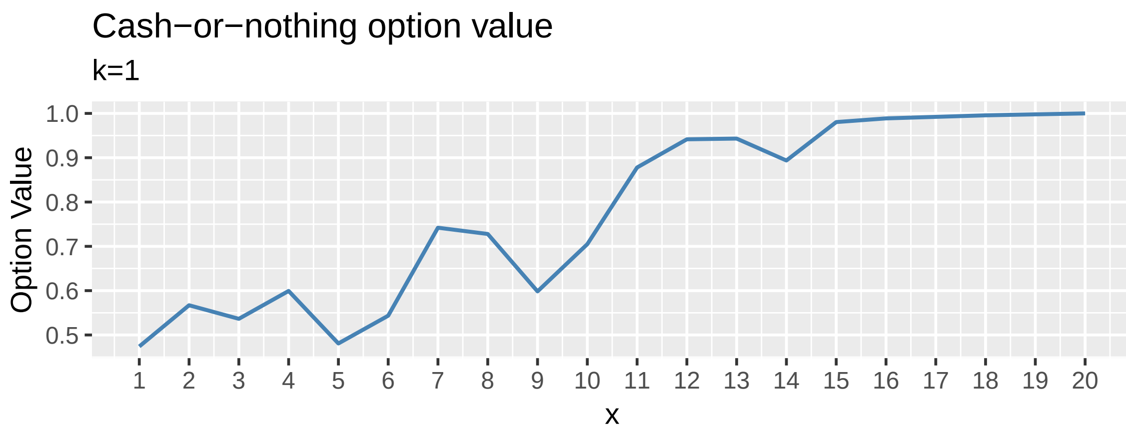
$$C_{cn} = ke^{-r\tau}N(d2)$$

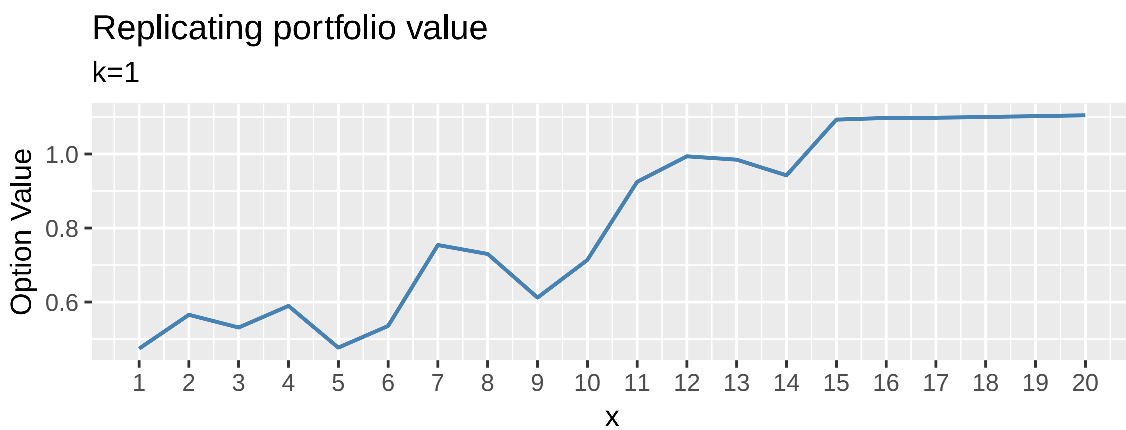
where S the initial stock price, K the strike price, T the time to maturity,σ the volatility and r the risk free interest rate, in thie section, we will set $k = 1$ for convenience.

```r
t <- 1e-6
C_N <- function(data,K){
  call_value <- double(0)             # calculate d1,d2,call option value,cashflow,replicating value
  d1 <- double(0)
  d2 <- double(0)
  delta <- double(0)
  for(i in 1:20){
    d1[i] <- (log(data[i]/K)+(r+0.5*sigma^2)*(20-i)*T/n)/(sigma*sqrt((20-i)*T/n))
    d2[i] <- (log(data[i]/K)+(r-0.5*sigma^2)*(20-i)*T/n)/(sigma*sqrt((20-i)*T/n))
    call_value[i] <- exp(-r*(20-i)*T/n)*pnorm(d2[i])
    delta[i] <- exp(-r*(20-i)*T/n)*dnorm(d2[i])/(data[i]*sigma*sqrt((20-i)*T/n))
  }
  delta[20] <- 0
  cashflow <- double(0)               # calculate the cash flow and replicating value
  bt <- double(0)
  replicate <- double(0)
  bt[1] <- -call_value[1]+data[1]*delta[1]
  cashflow[1] <- 0
  replicate[1] <- -bt[1]+data[1]*delta[1]
  for(i in 2:20){
    cashflow[i] <- data[i]*(delta[i]-delta[i-1])
    bt[i] <- bt[i-1]*exp(r*T/n)+cashflow[i]
    replicate[i] <- data[i]*delta[i]- bt[i]
  }
  return(data.frame(stock=data,delta=delta,bt=bt,replicate=replicate,CON=call_value))
}
call1 <- C_N(data,100)
print(call1)
```

```
##       stock        delta          bt replicate       CON
## 1  100.0000 1.277488e-02  0.80348097 0.4740067 0.4740067
## 2  107.2955 1.195375e-02  0.71707093 0.5655123 0.5670817
## 3  104.5432 1.284687e-02  0.81195158 0.5311005 0.5364410
## 4  109.2257 1.224385e-02  0.74779759 0.5895459 0.5993801
## 5  100.1430 1.447990e-02  0.97329816 0.4767620 0.4808052
## 6  104.3274 1.425622e-02  0.95201338 0.5353000 0.5434409
## 7  119.8154 1.011272e-02  0.45756562 0.7540944 0.7420235
## 8  117.5138 1.113968e-02  0.57921084 0.7298546 0.7279229
## 9  107.0416 1.531684e-02  1.02756165 0.6119770 0.5983227
## 10 113.8172 1.330817e-02  0.80110599 0.7135921 0.7049102
## 11 129.8152 6.667030e-03 -0.05932647 0.9248084 0.8779808
## 12 140.1362 3.261161e-03 -0.53673692 0.9937437 0.9415317
## 13 137.0039 3.596734e-03 -0.49189338 0.9846599 0.9431799
## 14 124.9284 7.921503e-03  0.04735655 0.9422643 0.8936870
## 15 143.9603 1.102573e-03 -0.93419878 1.0929255 0.9804641
## 16 146.2810 4.409836e-04 -1.03294553 1.0974531 0.9887096
## 17 142.5150 2.874868e-04 -1.05699806 1.0979692 0.9922017
## 18 141.8620 4.619730e-05 -1.09345546 1.1000091 0.9956319
## 19 141.0330 1.592226e-07 -1.10225278 1.1022752 0.9978967
## 20 144.8364 0.000000e+00 -1.10459882 1.1045988 1.0000000
```







The value of the replicating portfolio at time T

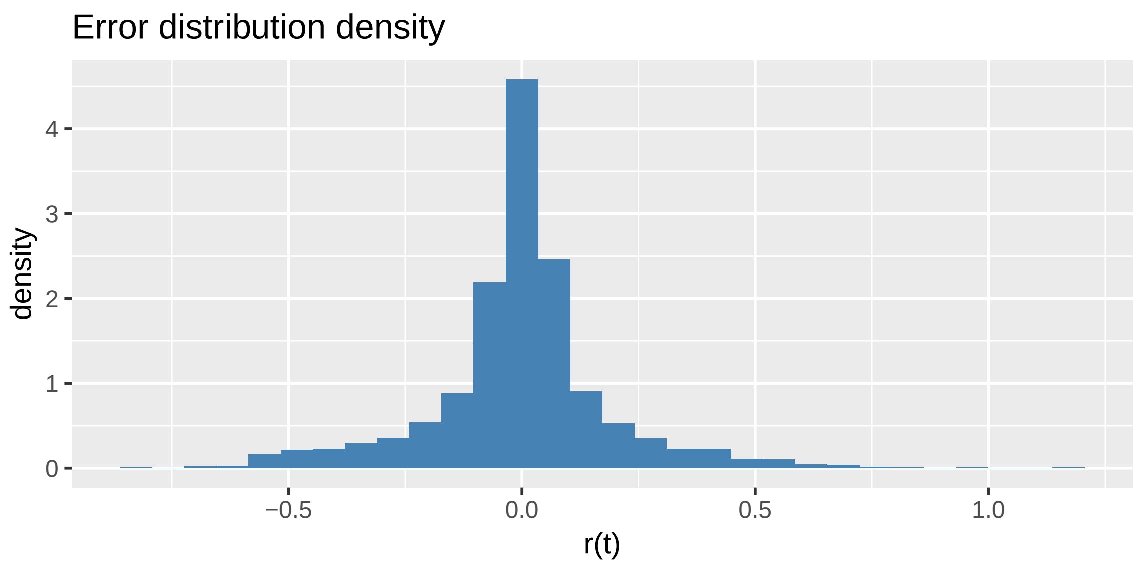
$$C = \triangle S + B$$

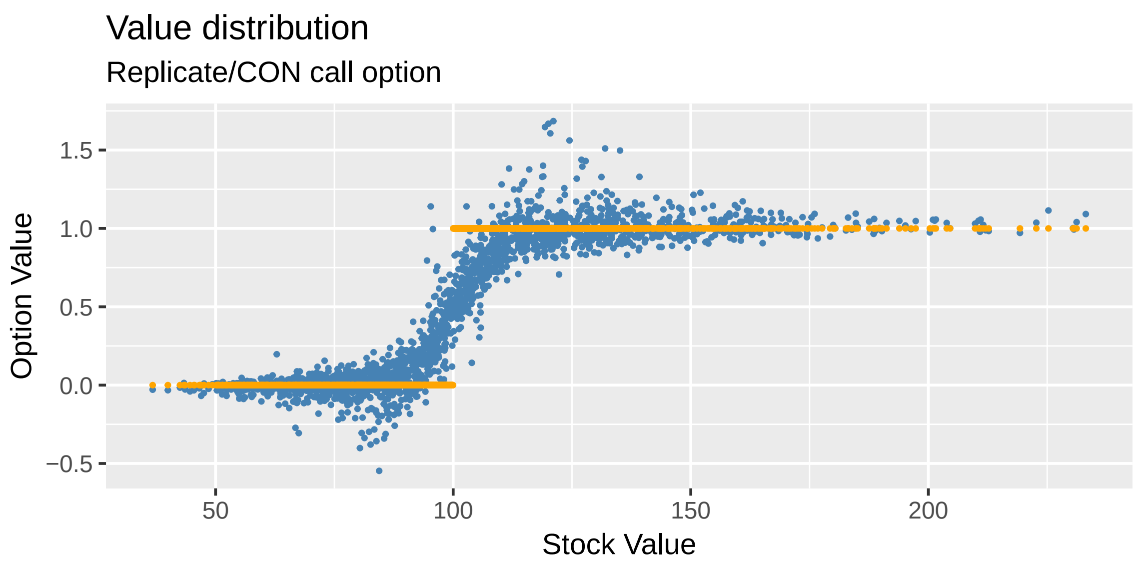
The terminal value of the call at time T

$$C = \frac{max(S-K,0)}{|S-K|}$$

Now repeat above process for 2000 times to generate the distribution of replicating error.

```
stock <- call <- replicate <- error <- double(0)
for(i in 1:2000){
  t <- C_N(S_T(s0,r,sigma,T),100)
  error[i] <- t$replicate[20]-t$CON[20]
  replicate[i] <- t$replicate[20]
  call[i] <- t$CON[20]
  stock[i] <- t$stock[20]
}
```





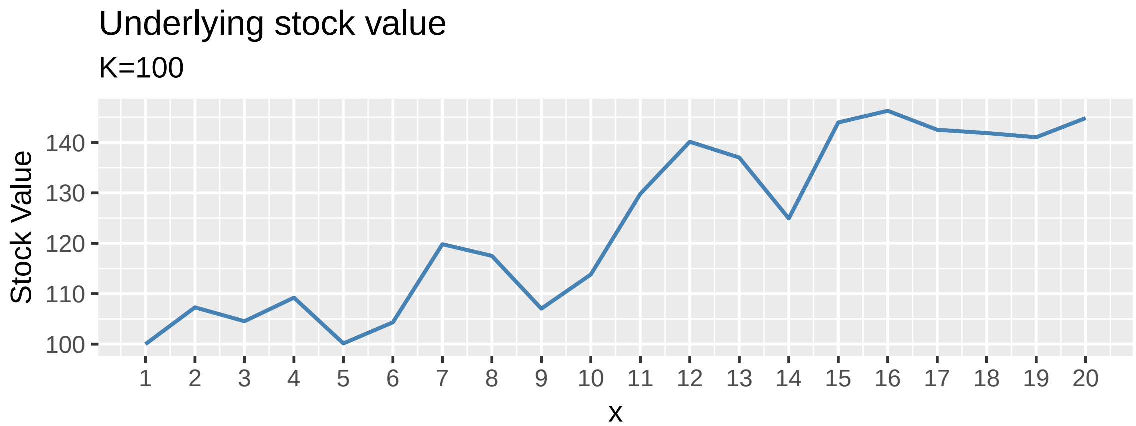
4 Asset-or-nothing call option

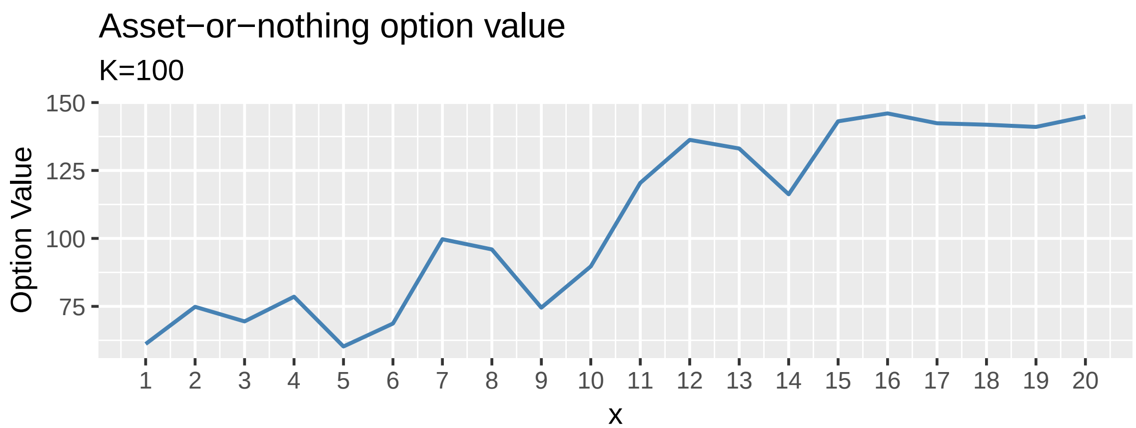
The asset-or-nothing option is basically the same, but your payment equals the price of the asset underlying the option.

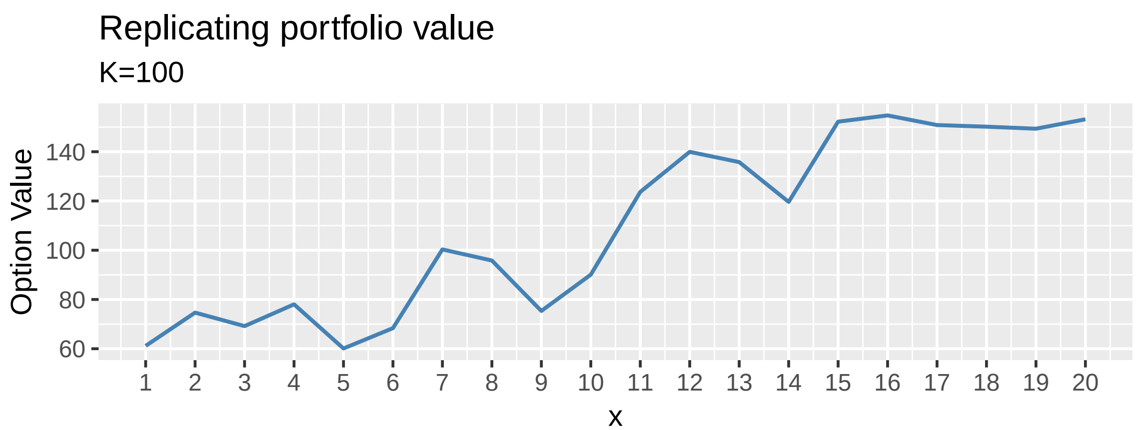
$$C_{an} = SN(d1)$$

```r
epsilon <- 0.01
A_N <- function(data,K){
  call_value <- double(0)             # calculate d1,d2,call option value,cashflow,replicating value
  d1 <- double(0)
  d2 <- double(0)
  delta <- double(0)
  for(i in 1:20){
    d1[i] <- (log(data[i]/K)+(r+0.5*sigma^2)*(20-i)*T/n)/(sigma*sqrt((20-i)*T/n))
    d2[i] <- (log(data[i]/K)+(r-0.5*sigma^2)*(20-i)*T/n)/(sigma*sqrt((20-i)*T/n))
    call_value[i] <- data[i]*pnorm(d1[i])
    delta[i] <- pnorm(d1[i])+dnorm(d1[i])/(sigma*sqrt((20-i)*T/n))
  }
  delta[20] <- pnorm(d1[i])
  cashflow <- double(0)               # calculate the cash flow and replicating value
  bt <- double(0)
  replicate <- double(0)
  bt[1] <- -call_value[1]+data[1]*delta[1]
  cashflow[1] <- 0
  replicate[1] <- -bt[1]+data[1]*delta[1]
  for(i in 2:20){
    cashflow[i] <- data[i]*(delta[i]-delta[i-1])
    bt[i] <- bt[i-1]*exp(r*T/n)+cashflow[i]
    replicate[i] <- data[i]*delta[i]- bt[i]
  }
  return(data.frame(stock=data,delta=delta,bt=bt,replicate=replicate,AON=call_value))
}
call2 <- A_N(data,100)
print(call2)
```

```
##       stock    delta         bt replicate       AON
## 1  100.0000 1.889027 127.748766  61.15393  61.15393
## 2  107.2955 1.892772 128.419833  74.66607  74.82758
## 3  104.5432 1.949114 134.580593  69.18592  69.46130
## 4  109.2257 1.943614 134.263508  78.02918  78.55834
## 5  100.1430 2.049442 145.144438  60.09285  60.23127
## 6  104.3274 2.083881 149.043267  68.36258  68.67450
## 7  119.8154 1.843469 120.552236 100.32370  99.70993
## 8  117.5138 1.930460 131.028950  95.82670  95.94910
## 9  107.0416 2.227949 163.148831  75.33439  74.52936
## 10 113.8172 2.119022 151.094904  90.08623  89.71133
## 11 129.8152 1.594393  83.308492 123.66800 120.42831
## 12 140.1362 1.298517  42.021058 139.94814 136.26852
## 13 137.0039 1.331021  46.562813 135.79222 133.07838
## 14 124.9284 1.722640  95.585241 119.62139 116.24455
## 15 143.9603 1.104271   6.766184 152.20500 143.09851
## 16 146.2810 1.042192  -2.300552 154.75341 146.00211
## 17 142.5150 1.027729  -4.366553 150.83343 142.36975
## 18 141.8620 1.004505  -7.670366 150.17143 141.84570
## 19 141.0330 1.000016  -8.319696 149.35489 141.03295
## 20 144.8364 1.000000  -8.339505 153.17588 144.83638
```







The value of the replicating portfolio at time T

$$C = \triangle S + B$$

The terminal value of the call at time T

$$C = \frac{max(S-K,0)}{|S-K|}S$$

Now repeat above process for 2000 times to generate the distribution of replicating error.

```
stock <- call <- replicate <- error <- double(0)
for(i in 1:2000){
  t <- A_N(S_T(s0,r,sigma,T),100)
  error[i] <- t$replicate[20]-t$AON[20]
  replicate[i] <- t$replicate[20]
  call[i] <- t$AON[20]
  stock[i] <- t$stock[20]
}
```