Class Project

Hukai Luo

03 December 2018

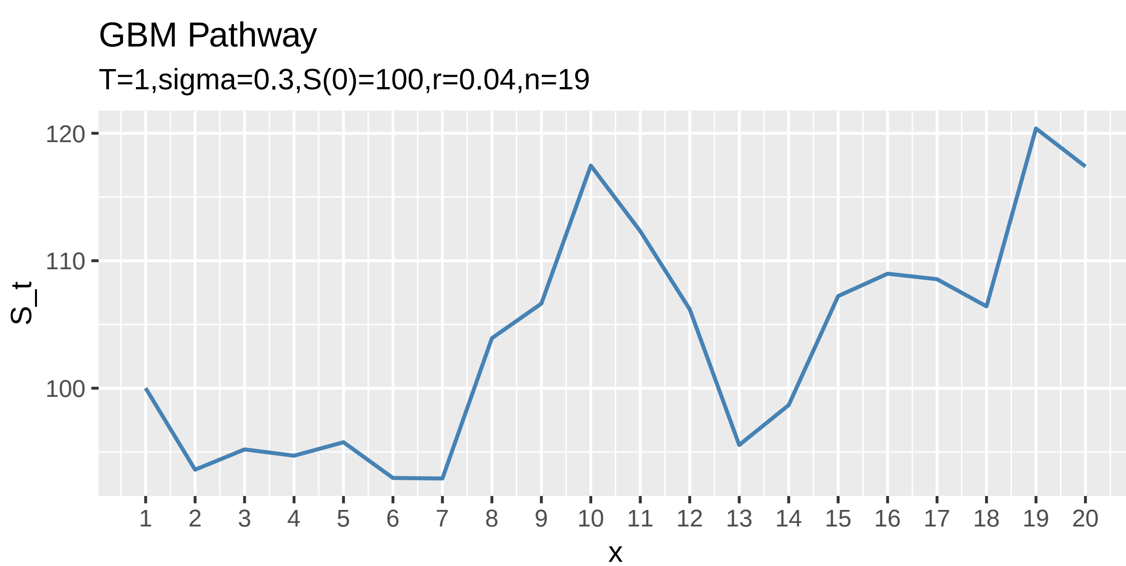
1 Simulate Geometric Brownian Motion

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")
plot1
```

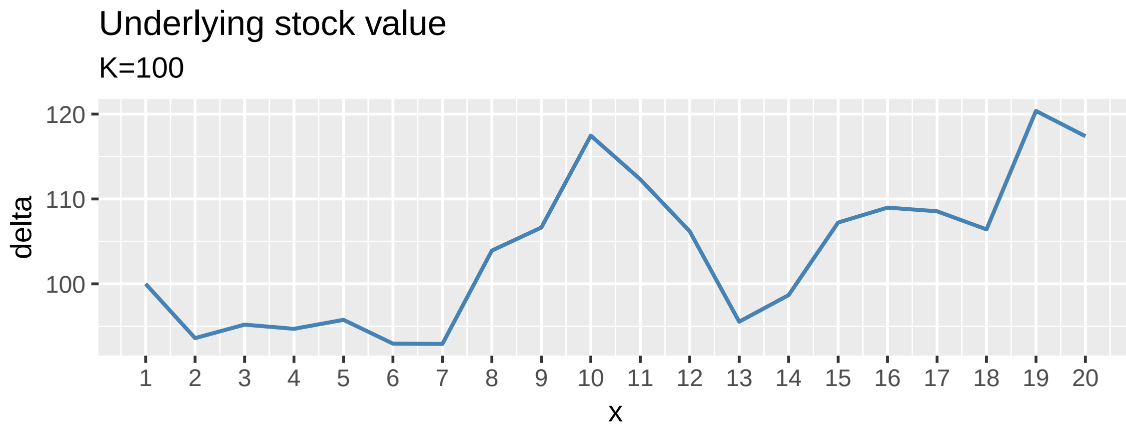


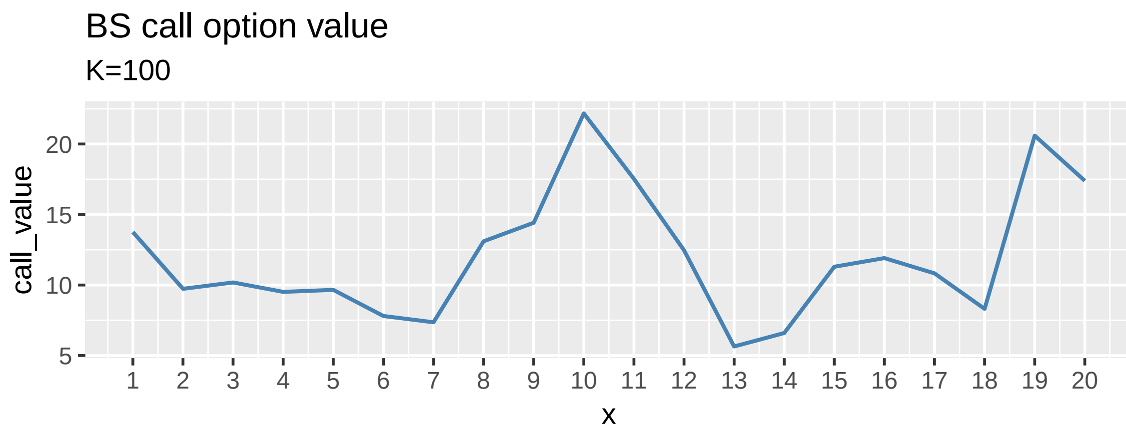
2 Vanilla black-scholes european call option

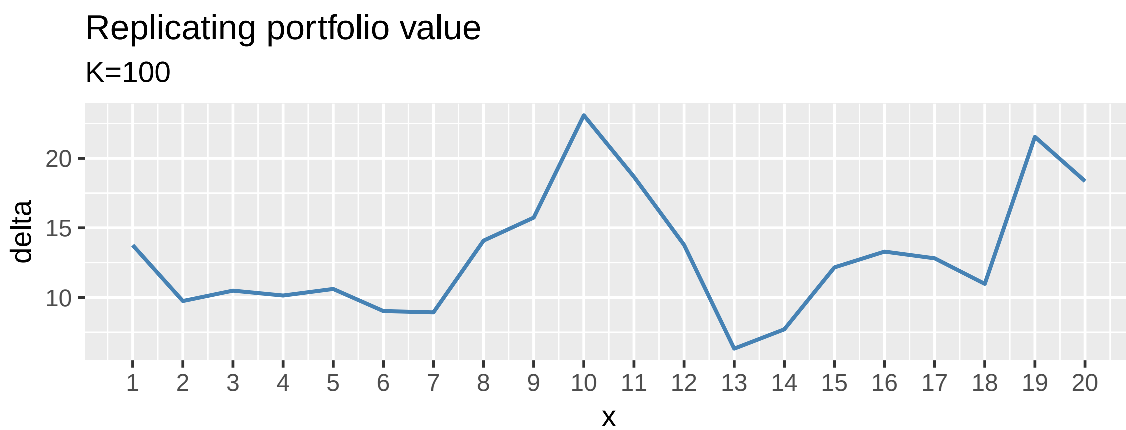
Set the $K = 100$

```
K=100                                        # Excercise price K
n <- 19
B_S <- function(data,K){                     # Build the Bs call option function
  call_value <- double(0)                    # calculate d1,d2,call option value,cashflow,replicating valu
  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 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  100.00000  0.28333333 0.6115393 47.40067 13.753265   13.753265
## 2   93.60228  0.04935243 0.5196808 38.90240  9.740909    9.732150
## 3   95.19400  0.09443981 0.5376201 40.69210 10.486110   10.180429
## 4   94.70188  0.06226924 0.5248258 39.56621 10.135782    9.513571
## 5   95.75865  0.08915920 0.5355223 40.67388 10.607013    9.656523
## 6   92.95430 -0.04050433 0.4838455 35.95602  9.019504    7.803149
## 7   92.91366 -0.06182384 0.4753516 35.24259  8.924062    7.358930
## 8  103.91996  0.38644655 0.6504170 53.50966 14.081650   13.101357
## 9  106.63619  0.49706653 0.6904289 57.88915 15.735564   14.416009
## 10 117.46566  0.94518457 0.8277177 74.13786 23.090544   22.170038
## 11 112.31592  0.75752132 0.7756312 68.44396 18.671768   17.526894
## 12 106.18858  0.49230996 0.6887499 59.36240 13.774969   12.466584
## 13  95.53987 -0.07859034 0.4686792 38.46199  6.315565    5.641730
## 14  98.67404  0.08004144 0.5318979 44.78108  7.703429    6.595620
## 15 107.22410  0.59857870 0.7252731 65.60994 12.156814   11.297607
## 16 108.98026  0.75475293 0.7748014 71.14582 13.292234   11.909723
## 17 108.55039  0.80083057 0.7883851 72.77028 12.809231   10.827006
## 18 106.41772  0.73098954 0.7676072 70.71251 10.974506    8.311856
## 19 120.37877  2.75986111 0.9971087 98.48864 21.542081   20.596425
## 20 117.39231         Inf 1.0000000 99.03562 18.356698   17.392313
```







The value of the replicating portfolio at time T

$$C = \triangle S + B$$

The terminal value of the call at time T

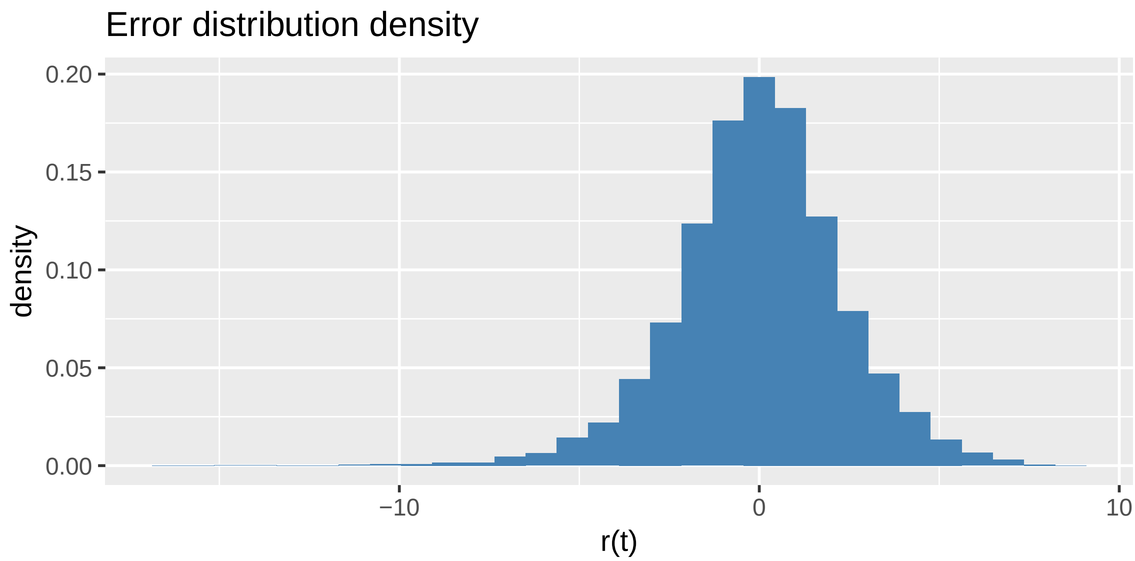
$$C = max(S - K, 0)$$

Now repeat above process for 10000 times to generate the distribution of replicating error.

```r
error <- double(0)
for(i in 1:10000){
```

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

plot4<- ggplot(data.frame(x = error), aes(x = x))+
  geom_histogram(aes(y=..density..),fill="steelblue")+labs(x = 'r(t)',
  y = 'density', title='Error distribution density')
plot4
```



3 Cash-or-nothing call option

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

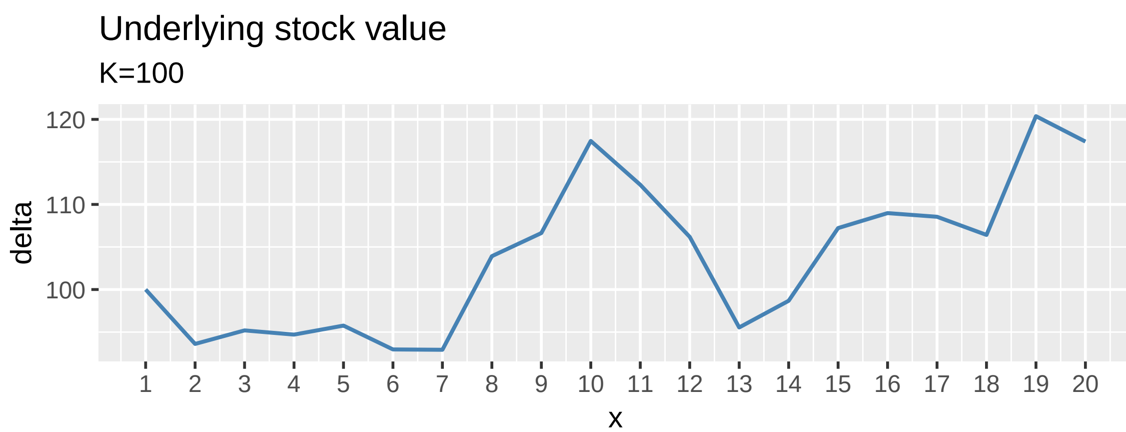
$$C_{cn} = e^{-r\tau}N(d2)$$

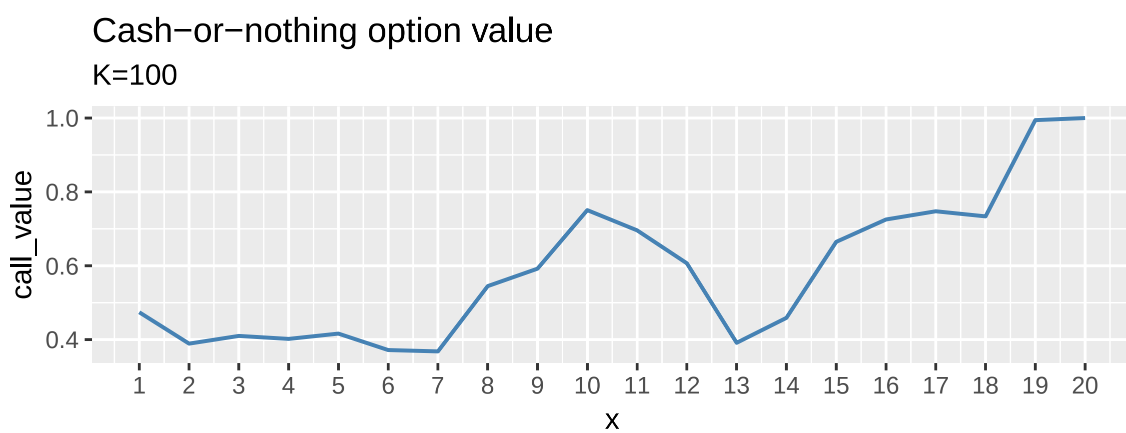
where S the initial stock price, K the strike price, T the time to maturity,σ the volatility and r the risk free interest rate. We can replicate it with two call option with different exercise price, we can buy call at $K = 100$ and sell call at $K = 100 + \epsilon$

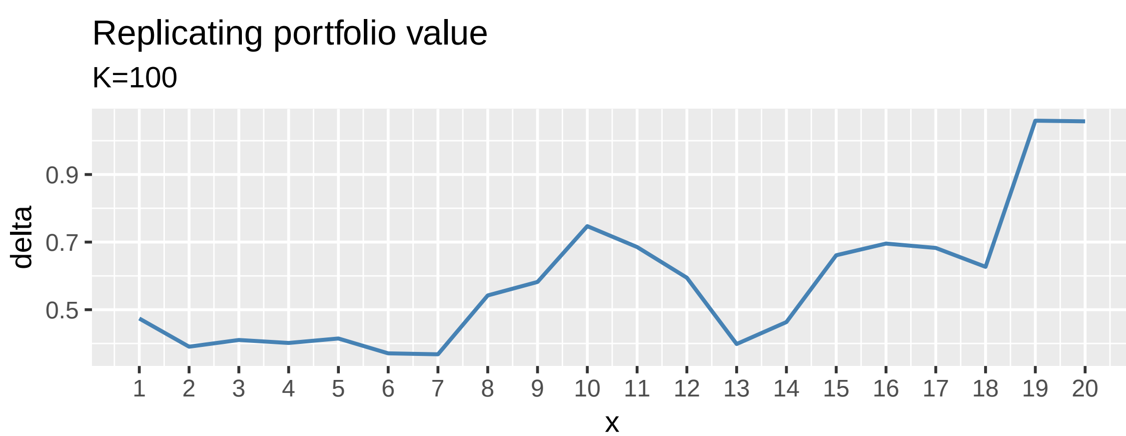
```
epsilon <- 0.01
C_N <- function(data,K){
  call_value <- double(0)                 # calculate d1,d2,call option value,cashflow,replicating valu
  d1 <- double(0)
  d2 <- double(0)
  d11 <- double(0)
  d21 <- double(0)
  for(i in 1:20){
    d1[i] <- (log(data[i]/K)+(r+0.5*sigma^2)*(20-i)*T/n)/(sigma*sqrt((20-i)*T/n))
    d2[i] <- (log(data[i]/K)+(r-0.5*sigma^2)*(20-i)*T/n)/(sigma*sqrt((20-i)*T/n))
    d11[i] <- (log(data[i]/(K+epsilon))+(r+0.5*sigma^2)*(20-i)*T/n)/(sigma*sqrt((20-i)*T/n))
    d21[i] <- (log(data[i]/(K+epsilon))+(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])
    }
```

```
  cashflow <- double(0)                    # calculate the cash flow and replicating value
  bt <- double(0)
  replicate <- double(0)
  bt[1] <- (K*exp(-r*T)*pnorm(d2[1])-(K+epsilon)*exp(-r*T)*pnorm(d21[1]))/epsilon
  cashflow[1] <- 0
  replicate[1] <- (data[1]*pnorm(d1[1])-data[1]*pnorm(d11[1]))/epsilon-bt[1]
  for(i in 2:20){
    cashflow[i] <- (data[i]*(pnorm(d1[i])-pnorm(d1[i-1]))-data[i]*(pnorm(d11[i])-pnorm(d11[i-1])))/epsi
    bt[i] <- bt[i-1]*exp(r*T/n)+cashflow[i]
    replicate[i] <- -bt[i]+(data[i]*pnorm(d1[i])-data[i]*pnorm(d11[i]))/epsilon
  }
  return(data.frame(stock=data,d1=d1,bt=bt,replicate=replicate,cash_or_nothing=call_value))
}
call1 <- C_N(data,100)
print(call1)
```

```
##        stock          d1         bt replicate cash_or_nothing
## 1  100.00000  0.28333333  0.8035413 0.4739428       0.4740067
## 2   93.60228  0.04935243  0.8867097 0.3905195       0.3891115
## 3   95.19400  0.09443981  0.9219221 0.4103702       0.4099778
## 4   94.70188  0.06226924  0.9680970 0.4015399       0.4018842
## 5   95.75865  0.08915920  1.0126551 0.4147832       0.4162437
## 6   92.95430 -0.04050433  1.0679187 0.3708457       0.3717237
## 7   92.91366 -0.06182384  1.1228237 0.3679660       0.3680772
## 8  103.91996  0.38644655  1.0716379 0.5421946       0.5448995
## 9  106.63619  0.49706653  1.0650874 0.5821180       0.5920870
## 10 117.46566  0.94518457  0.6305466 0.7471558       0.7505836
## 11 112.31592  0.75752132  0.9436286 0.6854280       0.6958884
## 12 106.18858  0.49230996  1.3334100 0.5945669       0.6067079
## 13  95.53987 -0.07859034  1.6881336 0.3984171       0.3913582
## 14  98.67404  0.08004144  1.8641949 0.4633084       0.4588889
## 15 107.22410  0.59857870  1.6629247 0.6610568       0.6646915
## 16 108.98026  0.75475293  1.6805816 0.6956152       0.7252833
## 17 108.55039  0.80083057  1.9542230 0.6827005       0.7475251
## 18 106.41772  0.73098954  2.7134381 0.6267749       0.7337516
## 19 120.37877  2.75986111 -0.9041700 1.0592623       0.9943429
## 20 117.39231         Inf -1.0573201 1.0573201       1.0000000
```







The value of the replicating portfolio at time T

$$C = \lim_{\epsilon \to 0} \frac{F(K) - F(K+\epsilon)}{\epsilon}$$

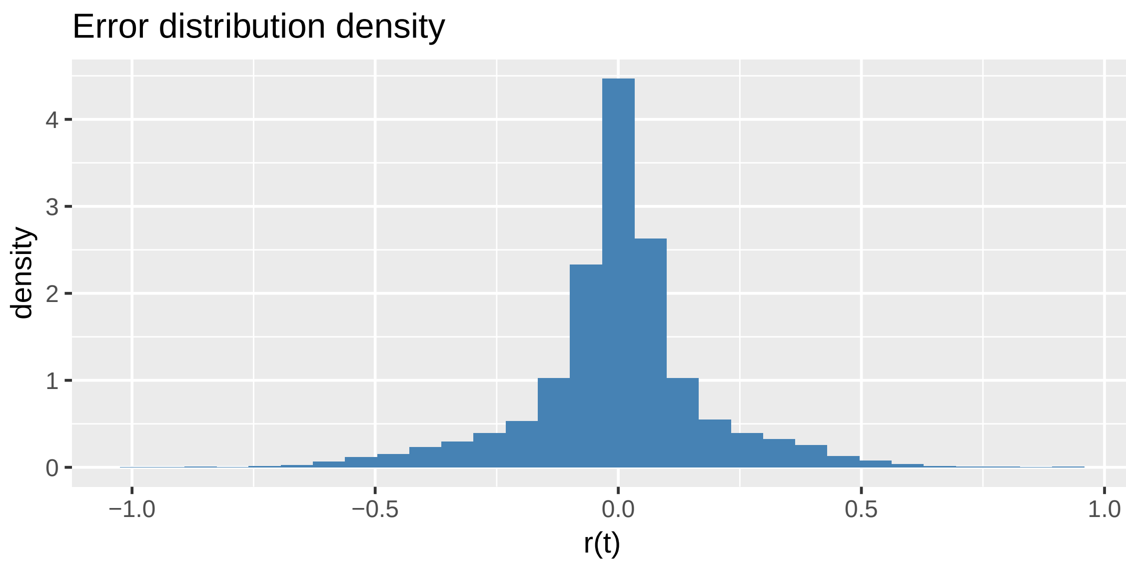
The terminal value of the call at time T

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

Now repeat above process for 10000 times to generate the distribution of replicating error.

```
error <- double(0)
for(i in 1:10000){
  t <- C_N(S_T(s0,r,sigma,T),100)
  error[i] <- t$replicate[20]-t$cash_or_nothing[20]
}

plot9<- ggplot(data.frame(x = error), aes(x = x))+
  geom_histogram(aes(y=..density..),fill="steelblue")+labs(x = 'r(t)',
  y = 'density', title='Error distribution density')
plot9
```



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.

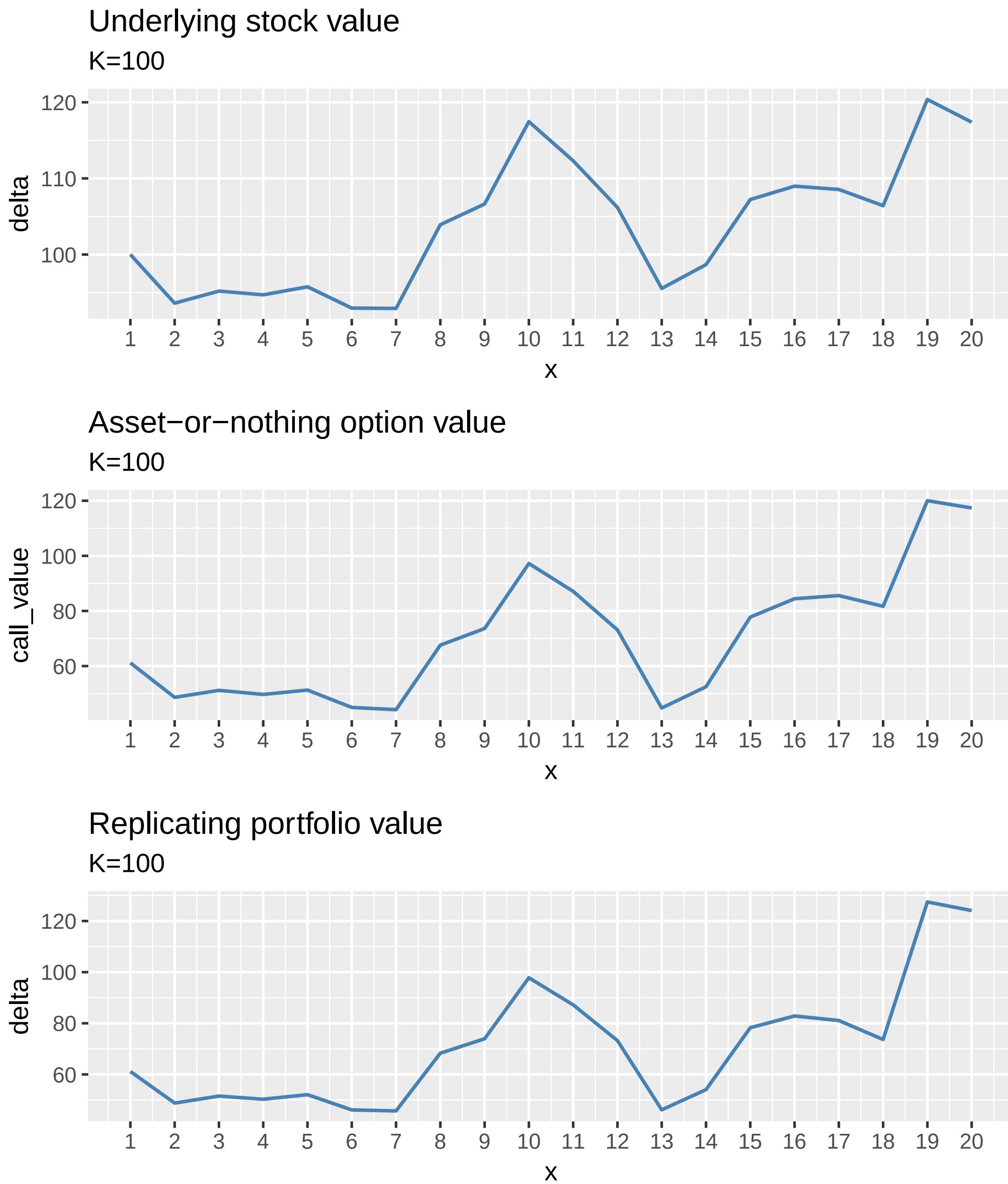
$$C_{an} = SN(d1)$$

In order to replicate this option, we can buy call at $K = 100$ and buy K units cash-or-nothing call at $K = 100$.

```
epsilon <- 0.01
A_N <- function(data,K){
  call_value <- double(0)                  # calculate d1,d2,call option value,cashflow,replicating valu
  d1 <- double(0)
  d2 <- double(0)
  d11 <- double(0)
  d21 <- 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))
    d11[i] <- (log(data[i]/(K+epsilon))+(r+0.5*sigma^2)*(20-i)*T/n)/(sigma*sqrt((20-i)*T/n))
    d21[i] <- (log(data[i]/(K+epsilon))+(r-0.5*sigma^2)*(20-i)*T/n)/(sigma*sqrt((20-i)*T/n))
    call_value[i] <- data[i]*pnorm(d1[i])
```

```r
    }
  cashflow <- double(0)                          # calculate the cash flow and replicating value
  bt <- double(0)
  replicate <- double(0)
  bt[1] <- K*exp(-r*T)*pnorm(d2[1])+K*(K*exp(-r*T)*pnorm(d2[1])-(K+epsilon)*exp(-r*T)*pnorm(d21[1]))/eps
  cashflow[1] <- 0
  replicate[1] <- -bt[1]+data[1]*pnorm(d1[1])+K*(data[1]*pnorm(d1[1])-data[1]*pnorm(d11[1]))/epsilon
  for(i in 2:20){
    cashflow[i] <- data[i]*(pnorm(d1[i])-pnorm(d1[i-1]))+K*(data[i]*(pnorm(d1[i])-pnorm(d1[i-1]))-data[
    bt[i] <- bt[i-1]*exp(r*T/n)+cashflow[i]
    replicate[i] <- -bt[i]+data[i]*pnorm(d1[i])+K*(data[i]*pnorm(d1[i])-data[i]*pnorm(d11[i]))/epsilon
  }
  return(data.frame(stock=data,d1=d1,bt=bt,replicate=replicate,asset_or_nothing=call_value))
}
call2 <- A_N(data,100)
print(call2)
```

```
##        stock          d1         bt replicate asset_or_nothing
## 1  100.00000  0.28333333 127.754796  61.14755         61.15393
## 2   93.60228  0.04935243 127.573363  48.79286         48.64330
## 3   95.19400  0.09443981 132.884302  51.52313         51.17821
## 4   94.70188  0.06226924 136.375910  50.28977         49.70199
## 5   95.75865  0.08915920 141.939386  52.08533         51.28089
## 6   92.95430 -0.04050433 142.747890  46.10408         44.97552
## 7   92.91366 -0.06182384 147.524957  45.72066         44.16665
## 8  103.91996  0.38644655 160.673444  68.30111         67.59131
## 9  106.63619  0.49706653 164.397882  73.94737         73.62471
## 10 117.46566  0.94518457 137.192522  97.80613         97.22840
## 11 112.31592  0.75752132 162.806819  87.21456         87.11573
## 12 106.18858  0.49230996 192.703405  73.23166         73.13737
## 13  95.53987 -0.07859034 207.275345  46.15727         44.77755
## 14  98.67404  0.08004144 231.200567  54.03426         52.48451
## 15 107.22410  0.59857870 231.902414  78.26249         77.76676
## 16 108.98026  0.75475293 239.203986  82.85375         84.43806
## 17 108.55039  0.80083057 268.192578  81.07928         85.57951
## 18 106.41772  0.73098954 342.056316  73.65200         81.68701
## 19 120.37877  2.75986111   8.071640 127.46831        120.03072
## 20 117.39231         Inf  -6.696399 124.08871        117.39231
```



The value of the replicating portfolio at time T

$$C = C(K) + K * CON(K)$$

The terminal value of the call at time T

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

Now repeat above process for 10000 times to generate the distribution of replicating error.

```r
error <- double(0)
for(i in 1:10000){
  t <- A_N(S_T(s0,r,sigma,T),100)
  error[i] <- t$replicate[20]-t$asset_or_nothing[20]
}

plot12<- ggplot(data.frame(x = error), aes(x = x))+
  geom_histogram(aes(y=..density..),fill="steelblue")+labs(x = 'r(t)',
  y = 'density', title='Error distribution density')
plot12
```