Pricing Financial Derivatives Project 1

Angus McKay, Laura Roman, Euan Dowers, Veronika Kyuchukova

February 2, 2017

Exercise 1

Consider a binomial model with $r = 0.04, T = 4, d = 0.98, S_0 = 20$ and a European call with maturity $T = 4$ and strike price $K = 20$ In the lecture notes we are given that in the binomial model the no-arbitrage condition is satisfied if and only if

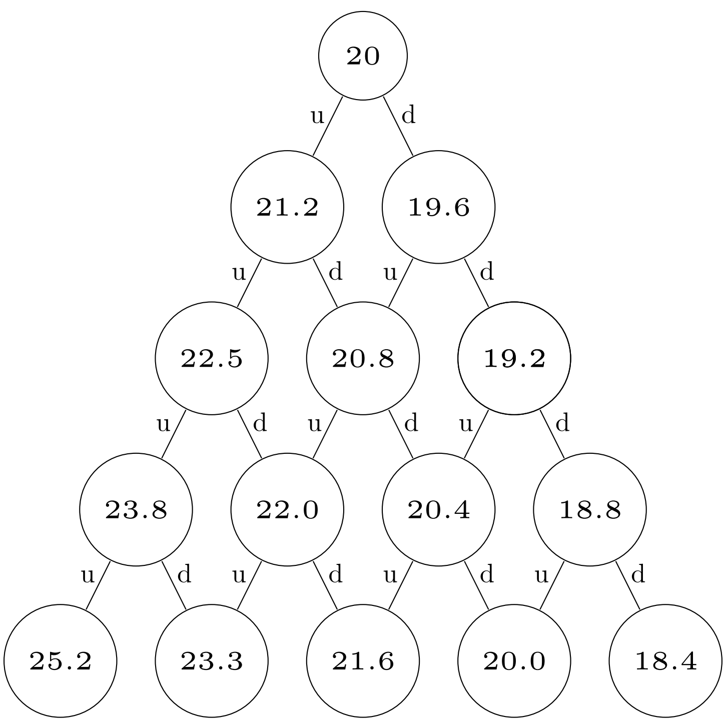
$$d < 1 + r < u$$

so as $0.98 < 1.04 < 1.06$, the no-arbitrage condition is satisfied. Furthemore, we are given that the risk-neutral probability is given by

$$q = \frac{1 + r - d}{u - d} = 0.75$$

So the risk-neutral probability is 0.75.

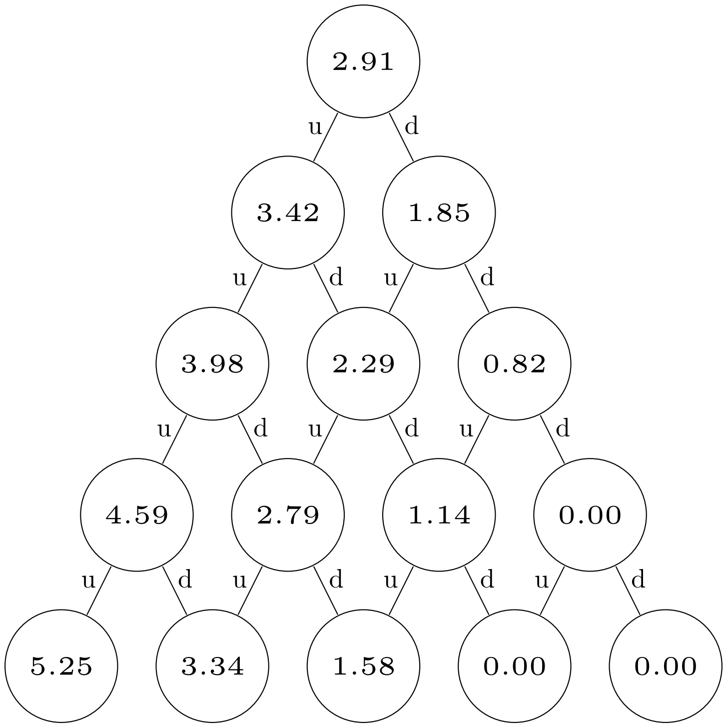
The following tree represents the binomial tree of S_n for $n \in \{0, 1, 2, 3, 4\}$. Note that prices are rounded to 1 decimal place for presentation, but rounding is done after evaluation to avoid numerical errors.



The values of the hedging portfolio at each stage are calculated using the backward recursion formula:

$$V_n = (1 + r)^{n-T} \widetilde{E}[h(S_T)|F_n]$$

The values of the hedging portfolio are shown in the binomial tree below:



The premium fee is calculated using the formula for the expectation of the payoff:

$$V_0 = (1+r)^{-T}\widetilde{E}[h(S_T)]$$

Which gives:

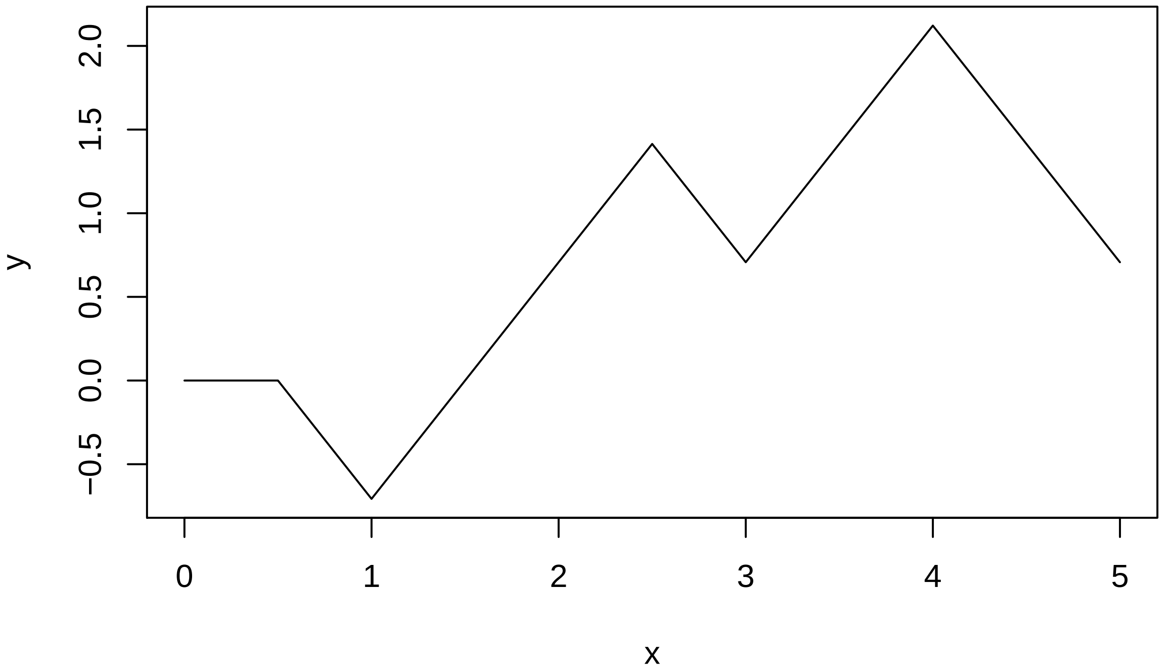
$$V_0 = 1.04^{-4} \times (0.75^4 \times 5.25 + 4 \times 0.75^3 \times 0.25 \times 3.34 + 6 \times 0.75^2 \times 0.25^2 \times 1.58 + 0 + 0) = 2.91$$

This is the same value as obtained for V_0 using backwards recursion.

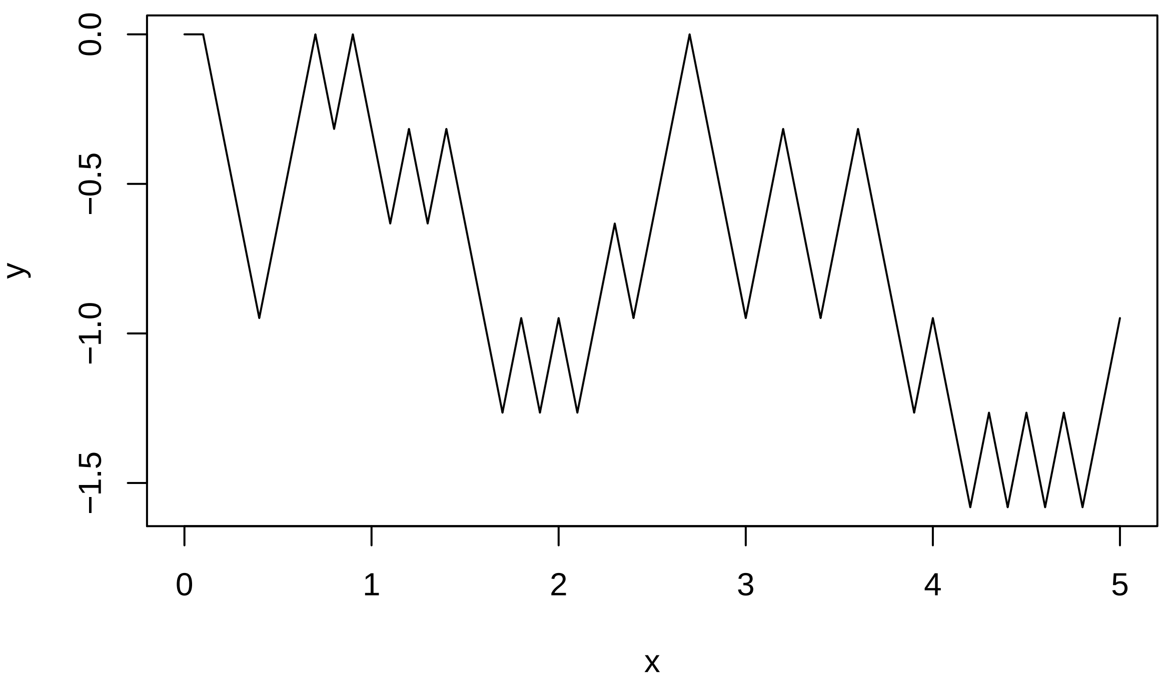
Exercise 2

```
rm(list=ls())
library(ggplot2)
##################
### QUESTION 2 ###
##################

# Part 1

Sn <- function(n,T = 5){
  # Create function that produces random walk with step size sqrt(T/n)
  stepsize <- sqrt(T/n)
  Z <- rbinom(n+1,1,0.5)
  Z[Z==1] <- stepsize
  Z[Z==0] <- -stepsize
  X <- rep(0,n+1)
  for ( i in 2:n+1) {
    X[i] = X[i-1] + Z[i] }
  return(X)
```

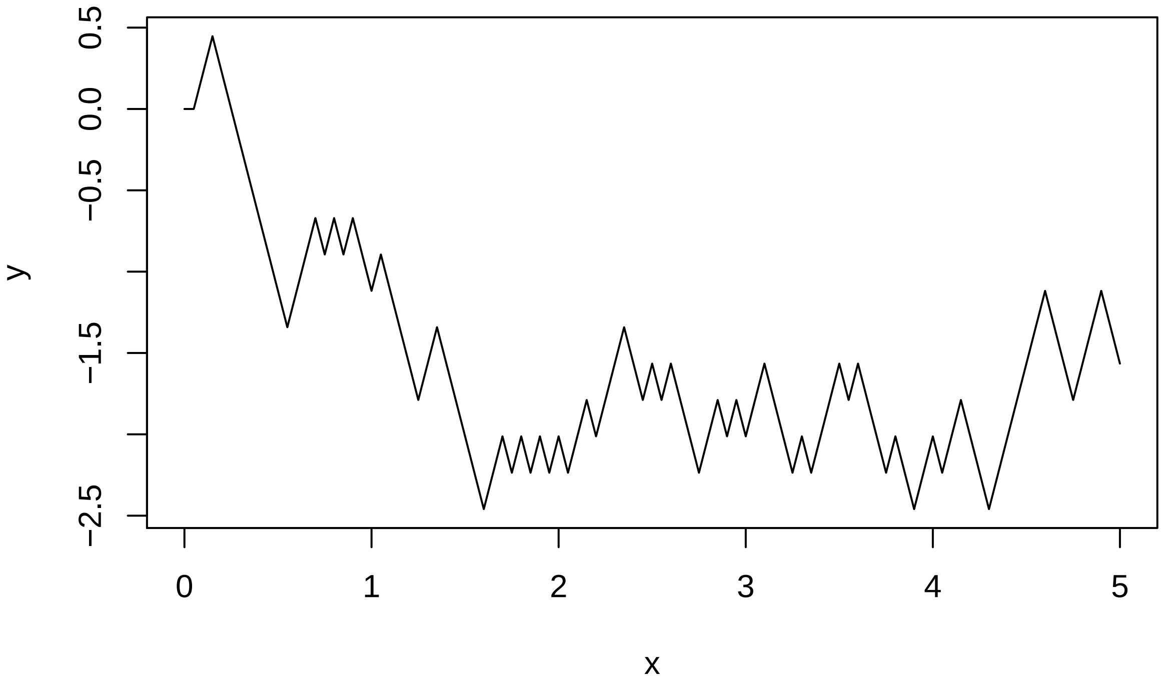
```r
}

# Now plot the process Sn(t) for n = 10,50,100,1000
y <- Sn(10)
x <- seq(0,5,length.out = length(y))
plot(y=y,x=x,type = 'l')
```



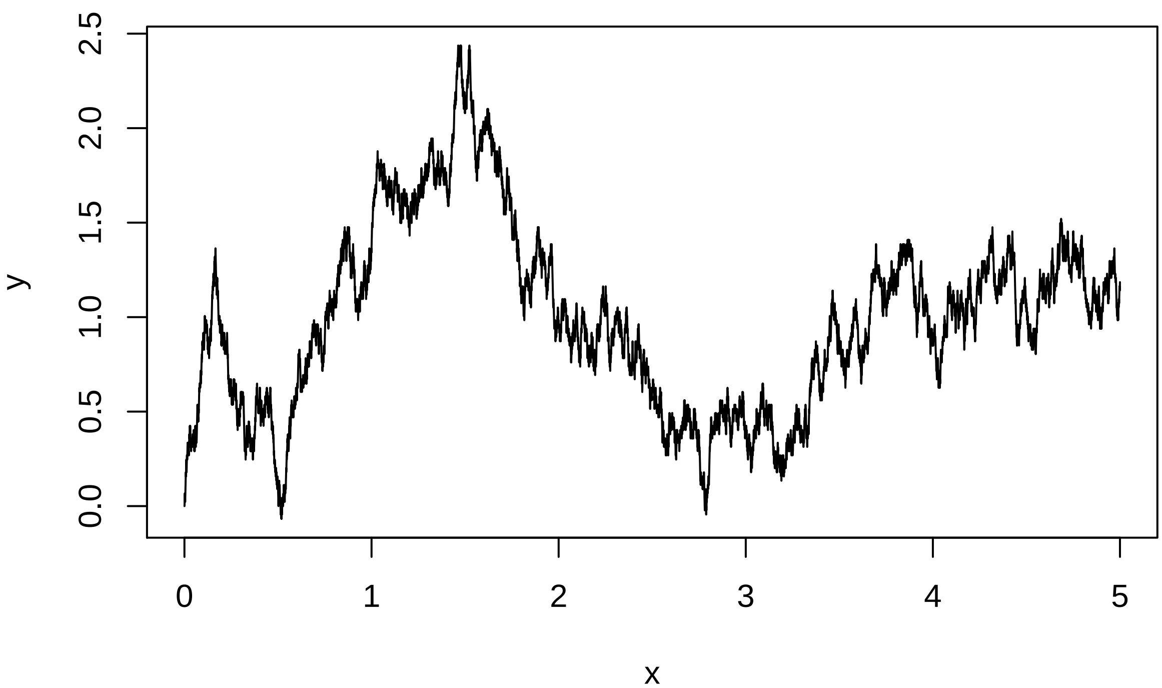

```r
y <- Sn(50)
x <- seq(0,5,length.out = length(y))
plot(y=y,x=x,type = 'l')
```



```r
y <- Sn(100)
x <- seq(0,5,length.out = length(y))
plot(y=y,x=x,type = 'l')
```




```r
y <- Sn(10000)
x <- seq(0,5,length.out = length(y))
plot(y=y,x=x,type = 'l')
```



```
#######################################
# PART 2
#######################################

geometric.brownian <- function(n,T,sigma,mu){
  B <- Sn(n,T)
  t <- seq(0,T,length.out = n+1)
  X <- sigma * B + mu * t
  return(exp(X))
}

brownian.drift <- function(n,T,sigma,mu){
  B <- Sn(n,T)
  t <- seq(0,T,length.out = n+1)
  X <- sigma * B + mu * t
  return(X)
}

brownian.bridge <- function(n,T){
  B <- Sn(n,T)
  t <- seq(0,1,length.out = n+1)
  X <- B - t * B[n]
}

martingale <- function(n,T) {
  B <- Sn(n,T)
  t <- seq(0,T,length.out = n+1)
  return(B^2 - t)
}

n <- 1000
T <- 1
t <- seq(0,T,length.out = n+1)

simulations <- as.data.frame(
  cbind(
    t,
    geometric.brownian(n,T,sigma = 1, mu = -0.5),
    geometric.brownian(n,T,sigma = 1, mu = 0.5),
    brownian.drift(n,T,sigma = 0.1, mu = 1),
    brownian.drift(n,T,sigma = 1, mu = 0.1),
    brownian.bridge(n,T),
    martingale(n,T)
  )
)

ggplot(data = simulations,aes(x=t,y=V2)) +
  labs(x='t', y='Value of process', title = 'Processes related to Brownian motion') +
  geom_line(aes(x=t,y=V2,colour='Geometric mu = -.5, sigma=1')) +
  geom_line(aes(x=t,y=V3,colour='Geometric mu = .5, sigma=1')) +
  geom_line(aes(x=t,y=V4,colour='Drift mu = 1, sigma = 0.1')) +
  geom_line(aes(x=t,y=V5,colour='Drift mu = 0.1, sigma = 1')) +
  geom_line(aes(x=t,y=V6,colour='Brownian Bridge')) +
```

```
geom_line(aes(x=t,y=V7,colour='Martingale'))
```