

Intro to Financial Engineering IEOR W4700

Homework 9

Linan Qiu
lq2137

November 30, 2015

Problem 1.

By put call parity,

$$C - P = S_0 e^{(-qT)} - K e^{(-rT)}$$

Then, given the price of the call, S_t and K and r and T ,

$$P = C - S_0 e^{(-qT)} + K e^{(-rT)} = 10 - 250 e^{(-0.04 * \frac{1}{r})} + 245 e^{(-0.06 * \frac{1}{4})} = 3.839967$$

Problem 2.

The adjusted stock price accounting for the dividend is

$$100 - 2e^{(-0.06 * \frac{1}{6})} = 98.0199$$

Evolving the stock price using a binomial tree as follows:

```
1 build_stock_tree = function(S, sigma, delta_t, N) {  
2   tree = matrix(0, nrow=N+1, ncol=N+1)  
3  
4   u = exp(sigma*sqrt(delta_t))  
5   d = exp(-sigma*sqrt(delta_t))  
6  
7   for (i in 1:(N+1)) {  
8     for (j in 1:i) {  
9       tree[i,j] = S * u^(j-1) * d^((i-1)-(j-1))  
}
```

```

10 }
11 }
12 return(tree)
13 }

```

Running the following code:

```

1 > stock_tree = build_stock_tree(S=100 - 2*exp(-0.06/6), 0.3, 1/12, 4)
2 > stock_tree
3      [,1]      [,2]      [,3]      [,4]      [,5]
4 [1,] 98.01990  0.00000  0.00000  0.00000  0.00000
5 [2,] 89.88832 106.88709  0.00000  0.00000  0.00000
6 [3,] 82.43132  98.01990 116.5564  0.00000  0.00000
7 [4,] 75.59294  89.88832 106.8871 127.1005  0.00000
8 [5,] 69.32186  82.43132  98.0199 116.5564 138.5984

```

This is the stock tree after adjusting for dividends.

Now we have to add the discounted dividend back to the tree. The first row gets $2e^{(-0.06*\frac{2}{12})}$ added, and the second row gets $2e^{(-0.06*\frac{1}{12})}$ back.

```

1 > stock_tree[1,1] = stock_tree[1,1] + 2*exp(-0.06*2/12)
2 > stock_tree
3      [,1]      [,2]      [,3]      [,4]      [,5]
4 [1,] 100.00000  0.00000  0.00000  0.00000  0.00000
5 [2,]  89.88832 106.88709  0.00000  0.00000  0.00000
6 [3,]  82.43132  98.01990 116.5564  0.00000  0.00000
7 [4,]  75.59294  89.88832 106.8871 127.1005  0.00000
8 [5,]  69.32186  82.43132  98.0199 116.5564 138.5984
9 > stock_tree[2,1] = stock_tree[2,1] + 2*exp(-0.06*1/12)
10 > stock_tree[2,2] = stock_tree[2,2] + 2*exp(-0.06*1/12)
11 > stock_tree
12      [,1]      [,2]      [,3]      [,4]      [,5]
13 [1,] 100.00000  0.00000  0.00000  0.00000  0.00000
14 [2,]  91.87834 108.87712  0.00000  0.00000  0.00000
15 [3,]  82.43132  98.01990 116.5564  0.00000  0.00000
16 [4,]  75.59294  89.88832 106.8871 127.1005  0.00000
17 [5,]  69.32186  82.43132  98.0199 116.5564 138.5984

```

The resulting binary tree is shown directly above.

Problem 3.

Let the value of the binary option at time 0 be B_0 .

$$B_0 = e^{(-rT)}E(\theta(S_T - K)) = e^{(-rT)}P(S_T \geq K)$$

Now the probability that $S_T \geq K$ can be found by looking at the distribution of $\log S$, since $P(S_T \geq K) = P(\log S_T \geq \log K)$.

We know that S_T is log-normally distributed, hence $\log S_T$ is normally distributed with the following statistics:

$$E(\log S_T) = \log S_0 + \left(\mu - \frac{1}{2}\sigma^2\right) T$$

$$\text{Var}(\log S_T) = \sigma^2 T$$

Then,

$$Q(S_T) = \frac{\log S_T - \log S_0 - \left(\mu - \frac{1}{2}\sigma^2\right) T}{\sigma\sqrt{T}} \sim \text{Norm}(0, 1)$$

Then,

$$\begin{aligned} P(S_T \geq K) &= P(\log S_T \geq \log K) \\ &= P\left(\frac{\log S_T - \log S_0 - \left(\mu - \frac{1}{2}\sigma^2\right) T}{\sigma\sqrt{T}} \geq \frac{\log K - \log S_0 - \left(\mu - \frac{1}{2}\sigma^2\right) T}{\sigma\sqrt{T}}\right) \\ &= 1 - P\left(\frac{\log S_T - \log S_0 - \left(\mu - \frac{1}{2}\sigma^2\right) T}{\sigma\sqrt{T}} < \frac{\log K - \log S_0 - \left(\mu - \frac{1}{2}\sigma^2\right) T}{\sigma\sqrt{T}}\right) \\ &= 1 - N(Q(K)) \\ &= N(-Q(K)) \end{aligned}$$

where N is the normal distribution function. Then the value B_0 is

$$B_0 = e^{(-rT)} N\left(\frac{\log S_0 - \log K + \left(r - \frac{1}{2}\sigma^2\right) T}{\sigma\sqrt{T}}\right) = e^{(-rT)} N(d_2)$$

where d_2 is as defined in Hull / slides. We know from risk neutral pricing that r is the risk free rate.

Problem 4.

The payoffs from this option is perfectly replicated by a portfolio of:

- long 1 vanilla call option with strike K
- long K digital call option with strike K

This is shown by the payoffs: let A be the binary asset-or-nothing call.

$$\begin{aligned} A &= e^{(-rT)}E(S\theta(S-K)) \\ &= e^{(-rT)}E((S-K)\theta(S-K) + K\theta(S-K)) \\ &= e^{(-rT)}E(\max(S-K, 0)) + Ke^{(-rT)}E(\theta(S-K)) \\ &= C + KB \\ &= S_0N(d_1) - Ke^{(-rT)}N(d_2) + Ke^{(-rT)}N(d_2) \\ &= S_0N(d_1) \end{aligned}$$

where d_1 is as defined in Hull / slides.

Problem 5.

The European Call value using BSM is calculated as:

```
1 d1 = function(S, K, r, sigma, T) {  
2   return((log(S/K) + (r+(sigma^2)/2)*T)/(sigma*sqrt(T)))  
3 }  
4  
5 d2 = function(S, K, r, sigma, T) {  
6   return(d1(S, K, r, sigma, T) - sigma*sqrt(T))  
7 }  
8  
9 bsm_call = function(sigma, K, S, r, T) {  
10  d1_val = d1(S, K, r, sigma, T)  
11  d2_val = d2(S, K, r, sigma, T)  
12  return(list(price=S*pnorm(d1_val) - K*exp(-r*T)*pnorm(d2_val), delta=  
13    pnorm(d1_val)))  
}
```

```

1 > bsm_call(sigma=0.1, T=1, r=0.08, K=51, S=50*exp(-0.1))
2 $price
3 [1] 1.066008
4
5 $delta
6 [1] 0.3639102

```

For an American option, let the dividend yield be d and the risk free probability be q .

$$qS_0u + (1 - q)S_0d = S_0e^{((r-d)\Delta t)}$$

Setting q to be the subject,

$$q = \frac{e^{((r-d)\Delta t)} - d}{u - d}$$

In other words, all we have to do to account for the continuous dividend yield is to set q to the new quantity.

Then, we modify the binomial program we used in the previous homework. The stock tree building procedure remains the same:

```

1 build_stock_tree = function(S, sigma, T, N) {
2   delta_t = T/N
3   tree = matrix(0, nrow=N+1, ncol=N+1)
4
5   u = exp(sigma*sqrt(delta_t))
6   d = exp(-sigma*sqrt(delta_t))
7
8   for (i in 1:(N+1)) {
9     for (j in 1:i) {
10      tree[i,j] = S * u^(j-1) * d^((i-1)-(j-1))
11    }
12  }
13  return(tree)
14 }

```

Particularly, q calculation now accounts for dividends.

```

1 q_prob = function(r, delta_t, sigma, div=0) {
2   u = exp(sigma*sqrt(delta_t))
3   d = exp(-sigma*sqrt(delta_t))
4
5   return((exp((r-div)*delta_t) - d)/(u-d))
6 }

```

As for building the option tree, I chose to use 4 trees (matrices) instead of a proper data structures (which I'd have used in any other language with proper data structures and

object orientedness like Python or Java or even JavaScript). Unfortunately, I am using R and am too lazy to switch languages.

```
1  delta_t = T / N
2  q = q_prob(r, delta_t, sigma, div=div)
3
4  underlying_tree = tree
5  value_ne_tree = matrix(0, nrow=nrow(tree), ncol=ncol(tree))
6  value_e_tree = matrix(0, nrow=nrow(tree), ncol=ncol(tree))
7  option_value_tree = matrix(0, nrow=nrow(tree), ncol=ncol(tree))
8
9  ## ONLY CALL OPTIONS NOW
10
11 # set not exercise value to 0
12 value_ne_tree[nrow(tree),] = 0
13 # set exercise value to S - X
14 value_e_tree[nrow(tree),] = tree[nrow(tree), ] - X
15 # set option value to whichever is higher
16 option_value_tree[nrow(tree),] = pmax(value_ne_tree[nrow(tree),], value
    _e_tree[nrow(tree),])
17
18
19 for (i in (nrow(tree)-1):1) {
20   for(j in 1:i) {
21     # not exercise value is probability weighted discounted
22     value_ne_tree[i, j] = ((1-q)*option_value_tree[i+1,j] + q*option_
        value_tree[i+1,j+1])/exp(r*delta_t)
23     # exercise value = S - X
24     value_e_tree[i, j] = tree[i, j] - X
25     # set option value to whichever is higher
26     option_value_tree[i, j] = max(value_ne_tree[i, j], value_e_tree[i,
        j])
27   }
28 }
29
30 delta = (option_value_tree[2, 2] - option_value_tree[2, 1])/(tree[2, 2]
    - tree[2, 1])
31
32 return(list(tree=tree, value_ne_tree=value_ne_tree, value_e_tree=value_
    e_tree, option_value_tree=option_value_tree, q=q, delta=delta))
33 }
```

Running the code using the following:

```
1 sigma=0.1
2 S=50
3 T=1
4 N=100
5 X=51
6 r=0.08
7 div=0.1
8
9 stock_tree = build_stock_tree(S=S, sigma=sigma, T=T, N=N)
```

```
10 option_tree = value_binomial_option(stock_tree, sigma=sigma, N=N, T=T, r=r,  
    X=X, type="call", div=div)
```

This gave the following results (compared with European BSM pricing):

- European Call using BSM: $P = 1.066008$, $\Delta = 0.3639102$
- American Call using Binomial Tree: $P = 1.157862$, $\Delta = 0.3698668$

The results are in the following pages:

Problem 5(a). Stock Tree

```

1 > round(option_tree$tree[1:11, 1:11], 2)
2      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
3 [1,] 50.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
4 [2,] 49.50 50.50  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
5 [3,] 49.01 50.00 51.01  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
6 [4,] 48.52 49.50 50.50 51.52  0.00  0.00  0.00  0.00  0.00  0.00  0.00
7 [5,] 48.04 49.01 50.00 51.01 52.04  0.00  0.00  0.00  0.00  0.00  0.00
8 [6,] 47.56 48.52 49.50 50.50 51.52 52.56  0.00  0.00  0.00  0.00  0.00
9 [7,] 47.09 48.04 49.01 50.00 51.01 52.04 53.09  0.00  0.00  0.00  0.00
10 [8,] 46.62 47.56 48.52 49.50 50.50 51.52 52.56 53.63  0.00  0.00  0.00
11 [9,] 46.16 47.09 48.04 49.01 50.00 51.01 52.04 53.09 54.16  0.00  0.00
12 [10,] 45.70 46.62 47.56 48.52 49.50 50.50 51.52 52.56 53.63 54.71  0.00
13 [11,] 45.24 46.16 47.09 48.04 49.01 50.00 51.01 52.04 53.09 54.16 55.26

```

∞

Problem 5(b). Not Exercise Value

```

1 > round(option_tree$value_ne_tree[1:11, 1:11], 2)
2      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
3 [1,] 1.16  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
4 [2,] 0.98  1.35  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
5 [3,] 0.82  1.15  1.56  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
6 [4,] 0.68  0.97  1.34  1.80  0.00  0.00  0.00  0.00  0.00  0.00  0.00
7 [5,] 0.56  0.81  1.13  1.55  2.07  0.00  0.00  0.00  0.00  0.00  0.00
8 [6,] 0.46  0.67  0.96  1.32  1.79  2.37  0.00  0.00  0.00  0.00  0.00
9 [7,] 0.37  0.55  0.80  1.12  1.54  2.06  2.70  0.00  0.00  0.00  0.00
10 [8,] 0.30  0.45  0.66  0.94  1.31  1.78  2.36  3.05  0.00  0.00  0.00
11 [9,] 0.23  0.36  0.54  0.79  1.11  1.53  2.05  2.69  3.45  0.00  0.00
12 [10,] 0.18  0.29  0.44  0.65  0.93  1.30  1.77  2.35  3.05  3.87  0.00
13 [11,] 0.14  0.23  0.35  0.53  0.78  1.10  1.52  2.04  2.68  3.44  4.34

```


Problem 5(c). Exercise Values

```

1 > round(option_tree$value_e_tree[1:11, 1:11], 2)
2      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
3 [1,] -1.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
4 [2,] -1.50 -0.50  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
5 [3,] -1.99 -1.00  0.01  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
6 [4,] -2.48 -1.50 -0.50  0.52  0.00  0.00  0.00  0.00  0.00  0.00  0.00
7 [5,] -2.96 -1.99 -1.00  0.01  1.04  0.00  0.00  0.00  0.00  0.00  0.00
8 [6,] -3.44 -2.48 -1.50 -0.50  0.52  1.56  0.00  0.00  0.00  0.00  0.00
9 [7,] -3.91 -2.96 -1.99 -1.00  0.01  1.04  2.09  0.00  0.00  0.00  0.00
10 [8,] -4.38 -3.44 -2.48 -1.50 -0.50  0.52  1.56  2.63  0.00  0.00  0.00
11 [9,] -4.84 -3.91 -2.96 -1.99 -1.00  0.01  1.04  2.09  3.16  0.00  0.00
12 [10,] -5.30 -4.38 -3.44 -2.48 -1.50 -0.50  0.52  1.56  2.63  3.71  0.00
13 [11,] -5.76 -4.84 -3.91 -2.96 -1.99 -1.00  0.01  1.04  2.09  3.16  4.26

```

6

Problem 5(d). Option Value

Option value is the max of exercise value or not exercise value at each stage.

```

1 > round(option_tree$option_value[1:11, 1:11], 2)
2      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
3 [1,]  1.16  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
4 [2,]  0.98  1.35  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
5 [3,]  0.82  1.15  1.56  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
6 [4,]  0.68  0.97  1.34  1.80  0.00  0.00  0.00  0.00  0.00  0.00  0.00
7 [5,]  0.56  0.81  1.13  1.55  2.07  0.00  0.00  0.00  0.00  0.00  0.00
8 [6,]  0.46  0.67  0.96  1.32  1.79  2.37  0.00  0.00  0.00  0.00  0.00
9 [7,]  0.37  0.55  0.80  1.12  1.54  2.06  2.70  0.00  0.00  0.00  0.00
10 [8,]  0.30  0.45  0.66  0.94  1.31  1.78  2.36  3.05  0.00  0.00  0.00
11 [9,]  0.23  0.36  0.54  0.79  1.11  1.53  2.05  2.69  3.45  0.00  0.00
12 [10,]  0.18  0.29  0.44  0.65  0.93  1.30  1.77  2.35  3.05  3.87  0.00
13 [11,]  0.14  0.23  0.35  0.53  0.78  1.10  1.52  2.04  2.68  3.44  4.34

```

Problem 6.

Problem 6(a).

The only part of the program we need to modify is:

```
1 value_binomial_option = function(tree, sigma, N, T, r, alpha=3, div=0) {
2   delta_t = T / N
3   q = q_prob(r, delta_t, sigma, div=div)
4
5   option_tree = matrix(0, nrow=nrow(tree), ncol=ncol(tree))
6
7   # option pays S^alpha at expiration
8   option_tree[nrow(tree),] = tree[nrow(tree), ]^alpha
9
10  for (i in (nrow(tree)-1):1) {
11    for(j in 1:i) {
12      option_tree[i, j] = ((1-q)*option_tree[i+1,j] + q*option_tree[i+1,j
13      +1])/exp(r*delta_t)
14    }
15  }
16
17  delta = (option_tree[2, 2] - option_tree[2, 1])/(tree[2, 2] - tree[2,
18  1])
19
20  return(list(tree=tree, option_tree=option_tree, price=option_tree[1,1],
21    q=q, delta=delta))
22 }
```

to account for the introduction of α and the different expiration price of power options.

Running this:

```
1 sigma=0.3
2 S=100
3 T=1/4
4 N=100
5 r=0.1
6 stock_tree = build_stock_tree(S=S, sigma=sigma, T=T, N=N)
7 option_tree = value_binomial_option(stock_tree, sigma=sigma, N=N, T=T, r=
  r, alpha=3)
```

We find that the price of the option is $P = 1124610.5$. We also obtain the following trees:

(i). Stock Tree

```

1 > round(option_tree$tree[1:11, 1:11], 2)
2      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
3 [1,] 100.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
4 [2,]  98.51 101.51  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
5 [3,]  97.04 100.00 103.05  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
6 [4,]  95.60  98.51 101.51 104.60  0.00  0.00  0.00  0.00  0.00  0.00  0.00
7 [5,]  94.18  97.04 100.00 103.05 106.18  0.00  0.00  0.00  0.00  0.00  0.00
8 [6,]  92.77  95.60  98.51 101.51 104.60 107.79  0.00  0.00  0.00  0.00  0.00
9 [7,]  91.39  94.18  97.04 100.00 103.05 106.18 109.42  0.00  0.00  0.00  0.00
10 [8,]  90.03  92.77  95.60  98.51 101.51 104.60 107.79 111.07  0.00  0.00  0.00
11 [9,]  88.69  91.39  94.18  97.04 100.00 103.05 106.18 109.42 112.75  0.00  0.00
12 [10,]  87.37  90.03  92.77  95.60  98.51 101.51 104.60 107.79 111.07 114.45  0.00
13 [11,]  86.07  88.69  91.39  94.18  97.04 100.00 103.05 106.18 109.42 112.75 116.18

```

11

(ii). Option Tree

```

1 > round(option_tree$option_tree[1:11, 1:11], 2)
2      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
3 [1,] 1124610.5  0.0  0.0  0.0  0  0  0  0  0  0  0
4 [2,] 1073862.9 1174993.2  0.0  0.0  0  0  0  0  0  0  0
5 [3,] 1025405.3 1121972.1 1227633.1  0.0  0  0  0  0  0  0  0
6 [4,]  979134.4 1071343.7 1172236.7 1282631.2  0  0  0  0  0  0  0
7 [5,]  934951.4 1022999.8 1119340.0 1224753.1 1340093  0  0  0  0  0  0
8 [6,]  892762.1  976837.3 1068830.3 1169486.6 1279622 1400130  0  0  0  0  0
9 [7,]  852476.6  932758.0 1020599.8 1116714.1 1221880 1336949 1462856  0  0  0  0
10 [8,]  814009.0  890667.7  974545.7 1066322.9 1166743 1276620 1396845 1528392  0  0  0
11 [9,]  777277.2  850476.7  930569.8 1018205.5 1114094 1219013 1333813 1459424 1596864  0  0
12 [10,]  742202.9  812099.4  888578.2  972259.4 1063821 1164006 1273625 1393568 1524806 1668404  0
13 [11,]  708711.3  775453.7  848481.5  928386.7 1015817 1111481 1216154 1330684 1456000 1593118 1743149

```

Problem 6(b).

Let P be the power option.

$$P = S^\alpha$$

$dS = \mu S dt + \sigma S dZ$. Recall Ito's lemma which states that

$$dP = \left(\frac{\delta P}{\delta S} \mu S + \frac{\delta P}{\delta t} + \frac{1}{2} \frac{\delta^2 P}{\delta S^2} \sigma^2 S^2 \right) dt + \frac{\delta P}{\delta S} \sigma S dZ$$

Now,

$$\begin{aligned} \frac{\delta P}{\delta S} &= \alpha S^{\alpha-1} \\ \frac{\delta^2 P}{\delta S^2} &= \alpha(\alpha-1) S^{\alpha-2} \\ \frac{\delta P}{\delta t} &= 0 \end{aligned}$$

Then,

$$\begin{aligned} dP &= \left(\mu \alpha S^\alpha + \frac{\alpha(\alpha-1)}{2} \sigma^2 S^\alpha \right) dt + \alpha \sigma S^\alpha dZ \\ &= \left(\mu \alpha P + \frac{\alpha(\alpha-1)}{2} \sigma^2 P \right) dt + \alpha \sigma P dZ \end{aligned}$$

$$\frac{dP}{P} = \left(\mu \alpha + \frac{\alpha(\alpha-1)}{2} \sigma^2 \right) dt + \alpha \sigma dZ$$

Then we know that $\frac{dP}{P}$ is a Wiener process with drift constant $\left(\mu \alpha + \frac{\alpha(\alpha-1)}{2} \sigma^2 \right)$ and variance $(\alpha \sigma)^2$.

Then, we know that the expected change between time 0 and time T is

$$E(P_T) = P_0 \exp \left[\left(\mu \alpha + \frac{\alpha(\alpha-1)}{2} \sigma^2 \right) T \right]$$

Let the value of the option be V .

$$V = e^{(-rT)}E(P_T) = e^{(-rT)}P_0 \exp \left[\left(\mu\alpha + \frac{\alpha(\alpha-1)}{2}\sigma^2 \right) T \right]$$

```

1 black_scholes_power = function(alpha, r, T, sigma, S) {
2   discount = exp(-r*T)
3   p0 = S^alpha
4   expectation = exp((r*alpha + (alpha * (alpha - 1)) / 2 * sigma^2)*T)
5   return(discount*p0*expectation)
6 }
7
8 black_scholes_power(alpha=3, r=0.1, T=0.25, sigma=0.3, S=100)

```

Now $P_0 = 100^3 = 1000000$, $r = 0.1$, $\mu = r = 0.1$ by risk neutral pricing, $T = 0.25$, $\alpha = 3$, $\sigma = 0.3$.

Then, $V = 1124682$