Prof. Dr. Natalie Packham
LV-Nr. 425093

Summer term 2019

Computational Finance and FinTech – Problem Set 5 with solutions

Exercise 1. Write a function that calculates the price of a contingent claim in a binomial tree. The function should return the option value at each node in the tree.

Solution 1.

```
def payoff(S):
    return max(S-160,0)

def calculate_price():
    X=np.zeros((T+1, T+1))
    qu=((1+r)-d)/(u-d)
    qd=(u-(1+r))/(u-d)
    z=T+1
    for k in range(z): # number of up moves
        X[k,T] = payoff(S0 * u**k * d**(T-k))
    for t in range(T-1, -1, -1):
        z = z - 1
        for k in range(z): # number of up moves
            X[k,t] = 1/(1+r) * (X[k+1,t+1] * qu + X[k,t+1] * qd)
    return np.flipud(X)
```

Exercise 2. Write a function that calculates the price of an option in the binomial tree model using risk-neutral expectation.
(Hint: Import the package `scipy.stats`.)

Solution 2.

```
import scipy.stats as scs

def payoff(S):
    return max(S-160,0)

def calculate_price():
    price=0;
    qu=((1+r)-d)/(u-d)
    for k in range(T+1):
        price = price + scs.binom.pmf(k,T,qu) * payoff(S0 * u**k * d**(T-k)
                                                       )
    price = 1/(1+r)**T * price
    return price
```

Exercise 3. Assume a CRR model with annual price jumps. Today's stock price is $S_0 = 100$ and assume an annualised volatility of 33%. Let $r = 5\%$ p.a.
Determine the price of a call option with strike $K = 105$ and maturity $T = 3$.

Solution 3.

```
import math
import scipy.stats as scs

S0=100
sigma=0.33
r=0.05
dt=1;
T=3;

def payoff(S):
    return max(S-105,0)

def calculate_price():
    u=math.exp(sigma*math.sqrt(dt));
    d=1/u
    qu=(math.exp(r*dt) - d) / (u - d)
    price=0
    N=T*dt
    for k in range(N+1):
        price = price + payoff(S0 * math.exp(sigma*math.sqrt(dt) * (2*k-N))
                                                    ) * scs.binom.pmf(k,N,qu)
    price = math.exp(-r*T) * price
    return price
```

The result is 27.968391481401202.

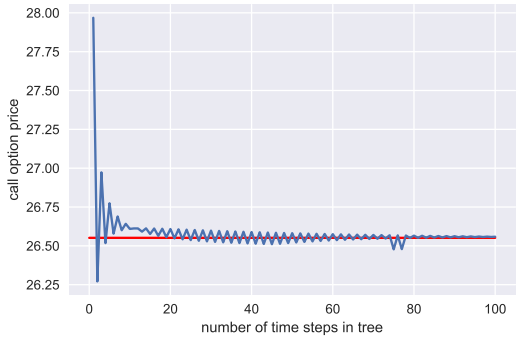
Exercise 4. Create a function that computes the price of a call option in the CRR model. Make the parameter Δt small, while keeping all other parameters fixed and show that the call price converges to the Black-Scholes price.

Solution 4.

```
import math
import scipy.stats as scs

S0=100
sigma=0.33
r=0.05
dt=1/252;
T=3;
K=105

def payoff(S):
    return max(S-K,0)

def calculate_price():
    u=math.exp(sigma*math.sqrt(dt));
    d=1/u
    qu=(math.exp(r*dt) - d) / (u-d)
    price=0
    N=int(T/dt)
    for k in range(N+1):
        price = price + payoff(S0 * math.exp(sigma*math.sqrt(dt) * (2*k-N))
                                                    ) * scs.binom.pmf(k,N,qu)
    price = math.exp(-r*T) * price
    return price
```

```
    def bs_price():
        d1 = (math.log(S0/K) + (r+0.5*sigma**2)*T) / (sigma*math.sqrt(T))
        d2 = (math.log(S0/K) + (r-0.5*sigma**2)*T) / (sigma*math.sqrt(T))
        return S0 * scs.norm.cdf(d1) - math.exp(-r*T) * K * scs.norm.cdf(d2)
```

The CRR price is 26.551828242079832, the Black-Scholes prices is 26.55157750961905.
The figure shows how the CRR price (blue) converges to the Black-Scholes price (red):



Exercise 5. Create a plot of the payoff of a call option as a function of the stock price S_T at expiry and the corresponding call option price.
The parameters are: $S_0 = 100$, $\sigma = 0.25$, $r = 0.05$, $T = 1$, $K = 100$.

Solution 5.

```
import scipy as sp
import scipy.stats as scs

sigma=0.25;
r=0.05;
T=1;
K=100;

def bs_price(S0):
    d1 = (sp.log(S0/K) + (r+0.5*sigma**2)*T) / (sigma*math.sqrt(T))
    d2 = (sp.log(S0/K) + (r-0.5*sigma**2)*T) / (sigma*math.sqrt(T))
    return S0 * scs.norm.cdf(d1) - math.exp(-r*T) * K * scs.norm.cdf(d2)

x=np.linspace(50,150,100)
plt.plot(x, (x>K) * (x-K));
plt.plot(x, bs_price(x));
plt.xlabel('$S_T$')
plt.title('Payoff vs. option price');
```

The output looks like this: