IEOR 221 Homework 8 part 2

Jannin Loïc

November 9, 2023

Introduction:

We want to price an American call option with the following parameters: $S_0 = 100.0$, $K = 100.0$, $T = 1.0$ year , $r = 6\%$, $q = 6\%$, $\sigma = 35\%$
To do so We will use a 100 step CRR binomial tree.
We have the following results:

1. Call option price : \$13.2413
2. $\rho = 33.9585$
3. $\Delta = 0.5481$
4. $\Gamma = 0.0112$
5. $\theta = -6.0564$
6. $\nu = 37.4531$

We used the following code :

```
import numpy as np
from math import log, sqrt, exp
from scipy.stats import norm

def option_payoff(S, K, option_type):
    payoff = 0.0
    if option_type == "call":
        payoff = max(S - K, 0)
    elif option_type == "put":
        payoff = max(K - S, 0)
    return payoff


def US_pricing(sigma,r):

    # initialisation of parameters
    S0 = 100
    K = 100
```

```
    T = 1.0
    r = r
    q = 0.06
    sigma = sigma
    option_type = "call"
    n_steps = 100  # number of steps
    dt = T/n_steps
    sqrt_dt = np.sqrt(dt)
    u = exp(sigma*sqrt(dt))
    d = 1/u
    p =(exp((r-q)*dt)-d)/(u-d)

    # Creates a matix with the payoffs
    payoffs = np.zeros((n_steps+1,n_steps+1))

    # value at time T
    for j in range(n_steps+1):
        stock_price = S0 * u**(j) * d**(n_steps-j)
        payoffs[n_steps,j] =
            option_payoff(stock_price,K,option_type)


    # Back propagation
    for i in range(n_steps-1, -1, -1):
        for j in range(0,i+1):
            early_exercise = S0 *(u**j)*(d**(i-j)) - K
            eu_price =
                exp(-r*dt)*(p*payoffs[i+1][j+1]+(1-p)*payoffs[i+1][j])
            payoffs[i,j] = max( early_exercise,eu_price )


    print(f"The price of the American call option is
        approximately: {payoffs[0,0]:.2f}")

    # Now we compute the greeks given the class formulas:
    delta = (payoffs[1,1]-payoffs[1,0])/(S0*u-S0*d)
    print(f"Delta = : {delta:.4f}")

    delta_2 = (payoffs[2,2]-payoffs[2,1])/(S0*u*u-S0)
    delta_1 = (payoffs[2,1]-payoffs[2,0])/(S0-S0*d*d)
    gamma = (delta_2-delta_1)/(S0*u-S0*d)
    print(f"Gamma = : {gamma:.4f}")

    theta = (payoffs[2,1]-payoffs[0,0])/(2*dt)
    print(f"Theta = : {theta:.4f}")

    return payoffs


sigma = 0.35
```

```
r = 0.06
US_pricing(sigma,r)

dsigma = 0.001
dr = 0.0001
vega =
    (US_pricing(sigma+dsigma,r)[0,0]-US_pricing(sigma,r)[0,0])/dsigma
print(f"Vega = : {vega:.4f}")
ro =
    (US_pricing(sigma,r+dr)[0,0]-US_pricing(sigma,r)[0,0])/dr
print(f"ro = : {ro:.4f}")#
```