**6233 Project Report**

**Topic: Barrier Options with Rebates**

**(up-and-out put with rebates)**

**Ziming Mao**

**Jinyang Li**

**Instructor: Roza Galeeva**

**Dec.17th 2019**

**Content**

**Introduction**: 3

1. **Set up** 3

2. **Problem and Solution (Theoretical Formula)** 3

2.1 Price the up-and-out put option with rebate 3

2.2 Calculate analytically Delta 8

3. **Formula Check** 10

3.1 Check the Option Pricing Formula using Monte Carlo 10

3.1.1 Obtain the CDF of standard normal distribution 10

3.1.2 Pricing Options using the deducted formula: 11

3.1.3 Conduct Monte Carl to check the formula of barrier with rebate 12

3.2 Compare analytical delta with numerical delta 13

4. **Implementation Practically in Python** 14

4.1 Calculate the value of the option 14

4.2 Determine the rebate 15

4.3 Calculate and plot daily PnL 15

**Conclusion:** 17

# Introduction:

The report is organized as follows: In section 1, we illustrate the set up of our project. In section 2, we develop the pricing formula of the up-and-out barrier put option with rebate and calculate analytical delta. We show all the steps of the derivation, including PDE, appropriate boundary conditions, reflection principle and derivations of the formulas with standard BS. In section 3, we check our pricing formula and analytical delta. In section 4, we choose a specific stock and price the specific option for it. Then we determine the rebate which makes the barrier with rebate price = vanilla call. Additionally, we construct a portfolio, calculate and plot the daily PnL based on the portfolio.

# Set up

Consider a standard vanilla barrier option which becomes activated (“in”) or worthless (“out”), if a price hits a barrier X: either rises “up” or falls “down”. Consider the case when the holder of the option gets a fixed rebate R if a barrier option becomes dead: so either knocks out or never becomes active. Assume that this rebate (if it happens) will be paid at the option expiry T. Assumptions.

1. The asset price follows the GBM wit constant volatility.
2. The interest rate is constant and positive, no dividends.
3. To avoid trivialities, we assume that the initial asset price S is on the appropriate site of the barrier, so that “in” or “out” event does not happen right at the beginning (for example S > X for a down-and-out option).

# 2. Problem and Solution (Theoretical Formula)

## 2.1 Price the up-and-out put option with rebate

We decompose the portfolio into a standard up-and-out barrier and a digital (“in”) barrier which pays a rebate at expiry, if the option becomes “dead” (knocks out or never knocks in).

**Firstly, we price for the standard up-and-out barrier put option:**

Using the approach we have learned from class, it is easy to develop the Partial Differential Equation as follows:

coupled with the terminal condition:

As long as S is smaller than X (barrier), the value of the option satisfies the Black-Scholes-Merton equation with final condition:

As S becomes smaller, the likelihood of the barrier being activated become negligible, so with no dividends:

The valuation differs in that the second boundary condition is applied at . If S ever reaches X:

We define the new variables and new function and define:

=K

Then in these new variables the equation for becomes:

with the initial condition:

We simplify it further by a change of variable:

for some constants and to be found, we have:

We can eliminate terms with and by choosing:

we have:

Then we have:

Then the problem becomes:

with the boundary condition:

Using reflection principle and the method of images, we obtain:

Then we have

Thus,

is a vanilla put option and after integration, we obtain :

For , we replace by , and it is equivalent to replace for , we obtain:

We obtain the price for the standard up-and-out barrier put option:

where

**Secondly, we develop the formula of a digital (“in”) barrier which pays a rebate at expiry:**

The digital term of the pricing model is an America digital call option with payout at expiry. The PDE of American digital option is as follows:

We need a time-independent solution of equation. Trying a solution of the form shows that two independent solutions are and , where and are the positive and negative roots respectively of the quadratic:

The option price then worth the present value of payoff $1 when the asset reaches the strike:

Suppose that it is a call, which must be valued for

We have European digital options with same strike X

when , call vanishes, does not.

Thus we have

Now we deduce the formula of European digital call:

where

Similarly, European Digital Put Option price should be

where

The payment is not made until expiry, the option value on the barrier is the present value of the rebate:

So the value of this pay-at-expiry American digital call is

Because of the definition of barrier option with rebate, we can replicate it by two parts: the first part has the appropriate payoff, but vanishes on the barrier; the second part has zero payoff but is equal to R at expiry as an American digital option which make payout at expiry.

**Thirdly, we add the standard up-and-out barrier put option and the digital barrier together and we obtain the formula of up-and-out barrier put with rebate :**

***+***

where

## 2.2 Calculate analytically Delta

*+*

Then,

Next,

()

where,

**

Where

()

=()

Thus,

where,



Above all,

***()()*+*-+***

# 3. Formula Check

In this section, we will check our formula by putting it in test. In order to check the price of barrier put option with rebate, we conduct Monte Carlo simulation and compare two prices. In order to check our deducted delta, we compare our analytical delta with numerical delta calculated by the given formula.

## 3.1 Check the Option Pricing Formula using Monte Carlo

Codes and Logic:

### 3.1.1 Obtain the CDF of standard normal distribution

**from** **math** **import** exp, log, pi,sqrt

**def** norm\_pdf(x):

**return** (1.0/((2\*pi)\*\*0.5))\*exp(-0.5\*x\*x)

**def** norm\_cdf(x):

k = 1.0/(1.0+0.2316419\*x)

k\_sum = k \* (0.319381530 + k \* (-0.356563782 + \

k \* (1.781477937 + k \* (-1.821255978 + 1.330274429 \* k))))

**if** x >= 0.0:

**return** (1.0-(1.0 /((2 \* pi)\*\*0.5)) \* exp(-0.5 \* x \* x) \* k\_sum)

**else**:

**return** 1.0 - norm\_cdf(-x)

### 3.1.2 Pricing Options using the deducted formula:

vanilla call, vanilla put, barrier without rebate, barrier with rebate

*# Define d1 and d2 in normal distribution. Here, d1 and d2 are for vanilla call/put.*

*# When i=1, di=d1, when i=2, di=d2*

**def** di(i,r,S,K,sigma,T):

**return** (log(S/K) + (r+(-1)\*\*(i-1)\*0.5\*sigma\*sigma)\*T)/(sigma\*(T\*\*0.5))

*# Here, d\_1 and d\_2 are for the barrier term of the option.*

*# Barrier term is the part of the up\_and\_out put without rebate exclude the vanilla put part*

*# When i=1, d\_i=d\_1, when i=2, d\_i=d\_2*

**def** d\_i(i,r,S,K,sigma,T,H):

**return** (log(H\*H/(K\*S)) + (r+(-1)\*\*(i-1)\*0.5\*sigma\*sigma)\*T)/(sigma\*(T\*\*0.5))

*# Calculat vanilla put option*

**def** vanilla\_put(r,S,K,sigma,T):

d1=di(1,r,S,K,sigma,T)

d2=di(2,r,S,K,sigma,T)

I1=K\*exp(-r\*T)\*norm\_cdf(-d2)

I2=S\*norm\_cdf(-d1)

**return** I1-I2

*# Calculat vanilla call option*

**def** vanilla\_call(r,S,K,sigma,T):

d1=di(1,r,S,K,sigma,T)

d2=di(2,r,S,K,sigma,T)

I1=S\*norm\_cdf(d1)

I2=K\*exp(-r\*T)\*norm\_cdf(d2)

**return** I1-I2

*# Calculate the barrier term of up\_and\_out put*

**def** barrier\_term(r,S,K,sigma,T,H):

d\_1=d\_i(1,r,S,K,sigma,T,H)

d\_2=d\_i(2,r,S,K,sigma,T,H)

k=2\*r/(sigma\*sigma)

I1=(S/H)\*\*(1-k)\*K\*exp(-r\*T)\*norm\_cdf(-d\_2)

I2=(S/H)\*\*(1-k)\*(H\*H/S)\*norm\_cdf(-d\_1)

**return** I1-I2

*# Obtain the up\_and\_out put without rebate by adding the vanilla put and barrier term*

**def** barrier\_up\_out\_put(r,S,K,sigma,T,H):

**return** vanilla\_put(r,S,K,sigma,T)-barrier\_term(r,S,K,sigma,T,H)

*# Calculate the Americam digital call (this term is for rebate)*

**def** digital\_term(r,S,K,sigma,T,H):

d2=(log(S/H) + (r-0.5\*sigma\*sigma)\*T)/(sigma\*(T\*\*0.5))

d2\_=(log(H/S) + (r-0.5\*sigma\*sigma)\*T)/(sigma\*(T\*\*0.5))

Cd=exp(-r\*T)\*norm\_cdf(d2)

k=2\*r/(sigma\*sigma)

a=0.5\*(1-k)

Pd=(S/H)\*\*(2\*a)\*exp(-r\*T)\*norm\_cdf(-d2\_)

**return** Cd+Pd

*#The final formula: barrier with rebate=barrier without rebate + digital option*

*# Here, we assume R=10*

**def** barrier\_rebate\_put(r,S,K,sigma,T,H):

**return** barrier\_up\_out\_put(r,S,K,sigma,T,H)+10\*digital\_term(r,S,K,sigma,T,H)

### 3.1.3 Conduct Monte Carl to check the formula of barrier with rebate

Here, we assume: sigma=0.25, T=1, S0=100, r=0.02, K=90, H=110, R=10

*# Simulate stock price*

**import** **numpy** **as** **np**

**def** simstockprice(sigma,T,S0,r):

stockprice = []

stockprice.append(S0)

**for** i **in** range(1,int(252\*T)):

stockprice.append(stockprice[-1]\*np.exp(r/252-0.5\*(sigma/np.sqrt(252))\*\*2+sigma/np.sqrt(252)\*np.random.randn()))

**return** stockprice

*#Pay-off function of barrier with rebate. Similarly, we assume rebate=10 here*

**def** barrier(stockprice,K,H,r,T):

**for** i **in** stockprice:

**if** i>=H:

**return** 10

**return** max(K-stockprice[-1],0)\*np.exp(-r\*T)

*# 10000 steps*

*#Here, we assume: sigma=0.25, T=1, S0=100, r=0.02, K=90, H=110*

barrierprice = []

**for** i **in** range(10000):

stockprice = simstockprice(0.25,1,100,0.02)

barrierprice.append(barrier(stockprice,90,110,0.02,1))

*#Calculate option price by Monte Carl*

print("Option price by Monte Carl: "+str(np.mean(barrierprice)))

*#Calculate option price by deducted formula*

print("Option price by deducted formula: "+str(barrier\_rebate\_put(0.02,100,90,0.25,1,110)))

The result shows that:

Option price by Monte Carl: 10.146191365730257

Option price by deducted formula: 10.167957723338581

The error is quite small. Thus, our formula has been proved.

## 3.2 Compare analytical delta with numerical delta

Here, we assume: sigma=0.25, T=6, S0=100, r=0.03, K=90, H=110, R=1

*# Calculate analytical delata based on our deducted formula*

**def** delta\_analytical(r,S,K,sigma,T,H,R):

d2=(log(S/H) + (r-0.5\*sigma\*sigma)\*T)/(sigma\*(T\*\*0.5))

delta1=R\*np.exp(-r\*T)/np.sqrt(2\*pi)\*np.exp(-d2\*\*2/2)/(S\*sigma\*np.sqrt(T))

k=2\*r/(sigma\*sigma)

a=0.5\*(1-k)

d2\_=(log(H/S) + (r-0.5\*sigma\*sigma)\*T)/(sigma\*(T\*\*0.5))

delta2=R\*2\*a\*(S/H)\*\*(2\*a-1)\*np.exp(-r\*T)\*norm\_cdf(-d2\_)

delta3=R\*(S/H)\*\*(2\*a)\*np.exp(-r\*T)/np.sqrt(2\*pi)\*np.exp(-d2\_\*\*2/2)/(S\*sigma\*np.sqrt(T))

d12=(log(S/K) + (r+0.5\*sigma\*sigma)\*T)/(sigma\*(T\*\*0.5))

delta4=norm\_cdf(d12)-1

d11=(log(H\*\*2/(S\*K)) + (r+0.5\*sigma\*sigma)\*T)/(sigma\*(T\*\*0.5))

d21=(log(H\*\*2/(S\*K)) + (r-0.5\*sigma\*sigma)\*T)/(sigma\*(T\*\*0.5))

delta5=-(k-1)\*(S/H)\*\*(-k)\*(K\*np.exp(-r\*T)\*norm\_cdf(-d21)-H\*\*2/S\*norm\_cdf(-d11))

delta6=(S/H)\*\*(-k+1)\*(norm\_cdf(d11)-1)

**return** delta1+delta2+delta3+delta4-delta5-delta6

*# The price of barrier with rebate, here we assume rebate R=1*

**def** barrier\_rebate1\_put(r,S,K,sigma,T,H):

**return** barrier\_up\_out\_put(r,S,K,sigma,T,H)+digital\_term(r,S,K,sigma,T,H)

*# Compare analytical delta with numerical delta*

analytical\_delta=delta\_analytical(0.03,100,90,0.25,6,110,1)

print(analytical\_delta)

delta\_num=(barrier\_rebate1\_put(0.03,100.1,90,0.25,6,110)-barrier\_rebate1\_put(0.03,99.9,90,0.25,6,110))/0.2

print(delta\_num)

*#Calculate the rate of error*

error\_rate=(delta\_num-analytical\_delta)/analytical\_delta

print(error\_rate)

The result show that:

Analytical Delta: -0.3658968356327496

Numerical Delta: -0.37987401999560966

Error Rate: 0.0381997956847294

The error is quite small. Thus, our analytical delta has been proved.

# 4. Implementation Practically in Python

In this section, we implement our pricing formula practically. We choose Goldman Sachs Stock from 2018.12.3 to 2019.12.13 and price the barrier put option with rebate for it. Then we compare the price with vanilla call and determine what is the rebate which makes the barrier with rebate price = vanilla call. After that, we construct a portfolio: 1000 contracts of the specified option and short position in stock (hedge) with position= of the option. Lastly, we use one last year of historical prices to calculate daily PnL of the portfolio and plot PnL.

We omit the codes for importing data and setting values here. Specifically, we set the strike price , (2018.12-2019.12) and the barrier , (2017.12-2018.12)

## 4.1 Calculate the value of the option

Including vanilla call, vanilla put, barrier without rebate, barrier with rebate.

Here, we assume: r=0.03, T=2, S0=stock price at Dec.3, 2018

*# The initial stock price on Dec.3 2018*

S0=df['Adj Close'][0]

*#Price the option on Dec.3 2018, including vanilla call, vanilla put, barrier without rebate, barrier with rebate*

Vanilla\_Call=vanilla\_call(0.03,S0,K,sigma,2)

Vanilla\_Put=vanilla\_put(0.03,S0,K,sigma,2)

Barrier\_Without\_Rebate=barrier\_up\_out\_put(0.03,S0,K,sigma,2,H)

Barrier\_With\_Rebate=barrier\_rebate\_put(0.03,S0,K,sigma,2,H)

print("Vanilla Call: "+str(Vanilla\_Call))

print("Vanilla Put: "+str(Vanilla\_Put))

print("Barrier Without Rebate: "+str(Barrier\_Without\_Rebate)) *# R=10*

print("Barrier With Rebate: "+str(Barrier\_With\_Rebate))

The result show that:

Vanilla Call: 30.61407272446901

Vanilla Put: 21.433112122556096

Barrier Without Rebate: 20.83231174350972

Barrier With Rebate: 23.927309607868594

## 4.2 Determine the rebate

R=(vanilla\_put(0.03,S0,K,sigma,2)-barrier\_up\_out\_put(0.03,S0,K,sigma,2,H))/digital\_term(0.03,S0,K,sigma,2,H)

print("Rebate: "+str(R))

The result show that:

Rebate: 1.9411980407645002

This is the rebate which makes the barrier with rebate price = vanilla call.

## 4.3 Calculate and plot daily PnL

Numerical Delta: =

Daily PnL:

**import** **matplotlib.pyplot** **as** **plt**

*#Store the stock price in a list (one year)*

S=[]

**for** i **in** range(0,date):

S.append(df['Adj Close'][i])

*#Store the barrier put with debate price in a list (one year)*

V=[]

**for** i **in** range(0,date):

V.append(barrier\_rebate\_put(0.03,S[i],K,sigma,2,H))

*# Construct option price with a slight increment regarding stock price*

V1=[]

**for** i **in** range(0,date):

V1.append(barrier\_rebate\_put(0.03,S[i]+0.1,K,sigma,2,H))

V2=[]

**for** i **in** range(0,date):

V2.append(barrier\_rebate\_put(0.03,S[i]-0.1,K,sigma,2,H))

*#Calculate the numerical delta using the given formula*

Delta=[]

**for** i **in** range(0,date):

Delta.append((V1[i]-V2[i])/0.2)

*# Calculate daily PnL and store them in a list*

PnL=[]

PnL.append(0)

**for** i **in** range(1,date):

pnl\_i=1000\*(V[i]-V[i-1]-Delta[i-1]\*(S[i]-S[i-1]))

PnL.append(pnl\_i)

*# Plot the PnL*

fig = plt.figure()

ax = fig.add\_subplot(111)

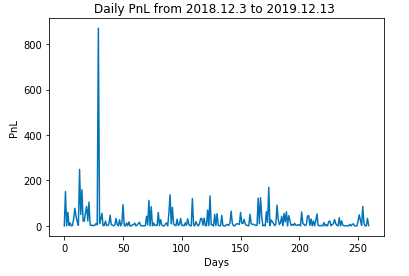
ax.set( title='Daily PnL from 2018.12.3 to 2019.12.13 ',

ylabel='PnL', xlabel='Days')

ax.plot(PnL)

plt.show()

The result:



# Conclusion:

**In our project, we first price the value of up-and-put barrier option with rebate. The formula is as follows:**

***+***

where

**Then, we deduct the value of analytical delta for this option. The formula is as follows:**

***()()*+*-+***

where

**Then, we check our formula in test.**

We conduct Monte Carl simulation to check our option price and we compare our analytical delta with the numerical delta given. The result shows that the differences of the two values are quite small, which proves our formula.

**After that, we choose Goldman Sachs stock from 2018.12.3 to 2019.12.13 to implement our formula.**

The result show that on Dec.3 2018:

Vanilla Call: 30.61407272446901

Vanilla Put: 21.433112122556096

Barrier Without Rebate: 20.83231174350972

Barrier With Rebate: 23.927309607868594

If we set Rebate= 1.9411980407645002, the barrier with rebate price = vanilla call.

**Finally, we calculate daily PnL and plot it based on the given portfolio.**

The result is:

