Análise Monte Carlo para Opções

February 13, 2021

[1]:
```python
import numpy as np
import pandas as pd
import yfinance
import matplotlib.pyplot as plt
```

[2]:
```python
data = yfinance.download(
    tickers = 'VALE3.SA',
    start = '2018-08-08',
    end = '2021-02-05',
    interval = '1d',
    treads = False
)
print(data)
```

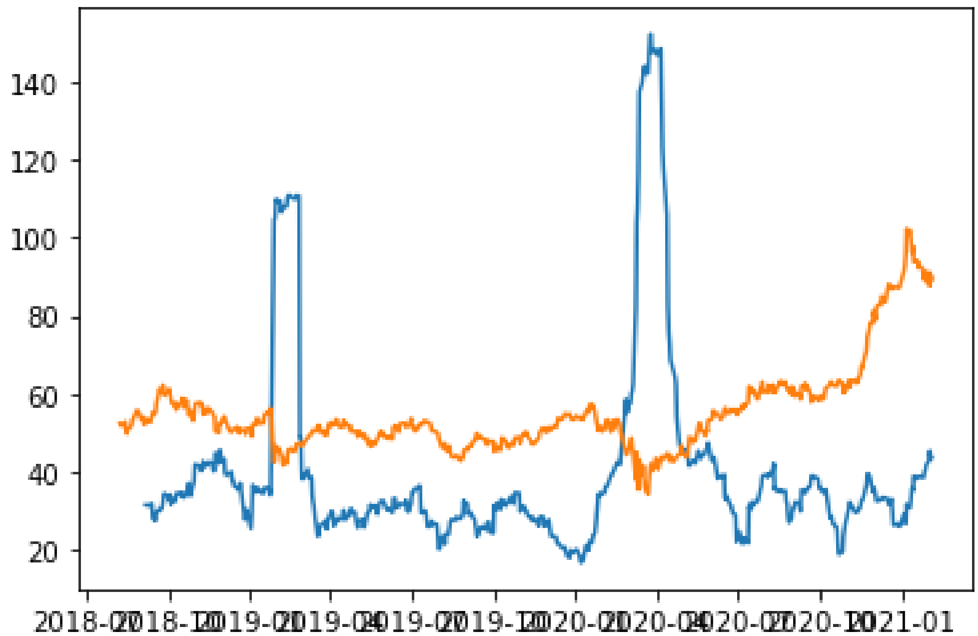
```
[*********************100%***********************]  1 of 1 completed
                 Open       High        Low      Close  Adj Close    Volume
Date
2018-08-08  53.299999  53.330002  52.259998  52.400002  48.999050  13076000
2018-08-09  52.500000  53.040001  51.730000  51.840000  48.475391  12752800
2018-08-10  51.410000  51.910000  50.720001  51.709999  48.353828  14630800
2018-08-13  52.070000  52.689999  51.740002  52.279999  48.886833  14720600
2018-08-14  52.099998  52.910000  51.770000  52.759998  49.335682  17475600
…                 …          …          …        …          …         …
2021-01-29  89.989998  90.849998  87.830002  88.669998  88.669998  23234000
2021-02-01  90.010002  91.269997  88.580002  91.269997  91.269997  27135500
2021-02-02  90.180000  90.580002  87.510002  87.660004  87.660004  34426700
2021-02-03  88.849998  91.419998  88.599998  90.430000  90.430000  34346400
2021-02-04  91.949997  92.870003  89.089996  89.290001  89.290001  29939700

[613 rows x 6 columns]
```

[20]:
```python
data['LogReturn'] = np.log(data['Close'] / data['Close'].shift(1)) # Retorno␣
 ↪Logarizado
data['Volatility'] = data['LogReturn'].rolling(21).std()*np.sqrt(252)
```

[19]:
```python
plt.plot(data['Volatility']*100)
plt.plot(data['Close'])
```

```
plt.show()
```



0.0.1 Exemplo de opções VALEB912

https://opcoes.net.br/opcoes2/bovespa/

Strike = 91,24 faltando 8 dias para o vencimento

[21]:
```
S0 = 92.69   # Valor da ação
K = 91.24    # Strike da opção
T = 8/252    # dias que faltam para o vencimento
r = 0.02     # taxa livre de juros
sigma = data['Volatility'][-1]
I = 1000000 # Número de simulações
```

0.0.2 The Equation

$S_T = S_0 exp((r - 1/2\sigma^2)T + \sigma\sqrt{T}z$

[22]:
```
z = np.random.standard_normal(I)

ST = S0 * np.exp((r - 0.5 * sigma ** 2) * T + sigma * np.sqrt(T) * z)
hT = np.maximum(ST - K, 0) # valores internos na maturidade
C0 = np.exp(-r * T) * np.sum(hT)/I # Estimação de Monte Calor

# Result Output
print("Valor teórico da Opção Européia %5.3f" % C0)
```

```
Valor teórico da Opção Européia 3.659
```

0.1 Comparando com Black-Scholes

[23]:
```python
import scipy.stats as si
import sympy as sy
from sympy.stats import Normal, cdf
from sympy import init_printing
init_printing()
```

[24]:
```python
def euro_call(S0, K, T, r, sigma):

    d1 = (np.log(S0/K) + (r + 0.5* sigma **2)*T) / (sigma*np.sqrt(T))
    d2 = (np.log(S0 / K) + (r - 0.5 * sigma ** 2) * T) / (sigma * np.sqrt(T))

    call = (S0 * si.norm.cdf(d1, 0.0, 1.0) - K * np.exp(-r * T) * si.norm.
 ↪cdf(d2, 0.0, 1.0))

    return call
```

[25]:
```python
BS = euro_call(92.69, 91.24, 8/252, 0.02, data['Volatility'][-1])
print("O valor teórico da Opção Européia é %5.3f" % BS)
```

```
O valor teórico da Opção Européia é 3.658
```