EstimationParamH

September 21, 2021

[7]:
```
                                                                            ␣
 ↪#EstimationParamètresH
import numpy as np
import math
import pandas as pd
import matplotlib.pyplot as plt
from MFDFA import MFDFA
from MFDFA import fgn
import datetime as dt
import matplotlib.pyplot as plt
import matplotlib.mlab as mlab
import yfinance as yf
import pandas_datareader.data as pdr
import requests
import zipfile as zi
from sklearn import datasets, linear_model
import scipy.special as scsp
import statsmodels.api as sm
import seaborn as sns; sns.set()
%matplotlib inline
import io as sio
```

[9]:
```
                                                                   #Etapes␣
 ↪Estimation du paramètre de Hurst H

                                                                            ␣
 ↪#Estimation du paramètre H pour un seul indice
df = pd.read_csv('oxfordmanrealizedvolatilityindices.csv')
volS = df[df["Symbol"] == ".FCHI"]["rv10"]
```

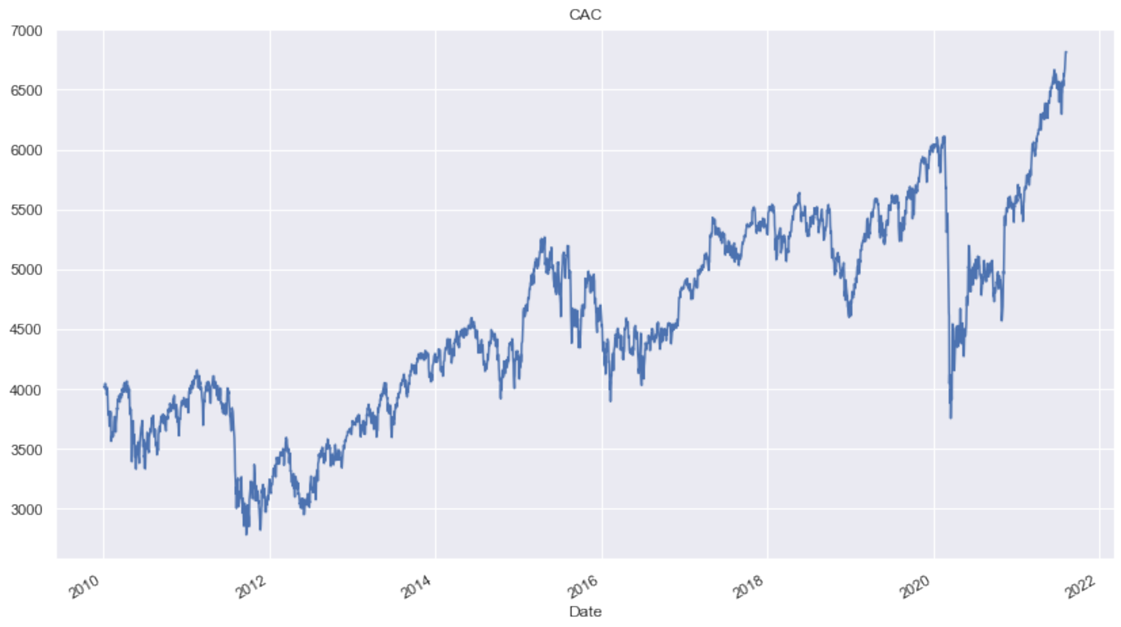
[10]:
```
##################################################################################
yf.pdr_override()
start_date = '01-01-2010'
end_date = '10-08-2021'

def download_data(symbol, source, start_date, end_date):
    start = dt.datetime.strptime(start_date, '%d-%m-%Y')
    end = dt.datetime.strptime(end_date, '%d-%m-%Y')
    df = pdr.get_data_yahoo(symbol, data_source=source, start=start, end=end)
    return df
```

[11]:
```
cours = download_data("^FCHI", "yahoo", start_date, end_date)
cours = cours['Adj Close']
cours.plot(title='CAC',figsize=(14, 8))
##########################################################################################
```

```
[*********************100%***********************]  1 of 1 completed
```

[11]:
```
<AxesSubplot:title={'center':'CAC'}, xlabel='Date'>
```



[12]:
```
dfVol = pd.DataFrame()
dfVol['sqrt']= np.sqrt(volS)
dfVol['log_sqrt'] = np.log(dfVol['sqrt'])

def del_Raw(q, x, df):
    return [np.mean(np.abs(df - df.shift(lag)) ** q) for lag in x]
```

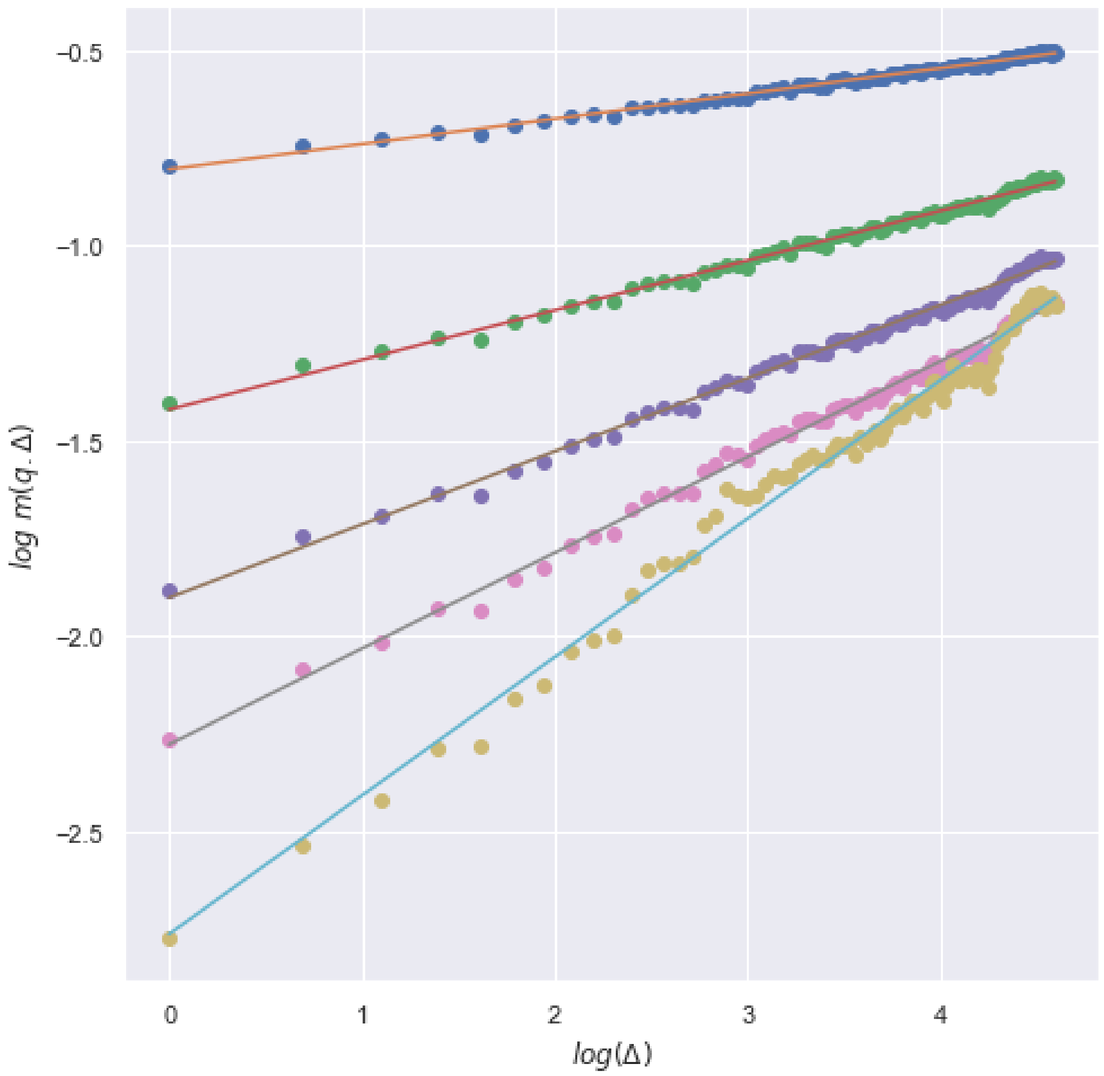
[13]:
```
fig = plt.figure(figsize=(8, 8))
ax = fig.add_axes([0.1,0.1,0.75,0.75])
ax.set_xlabel('$log(\Delta)$')
ax.set_ylabel('$log\  m(q.\Delta)$')
ax.ylim=(-3, -.5)

zeta_q = list()
qVec = np.array([.5, 1, 1.5, 2, 3])
```

```
x = np.arange(1, 100)
for q in qVec:
    ax.plot(np.log(x), np.log(del_Raw(q, x, dfVol['log_sqrt'])), 'o')
    model = np.polyfit(np.log(x), np.log(del_Raw(q, x, dfVol['log_sqrt'])), 1)
    ax.plot(np.log(x), np.log(x) * model[0] + model[1])
    zeta_q.append(model[0])

print (zeta_q)
fig.savefig('logM(q,delta)logdelta.png', dpi=300, bbox_inches='tight')
```

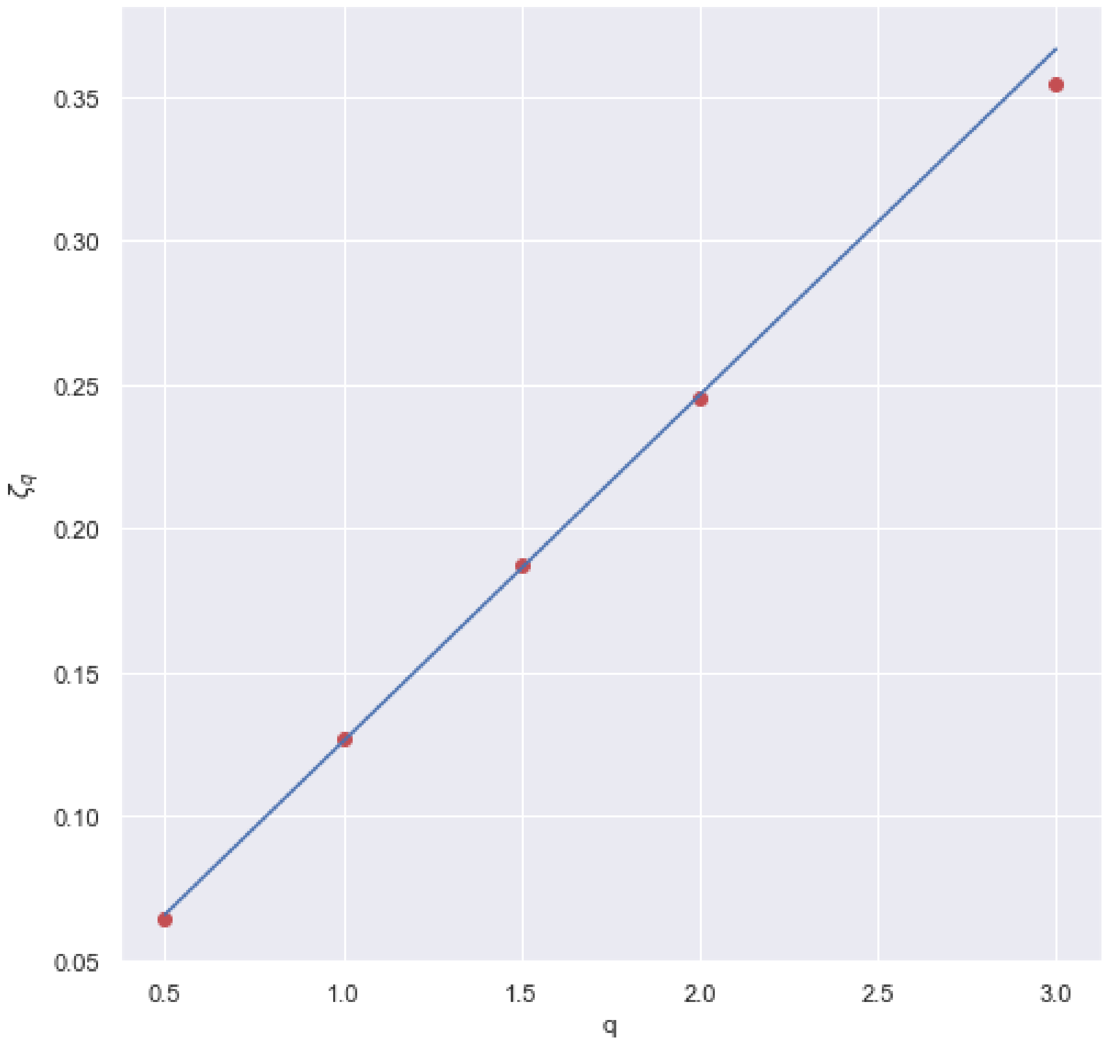
```
[0.06465910481822533, 0.12704699654087392, 0.1871885659766945,
0.24513591254023603, 0.3542774580370526]
```



[14]:
```
plt.figure(figsize=(8,8))
plt.xlabel('q')
plt.ylabel('$\zeta_{q}$')
plt.plot(qVec, zeta_q, 'or')

line = np.polyfit(qVec[:4], zeta_q[:4],1)
plt.plot(qVec, line[0] * qVec + line[1])
h_est= line[0]
print(h_est)
```

```
0.12031439852037054
```



[17]:
```
                                                                         ␣
 ↪#Estimation paramètre H pour tous les indices
df = pd.read_csv('oxfordmanrealizedvolatilityindices.csv')
```

```
indexes = df["Symbol"].value_counts().index
vol = pd.DataFrame()
for i in range (len(indexes)):
    s = df[df["Symbol"] == indexes[i]]["rv10"]
    vol = pd.concat([vol, s], axis=1)
vol.columns = indexes

def dlsig2(sic, x, pr=False):
    if pr:
        a= np.array([(sig-sig.shift(lag)).dropna() for lag in x])
        a=a ** 2
        print (a.info())
    return [np.mean((sig-sig.shift(lag)).dropna() ** 2) for lag in x]
```

```
[18]: h = list()
nu = list()

for col in vol.columns:
    sig = vol[col]
    sig = np.log(np.sqrt(sig))
    sig = sig.dropna()
    model = np.polyfit(np.log(x), np.log(dlsig2(sig, x)), 1)
    nu.append(np.sqrt(np.exp(model[1])))
    h.append(model[0]/2.)

est = pd.DataFrame({'Indices':vol.columns, 'Estimation du paramètre H': h,␣
 ↪'Estimation du paramètre nu': nu})
```

```
/Users/reda/opt/anaconda3/lib/python3.8/site-
packages/pandas/core/arraylike.py:364: RuntimeWarning: divide by zero
encountered in log
  result = getattr(ufunc, method)(*inputs, **kwargs)
```

```
[19]: est
estççç.dropna()
```

```
[19]:      Indices  Estimation du paramètre H  Estimation du paramètre nu
0        .FCHI                   0.122568                    0.320464
1         .AEX                   0.133145                    0.320414
2    .STOXX50E                   0.100548                    0.397587
3         .BFX                   0.125937                    0.306290
4        .IBEX                   0.112744                    0.314685
5       .GDAXI                   0.122359                    0.331459
6        .AORD                   0.071875                    0.459212
7        .FTSE                   0.109180                    0.365529
8         .MXX                   0.079091                    0.414202
9        .IXIC                   0.123072                    0.350766
10        .SPX                   0.131715                    0.381304
```

```
11       .RUT            0.110748            0.387228
12      .SSMI            0.149400            0.278083
13       .DJI            0.123347            0.390174
16      .KS11            0.102043            0.319863
17      .BVSP            0.127826            0.333823
18       .HSI            0.084487            0.327221
19       .KSE            0.099285            0.444501
20      .N225            0.105592            0.372272
21      .SSEC            0.105825            0.360319
22     .OSEAX            0.107394            0.369890
23    .GSPTSE            0.110107            0.391804
24      .SMSI            0.109951            0.335748
25    .OMXHPI            0.107753            0.356045
26    .OMXSPI            0.116433            0.354460
27    .OMXC20            0.098222            0.364126
29     .FTMIB            0.114010            0.345525
```

```
[ ]:
```