TP1 SERIES TEMPS

Réaliser par MEKA MOISE CHRISTIAN JUNIOR 21T2561

Il s'agit de l'implémentation du TP1 sur le dataset des données des bitcoin

In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:

```
FILEPATH="BTC-EUR.csv"
extension=FILEPATH.split(".")[-1]
SEP=","
if extension=="csv":
    df=pd.read_csv(FILEPATH, sep=SEP)
else:
    df=pd.read_excel(FILEPATH, index_col=0)
```

In [3]:

```
extension
```

Out[3]:

```
'csv'
```

In [4]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3490 entries, 0 to 3489
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Date       3490 non-null   object 
 1   Open       3490 non-null   float64
 2   High       3490 non-null   float64
 3   Low        3490 non-null   float64
 4   Close      3490 non-null   float64
 5   Adj Close  3490 non-null   float64
 6   Volume     3490 non-null   int64  
dtypes: float64(5), int64(1), object(1)
memory usage: 191.0+ KB
```

In [5]:

```
df.head()
```

Out[5]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	2014-09-17	359.546204	361.468506	351.586884	355.957367	355.957367	16389166
1	2014-09-18	355.588409	355.505402	319.789459	328.539368	328.539368	26691849
2	2014-09-19	328.278503	330.936707	298.921021	307.761139	307.761139	29560103
3	2014-09-20	307.665253	329.978180	303.931244	318.758972	318.758972	28736826
4	2014-09-21	318.120514	321.504517	306.502197	310.632446	310.632446	20702625

In [6]:

```
metrique="High"
period="Date"
```

In [7]:

```
df[metrique].describe()
```

Out[7]:

```
count     3490.000000
mean     14260.255169
std      15680.520917
min        183.047470
25%        955.831451
50%       7678.308106
75%      24935.772949
max      67416.492188
Name: High, dtype: float64
```

In [8]:

```
df=df.dropna()
```

Question 1

In [9]:

```
#moyenne
np.mean(df[metrique])
```

Out[9]:

```
14260.25516856132
```

In [10]:

```
#varaince
np.std(df[metrique])**2
```

Out[10]:

```
245808283.8535156
```

In [11]:

```
#ecart-type
np.std(df[metrique])
```

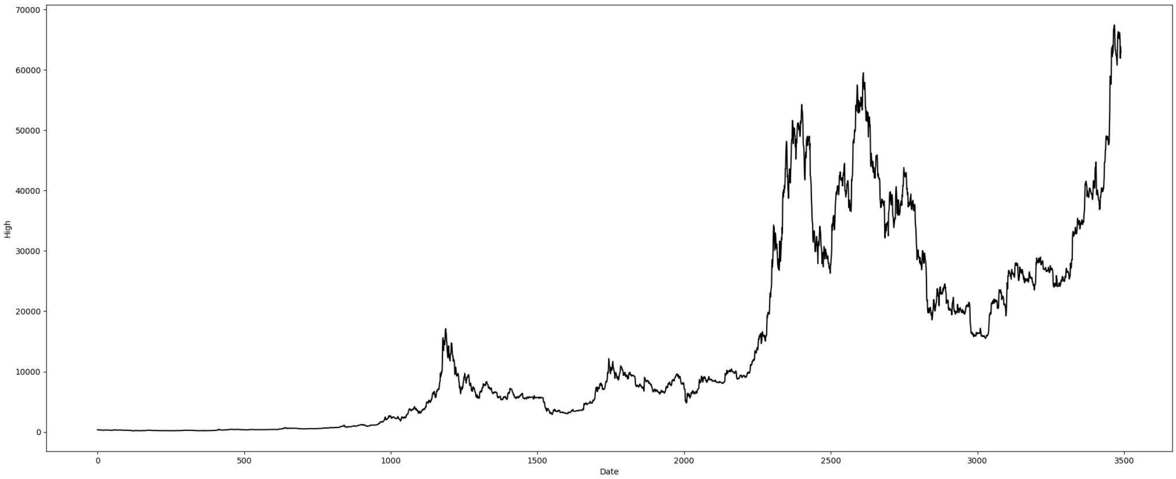
Out[11]:

```
15678.274262606697
```

Question 2

In [12]:

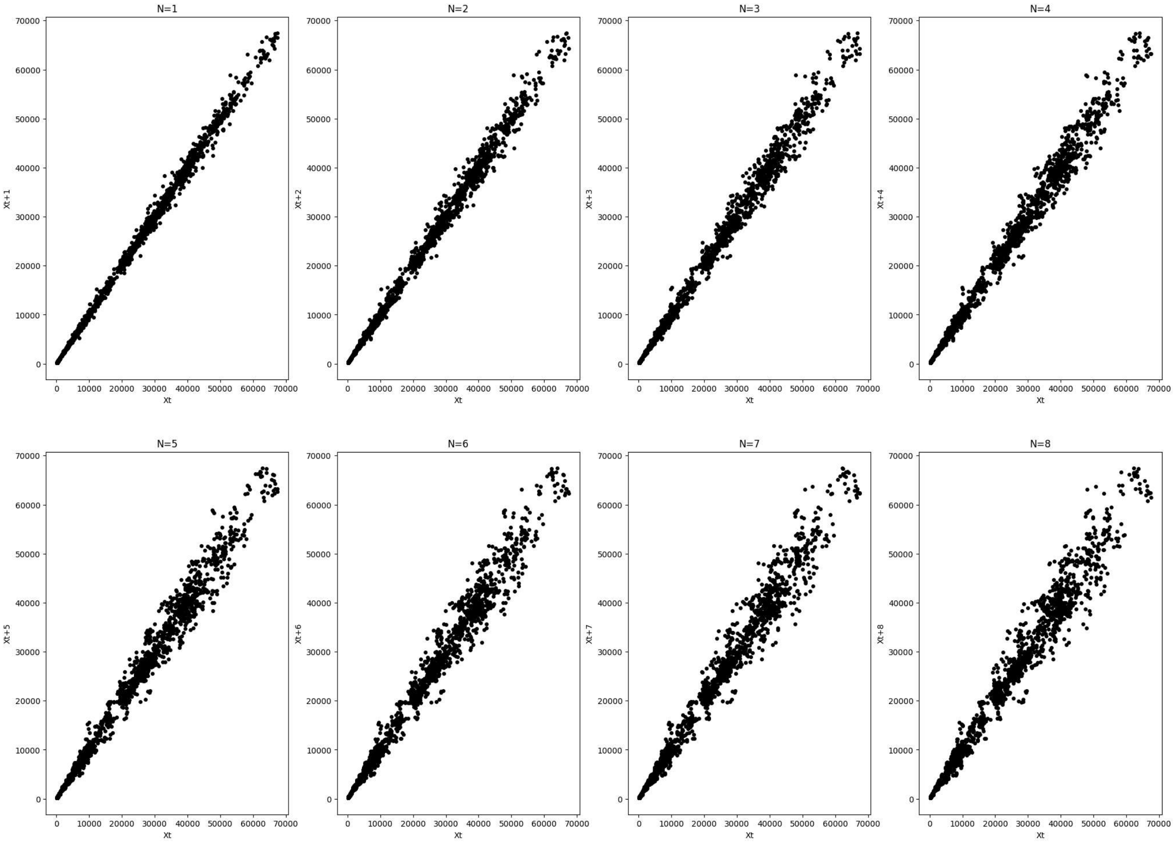
```
plt.figure(figsize=[25,10])
plt.plot(df[metrique], color='black',)
plt.xlabel(period)
plt.ylabel(metrique)
plt.show()
```



Question 3

In [13]:

```
###NX
plt.figure(figsize=[25,18])
for N in range(1,9):
    plt.subplot(2,4,N)
    debut=0
    fin=df.shape[0]
    plt.scatter(df[metrique][debut:fin-N], df[metrique][debut+N:fin], color='bla
    plt.xlabel("Xt")
    plt.ylabel(f"Xt+{N}")
    plt.title(f"N={N}")
plt.show()
```



Question 4

```
In [14]: #Calcul l'auto-Covariance empirique
         def auto_cov(data, K, moy):
             debut=0
             fin=len(data)
             Xt=data[debut:fin-K]
             Xt_k=data[debut+K:fin]

             cov=0
             for i in range(fin-K):
                 cov+=(Xt[i]-moy)*(Xt_k[i]-moy)
             return cov/(fin-K)
```

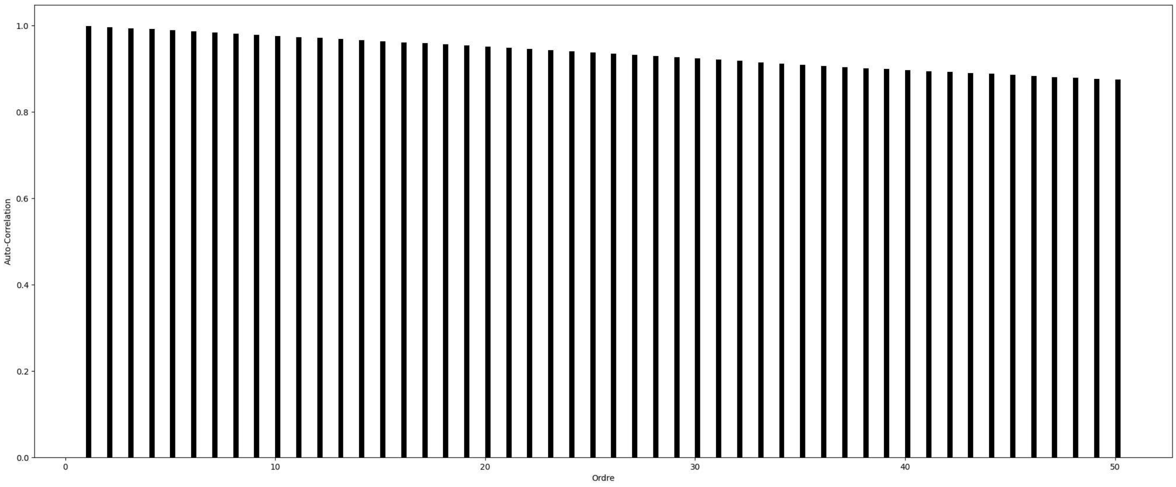
```
In [15]: def auto_cor(data, K):
             moy=np.mean(data)
             cov_0=auto_cov(data, 0, moy)
             cov_K=auto_cov(data, K, moy)
             return cov_K/cov_0
```

```
In [16]: from tqdm import tqdm
         auto_cor_all=list()
         data=list(df[metrique])
         for i in tqdm(range(1, 51)):
             auto_cor_all.append(auto_cor(data, i))
```

```
100%|██████████████████████████████████████████████████
████| 50/50 [00:00<00:00, 234.68it/s]
```

```
In [17]: indexes=[i for i in range(1,51)]
         plt.figure(figsize=[25,10])
         plt.bar(indexes,auto_cor_all, align='edge', width= 0.25, color='black')
         plt.xlabel("Ordre")
         plt.ylabel("Auto-Correlation")
```

```
Out[17]: Text(0, 0.5, 'Auto-Correlation')
```


Auto-Correlation
Ordre