# Asset Pricing and Portfolio Management

Suppose that an investor owns, on September 25th, 2008 a portfolio worth $10 million consisting of investments in four stock indices: Dow Jones Industrial Average (DJIA) in the United States, the FTSE 100 in the United Kingdom, the CAC 40 in France, and the Nikkei 225 in Japan. So, let's suppose that today is September 25th, 2008. The value of the investment in each index on September 25, 2008, is (in $ 000s): $4000 in DJIA, $3000 in FTSE, $1000 in CAC and $2000 in NIKKEI.

## Question 1 : Using a GARCH model (1,1), estimate the tomorrow's volatility of each of the four indices. Compare the values obtained. Is the result in line with your expectations ?

The $garch(p,q)$ model calculates the daily volatility forecast from the $p$ most recent observations $u$ and the $q$ most recent variances $\sigma^2$ :

$$\sigma_t^2 = \omega + \sum_{i=0}^{p} \alpha_i u_{t-i}^2 + \sum_{i=0}^{q} \beta_i \sigma_{t-i}^2 + \epsilon_t$$

$$\sigma_t^2 = \omega + \alpha_0 u_t^2 + \alpha_1 u_{t-1}^2 + \ldots + \alpha_p u_{t-p}^2 + \beta_0 \sigma_t^2 + \beta_1 \sigma_{t-1}^2 + \ldots + \beta_q \sigma_{t-q}^2 + \epsilon_t$$

With the simplest and most popular $garch(1,1)$ model :

$$\sigma_t^2 = \omega + \alpha u_{t-1}^2 + \beta \sigma_{t-1}^2 + \epsilon_t \tag{A.1}$$

with :

- $u_{t-1}$ the market latest news
- $\sigma_{t-1}$ the lastest standard deviation
- $\epsilon_t$ the residual error

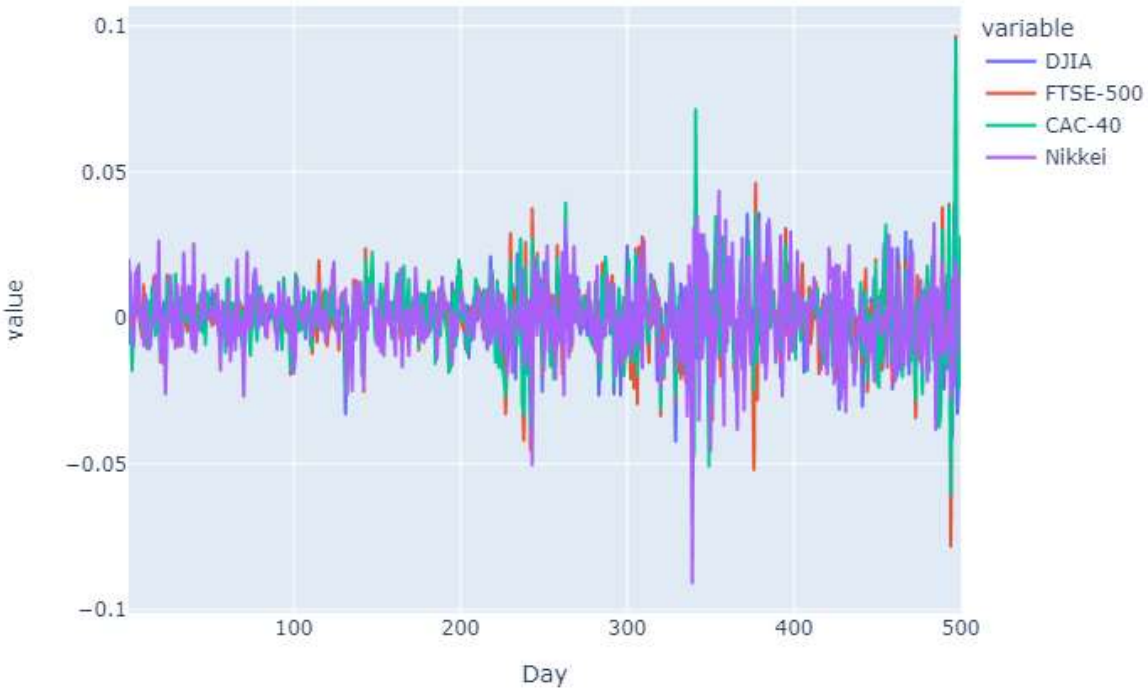
In [1]:
```
#import matplotlib.pyplot as plt
import plotly.express as px
import numpy as np
import pandas as pd
import datetime
```

In [2]:
```
indexes = pd.read_excel('stock_indexes.xls','for_python',index_col=0)
print(indexes)

fig_indexes = px.line(indexes)
fig_indexes.show(renderer="png")
```

```
          DJIA     FTSE-500       CAC-40      Nikkei
Day
0     11219.38  11131.84224  6373.894033  131.774435
1     11173.59  11096.28032  6378.161510  134.381821
2     11076.18  11185.35030  6474.040196  135.943301
3     11124.37  11016.70812  6357.485948  135.438090
4     11088.02  11040.72970  6364.764458  134.100284
..         ...          ...          ...         ...
496   11019.69   8878.18400  5689.850489  109.547101
497   11388.44   9734.01951  6230.005762  111.618539
498   11015.69   9656.26083  6181.952576  113.228975
499   10825.17   9438.57988  6033.934595  114.260398
500   11022.06   9599.89840  6200.396069  112.822120

[501 rows x 4 columns]
```

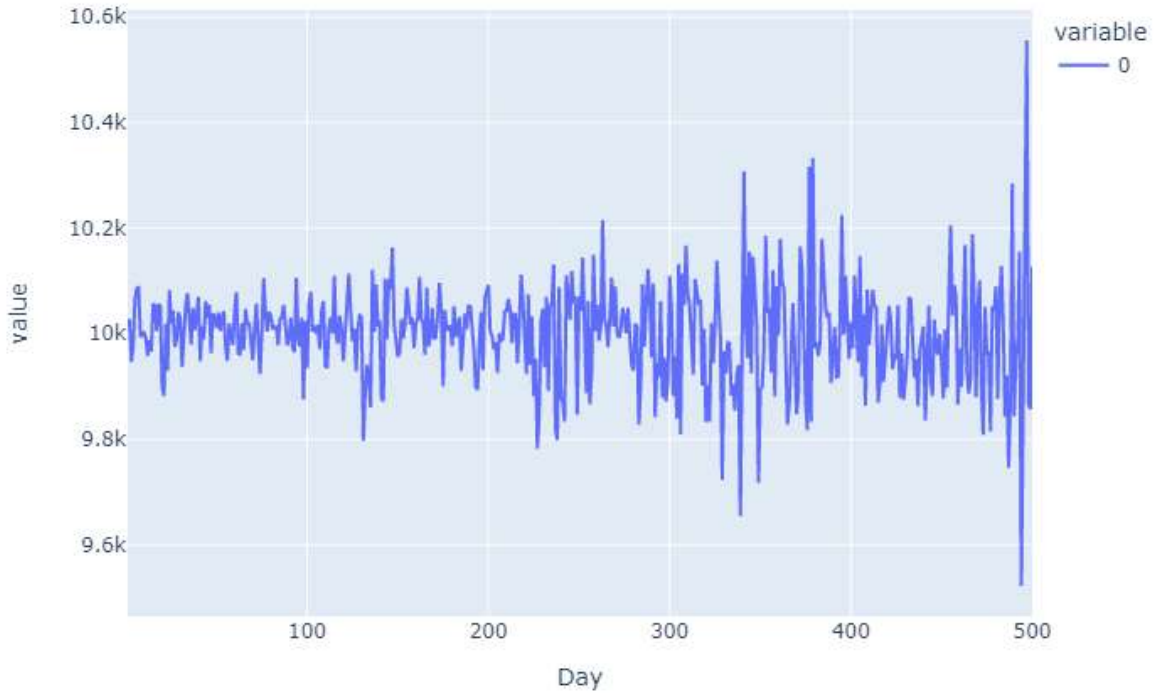

In [290…

```
# We start with the calculation of returns (we have the index levels and not the
returns = indexes.apply(pd.DataFrame.pct_change).dropna()
fig_returns = px.line(returns)
fig_returns.show(renderer="png")
```




In [291…

```
portfolio  = [4000,3000,1000,2000]
lambdas = portfolio / np.sum(portfolio)
portfolio_valuation = np.sum(portfolio * (1 + returns), axis=1)
fig_portfolio = px.line(portfolio_valuation)
fig_portfolio.show(renderer="png")
```




In [292…

```
from arch.univariate import arch_model

# We use normalized returns for the GARCH model to focus on the variance
```

```
normalized_returns = returns - np.mean(returns)

models = dict()
# p=1, q=1 : forecast lengths. Both at 1 for garch(1, 1)
# o=0 : no integration (different from IGARCH)
for index in indexes.columns:
    am = arch_model(100 * np.array(normalized_returns[index]), mean='zero',p=1,
    models[index] = am.fit()
    print(models[index].summary())
```

```
Iteration:      1,   Func. Count:      5,   Neg. LLF: 269383.28894731007
Iteration:      2,   Func. Count:     11,   Neg. LLF: 701.4429284931334
Iteration:      3,   Func. Count:     16,   Neg. LLF: 2424.8164589137905
Iteration:      4,   Func. Count:     21,   Neg. LLF: 806.0475930615592
Iteration:      5,   Func. Count:     26,   Neg. LLF: 700.0291007477424
Iteration:      6,   Func. Count:     31,   Neg. LLF: 268951.13036775636
Iteration:      7,   Func. Count:     38,   Neg. LLF: 700.84774721865
Iteration:      8,   Func. Count:     43,   Neg. LLF: 1082.8148780953195
Iteration:      9,   Func. Count:     48,   Neg. LLF: 694.9524199734253
Iteration:     10,   Func. Count:     53,   Neg. LLF: 689.9671603400541
Iteration:     11,   Func. Count:     57,   Neg. LLF: 689.6884700339114
Iteration:     12,   Func. Count:     61,   Neg. LLF: 689.6826324945141
Iteration:     13,   Func. Count:     65,   Neg. LLF: 689.6810553802809
Iteration:     14,   Func. Count:     69,   Neg. LLF: 1922.7461717785827
Iteration:     15,   Func. Count:     79,   Neg. LLF: 689.8009654003046
Optimization terminated successfully    (Exit mode 0)
            Current function value: 689.6810535127235
            Iterations: 17
            Function evaluations: 83
            Gradient evaluations: 15
                      Zero Mean - GARCH Model Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.000
Mean Model:                 Zero Mean   Adj. R-squared:                  0.002
Vol Model:                      GARCH   Log-Likelihood:               -689.681
Distribution:                  Normal   AIC:                           1385.36
Method:            Maximum Likelihood   BIC:                           1398.01
                                        No. Observations:                  500
Date:                Fri, Sep 30 2022   Df Residuals:                      497
Time:                        23:11:04   Df Model:                            3
                            Volatility Model
==============================================================================
                 coef    std err          t      P>|t|       95.0% Conf. Int.
------------------------------------------------------------------------------
omega      3.5933e-03  9.384e-03      0.383      0.702 [-1.480e-02,2.198e-02]
alpha[1]       0.0226  2.910e-02      0.777      0.437 [-3.441e-02,7.965e-02]
beta[1]        0.9774  4.015e-02     24.343 6.793e-131      [  0.899,  1.056]
==============================================================================

Covariance estimator: robust
Iteration:      1,   Func. Count:      5,   Neg. LLF: 2515.5418784142275
Iteration:      2,   Func. Count:     14,   Neg. LLF: 819.5334132341468
Iteration:      3,   Func. Count:     19,   Neg. LLF: 797.2806802380792
Iteration:      4,   Func. Count:     24,   Neg. LLF: 790.2650387977121
Iteration:      5,   Func. Count:     28,   Neg. LLF: 790.3712937223756
Iteration:      6,   Func. Count:     33,   Neg. LLF: 790.2441907196769
Iteration:      7,   Func. Count:     38,   Neg. LLF: 790.2430501522565
Iteration:      8,   Func. Count:     41,   Neg. LLF: 790.2430495548301
Optimization terminated successfully    (Exit mode 0)
            Current function value: 790.2430501522565
            Iterations: 8
            Function evaluations: 41
            Gradient evaluations: 8
                      Zero Mean - GARCH Model Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.000
Mean Model:                 Zero Mean   Adj. R-squared:                  0.002
Vol Model:                      GARCH   Log-Likelihood:               -790.243
Distribution:                  Normal   AIC:                           1586.49
Method:            Maximum Likelihood   BIC:                           1599.13
```

```
                                    No. Observations:                  500
Date:                Fri, Sep 30 2022   Df Residuals:                      497
Time:                        23:11:04   Df Model:                            3
                           Volatility Model
============================================================================
                 coef    std err          t      P>|t|       95.0% Conf. Int.
----------------------------------------------------------------------------
omega          0.0249  2.059e-02      1.211      0.226 [-1.543e-02,6.529e-02]
alpha[1]       0.1627  4.263e-02      3.817  1.352e-04    [7.916e-02,  0.246]
beta[1]        0.8373  4.565e-02     18.341  3.897e-75      [  0.748,  0.927]
============================================================================

Covariance estimator: robust
Iteration:      1,   Func. Count:      5,   Neg. LLF: 494088.6594260849
Iteration:      2,   Func. Count:     11,   Neg. LLF: 915.7838089176047
Iteration:      3,   Func. Count:     18,   Neg. LLF: 799.1211304991707
Iteration:      4,   Func. Count:     22,   Neg. LLF: 799.620510980104
Iteration:      5,   Func. Count:     27,   Neg. LLF: 875.8578570887162
Iteration:      6,   Func. Count:     34,   Neg. LLF: 799.0639022722684
Iteration:      7,   Func. Count:     38,   Neg. LLF: 799.0638899705226
Iteration:      8,   Func. Count:     41,   Neg. LLF: 799.063889970467
Optimization terminated successfully    (Exit mode 0)
            Current function value: 799.0638899705226
            Iterations: 8
            Function evaluations: 41
            Gradient evaluations: 8
                      Zero Mean - GARCH Model Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.000
Mean Model:                 Zero Mean   Adj. R-squared:                  0.002
Vol Model:                      GARCH   Log-Likelihood:               -799.064
Distribution:                  Normal   AIC:                           1604.13
Method:            Maximum Likelihood   BIC:                           1616.77
                                        No. Observations:                  500
Date:                Fri, Sep 30 2022   Df Residuals:                      497
Time:                        23:11:04   Df Model:                            3
                           Volatility Model
============================================================================
                 coef    std err          t      P>|t|       95.0% Conf. Int.
----------------------------------------------------------------------------
omega          0.0253  2.045e-02      1.238      0.216 [-1.477e-02,6.540e-02]
alpha[1]       0.1304  3.267e-02      3.992  6.545e-05    [6.639e-02,  0.194]
beta[1]        0.8680  3.456e-02     25.118 3.169e-139      [  0.800,  0.936]
============================================================================

Covariance estimator: robust
Iteration:      1,   Func. Count:      5,   Neg. LLF: 1240.236410226374
Iteration:      2,   Func. Count:     15,   Neg. LLF: 1154.397166658533
Iteration:      3,   Func. Count:     21,   Neg. LLF: 901.6473401572973
Iteration:      4,   Func. Count:     28,   Neg. LLF: 834.0680729647509
Iteration:      5,   Func. Count:     32,   Neg. LLF: 834.0612157680425
Iteration:      6,   Func. Count:     36,   Neg. LLF: 834.0610737422608
Iteration:      7,   Func. Count:     40,   Neg. LLF: 834.0610703443177
Iteration:      8,   Func. Count:     43,   Neg. LLF: 834.0610703443479
Optimization terminated successfully    (Exit mode 0)
            Current function value: 834.0610703443177
            Iterations: 8
            Function evaluations: 43
            Gradient evaluations: 8
                      Zero Mean - GARCH Model Results
```

```
================================================================================
Dep. Variable:                          y   R-squared:                       0.000
Mean Model:                      Zero Mean   Adj. R-squared:                  0.002
Vol Model:                           GARCH   Log-Likelihood:               -834.061
Distribution:                       Normal   AIC:                           1674.12
Method:                 Maximum Likelihood   BIC:                           1686.77
                                             No. Observations:                  500
Date:                    Fri, Sep 30 2022    Df Residuals:                      497
Time:                            23:11:04    Df Model:                            3
                              Volatility Model
================================================================================
                 coef    std err          t      P>|t|     95.0% Conf. Int.
--------------------------------------------------------------------------------
omega          0.0637  3.703e-02      1.721  8.522e-02 [-8.843e-03,  0.136]
alpha[1]       0.1343  7.059e-02      1.903  5.705e-02 [-4.027e-03,  0.273]
beta[1]        0.8414  6.196e-02     13.580  5.271e-42    [  0.720,  0.963]
================================================================================

Covariance estimator: robust
```

We stored the models into a dictionary and can access the parameters for each index :

In [293…
```
print(models["DJIA"].params)
```

```
omega       0.003593
alpha[1]    0.022616
beta[1]     0.977384
Name: params, dtype: float64
```

We can then compute tomorrow's daily volatility for each index using the GARCH formula $(A.1)$ :

In [294…
```
garch_vol_tomorrow = dict()
garch_vol_forecasted_tomorrow = dict()
for index in indexes.columns :
    previous_return = returns[index].iloc[-1]
    previous_vol = models[index].conditional_volatility[-1]
    garch_vol_tomorrow[index] = np.sqrt(models[index].params[0] + models[index].
    print(f"Tomorrow's volatility for {index} computed by hand : {garch_vol_tomo

    garch_vol_forecasted_tomorrow[index] = np.sqrt(models[index].forecast().vari
    print(f"Tomorrow's volatility for {index} forecasted by the module : {float(
```

```
Tomorrow's volatility for DJIA computed by hand : 1.80
Tomorrow's volatility for DJIA forecasted by the module : 1.82

Tomorrow's volatility for FTSE-500 computed by hand : 3.87
Tomorrow's volatility for FTSE-500 forecasted by the module : 3.94

Tomorrow's volatility for CAC-40 computed by hand : 3.58
Tomorrow's volatility for CAC-40 forecasted by the module : 3.72

Tomorrow's volatility for Nikkei computed by hand : 1.52
Tomorrow's volatility for Nikkei forecasted by the module : 1.59
```

We check in the summaries that, for each index, the P values of the $\omega$, $\alpha$ and $\beta$ parameters are acceptable, ie that for each : $(P > |z|) < 0.05$. It is not always the case

here.

In [296…
```
threshold = 0.05
corrected_models = models

for index in indexes.columns:
    print(f"{index}:")
    for i, param in enumerate(models[index].params.keys()):
        if models[index].pvalues[i] < threshold :
            print(f" * {param} = {models[index].params[param]:.4f}, p-value = {m
        else :
            print(f" * {param} = {models[index].params[param]:.4f}, p-value = {m
            corrected_models[index].params[param] = 0
```

```
DJIA:
 * omega = 0.0036, p-value = 0.7018 > 0.05 : the error margin is too high, the co
efficient is dropped
 * alpha[1] = 0.0226, p-value = 0.4370 > 0.05 : the error margin is too high, the
coefficient is dropped
 * beta[1] = 0.9774, p-value = 0.0000 < 0.05 : the error margin is low, the coeff
icient is significant
FTSE-500:
 * omega = 0.0249, p-value = 0.2261 > 0.05 : the error margin is too high, the co
efficient is dropped
 * alpha[1] = 0.1627, p-value = 0.0001 < 0.05 : the error margin is low, the coef
ficient is significant
 * beta[1] = 0.8373, p-value = 0.0000 < 0.05 : the error margin is low, the coeff
icient is significant
CAC-40:
 * omega = 0.0253, p-value = 0.2158 > 0.05 : the error margin is too high, the co
efficient is dropped
 * alpha[1] = 0.1304, p-value = 0.0001 < 0.05 : the error margin is low, the coef
ficient is significant
 * beta[1] = 0.8680, p-value = 0.0000 < 0.05 : the error margin is low, the coeff
icient is significant
Nikkei:
 * omega = 0.0637, p-value = 0.0852 > 0.05 : the error margin is too high, the co
efficient is dropped
 * alpha[1] = 0.1343, p-value = 0.0571 > 0.05 : the error margin is too high, the
coefficient is dropped
 * beta[1] = 0.8414, p-value = 0.0000 < 0.05 : the error margin is low, the coeff
icient is significant
```

In [297…
```
print(f"Tomorrow's corrected volatility :")
volatility_prevision = dict()
for index in indexes.columns :
    previous_return = returns[index].iloc[-1]
    previous_vol = corrected_models[index].conditional_volatility[-1]
    volatility_prevision[index] = corrected_models[index].params[0] + corrected_
    print(f" * {index} : {volatility_prevision[index]:.2f}")
```

```
Tomorrow's corrected volatility :
 * DJIA : 1.78
 * FTSE-500 : 3.55
 * CAC-40 : 3.34
 * Nikkei : 1.38
```

For the DJIA and the Nikkei, we dropped more than one coefficient. This means that the GARCH model is not a good fit. Instead we will compute the volatility with the basic

approach.

In [298…
```
kernel_window = 500
for index in ["DJIA", "Nikkei"] :
    kw_sigmas = normalized_returns[index][-kernel_window:]
    volatility_prevision[index] = np.std(kw_sigmas) * 100
    print(f" * {index}'s average volatility over the last {kernel_window} days :
```

```
* DJIA's average volatility over the last 500 days : 1.11
* Nikkei's average volatility over the last 500 days : 1.38
```

# Question 2: Calculate tomorrow's portfolio volatility.

We use the formula to compute the variance of a sum of variables :

$$\sigma_p^2 = \sum_i \lambda_i \sigma_i^2 + \sum_i \sum_j \lambda_i \lambda_j Cov(X_i, X_j)$$

with

- $\sigma_p$ the variance of portfolio $p = \sum_i \lambda_i X_i$
- $\sigma_i$ the variance of asset $X_i$
- $\lambda_i$ the weight of $X_i$ in the portfolio, with $0 \leq \lambda_i \leq 1$ and $\sum_i \lambda_i = 1$

In [299…
```
covariances = np.cov([returns[i] for i in indexes.columns])
covariances
```

Out[299…
```
array([[ 1.22952438e-04,  7.69659054e-05,  7.68251360e-05,
        -9.49348777e-06],
       [ 7.69659054e-05,  2.01397345e-04,  1.82107644e-04,
         3.94430218e-05],
       [ 7.68251360e-05,  1.82107644e-04,  1.95350635e-04,
         4.07812976e-05],
       [-9.49348777e-06,  3.94430218e-05,  4.07812976e-05,
         1.91312901e-04]])
```

In [310…
```
def portfolio_volatility(indexes_volatility) :
    portfolio_variance = 0
    for i, index in enumerate(indexes.columns):
        portfolio_variance += lambdas[i] * indexes_volatility[index]**2
        for j in range(len(indexes.columns)):
            if j != i :
                portfolio_variance += lambdas[i] * lambdas[j] * covariances[i][j
    return np.sqrt(portfolio_variance)
```

In [314…
```
# Pass "garch_vol_tomorrow" as variable to compute with non-corrected volatility
portfolio_vol_tomorrow = portfolio_volatility(volatility_prevision)
print(f"Tomorrow's portfolio volatility is {portfolio_vol_tomorrow:.4f}.")
```

```
Tomorrow's portfolio volatility is 2.3995.
```

# Question 3: Deduce an estimate of the daily VaR at the 99% confidence level of this portfolio (use the volatility estimate obtained with the

# GARCH(1,1) model in the previous question). Which assumptions did you have to use to obtain this estimate?

Assuming that the returns are normally distributed with a mean of 0, we can use the density probability function to calculate the daily 99% VaR.

In [302…
```
portfolio_returns = np.sum(returns * lambdas, axis=1)
```

In [303…
```
from scipy.stats import norm

x_level = 0.99
# ppf(q, loc=0, scale=1) with loc = mean and scale = std_dev
VaR_1d_99n = - norm.ppf(1 - x_level, loc=np.mean(portfolio_returns), scale=portf

print(f"The daily VaR at the 99% confidence level is ${VaR_1d_99n:.2f}.")
```

```
The daily VaR at the 99% confidence level is $558.21.
```

A positive VaR mean that we need to put some money aside to hedge the portfolio. Here, we are 99% confident that the portfolio will lose at most the VaR value until the next day.

# Question 4: Is this estimate significantly different from the one you would have obtained without using a GARCH model (1,1)?

Without using a GARCH model (1,1), we could have assumed that the volatility is constant over time and compute it with the basic approach. We start with each index, then compute the portfolio volatility, then the portfolio VaR.

In [318…
```
kernel_window = 20

basic_vol_daily = {i : np.std(returns[i][-kernel_window:]) * 100 for i in indexe
basic_portfolio_daily_vol = portfolio_volatility(basic_vol_daily)
x_level = 0.99
basic_VaR_1d_99n = - norm.ppf(1 - x_level, loc=np.mean(portfolio_returns), scale
print(f"The daily VaR at the 99% confidence level with the basic approach is ${b
```

```
The daily VaR at the 99% confidence level with the basic approach is $619.89.
```

The estimates obtained with the basic approach and the GARCH model (1,1) are significantly different. While the basic approach is easier to understand, it puts the same weight on all previous coefficients while the garch model showed at least for some indexes that the previous return ($\alpha$ coefficient) and the previous volatility ($\beta$ coefficient) can have a different influence over tomorrow's returns.

# Question 5: Using the same method, give an estimate of the daily VaR at 99.999%.

In [319…
```
x_level = 0.99999
# ppf(q, loc=0, scale=1) with loc = mean and scale = std_dev
VaR_1d_99_999n = - norm.ppf(1 - x_level, loc=np.mean(portfolio_returns), scale=p

print(f"The daily VaR at the 99.999% confidence level is ${VaR_1d_99_999n:.2f}."
```

```
The daily VaR at the 99.999% confidence level is $1023.36.
```

The 99.999% VaR is higher than the 99% VaR, because with a higher level of confidence, we consider more dreadful - hence costly - scenarios.

# Question 6: Again using the results of the GARCH model (1,1), estimate a 10-day Var at the 99% confidence level. Is the result significantly different from that obtained without using a GARCH model?

Assuming that changes in the portfolio on successive days have independent identical normal distributions with mean zero, we can use the following formula to compute the 10-day VaR :

$$VaR(T) = VaR(1day)\sqrt{T}$$

hence :

$$VaR(10day) = VaR(1day)\sqrt{10}$$

In [320…
```
VaR_10d_99n = VaR_1d_99n * np.sqrt(10)
print(f"The 10-day VaR at the 99.999% confidence level is ${VaR_10d_99n:.2f}.")
```

```
The 10-day VaR at the 99.999% confidence level is $1765.21.
```