LECTURE 6 - MAY 8, 2024

TEST OF NORMALITY OF LOG-RETURNS

THE ESTIMATION OF HISTORICAL VOLATILITY IS BASED ON THE ASSUMPTION THAT LOG-RETURNS ARE NORMAL.

PROBLEM: TEST IF THE NORMALITY ASSUMPTION IS REALISTIC.

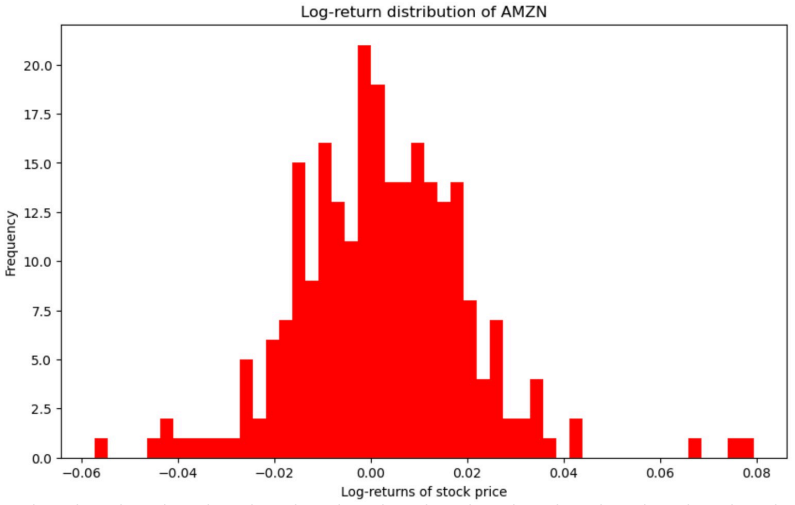
WE FIRST COMPUTE LOG-RETURNS OF STOCK AMZN IN THE LAST YEAR.

[]:
```
# Import the libraries
import yfinance as yf
import pandas as pd
import numpy as np
```

[]:
```
# Create an object to access AMZN stock
stock = yf.Ticker('AMZN')

# Load AMZN's historical data
hist_data = stock.history(period='1y')

# Extract closing prices
prices = hist_data['Close']

# Compute the log-return and remove NaN
log_rets = np.log(prices / price.shift(1))
log_rets = log_rets.dropna()
```

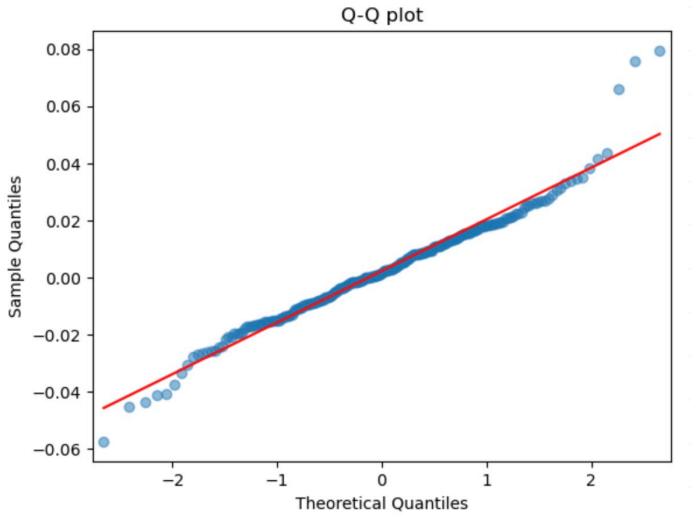
WE WANT TO PLOT THE HISTOGRAM OF LOG-RETURNS

[ ]:
```
# Plot the histogram of log-returns
import matplotlib.pyplot as plt

plt.figure(figsize = (10, 6))
plt.hist(log_rets, bins = 50, color = 'red')
plt.xlabel('Log-returns of stock price')
plt.ylabel('Frequency')
plt.title('Log-return distribution of AMZN')
```



THE QUANTILE-QUANTILE (Q-Q) PLOT ALLOWS TO VISUALIZE NORMALITY AS WELL: IN CASE OF NORMALITY, POINTS SHOULD BE CLOSE TO THE RED LINE.

[]:
```
# Plot the Q-Q plot
import statsmodels.api as sm

fig = sm.qqplot(log_rets, line='s', alpha=0.5)
plt.title('Q-Q plot')
```



WE THEN EXECUTE SKEW, KURTOSIS AND NORMALITY TESTS: A p-VALUE BELOW 0.05 IS GENERALLY CONSIDERED TO BE A COUNTER-INDICATOR FOR NORMALITY.

[]:
```
# Compute skew and kurtosis
import scipy.stats as scs

skew = scs.skew(log_rets)
kurtosis = scs.kurtosis(log_rets)
print(f'Skew of sample log-returns {skew:.6f}')
print(f'Kurtosis of sample log returns {kurtosis:.6f}')
```

FORMATTED PRINT: 6 DECIMALS

```
[]: # Execute the skew test
     p_value = scs.skewtest(log_rets)[1]
     print(f'Skew Normal test {p_value:.6f}')
```

TO EXTRACT THE p-VALUE

```
[]: # Execute the kurtosis test
     p_value = scs.kurtosistest(log_rets)[1]
     print(f'Kurtosis Normal test {p_value:.6f}')
```

```
[]: # Execute the normal test
     p_value = scs.normaltest(log_rets)[1]
     print(f'Normal test {p_value:.6f}')
```

← SKEW AND KURTOSIS COMBINED TEST

HOMEWORK (SOLUTION ON UniStudium)

① WRITE A FUNCTION THAT, GIVEN log_rets, COMPUTES THE HISTORICAL VOLATILITY (THE FUNCTION RETURNS THE HISTORICAL VOLATILITY THAT MUST BE PRINTED).

② WRITE A FUNCTION THAT, GIVEN log_rets, SHOWS THE HISTOGRAM (THE FUNCTION HAS NO return STATEMENT).

③ WRITE A FUNCTION THAT, GIVEN log_rets, EXECUTE THE SKEW, KURTOSIS AND NORMALITY TESTS (THE FUNCTION HAS NO return STATEMENT).

GIVEN THE LIST OF TICKERS:

tickers = ['AMZN', 'TSLA', 'BTC-USD', 'ETH-USD']

FOR EACH TICKER COMPUTE THE DAILY LOG-RETURNS IN THE VARIABLE log_rets AND EXECUTE THE FUNCTIONS OF POINTS ① - ③ ON log_rets.

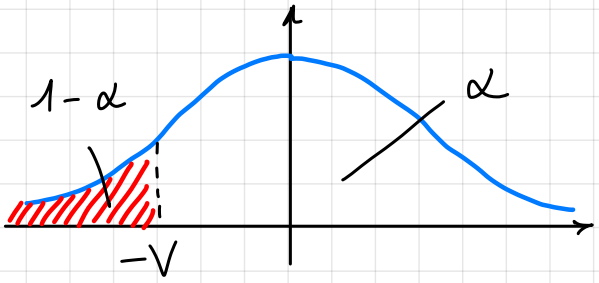
VALUE AT RISK

GIVEN A PORTFOLIO, THE **VALUE AT RISK** VaR IS A RISK MEASURE EXPRESSED BY A SINGLE NUMBER V THAT ANSWERS TO THE FOLLOWING QUESTION:

> "WE ARE CERTAIN AT LEVEL α THAT WE WILL NOT LOSE MORE THAN V MONETARY UNITS IN TIME T"

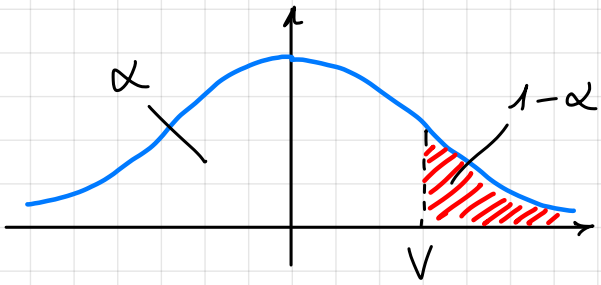
WE CAN ADOPT TWO POINTS OF VIEW IN THE COMPUTATION OF VaR:

PROB. DIST. OF PORTFOLIO GAIN



LEFT TAIL

PROB. DIST. OF PORTFOLIO LOSS



RIGHT TAIL

ADOPTING THE POINT OF VIEW OF GAIN, WHICH IS A RANDOM VARIABLE X WITH CDF $F(x)$

$$VaR_\alpha(X) = -q_{1-\alpha}(X) = -\text{"}(1-\alpha)\text{-TH QUANTILE OF } X\text{"}$$

IN PARTICULAR, IF X IS A CONTINUOUS R.V. WE HAVE

$$F(q_{1-\alpha}(X)) = 1-\alpha$$

TODAY WE WILL CONSIDER THE COMPUTATION OF VaR FOR **STOCK RETURNS** (LEFT TAILS).

IN THE PREVIOUS LECTURE WE HAVE ESTIMATED VOLATILITY WORKING WITH LOG-RETURNS THAT USUALLY ARE ASSUMED TO BE NORMAL.

GIVEN $n+1$ STOCK PRICE DAILY OBSERVATIONS

$$S_0, S_1, \dots, S_n$$ ← IN THE PAST

WE HAVE COMPUTED DAILY LOG-RETURNS

$$u_i = \ln\left(\frac{S_i}{S_{i-1}}\right) \qquad i = 1, 2, \dots, n$$

WHEN WE COMPUTE VaR IT IS COMMON TO REFER TO DAILY RETURN RATES

$$u_i = \frac{S_i - S_{i-1}}{S_{i-1}} \qquad i = 1, 2, \dots, n$$

AND ESTIMATE THE STANDARD DEVIATION OF u_i WITH THE BIASED ESTIMATOR

$$\sigma = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (u_i - \bar{u})^2}$$

REMARK: ON A DAILY BASIS $\ln\left(\frac{S_i}{S_{i-1}}\right) \approx \frac{S_i - S_{i-1}}{S_{i-1}}$

AND THE BIASED ESTIMATOR IS CLOSE TO THE UNBIASED ONE.

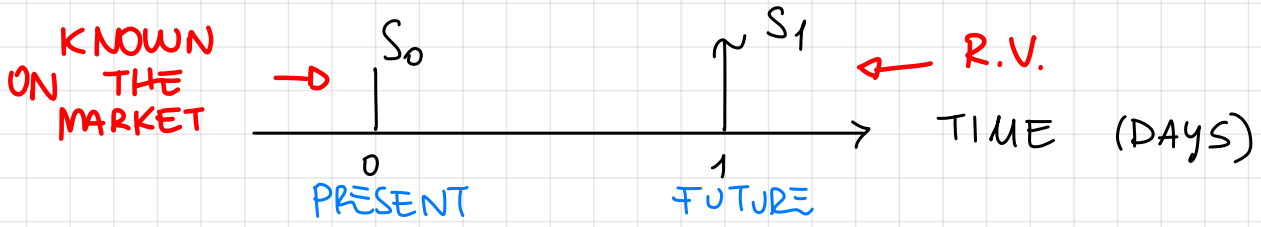
WE FIRST CONSIDER A PORTFOLIO MADE UP OF A SINGLE STOCK: THE AMZN STOCK.

WE HAVE TWO WAYS FOR ESTIMATING THE VaR:

- NON-PARAMETRIC APPROACH
- PARAMETRIC APPROACH

WE FIX THE TIME HORIZON TO 1 DAY AND CONSIDER THE CHANGE OF VALUE OF THE STOCK.

LOOKING ONE DAY AHEAD WE PUT OURSELVES AT TIME 0



THE RATE OF RETURN OF THE STOCK IS A R.V.

$$\frac{S_1}{S_0} - 1 = \frac{S_1 - S_0}{S_0} = \frac{\Delta S}{S_0}$$

WITH CDF $F(x)$, MEAN μ AND STANDARD DEVIATION σ ON A DAILY BASIS

$$VaR_\alpha(\Delta S) = S_0 \left[\mu - q_{1-\alpha}\left(\frac{\Delta S}{S_0}\right) \right]$$

THIS IS FOR 1 SHARE OF STOCK. IF WE HAVE N SHARES, THE GLOBAL VaR IS

$$N \cdot VaR_\alpha(\Delta S) = N \cdot S_0 \cdot \left[\mu - q_{1-\alpha}\left(\frac{\Delta S}{S_0}\right) \right]$$

① SINGLE STOCK: NON-PARAMETRIC ESTIMATION

WE DO NOT ASSUME ANY PARTICULAR DISTRIBUTION FOR THE R.V. $\frac{\Delta S}{S_0}$. WE ESTIMATE THE CDF $F(x)$ FROM HISTORICAL DATA AND COMPUTE $q_{1-\alpha}\left(\frac{\Delta S}{S_0}\right)$ AS THE $(1-\alpha)$-TH ORDER STATISTICS OF HISTORICAL RATE OF RETURNS: WE ORDER THE u_i'S INCREASINGLY

$$u_{j_1} \leq u_{j_2} \leq \cdots \leq u_{j_k} \leq \cdots \leq u_{j_{n-1}} \leq u_{j_n}$$

$(1-\alpha) \cdot n$ $\qquad$ $\alpha \cdot n$

QUANTILE

[ ]:
```
# Import the libraries
import numpy as np
import pandas as pd
import yfinance as yf
```

[ ]:
```
# Load the last 3 years of data for AMZN
ticker = 'AMZN'
stock = yf.Ticker(ticker)
hist_data = stock.history(period='3y')
prices = hist_data['Close']
prices
```

3 YEARS

[ ]:
```
# Fix the parameters for the risk analysis
alpha = 0.99
num_shares = 5000
on_date = '2024-05-08'
share_price = prices[on_date]
```

← N

← S_0

TO COMPUTE $u_i = \frac{S_i}{S_{i-1}} - 1$ WE USE pct_change()

[ ]:
```
# Compute the return rates
return_rates = prices.pct_change()
return_rates
```

[ ]:
```
# Delete the NaN
return_rates = return_rates.dropna()
return_rates
```

```
[]: # Get the quantile
   quantile = np.quantile(return_rates, q=1-alpha,
                          method='inverted_cdf')
```

THERE ARE DIFFERENT METHODS TO COMPUTE THE QUANTILE

```
[]: # Compute the mean of return rates
   mean_return_rate = return_rates.mean()
```
← μ

NOW WE CAN COMPUTE THE VALUE AT RISK AS

$$N \cdot S_0 \cdot \left[\mu - q_{1-\alpha}\left(\frac{\Delta S}{S_0}\right)\right]$$

```
[]: # Compute the non-parametric VaR
   VaR_np = num_shares * share_price *
            (mean_return_rate - quantile)
   print('The n.p. VaR is:', Var_np)
```

num_shares → N; share_price → S_0; mean_return_rate → μ; quantile → $q_{1-\alpha}\left(\frac{\Delta S}{S_0}\right)$

(2) SINGLE STOCK: PARAMETRIC ESTIMATION

WE ASSUME THAT

$$\frac{\Delta S}{S_0} \sim N(\mu, \sigma^2)$$

ON A DAILY BASIS

$$\varepsilon = \frac{(\Delta S/S_0) - \mu}{\sigma} \sim N(0,1)$$ ← STANDARD NORMAL

$$\frac{\Delta S}{S_0} = \mu + \sigma \cdot \varepsilon$$

THE QUANTILE OF $\frac{\Delta S}{S_0}$ IS DERIVED FROM THE QUANTILE OF THE STANDARD NORMAL

INVERSE OF THE STANDARD NORMAL CDF

$$q_{1-\alpha}\left(\frac{\Delta S}{S_0}\right) = \mu + \sigma \cdot N^{-1}(1-\alpha)$$

$N^{-1}(1-\alpha) = z_{1-\alpha}$

```
[]: # VaR PARAMETRIC VERSION
    from scipy.stats import norm
```

NORMAL DISTRIBUTION

```
[]: # Normal return assumption
    # Use the z-value, mean and variance to
    # calculate VaR
    z_value = norm.ppf(1 - alpha)
```

NORMAL QUANTILE FUNCTION: BY DEFAULT A STANDARD NORMAL

```
[]: # Mean and standard deviation of return rates
    mean_return_rate = return_rates.mean()
    std_return_rate = return_rates.std()
```

```
[]: # Compute the (1-alpha)-th quantile
    quantile = mean_return_rate +
                    std_return_rate * z_value
```

TO COMPUTE IT DIRECTLY AVOIDING THE STANDARD NORMAL, WRITE:

```
quantile = norm.ppf(1-alpha, mean_return_rate,
                    std_return_rate)
```

```
[]: # Compute the VaR under normality
    VaR_normal = num_snares * share_price *
                    (mean_return_rate - quantile)
    print('The normal VaR is:', Var_normal)
```

THE VaR WE HAVE COMPUTED IS THE SO-CALLED "RELATIVE VaR". IF mean_return_rate IS REPLACED BY 0 WE GET THE SO-CALLED "ABSOLUTE VaR".

FOR 1 SHARE, THE ABSOLUTE VaR IS

$$VaR_\alpha(\Delta S) = S_0 \left[0 - q_{1-\alpha}\left(\frac{\Delta S}{S_0}\right)\right] = - S_0 \cdot q_{1-\alpha}\left(\frac{\Delta S}{S_0}\right)$$

FOR N SHARES, THE ABSOLUTE VaR IS

$$N \cdot VaR_\alpha(\Delta S) = N \cdot S_0 \left[0 - q_{1-\alpha}\left(\frac{\Delta S}{S_0}\right)\right] = - N \cdot S_0 \cdot q_{1-\alpha}\left(\frac{\Delta S}{S_0}\right)$$

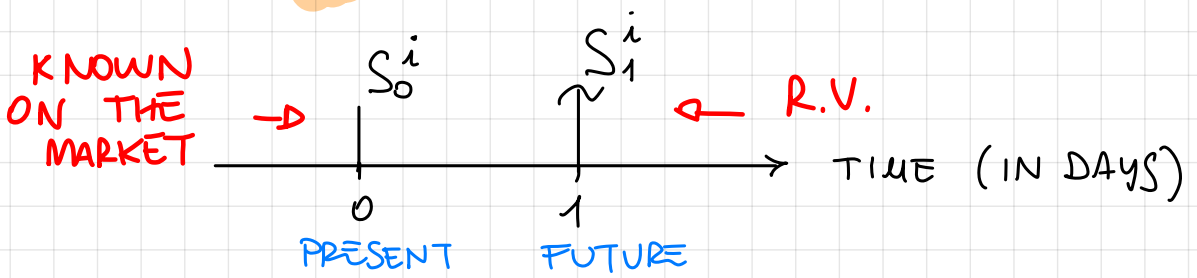
[]:
```
# Compute the absolute VaR
VaR_normal_abs = num_shares * share_price
                 * (0 - quantile)

VaR_normal_abs
```

WE CAN SIMPLY WRITE

```
VaR_normal_abs = - num_shares * share_price
                 * quantile
```

③ PORTFOLIO OF STOCKS: VARIANCE-COVARIANCE METHOD

WE CONSIDER m STOCKS EACH WITH PRICE



A PORTFOLIO IS A VECTOR $\underline{\theta} = [\theta_1, \theta_2, \dots, \theta_m]^T$

WHERE θ_i IS THE NUMBER OF OWNED SHARES.

$$P_1 = \theta_1 S_1^1 + \theta_2 S_1^2 + \dots + \theta_m S_1^m \quad \leftarrow \text{FUTURE VALUE}$$

$$P_0 = \theta_1 S_0^1 + \theta_2 S_0^2 + \dots + \theta_m S_0^m \quad \leftarrow \text{PRESENT VALUE}$$

WE USUALLY FIX P_0 AND SWITCH FROM SHARES TO WEIGHTS: $\underline{w} = [w_1, w_2, \dots, w_m]^T$ ← THEY SUM UP TO 1

$$w_i = \frac{\theta_i S_0^i}{P_0}$$

ASSUMPTION: THE DAILY RETURN RATES ARE JOINTLY NORMAL DISTRIBUTED WITH MEAN μ_i, σ_i, AND COVARIANCES $Cov_{ij} = \rho_{ij}\sigma_i\sigma_j$ $(i \neq j)$.

THE RETURN RATE OF THE PORTFOLIO IS

$$\frac{\Delta P}{P_0} = \frac{P_1 - P_0}{P_0} = \frac{P_1}{P_0} - 1$$

IT IS NORMALLY DISTRIBUTED: $\frac{\Delta P}{P_0} \sim N(\mu_{port}, \sigma^2_{port})$

GOAL: ESTIMATE $VaR_\alpha(\Delta P)$.

WE BUILD A PORTFOLIO FORMED BY

- STOCKS: AMZN, TSLA, AAPL, GOOG
- WEIGHTS: 0.25, 0.3, 0.15, 0.3
- INITIAL INVESTMENT: 1000000

[]:
```
# Import libraries
import numpy as np
import pandas as pd
import yfinance as yf
from scipy.stats import norm
```

[]:
```
# Create a portfolio of equities
tickers = ['AMZN', 'TSLA', 'AAPL', 'GOOG']
# Set the portfolio weights (summing up to 1)
weights = np.array([0.25, 0.3, 0.15, 0.3])
# Set the initial investment
initial_investment = 1000000
```

← P_0

WITH "S"

TO READ ALL TICKERS AT ONCE WE USE yf.Tickers()

[]:
```
# Read all the tickers at once
stocks = yf.Tickers(tickers)
hist_data = stocks.history(period='3y')['Close']
hist_data
```

WE CONTINUE THIS ANALYSIS IN THE NEXT LECTURE.