Performance and Risk Analysis of UBS and the ETF MSCI World

Descriptive Statistics, Ratios, Stylized Facts, Comparisons and VaR Based Portfolio Management of The Two assets

Tickers: UBSG.SW and IWDC.SW

Matteo Cunsolo

March 1, 2026

Introduction

This paper studies the risk and performance of UBS compared with a global market benchmark, the iShares MSCI World CHF Hedged UCITS ETF Accumulation. The aim is to understand how a single banking stock behaves relative to the overall equity market, and how its volatility and downside risk can be measured and managed.

UBS is an interesting case because it is one of the largest European banking groups and it experienced strong price movements during recent events such as the Covid crisis and the Credit Suisse takeover in 2023. These episodes created large volatility and make UBS a useful example for risk modelling.

To keep the comparison consistent, **both assets are analysed in the same currency, Swiss francs.** UBS is traded in CHF, and the chosen ETF is a CHF hedged global market ETF. This avoids distortions caused by exchange rate movements. If one asset were in euros or dollars and the other in Swiss francs, part of the measured return differences would come from currency changes rather than from stock performance.

Daily adjusted closing prices are used for both assets, since they account for dividends and stock splits. The sample covers January 2015 to December 2025 and includes very different market regimes, from calm periods to major stress episodes. The data is retrieved from Yahoo Finance and processed in R. After aligning trading days and computing daily log returns, we compare UBS and the market ETF using both descriptive measures and risk metrics.

The analysis is organized in five steps. First, we compare UBS and the market ETF using basic performance and risk measures: annualized return, annualized volatility, skewness, kurtosis, correlation, and other risk measures (Sharpe, Sortino, maximum drawdown, and Calmar). This gives a quick picture of risk return.

Second, we check the main stylized facts of returns. We show that daily returns have little autocorrelation, but volatility is persistent and we show non normal tails using density and QQ plots.

Third, we compute 95% Value at Risk with simple methods (Historical, Gaussian, and Cornish–Fisher) and compare the results.

Fourth, we move to time varying risk by forecasting volatility with a rolling GARCH model and producing one day ahead 95% VaR forecasts. We evaluate each VaR model with backtesting, counting how often returns fall below the VaR line.

Finally, we use the VaR forecasts to build a simple risk managed portfolio. We allocate more weight to the asset with lower predicted VaR and compare the managed portfolio to holding UBS or the ETF alone, focusing on the trade off between lower risk and lower raw return.

Contents

1 Data processing and visualization

1.1 Chosen instruments: UBS and MSCI World CHF ETF

The analysis focuses on the stock of **UBS Group AG** and on a global market benchmark represented by the **iShares MSCI World CHF Hedged UCITS ETF**. UBS is one of the largest European banking groups and is used as the risky asset in the portfolio management section. The ETF represents a diversified global equity portfolio and serves as a market benchmark; thus, it is sometimes referred to here as just "Market ETF".

Both instruments are analysed in Swiss francs in order to avoid distortions caused by exchange rate movements. Using a CHF hedged ETF ensures that differences in performance reflect equity risk rather than currency risk.

1.2 Sample period, frequency, and data source

Daily adjusted closing prices are used from January 2015 to December 2025. This period includes different market conditions, such as the COVID-19 crisis in 2020 and banking sector turbulence in 2023.

Data is downloaded from Yahoo Finance using R packages designed for financial data retrieval. **Adjusted closing prices** are preferred because they account for dividends, stock splits, and other corporate actions.

1.3 Cleaning, alignment, and missing values

Trading calendars of UBS and the iShares MSCI World ETF do not perfectly overlap because of different exchange holidays. Therefore, the datasets are aligned by keeping only common trading days. Missing observations are removed, and duplicated entries are checked and eliminated if necessary.

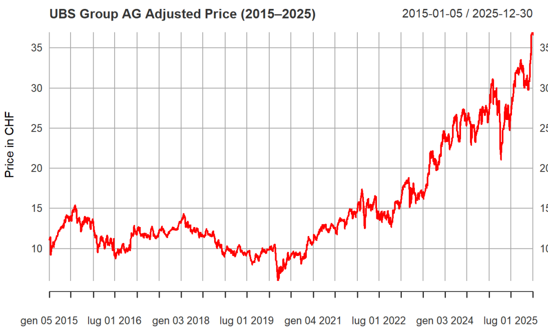
Table 1: Last available trading days for UBSG.SW and IWDC.SW adjusted prices

IWDC.SW		**UBSG.SW**	
Date	Adjusted Price	Date	Adjusted Price
2025-12-18	85.82	2025-12-18	36.70
2025-12-19	86.25	2025-12-19	36.81
2025-12-22	86.52	2025-12-22	36.78
2025-12-23	86.58	2025-12-23	36.94
2025-12-29	86.68	2025-12-29	36.63
2025-12-30	86.91	2025-12-30	36.96

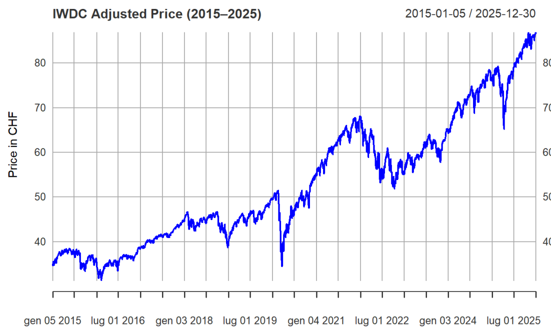
After alignment, both price series contain the same number of observations, which ensures that correlations, volatility measures, and Value at Risk estimates are computed consistently.

1.4 Initial plots: prices and returns

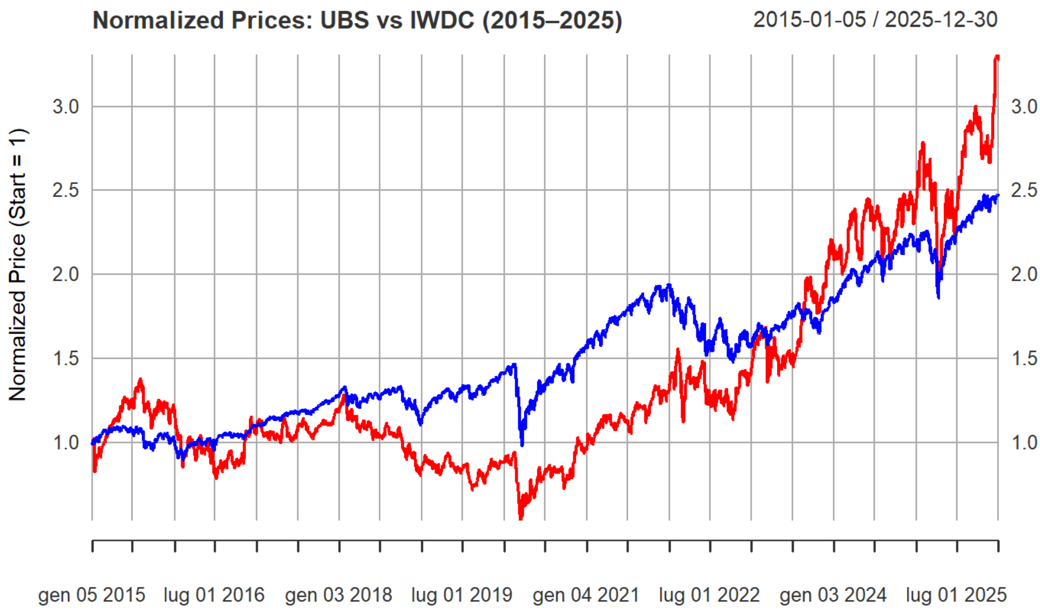
As a first step, price series and return series are plotted for both assets. Price plots show long term trends and major market events, while return plots highlight volatility differences between UBS and the MSCI world ETF taken as the marlet benchmark. These visualizations provide an initial understanding of the data and help identify structural breaks or abnormal observations before doing more analysis



(a) Adjusted closing price of UBSG.SW.

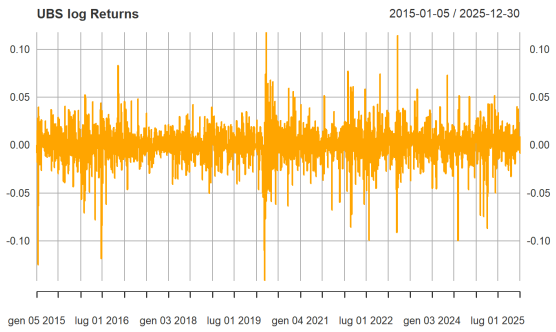


(b) Adjusted closing price of IWDC.SW.

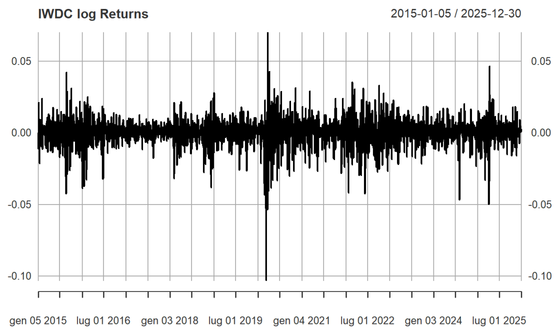


(c) Normalized prices with common starting value.

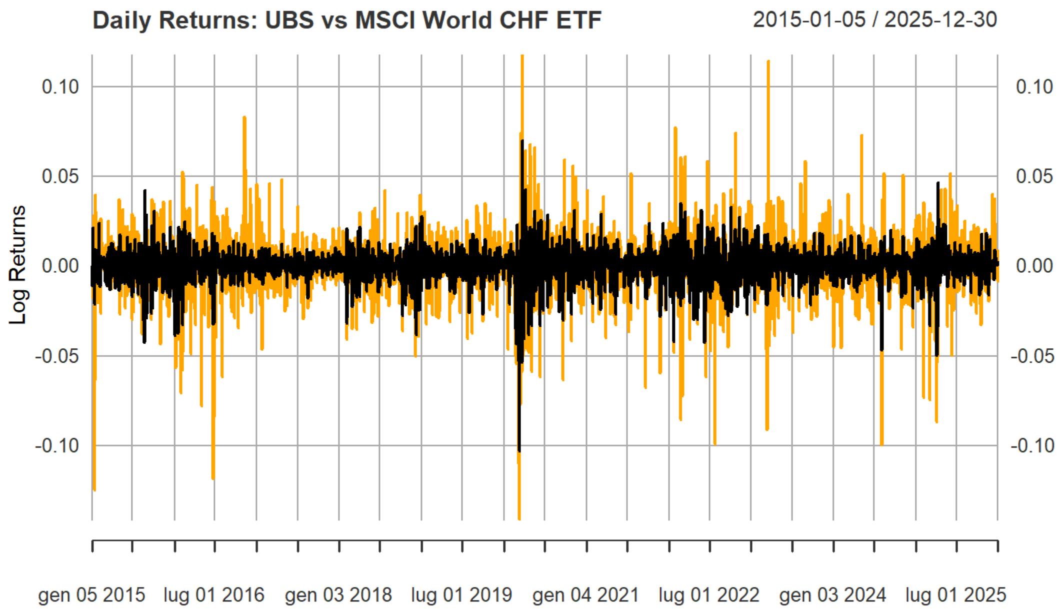
Figure 1: Price evolution of UBSG.SW and IWDC.SW from 2015 to 2025.



(a) Daily log returns of UBSG.SW.



(b) Daily log returns of IWDC.SW.



(c) Comparison of daily log returns of UBSG.SW and IWDC.SW.

Figure 2: Daily log returns of UBSG.SW and IWDC.SW from 2015 to 2025.

After plotting the price graphs and the return series, we can already see some key insights before doing further analysis. UBS appears to have performed better than the MSCI World ETF over the sample period, since its price series is generally higher. Looking at the return series, UBS also appears to show larger and more frequent swings, suggesting higher volatility compared to the ETF. We now move on to test these observations empirically using descriptive statistics and further data analysis, rather than relying only on visual inspection of the graphs.

2 Descriptive statistics and performance metrics

2.1 Mean, Std.Deviation, Ann.Volatility, Ann.Returns, Skewness, Kurtosis

Return definition and data frequency All computations are based on daily returns computed from adjusted prices, after aligning the two instruments on a common trading calendar and removing missing observations. Daily volatility and returns are annualized assuming 252 trading days per year.

Distribution shape: skewness and kurtosis Skewness and kurtosis are reported to assess departures from normality. Negative skewness indicates a heavier left tail (more extreme downside moves), while excess kurtosis signals fat tails. These moments motivate the use of non Gaussian risk measures and VaR models beyond the normal assumption.

Interpretation for risk analysis Mean and annualized return describe central tendency, while standard deviation and annualized volatility quantify risk. Together with skewness and kurtosis, these statistics provide a first diagnostic of asymmetry and tail risk, which is then explored more formally in the Stylized Facts and VaR sections.

Table 2: Descriptive Statistics of Daily Returns

	UBS	MSCI World ETF
Mean	0.0433%	0.0328%
Standard Deviation	1.8394%	0.9822%
Annualized Volatility	29.20%	15.59%
Annualized Return	10.91%	8.28%
Skewness	-0.607	-0.841
Kurtosis	9.88	11.30

Extreme daily returns:

- **UBS**: minimum = -14.16% on 12 March 2020, maximum = 11.80% on 24 March 2020
- **Ishares MSCI World ETF**: minimum = -10.31% on 12 March 2020, maximum = 7.02% on 24 March 2020

2.2 Correlation between UBS and MSCI World ETF

The linear correlation between the asset returns and the market ETF returns is defined as:

$$\rho_{X,Y} = \frac{\mathrm{Cov}(X,Y)}{\sigma_X\,\sigma_Y} \tag{1}$$

where:

- X is the return series of UBS, Y of ETF
- $\mathrm{Cov}(X,Y)$ is the covariance between the two return series
- σ_X and σ_Y are the standard deviations of the two return series

In practice, the correlation coefficient is computed in R using the builtin function `cor()` applied to the daily return series. The result is positive, showing a moderate correlation.

Table 3: Correlation between UBS and MSCI World ETF Returns

	Correlation
UBS vs MSCI World ETF	0.649

2.3 Risk adjusted metrics: Sharpe ratio and Sortino ratio

Raw returns do not account for the amount of risk taken to achieve them, so they are not sufficient to compare performance across assets. For this reason, we compute the Sharpe ratio and the Sortino ratio. The Sharpe ratio measures the average excess return over the risk free rate per unit of total volatility, treating both positive and negative fluctuations as risk. The Sortino ratio instead replaces total volatility with downside deviation, so it measures excess return relative only to negative returns. Together, these metrics provide a clearer comparison of performance by relating returns to the level and type of risk taken.

As the risk free rate, we use the 13 week US Treasury Bill yield (`ÎRX`) from Yahoo Finance, applied consistently to both UBS and the MSCI World ETF. For this we will compute it manually in R as follow, because we already have all the data needed:

Sharpe Ratio

$$\text{Sharpe} = \frac{R - r_f}{\sigma_R} \tag{2}$$

where:

- R is the return series
- r_f is the risk free rate
- σ_R is the standard deviation of returns

Table 4: Sharpe Ratio Comparison

	UBS	MSCI World ETF
Sharpe Ratio	0.251	0.362

The MSCI World ETF shows a higher Sharpe ratio than UBS, indicating better returns per unit of volatility.

Sortino Ratio

$$\text{Sortino} = \frac{R - r_f}{\sigma_D} \tag{3}$$

where:

- R is the return series
- r_f is the risk free rate
- σ_D is the standard deviations of negative returns

Table 5: Risk adjusted Performance Metrics

	UBS	MSCI World ETF
Sortino Ratio	0.343	0.486

The MSCI World ETF exhibits a higher Sortino ratio than UBS, indicating better downside risk adjusted performance. This result reflects the diversification benefits of the global ETF compared to the higher firm specific risk of UBS.

2.4 Drawdown and Calmar ratio

Risk adjusted ratios based on volatility do not fully capture the magnitude of large losses. For this reason, we also compute the maximum drawdown and the Calmar ratio. Maximum drawdown measures the largest peak to trough decline in the cumulative performance of the asset, while the Calmar ratio relates the annual return to this worst observed loss.

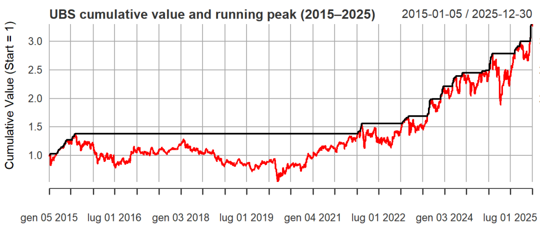
Drawdown

$$\text{Drawdown}_t = \frac{\text{Peak Value}_t - \text{Trough Value}_t}{\text{Peak Value}_t} \tag{4}$$

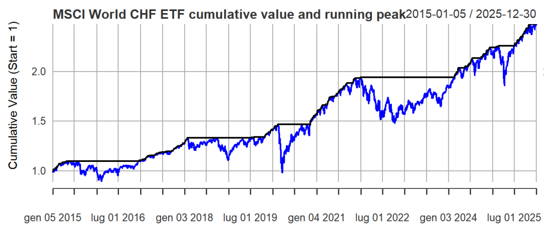
$$\text{Maximum Drawdown} = \max_t(\text{Drawdown}_t) \tag{5}$$

where:

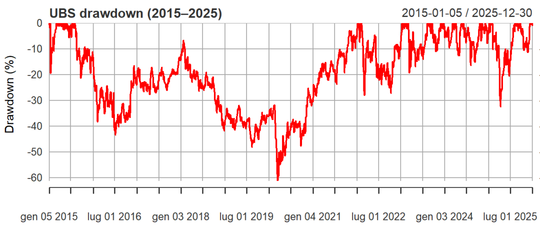
- Peak Value_t is the highest cumulative value reached before time t
- Trough Value_t is the lowest value after the peak
- Maximum Drawdown is the largest observed loss from a peak to a trough



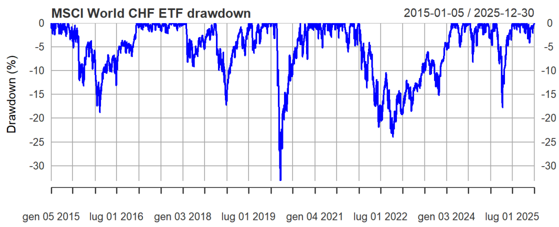
(a) UBS cumulative value and peaks



(b) MSCI World cumulative value and peaks



(c) UBS drawdown



(d) MSCI World drawdown

Figure 3: Cumulative performance, running peaks and drawdowns for UBS and Ishares MSCI World ETF

Table 6: Maximum Drawdown and Dates

	UBS	MSCI World ETF
Maximum Drawdown	-67.26%	-33.98%
Worst Day	16 March 2020	23 March 2020

Calmar Ratio

$$\text{Calmar} = \frac{R_{\text{ann}}}{|\text{MDD}|} \tag{6}$$

where:

- R_{ann} is the annualized returns
- $|\text{MDD}|$ is the absolute value of maximum drawdown

Given that we now have everything we need to compute the Calmar Ratio, the result is straight forwardly computed in R for both assets.

Table 7: Risk Adjusted Performance based on Maximum Drawdown

	UBS	MSCI World ETF
Calmar Ratio	0.179	0.250

The MSCI World ETF exhibits a higher Calmar ratio than UBS, indicating a better trade off between annual returns and extreme downside risk, which reflects the diversification benefits of the global ETF compared to a single stock.

2.5 Overall comparison: UBS vs MSCI World ETF

This section summarizes the main results from Tables 2 to 7.

Returns UBS shows a higher average return over the sample. The annualized return is 10.91% for UBS and 8.28% for the MSCI World ETF.

Risk UBS is much more volatile. The annualized volatility is 29.20% for UBS and 15.59% for the ETF. This is consistent with UBS being a single stock, while the ETF is diversified.

Risk adjusted performance All risk adjusted ratios favor the MSCI World ETF:

- Sharpe ratio: 0.251 (UBS) vs 0.362 (ETF)
- Sortino ratio: 0.343 (UBS) vs 0.486 (ETF)
- Calmar ratio: 0.179 (UBS) vs 0.250 (ETF)

These results indicate that the ETF delivers better performance per unit of risk, especially when focusing on downside risk and large losses.

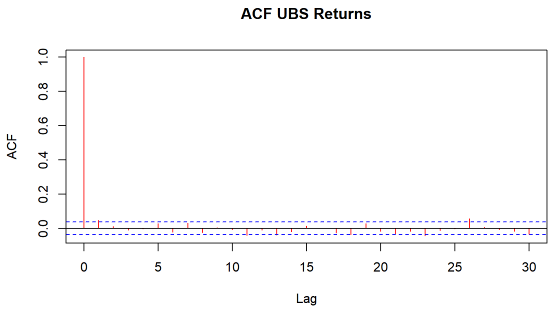
Tails and drawdowns Both series show fat tails and negative skewness, but UBS has higher downside risk. UBS experienced a maximum drawdown of 67.26% (16 March 2020), while the ETF reached a maximum drawdown of 33.98% (23 March 2020). This large difference explains why the Calmar ratio is lower for UBS.

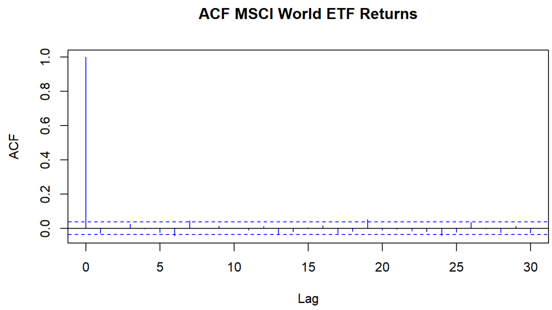
Correlation The correlation between UBS and the MSCI World ETF is 0.649. This means UBS is strongly linked to the global equity market movements, but it still has additional risk.

3 Stylized facts of assets returns

3.1 Unpredictability and Autocorrelation of Returns

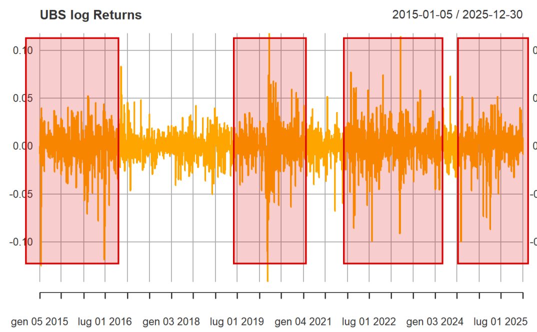
Returns typically show little or no linear autocorrelation, meaning that past returns do not provide a reliable signal about the direction of future returns. This is consistent with the weak form of Fama's efficient market hypothesis, where current prices already incorporate information from past prices. Autocorrelation can be examined using the autocorrelation function, which reports the Pearson correlation between today's return and returns from previous days at different lags. If these correlations were large and statistically significant, this would suggest predictability in returns. In practice, for liquid assets, the autocorrelations of daily returns are usually close to zero, so forecasting whether the price will go up or down in the next period is not systematically possible. This can be shown by this plots generated in r with the autocorrelation function.



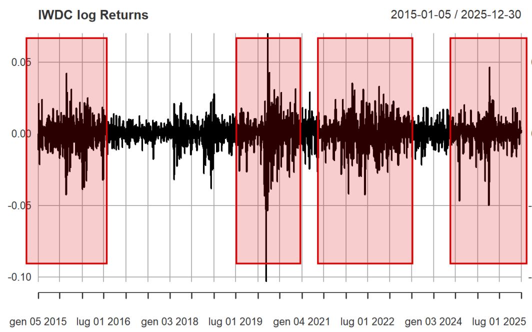


3.2 Volatility clustering and Autocorrelation of absolute Returns

What we found does not mean that everything is random: the magnitude of returns often shows dependence. In particular absolute returns can exhibit autocorrelation, which is linked to volatility clustering, where high volatility tends to be followed by high volatility and low volatility tends to be followed by low volatility:



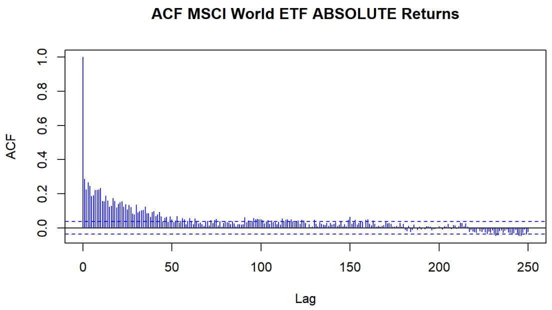
(a) UBS log returns with volatility clustering



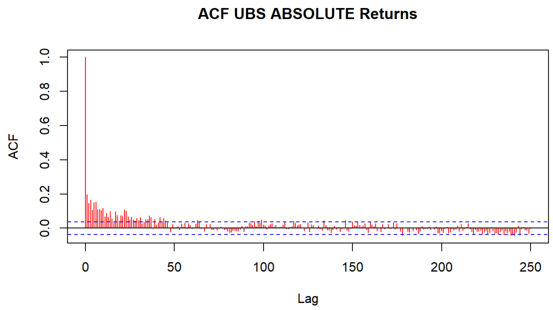
(b) IWDC log returns with volatility clustering

Figure 5: Volatility clustering in log returns of UBS and IWDC.

Turning back to the autocorrelations function that we used before for the linear returns, we can now apply the same function to the absolute returns and we end up with this graphs:



(a) ACF absolute returns IWDC



(b) ACF absolute returns UBS

Figure 6: Persistence of volatility confirmed by autocorrelation of absolute returns.

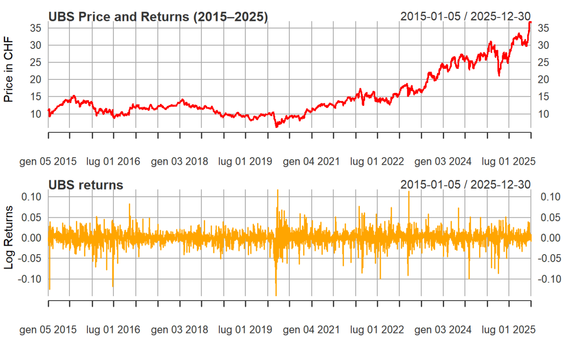
3.3 Leverage effect and asymmetric volatility

The leverage effect describes a negative relationship between an asset's returns and its volatility. When returns are negative, volatility usually increases, often sharply, and it can stay high for some time.

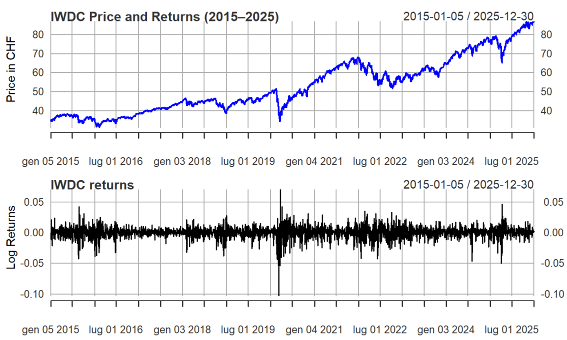
This reaction is asymmetric. Price drops are more likely to be followed by large volatility spikes than price rises of the same size. When prices move up, volatility is often lower and more stable.

To show this clearly, we include 2 figures with two panels. In the top panels we plot the price level over time. In the bottom panels we plot the returns over the same dates. With this layout we can easily compare the two series.

When the price trends downward, the return series below usually shows larger swings and stronger clustering of high volatility. When the price trend is upward or stable, the return series is often calmer, with smaller fluctuations. This visual check does not measure the effect with a formal test, but it gives a clear and intuitive picture of the pattern we expect:



(a) UBS price and return series (2015–2025)



(b) IWDC price and return series (2015–2025)

3.4 Non Normality of Returns

Many statistical and econometric models assume that financial data follow a normal distribution. This assumption rarely holds in practice. Empirical evidence shows that asset returns tend to exhibit heavy tails and excess kurtosis, meaning that extreme events occur more frequently than predicted by a Gaussian model.

To illustrate this feature, we compare the kernel density of log returns for UBS and the MSCI World CHF ETF (IWDC) with a normal distribution having the same mean and standard deviation.

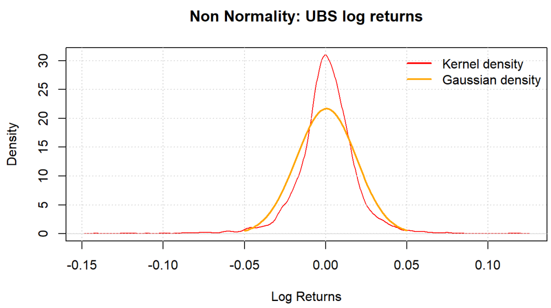


Figure 8: UBS: kernel vs Gaussian density

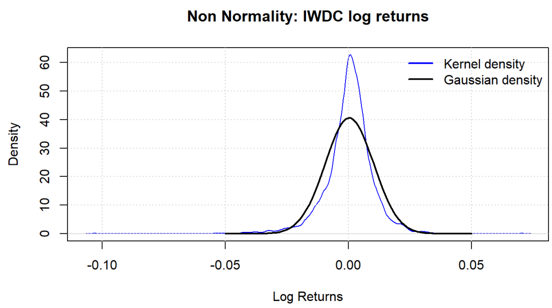


Figure 9: IWDC: kernel vs Gaussian density

The kernel densities are clearly more peaked around the mean and display thicker tails compared to the Gaussian benchmark. This indicates excess kurtosis, that we had already found before numerically. These results have important implications for risk management. If returns were normally distributed, extreme losses would be very unlikely. However, the presence of fat tails implies that large market moves occur more often than predicted by standard models.

3.5 QQ plots of the returns

Another way to assess normality is through quantile vs quantile plots. A QQ plot compares the empirical quantiles of the returns with the theoretical quantiles of a normal distribution. If returns were normally distributed, the points would lie approximately on a straight line. We generate these plots in `R` using the functions `qqnorm()` and `qqline()`.

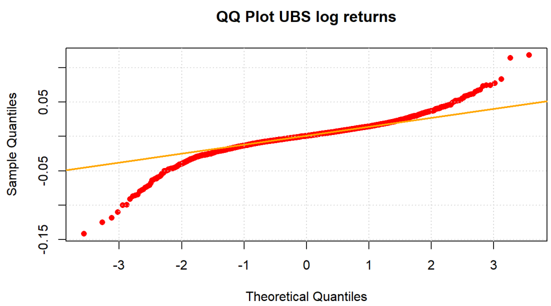


Figure 10: QQ plot of UBS log returns

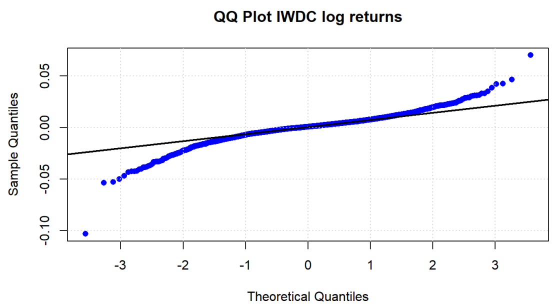


Figure 11: QQ plot of IWDC log returns

Both series deviate from the straight line in the tails, as we would expect by having looked at the density plots. The deviation is more for UBS, reflecting the higher exposure of an individual stock to market and sector specific shocks, while the globally diversified ETF shows a less strong but still clear non normality.

4 Value at Risk (95% confidence level) with different methods

In this section we will go through computing the VaR from the data in three simple different ways and then showcasing the differences in the results. Value at Risk measures the potential loss of an asset over a given time period at a chosen confidence level. A daily VaR at 95 percent means there is a 95 percent probability that the loss in one day will stay below a certain threshold, and a 5 percent probability that it will exceed it.

4.1 Historical VaR

The Historical Var is extracted from the sample data by looking at the 5 percent of daily returns.

	Historical VaR (95%)	**Historical VaR (95%) in %**
UBS	0.027076	2.7076%
IWDC	0.015758	1.5758%

Table 8: Historical Value at Risk computed using the `quantile()` function in R

4.2 Gaussian VaR

Even though returns are not normally distributed, we compute anyways the Gaussian VaR as a benchmark. We then use Cornish–Fisher VaR, which corrects the normal quantile for skewness and kurtosis, providing a more realistic estimate of tail risk.

$$\mathrm{VaR}_{Gaussian} = -\left(\mu + \sigma\, z\right)$$

where μ is the mean of returns, σ is the standard deviation of returns, and z is the z-score of the standard normal distribution. For a 95% confidence level, z= 1,645

Translating this into R gives us:

	Gaussian VaR (95%)	**Gaussian VaR (95%) in %**
UBS	0.029823	2.9823%
IWDC	0.015827	1.5827%

Table 9: Gaussian Value at Risk at the 95% confidence level, computed using the mean and standard deviation of log returns with the normal quantile in R.

4.3 Modified Cornish-Fisher VaR

The Modified Cornish-Fisher VaR adjusts the z score based on the real values of the skewness S and kurtosis K of our returns and thus it is not based on the normal distribution default values.

$$\mathrm{VaR}_{CF} = -\left(\mu + \sigma\, \tilde{z}_\alpha\right)$$

where:

$$\tilde{z}_\alpha = z_\alpha + \frac{1}{6}(z_\alpha^2 - 1)S + \frac{1}{24}(z_\alpha^3 - 3z_\alpha)(K-3) - \frac{1}{36}(2z_\alpha^3 - 5z_\alpha)S^2$$

Going into R and using our before found Skewness and Kurtosis we obtain:

	Cornish-Fisher VaR (95%)	Cornish-Fisher VaR (95%) in %
UBS	0.030316	3.0316%
IWDC	0.016407	1.6407%

Table 10: Cornish-Fisher Value at Risk at the 95% confidence level, computed using skewness and kurtosis adjustments in R.

4.4 Three VaRs Comparison

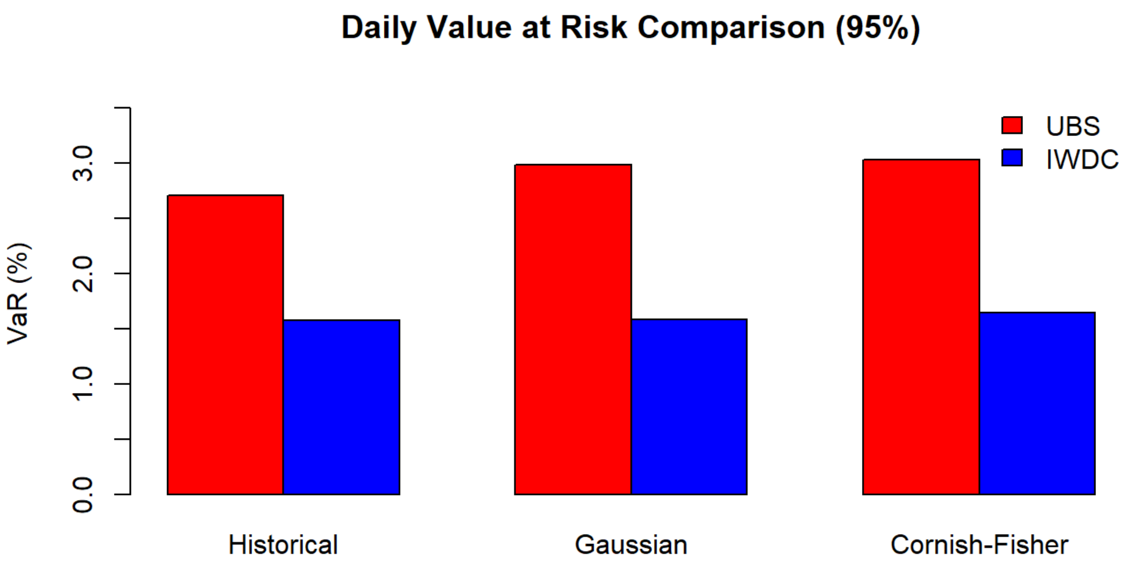


Figure 12: Comparison of the three VaRs for UBS and MSCI World CHF ETF.

	Historical	Gaussian	Cornish-Fisher
UBS	2.71%	2.98%	**3.03%**
IWDC	1.58%	1.58%	**1.64%**

Table 11: Comparison of Historical, Gaussian and Cornish-Fisher Value at Risk in percentage terms. The highest VaR for each asset is highlighted in bold.

UBS has a higher VaR than IWDC with all three methods, so it has higher downside risk. For both assets, Cornish Fisher VaR is the largest, Gaussian VaR is smaller, and Historical VaR is usually in between. This shows that the IWDC is safer and that adjusting for skewness and fat tails increases the estimated risk.

5 Volatility Forecasting and GARCH based VaR

In financial data, volatility is not constant and tends to cluster, as we already observed in the returns and autocorrelation plots. For this reason, we first estimate volatility and then use it to build VaR forecasts.

5.1 Forecasting Volatility with Sample Volatility

Before estimating volatility let's **just plot the actual one from the two assets on a 90 day rolling window:**

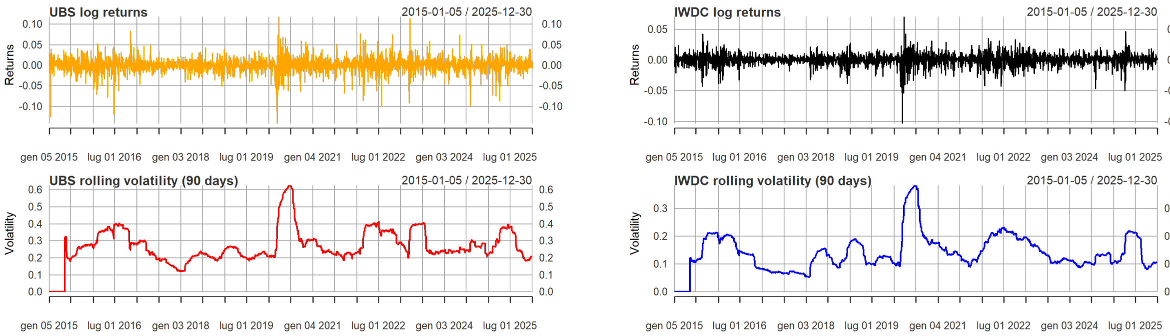
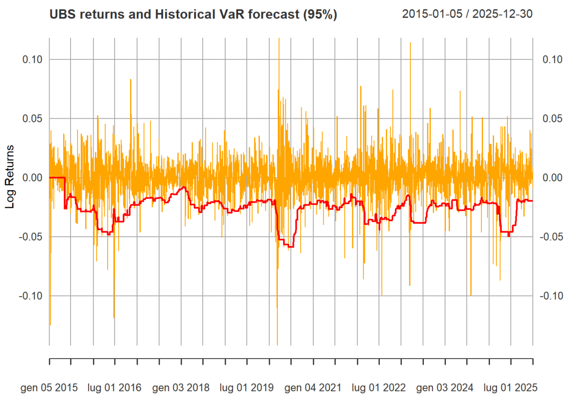


Figure 13: Rolling volatility with a 90 day window for UBS and IWDC. Periods of high volatility tend to cluster in time.

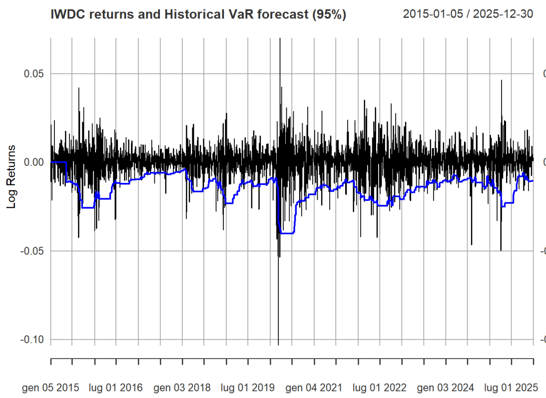
What is the procedure about:
We compute VaR forecasts under different methods all with a rolling window, using the Historical, Gaussian, Cornish Fisher, and GARCH based Var. We then compare their performance. To evaluate the forecasts, we count exceedances, that is, how often the realized return is below the predicted VaR threshold. For a 95 percent VaR, exceedances should occur about 5 percent of the time. If violations are too frequent or clustered in time, the model is inadequate.

Violations occure when the returns are under the VaR line.

5.2 Historical VaR: Forecast and Violation Analysis



UBS Historical VaR Forecast (95%)



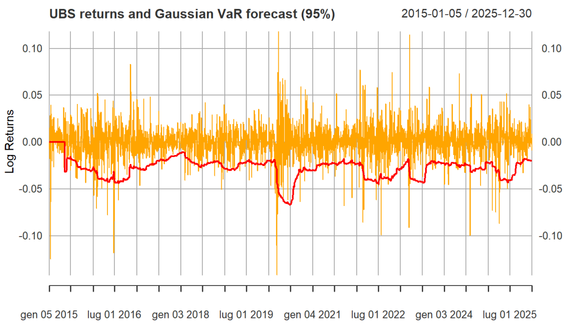
IWDC Historical VaR Forecast (95%)

Figure 14: Historical VaR forecasts for UBS and MSCI World ETF.

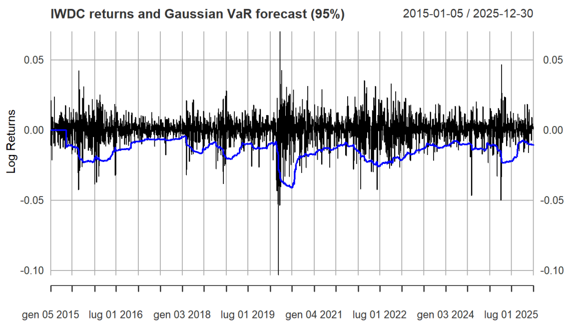
Historical VaR Backtesting Results		
	Violation Rate	**Number of Violations**
UBS	5.83%	156
IWDC	5.72%	153

Table 12: The violation rate is the proportion of days where losses exceeded the 95% VaR estimate. Since the confidence level is 95%, the expected violation rate is about 5%; values above 5% indicate that VaR underestimates risk.

5.3 Gaussian VaR: Forecast and Violation Analysis



UBS Gaussian VaR Forecast (95%)



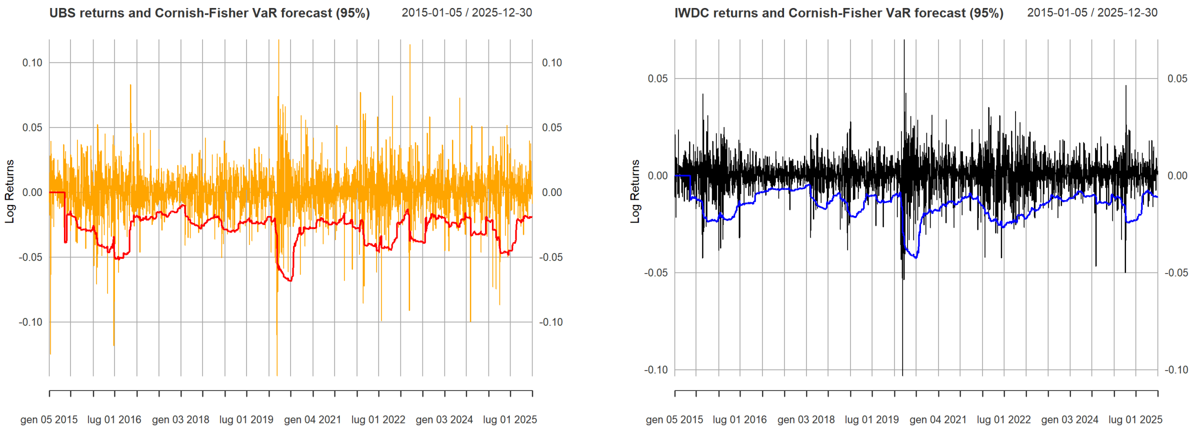
IWDC Gaussian VaR Forecast (95%)

Figure 15: Gaussian VaR forecasts for UBS and MSCI World ETF.

Gaussian VaR Backtesting Results		
	Violation Rate	**Number of Violations**
UBS	5.78%	157
IWDC	6.25%	167

Table 13: The violation rate is the proportion of days where losses exceeded the 95% Gaussian VaR estimate. Since the confidence level is 95%, the expected violation rate is about 5%; values above 5% indicate that the Gaussian VaR underestimates risk.

5.4 Cornish-Fisher VaR: Forecast and Violation Analysis



UBS Cornish-Fisher VaR Forecast (95%) IWDC Cornish-Fisher VaR Forecast (95%)

Figure 16: Cornish-Fisher VaR forecasts for UBS and MSCI World ETF.

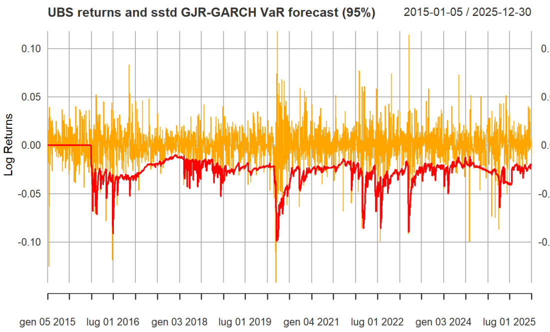
Cornish-Fisher VaR Backtesting Results		
	Violation Rate	**Number of Violations**
UBS	5.57%	149
IWDC	5.39%	144

Table 14: The violation rate is the proportion of days where losses exceeded the 95% Cornish-Fisher VaR estimate. Since the confidence level is 95%, the expected violation rate is about 5%; values above 5% indicate that VaR underestimates risk.

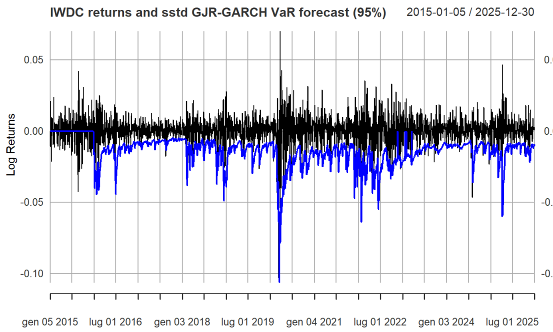
5.5 GARCH Based VaR: Forecast and Violation Analysis

Unlike Historical and Gaussian VaR, which assume constant volatility within the rolling window, GARCH models allow volatility to change over time. Financial returns such as UBS and the MSCI World ETF exhibit volatility clustering, making GARCH a more realistic framework for risk forecasting.

We estimate a rolling GARCH(1,1) model with skewed Student-t innovations using the `rugarch` package and compute one-day-ahead 95% VaR forecasts. These forecasts are evaluated using violation rates, as in the previous sections.



UBS GARCH VaR Forecast (95%)



IWDC GARCH VaR Forecast (95%)

Figure 17: GARCH based VaR forecasts for UBS and MSCI World ETF.

GARCH VaR Backtesting Results		
	Violation Rate	**Number of Violations**
UBS	5.2%	131
IWDC	5.5%	138

Table 15: The violation rate is the proportion of days where losses exceeded the 95% GARCH VaR estimate. Since the confidence level is 95%, the expected violation rate is about 5%; values above 5% indicate that VaR underestimates risk.

Comparison of VaR Models				
	UBS		**IWDC ETF**	
Model	Violation Rate	Violations	Violation Rate	Violations
Historical VaR	5.83%	156	5.72%	153
Gaussian VaR	5.78%	157	6.25%	167
Cornish-Fisher VaR	5.57%	149	5.39%	144
GARCH VaR	5.20%	131	5.50%	138
Expected (95% VaR)	5% violation rate			

Table 16: Comparison of Historical, Gaussian, Cornish-Fisher and GARCH VaR models for UBS and MSCI World ETF.

5.6 Interpretation of differences

Across all models, violation rates are slightly above the theoretical 5% level, indicating that each VaR specification underestimates tail risk to some extent. The Gaussian VaR performs worst, especially for the MSCI World ETF, reflecting the inability of the normal distribution to capture fat tails. Cornish-Fisher adjustments improve the results by incorporating skewness and kurtosis. The GARCH-based VaR provides the most accurate forecasts, with violation rates closest to 5%, confirming that modelling time varying volatility is essential for assets such as UBS and the MSCI World ETF, which exhibit volatility clustering.

6 Portfolio Management: VaR-Based Allocation Between UBS and the MSCI World ETF

After estimating volatility dynamics and computing Value at Risk forecasts for UBS and the MSCI World CHF ETF, we implement a simple risk-based portfolio allocation rule. The goal is not to maximize returns, but to dynamically allocate capital according to the predicted risk of each asset.

This approach links econometric modeling with practical investment decisions: assets with lower expected downside risk receive a higher allocation, while assets with higher predicted risk are reduced in weight.

6.1 Strategy Definition: VaR Target and Rebalancing Rule

We construct a two-asset portfolio composed of UBS stock and the MSCI World CHF ETF. At each rebalancing date, we estimate rolling GARCH(1,1) models with Student-t innovations over a window of 750 trading days.

Using the one-day-ahead forecasts of conditional mean and volatility, we compute the parametric VaR at the 95% confidence level for each asset. Portfolio weights are then set inversely proportional to the predicted VaR:

$$w_{i,t} = \frac{1/\mathrm{VaR}_{i,t}}{\sum_j 1/\mathrm{VaR}_{j,t}}$$

This rule assigns larger weights to assets with lower expected downside risk. The portfolio is rebalanced daily in the baseline specification, although less frequent rebalancing could reduce transaction costs. For this paper: No transaction costs are included, Portfolio weights are constrained to be positive and to sum to one. Cash is not explicitly modeled; instead, all capital is allocated between UBS and the ETF.

6.2 Visualization of the Results

To illustrate the performance of the risk-managed strategy, we plot the cumulative value of the VaR-based portfolio constructed from UBS and the MSCI World CHF ETF. The portfolio starts with an initial value of one and evolves according to daily returns generated by the GARCH–VaR allocation rule.

The equity curve reflects the dynamic adjustment of portfolio weights in response to changes in predicted risk. During periods of higher estimated volatility, exposure shifts toward the less risky asset, leading to smoother performance. During calmer periods, the strategy increases exposure to the riskier asset, allowing participation in positive market movements.

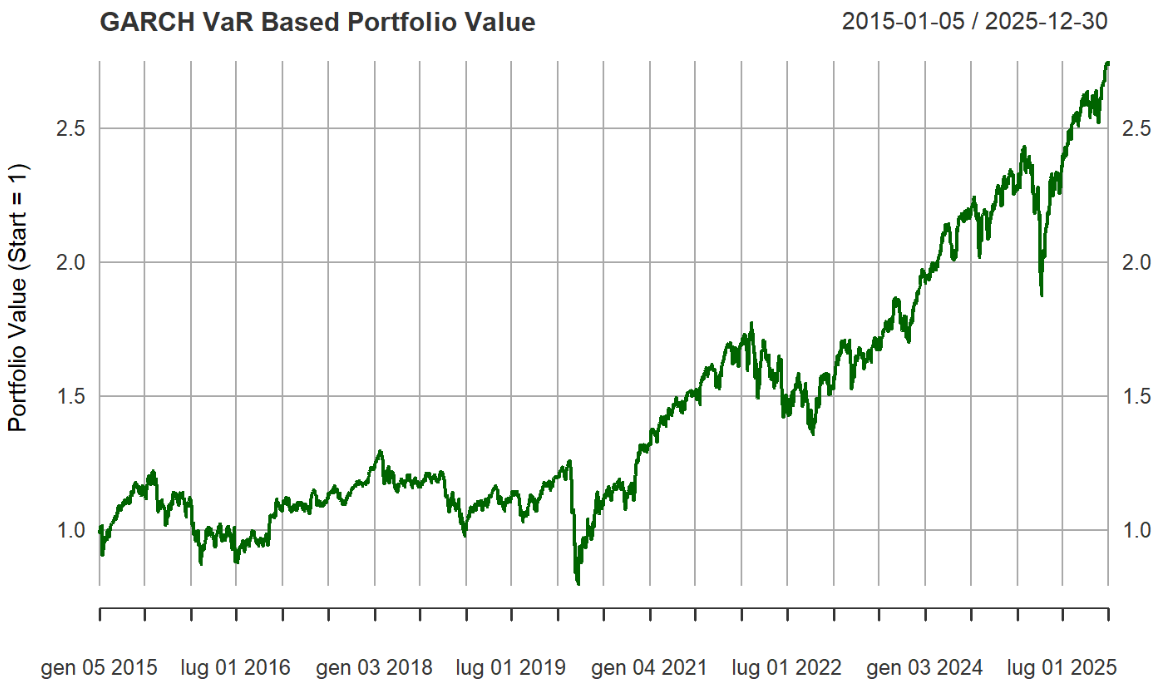


Figure 18: Portfolio of UBS and ETF performance

6.3 Performance Evaluation: Managed Portfolio vs Single Assets

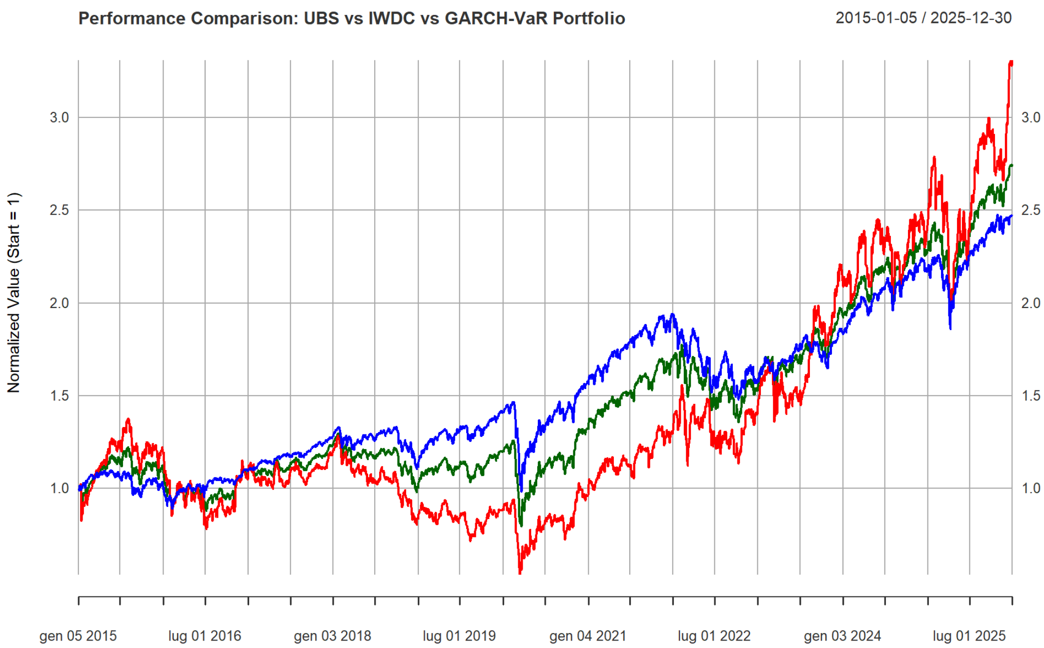


Figure 19: Cumulative value of the GARCH–VaR based portfolio built from UBS and the MSCI World CHF ETF over the period 2015–2025. The portfolio is rebalanced using rolling GARCH forecasts and Value at Risk estimates. The initial value is normalized to one.

Color mapping in the plot:

GARCH–VaR portfolio, UBS, MSCI World ETF (IWDC)

UBS exhibits the highest cumulative return, reflecting its higher exposure to firm-specific and sector-specific risk. However, its equity curve shows pronounced drawdowns and strong volatility, particularly during periods of market stress. The MSCI World ETF displays smoother dynamics, consistent with global diversification, but delivers lower long-term growth compared to UBS.

The GARCH–VaR portfolio lies between the two assets. Its performance is lower than UBS in terms of raw cumulative return, which is expected because the strategy reduces exposure when predicted volatility increases. During turbulent periods, capital shifts toward the less risky asset, limiting losses and stabilizing the portfolio path. As a result, the VaR-based strategy produces a smoother equity curve and smaller drawdowns than UBS, while still participating in market uptrends.

These results are consistent with the objective of VaR-based allocation. The strategy is designed to control downside risk rather than maximize returns. Therefore, performance should be evaluated not only through cumulative return but also through volatility, Sharpe ratio, maximum drawdown, and VaR violations. In many cases, a lower return combined with significantly reduced risk represents an economically meaningful improvement.

Overall, the evidence suggests that volatility forecasting through GARCH models can be translated into practical portfolio allocation rules that improve risk control, even if they do not always outperform a high-risk asset such as UBS in terms of raw returns.

6.4 Limitations of the Method

Several limitations should be noted.

First, the allocation rule ignores time varying correlations between UBS and the ETF. A more realistic approach would estimate a dynamic covariance matrix or simulate joint return distributions.

Second, the strategy does not include transaction costs, bid ask spreads, or liquidity constraints. Frequent rebalancing may significantly reduce net performance.

Third, VaR is not a coherent risk measure and does not capture tail risk beyond the chosen confidence level. Extreme events may still generate large losses.

Finally, GARCH models rely on historical data and may fail to predict structural breaks or regime changes.

7 Conclusion

This study analyzes the performance and risk characteristics of UBS relative to a global equity benchmark and demonstrates how volatility modeling can be integrated into portfolio allocation.

We show that stylized facts such as volatility clustering motivate the use of GARCH models, which in turn provide forecasts for Value at Risk. These forecasts can be translated into practical allocation rules that dynamically adjust exposure to risky assets.

While the VaR-based portfolio improves risk control compared to a pure UBS investment, its performance depends on model assumptions and market conditions. Future research could extend this framework by incorporating multivariate GARCH models, transaction costs, or alternative risk measures or introducing cash and bonds for "risk free" assets.

A R Code used in this paper

This is the R Code Used:

```
#####SETUP#####

library(quantmod)

#Downloading tickers:UBSG.SW and IWDC.SW from yahoo
getSymbols("UBSG.SW", src="yahoo")
getSymbols("IWDC.SW", src="yahoo")

# Renaming the Dataset and removing not necessary columns,
# after checking the columns name i keep only the adjusted prices
# to account for stock splits, dividends and reverse split.
UBS = UBSG.SW
rm(UBSG.SW)
colnames(UBS)
UBS <- UBS[, -c(1:5)]

IWDC = IWDC.SW
rm(IWDC.SW)
colnames(IWDC)
IWDC <- IWDC[, -c(1:5)]

#setting start and finish date
IWDC <- IWDC["2015-01-01/2025-12-31"]
UBS  <- UBS["2015-01-01/2025-12-31"]

tail(UBS)
tail(IWDC)

#checking if all the dates are the same
all(index(IWDC) == index(UBS))
#They are not the same so we keep only the same dates
common_dates <- intersect(index(IWDC), index(UBS))
IWDC <- IWDC[common_dates]
UBS  <- UBS[common_dates]
# Last check
all(index(IWDC) == index(UBS))
rm(common_dates)
#given TRUE output, we now have the dataset with just the common dates

#plotting the adj. prices
plot(UBS, col="red", lwd=2, main="UBS Group AG Adjusted Price (2015-2025)",
    ylab="Price in CHF", xlab="Date")
grid()

plot(IWDC, col="blue", lwd=2, main="IWDC Adjusted Price (2015-2025)", ylab="
    Price in CHF", xlab="Date")
grid()

#Plotting the normalized prices together
plot(UBS/UBS[[1]], col="red", lwd=2, main="Normalized Prices: UBS vs IWDC
    (2015-2025)", ylab="Normalized Price (Start = 1)", xlab="Date")
grid()
lines(IWDC/IWDC[[1]], col="blue", lwd=2)

#calculating returns
UBS_returns = dailyReturn(UBS, type="log", leading=TRUE)
IWDC_returns = dailyReturn(IWDC, type="log", leading=TRUE)

#plotting returns
```

```
plot(UBS_returns, col="orange", main="UBS log Returns")
plot(IWDC_returns, col="black", main="IWDC log Returns")

#together
plot(UBS_returns, col="orange", lwd=2, main="Daily Returns: UBS vs MSCI World
    CHF ETF", ylab="Log Returns", xlab="Date")
grid()
lines(IWDC_returns, col="black", lwd=2)

##### DESCRIPTIVE STATISTICS #####

#mean daily return
UBS_mean_returns = mean(UBS_returns)
IWDC_mean_returns = mean(IWDC_returns)
print(paste("UBS mean return -->", UBS_mean_returns))
print(paste("IWDC mean return -->", IWDC_mean_returns))

#standard deviation
UBS_sd = sd(UBS_returns)
IWDC_sd = sd(IWDC_returns)
print(paste("UBS standard deviation -->", UBS_sd))
print(paste("IWDC standard deviation -->", IWDC_sd))

#annualized volatility
UBS_ann_vol = UBS_sd * sqrt(252)
IWDC_ann_vol = IWDC_sd * sqrt(252)
print(paste("UBS annualized volatility -->", UBS_ann_vol))
print(paste("IWDC annualized volatility -->", IWDC_ann_vol))

#annualized return linear
UBS_ann_return = UBS_mean_returns * 252
IWDC_ann_return = IWDC_mean_returns * 252
print(paste("UBS annualized return -->", UBS_ann_return))
print(paste("IWDC annualized return -->", IWDC_ann_return))

#annualized return compounded
UBS_ann_return_comp = prod(1 + UBS_returns)^(252/length(UBS_returns)) - 1
IWDC_ann_return_comp = prod(1 + IWDC_returns)^(252/length(IWDC_returns)) - 1
print(paste("UBS annualized return compounded -->", UBS_ann_return_comp))
print(paste("IWDC annualized return compounded -->", IWDC_ann_return_comp))

#min and max returns with dates
UBS_min = min(UBS_returns)
IWDC_min = min(IWDC_returns)
UBS_max = max(UBS_returns)
IWDC_max = max(IWDC_returns)

UBS_min_date = index(UBS_returns)[which.min(UBS_returns)]
IWDC_min_date = index(IWDC_returns)[which.min(IWDC_returns)]
UBS_max_date = index(UBS_returns)[which.max(UBS_returns)]
IWDC_max_date = index(IWDC_returns)[which.max(IWDC_returns)]

print(paste("UBS min return -->", UBS_min, "on", UBS_min_date))
print(paste("IWDC min return -->", IWDC_min, "on", IWDC_min_date))
print(paste("UBS max return -->", UBS_max, "on", UBS_max_date))
print(paste("IWDC max return -->", IWDC_max, "on", IWDC_max_date))

#before calulating skewnewss an kurtosis
#i am installing a library to not manually calculate the results
#install.packages("moments")
library(moments)

#skewness
```

```
UBS_skew = skewness(UBS_returns)
IWDC_skew = skewness(IWDC_returns)
print(paste("UBS skewness -->", UBS_skew))
print(paste("IWDC skewness -->", IWDC_skew))

#kurtosis (NON EXCESS, so i have to detract 3)
UBS_kurt = kurtosis(UBS_returns)
IWDC_kurt = kurtosis(IWDC_returns)
print(paste("UBS kurtosis -->", UBS_kurt))
print(paste("IWDC kurtosis -->", IWDC_kurt))

#correlation between UBS and ETF
correlation = cor(UBS_returns, IWDC_returns)
print(paste("Correlation UBS vs MSCI World ETF -->", correlation))

#####Risk Adjusted Metrics ######

#risk free rate from Yahoo Finance
#(13 week T bill of USA, because of global etf)
getSymbols("^IRX", src="yahoo")
rf_annual = Cl(IRX)/100
rf_daily = (1 + rf_annual)^(1/252) - 1

#align dates
data = na.omit(merge(UBS_returns, IWDC_returns, rf_daily))
UBS_r = data[,1]
IWDC_r = data[,2]
rf = data[,3]

#excess returns
UBS_excess = UBS_r - rf
IWDC_excess = IWDC_r - rf

#Sharpe ratio
UBS_sharpe = mean(UBS_excess)/sd(UBS_excess)*sqrt(252)
IWDC_sharpe = mean(IWDC_excess)/sd(IWDC_excess)*sqrt(252)
print(paste("UBS Sharpe ratio -->", UBS_sharpe))
print(paste("IWDC Sharpe ratio -->", IWDC_sharpe))

#Sortino ratio
UBS_downside = UBS_excess
UBS_downside[UBS_downside > 0] = 0
UBS_downside_dev = sqrt(mean(UBS_downside^2))

IWDC_downside = IWDC_excess
IWDC_downside[IWDC_downside > 0] = 0
IWDC_downside_dev = sqrt(mean(IWDC_downside^2))

UBS_sortino = mean(UBS_excess)/UBS_downside_dev*sqrt(252)
IWDC_sortino = mean(IWDC_excess)/IWDC_downside_dev*sqrt(252)
print(paste("UBS Sortino ratio -->", UBS_sortino))
print(paste("IWDC Sortino ratio -->", IWDC_sortino))

#####Drawdowns and Peaks and calmar######

#cumulative value (starting from 1)
UBS_value = exp(cumsum(UBS_returns))
IWDC_value = exp(cumsum(IWDC_returns))

#drawdown series
UBS_peak = cummax(UBS_value)
IWDC_peak = cummax(IWDC_value)

```

```
UBS_dd = (UBS_value - UBS_peak)/UBS_peak
IWDC_dd = (IWDC_value - IWDC_peak)/IWDC_peak

#maximum drawdown and dates
UBS_mdd = min(UBS_dd)
IWDC_mdd = min(IWDC_dd)

UBS_mdd_date = index(UBS_dd)[which.min(UBS_dd)]
IWDC_mdd_date = index(IWDC_dd)[which.min(IWDC_dd)]

print(paste("UBS max drawdown -->", UBS_mdd, "on", UBS_mdd_date))
print(paste("IWDC max drawdown -->", IWDC_mdd, "on", IWDC_mdd_date))

#####PLOTS PEAKS AND DRAWDOWNS

#UBS: cumulative value with running peak
plot(UBS_value, col="red", lwd=2, main="UBS cumulative value and running peak
    (2015-2025)", ylab="Cumulative Value (Start = 1)", xlab="Date")
grid()
lines(UBS_peak, col="black", lwd=2)

#IWDC: cumulative value with running peak
plot(IWDC_value, col="blue", lwd=2, main="MSCI World CHF ETF cumulative value
    and running peak", ylab="Cumulative Value (Start = 1)", xlab="Date")
grid()
lines(IWDC_peak, col="black", lwd=2)

#UBS drawdown
plot(UBS_dd*100, col="red", lwd=2, main="UBS drawdown (2015-2025)", ylab="
    Drawdown (%)", xlab="Date")
grid()
abline(h=0)

#IWDC drawdown
plot(IWDC_dd*100, col="blue", lwd=2, main="MSCI World CHF ETF drawdown", ylab
    ="Drawdown (%)", xlab="Date")
grid()
abline(h=0)

#Calmar ratio using annual return
UBS_calmar = UBS_ann_return/abs(UBS_mdd)
IWDC_calmar = IWDC_ann_return/abs(IWDC_mdd)
print(paste("UBS Calmar ratio -->", UBS_calmar))
print(paste("IWDC Calmar ratio -->", IWDC_calmar))

#####Autocorrelation#####
#normal returns
acf(UBS_returns, lag=30, main="ACF UBS Returns", col="red")
acf(IWDC_returns, lag=30, main="ACF MSCI World ETF Returns", col="blue")

#absolute returns
acf(abs(UBS_returns), lag=250, main="ACF UBS ABSOLUTE Returns", col="red")
acf(abs(IWDC_returns), lag=250, main="ACF MSCI World ETF ABSOLUTE Returns",
    col="blue")

#####Leverage Effect#####

#IWDC
par(mfrow=c(2,1), mar=c(3,4,2,2))
plot(IWDC, col="blue", lwd=2, main="IWDC Price and Returns (2015-2025)", ylab
    ="Price in CHF", xlab="")
grid()
plot(IWDC_returns, main="IWDC returns", col="black", lwd=1.5, ylab="Log
```

```
    Returns", xlab="Date")
grid()
abline(h=0)
par(mfrow=c(1,1))

#UBS
par(mfrow=c(2,1), mar=c(3,4,2,2))
plot(UBS, col="red", lwd=2, main="UBS Price and Returns (2015-2025)", ylab="
    Price in CHF", xlab="")
grid()
plot(UBS_returns, main="UBS returns", col="orange", lwd=1.5, ylab="Log
    Returns", xlab="Date")
grid()
abline(h=0)
par(mfrow=c(1,1))

##### NON NORMALITY #####

# GAUSSIAN DENSITY UBS
mu_UBS = UBS_mean_returns
sigma_UBS = UBS_sd

x_UBS = seq(-0.05, 0.05, 0.001)
Gaussian_density_UBS = dnorm(x_UBS, mu_UBS, sigma_UBS)
kernel_density_UBS = density(UBS_returns)

plot(kernel_density_UBS, main="Non Normality: UBS log returns", xlab="Log
    Returns", ylab="Density", col="red")
grid()
lines(x_UBS, Gaussian_density_UBS, col="orange", lwd=2)
legend("topright", legend=c("Kernel density", "Gaussian density"), col=c("red
    ","orange"), lwd=2, bty="n")

# GAUSSIAN DENSITY IWDC
mu_IWDC = IWDC_mean_returns
sigma_IWDC = IWDC_sd

x_IWDC = seq(-0.05, 0.05, 0.001)
Gaussian_density_IWDC = dnorm(x_IWDC, mu_IWDC, sigma_IWDC)
kernel_density_IWDC = density(IWDC_returns)

plot(kernel_density_IWDC, main="Non Normality: IWDC log returns", xlab="Log
    Returns", ylab="Density", col="blue")
grid()
lines(x_IWDC, Gaussian_density_IWDC, col="black", lwd=2)
legend("topright", legend=c("Kernel density","Gaussian density"), col=c("blue
    ","black"), lwd=2, bty="n")

#QQ Plots
#UBS QQ
qqnorm(UBS_returns, main="QQ Plot UBS log returns", col="red", pch=16)
grid()
qqline(UBS_returns, col="orange", lwd=2)

#IWDC QQ
qqnorm(IWDC_returns, main="QQ Plot IWDC log returns", col="blue", pch=16)
grid()
qqline(IWDC_returns, col="black", lwd=2)

#####Value at Risk sample data #####

#Historical Var
hist_var_UBS = -quantile(UBS_returns, 0.05)
```

```
hist_var_IWDC = -quantile(IWDC_returns, 0.05)
print(paste("Hist VaR of UBS -->", hist_var_UBS))
print(paste("Hist VaR of IWDC -->", hist_var_IWDC))

# Gaussian VaR
gauss_var_UBS = -(mean(UBS_returns) + sd(UBS_returns)*qnorm(0.05))
gauss_var_IWDC = -(mean(IWDC_returns) + sd(IWDC_returns)*qnorm(0.05))
print(paste("Gaussian VaR of UBS  -->", gauss_var_UBS))
print(paste("Gaussian VaR of IWDC -->", gauss_var_IWDC))

#Cornish-Fisher VaR
alpha = 0.05
z_alpha = qnorm(alpha)

z_tilde_UBS = z_alpha + (1/6)*(z_alpha^2 - 1)*UBS_skew + (1/24)*(z_alpha^3 -
    3*z_alpha)*(UBS_kurt - 3) - (1/36)*(2*z_alpha^3 - 5*z_alpha)*(UBS_skew^2)
z_tilde_IWDC = z_alpha + (1/6)*(z_alpha^2 - 1)*IWDC_skew + (1/24)*(z_alpha^3
    - 3*z_alpha)*(IWDC_kurt - 3) - (1/36)*(2*z_alpha^3 - 5*z_alpha)*(IWDC_skew
    ^2)

cf_var_UBS = -(UBS_mean_returns + UBS_sd*z_tilde_UBS)
cf_var_IWDC = -(IWDC_mean_returns + IWDC_sd*z_tilde_IWDC)

print(paste("Cornish-Fisher VaR of UBS  -->", cf_var_UBS))
print(paste("Cornish-Fisher VaR of IWDC -->", cf_var_IWDC))

# VaR Comparison
var_matrix = rbind(c(hist_var_UBS, gauss_var_UBS, cf_var_UBS), c(hist_var_
    IWDC, gauss_var_IWDC, cf_var_IWDC))
colnames(var_matrix) = c("Historical", "Gaussian", "Cornish-Fisher")
rownames(var_matrix) = c("UBS", "IWDC")

barplot(var_matrix*100, beside=TRUE, col=c("red","blue"), ylim=c(0, max(var_
    matrix*100)*1.2), main="Daily Value at Risk Comparison (95%)", ylab="VaR
    (%)", legend.text=TRUE, args.legend=list(x="topright", bty="n"))
grid()

##### Forecasting Volatility with Sample Volatility #####

#FIRST WE PLOT ACTUAL VOALTILITY
rw_size = 90

# UBS rolling volatility
UBS_sample_vol = UBS_returns*0
for(i in rw_size:length(UBS_returns)){ sample_returns = UBS_returns[(i-rw_
    size+1):i]; UBS_sample_vol[i] = sd(sample_returns)*sqrt(252) }

# IWDC rolling volatility
IWDC_sample_vol = IWDC_returns*0
for(i in rw_size:length(IWDC_returns)){ sample_returns = IWDC_returns[(i-rw_
    size+1):i]; IWDC_sample_vol[i] = sd(sample_returns)*sqrt(252) }

#UBS GRAPH
par(mfrow=c(2,1), mar=c(3,4,2,2))
plot(UBS_returns, col="orange", lwd=1.5, main="UBS log returns", ylab="
    Returns", xlab="")
grid()
abline(h=0)
plot(UBS_sample_vol, col="red", lwd=2, main="UBS rolling volatility (90 days)
    ", ylab="Volatility", xlab="Date")
grid()
par(mfrow=c(1,1))
```

```r
# IWDC GRAPH
par(mfrow=c(2,1), mar=c(3,4,2,2))
plot(IWDC_returns, col="black", lwd=1.5, main="IWDC log returns", ylab="Returns", xlab="")
grid()
abline(h=0)
plot(IWDC_sample_vol, col="blue", lwd=2, main="IWDC rolling volatility (90 days)", ylab="Volatility", xlab="Date")
grid()
par(mfrow=c(1,1))

##### Historical VaR forecast (rolling) + violations #####
rw_size = 90

##### UBS #####
VaR_Forecast_UBS_hist = UBS_returns*0
for(i in rw_size:(length(UBS_returns)-1)){ Sample_Returns = UBS_returns[(i-rw_size+1):i]; VaR_Forecast_UBS_hist[i+1] = quantile(Sample_Returns, 0.05) }

plot(UBS_returns, col="orange", lwd=1, main="UBS returns and Historical VaR forecast (95%)", ylab="Log Returns", xlab="Date")
grid()
lines(VaR_Forecast_UBS_hist, col="red", lwd=2)
abline(h=0)

violations_UBS_hist = 0
for(i in rw_size:(length(UBS_returns)-1)){ violations_UBS_hist = violations_UBS_hist + (as.numeric(UBS_returns[i+1]) < as.numeric(VaR_Forecast_UBS_hist[i+1])) }
totals_UBS_hist = (length(UBS_returns)-1) - rw_size + 1
violation_rate_UBS_hist = violations_UBS_hist/totals_UBS_hist
print(paste("UBS historical VaR violations -->", violations_UBS_hist))
print(paste("UBS historical VaR violation rate -->", violation_rate_UBS_hist))

##### IWDC #####
VaR_Forecast_IWDC_hist = IWDC_returns*0
for(i in rw_size:(length(IWDC_returns)-1)){ Sample_Returns = IWDC_returns[(i-rw_size+1):i]; VaR_Forecast_IWDC_hist[i+1] = quantile(Sample_Returns, 0.05) }

plot(IWDC_returns, col="black", lwd=1, main="IWDC returns and Historical VaR forecast (95%)", ylab="Log Returns", xlab="Date")
grid()
lines(VaR_Forecast_IWDC_hist, col="blue", lwd=2)
abline(h=0)

violations_IWDC_hist = 0
for(i in rw_size:(length(IWDC_returns)-1)){ violations_IWDC_hist = violations_IWDC_hist + (as.numeric(IWDC_returns[i+1]) < as.numeric(VaR_Forecast_IWDC_hist[i+1])) }
totals_IWDC_hist = (length(IWDC_returns)-1) - rw_size + 1
violation_rate_IWDC_hist = violations_IWDC_hist/totals_IWDC_hist
print(paste("IWDC historical VaR violations -->", violations_IWDC_hist))
print(paste("IWDC historical VaR violation rate -->", violation_rate_IWDC_hist))

##### Gaussian VaR forecast (rolling) + violations #####
rw_size = 90

## UBS
VaR_Forecast_UBS_gauss = UBS_returns*0
for(i in rw_size:(length(UBS_returns)-1)){ Sample_Returns = UBS_returns[(i-rw
```

```
_size+1):i]; mu = mean(Sample_Returns); sigma = sd(Sample_Returns); VaR_Forecast_UBS_gauss[i+1] = mu + sigma*qnorm(0.05) }

plot(UBS_returns, col="orange", lwd=1, main="UBS returns and Gaussian VaR forecast (95%)", ylab="Log Returns", xlab="Date")
grid()
lines(VaR_Forecast_UBS_gauss, col="red", lwd=2)
abline(h=0)

violations_UBS_gauss = 0
for(i in rw_size:(length(UBS_returns)-1)){ violations_UBS_gauss = violations_UBS_gauss + (as.numeric(UBS_returns[i+1]) < as.numeric(VaR_Forecast_UBS_gauss[i+1])) }
totals_UBS_gauss = (length(UBS_returns)-1) - rw_size + 1
violation_rate_UBS_gauss = violations_UBS_gauss/totals_UBS_gauss
print(paste("UBS gaussian VaR violations -->", violations_UBS_gauss))
print(paste("UBS gaussian VaR violation rate -->", violation_rate_UBS_gauss))

## IWDC
VaR_Forecast_IWDC_gauss = IWDC_returns*0
for(i in rw_size:(length(IWDC_returns)-1)){ Sample_Returns = IWDC_returns[(i-rw_size+1):i]; mu = mean(Sample_Returns); sigma = sd(Sample_Returns); VaR_Forecast_IWDC_gauss[i+1] = mu + sigma*qnorm(0.05) }

plot(IWDC_returns, col="black", lwd=1, main="IWDC returns and Gaussian VaR forecast (95%)", ylab="Log Returns", xlab="Date")
grid()
lines(VaR_Forecast_IWDC_gauss, col="blue", lwd=2)
abline(h=0)

violations_IWDC_gauss = 0
for(i in rw_size:(length(IWDC_returns)-1)){ violations_IWDC_gauss = violations_IWDC_gauss + (as.numeric(IWDC_returns[i+1]) < as.numeric(VaR_Forecast_IWDC_gauss[i+1])) }
totals_IWDC_gauss = (length(IWDC_returns)-1) - rw_size + 1
violation_rate_IWDC_gauss = violations_IWDC_gauss/totals_IWDC_gauss
print(paste("IWDC gaussian VaR violations -->", violations_IWDC_gauss))
print(paste("IWDC gaussian VaR violation rate -->", violation_rate_IWDC_gauss))

##### Cornish-Fisher VaR forecast (rolling skewness and kurtosis) + violations #####
rw_size = 90
alpha = 0.05
z_alpha = qnorm(alpha)

##### UBS #####
VaR_Forecast_UBS_cf = UBS_returns*0
for(i in rw_size:(length(UBS_returns)-1)){ x = UBS_returns[(i-rw_size+1):i]; mu = mean(x); sigma = sd(x); s = mean((x-mu)^3)/sigma^3; k = mean((x-mu)^4)/sigma^4; z_tilde = z_alpha + (1/6)*(z_alpha^2-1)*s + (1/24)*(z_alpha^3-3*z_alpha)*(k-3) - (1/36)*(2*z_alpha^3-5*z_alpha)*(s^2); VaR_Forecast_UBS_cf[i+1] = mu + sigma*z_tilde }

plot(UBS_returns, col="orange", lwd=1, main="UBS returns and Cornish-Fisher VaR forecast (95%)", ylab="Log Returns", xlab="Date")
grid()
lines(VaR_Forecast_UBS_cf, col="red", lwd=2)
abline(h=0)

violations_UBS_cf = 0
for(i in rw_size:(length(UBS_returns)-1)){ violations_UBS_cf = violations_UBS_cf + (as.numeric(UBS_returns[i+1]) < as.numeric(VaR_Forecast_UBS_cf[i+1])
```

```
    ) }
totals_UBS_cf = (length(UBS_returns)-1) - rw_size + 1
violation_rate_UBS_cf = violations_UBS_cf/totals_UBS_cf
print(paste("UBS cornish-fisher VaR violations -->", violations_UBS_cf))
print(paste("UBS cornish-fisher VaR violation rate -->", violation_rate_UBS_
    cf))

##### IWDC #####
VaR_Forecast_IWDC_cf = IWDC_returns*0
for(i in rw_size:(length(IWDC_returns)-1)){ x = IWDC_returns[(i-rw_size+1):i
    ]; mu = mean(x); sigma = sd(x); s = mean((x-mu)^3)/sigma^3; k = mean((x-mu
    )^4)/sigma^4; z_tilde = z_alpha + (1/6)*(z_alpha^2-1)*s + (1/24)*(z_alpha
    ^3-3*z_alpha)*(k-3) - (1/36)*(2*z_alpha^3-5*z_alpha)*(s^2); VaR_Forecast_
    IWDC_cf[i+1] = mu + sigma*z_tilde }

plot(IWDC_returns, col="black", lwd=1, main="IWDC returns and Cornish-Fisher
    VaR forecast (95%)", ylab="Log Returns", xlab="Date")
grid()
lines(VaR_Forecast_IWDC_cf, col="blue", lwd=2)
abline(h=0)

violations_IWDC_cf = 0
for(i in rw_size:(length(IWDC_returns)-1)){ violations_IWDC_cf = violations_
    IWDC_cf + (as.numeric(IWDC_returns[i+1]) < as.numeric(VaR_Forecast_IWDC_cf
    [i+1])) }
totals_IWDC_cf = (length(IWDC_returns)-1) - rw_size + 1
violation_rate_IWDC_cf = violations_IWDC_cf/totals_IWDC_cf
print(paste("IWDC cornish-fisher VaR violations -->", violations_IWDC_cf))
print(paste("IWDC cornish-fisher VaR violation rate -->", violation_rate_IWDC
    _cf))

############GARCHVAR###########################
library(rugarch)

rw_size = 250
alpha = 0.05
dist_choice = "sstd"

garch.setup = ugarchspec(mean.model=list(armaOrder=c(0,0), include.mean=TRUE)
    , variance.model=list(model="gjrGARCH", garchOrder=c(1,1)), distribution.
    model=dist_choice)

##### UBS #####
VaR_Forecast_UBS_garch = rep(0, length(UBS_returns))
valid_UBS = 0
violations_UBS_garch = 0

for(i in rw_size:(length(UBS_returns)-1)){
  Sample_Returns = UBS_returns[(i-rw_size+1):i]

  fit = tryCatch(ugarchfit(spec=garch.setup, data=Sample_Returns, solver="
      hybrid"), error=function(e) NULL)

  if(!is.null(fit)){
    fc = ugarchforecast(fit, n.ahead=1)

    mu_hat = as.numeric(fitted(fc))[1]
    sigma_hat = as.numeric(sigma(fc))[1]

    # get distribution parameters from the fitted model
    pars = coef(fit)
    shape = if("shape" %in% names(pars)) pars["shape"] else NA
    skew  = if("skew"  %in% names(pars)) pars["skew"]  else NA
```

```

    # alpha-quantile of standardized innovations
    if(dist_choice == "norm"){ q_alpha = qnorm(alpha) } else if(dist_choice
        == "std"){ q_alpha = qdist("std", p=alpha, mu=0, sigma=1, shape=shape)
         } else if(dist_choice == "sstd"){ q_alpha = qdist("sstd", p=alpha, mu
        =0, sigma=1, skew=skew, shape=shape) } else { stop("Unsupported
        distribution choice") }

    VaR_Forecast_UBS_garch[i+1] = mu_hat + sigma_hat*q_alpha

    valid_UBS = valid_UBS + 1
    violations_UBS_garch = violations_UBS_garch + (as.numeric(UBS_returns[i
        +1]) < as.numeric(VaR_Forecast_UBS_garch[i+1]))
  }

  if(i %% 100 == 0) print(paste("Iteration:", i))
}

VaR_plot = VaR_Forecast_UBS_garch
VaR_plot[VaR_plot == 0] = NA

plot(UBS_returns, col="orange", lwd=1, main=paste("UBS returns and", dist_
    choice, "GJR-GARCH VaR forecast (95%)"), ylab="Log Returns", xlab="Date")
grid()
lines(VaR_plot, col="red", lwd=2)
abline(h=0)

violation_rate_UBS_garch = violations_UBS_garch/valid_UBS

print(paste("UBS", dist_choice, "GARCH VaR violations -->", violations_UBS_
    garch))
print(paste("UBS", dist_choice, "GARCH VaR valid forecasts -->", valid_UBS))
print(paste("UBS", dist_choice, "GARCH VaR violation rate -->", round(
    violation_rate_UBS_garch, 4)))

##### VAR BASED PORTOFOLIO
##### GARCH VaR BASED PORTFOLIO (UBS vs IWDC) #####
library(rugarch)

window <- 750
alpha  <- 0.05
rebalance_every <- 1

UBS_r  <- na.omit(UBS_returns)
IWDC_r <- na.omit(IWDC_returns)

data_all <- na.omit(merge(UBS_r, IWDC_r))
UBS_r  <- data_all[,1]
IWDC_r <- data_all[,2]
rm(data_all)

n <- length(UBS_r)

VaR_UBS  <- rep(NA, n)
VaR_IWDC <- rep(NA, n)

w_UBS  <- rep(NA, n)
w_IWDC <- rep(NA, n)

spec <- ugarchspec(variance.model=list(model="sGARCH", garchOrder=c(1,1)),
    mean.model=list(armaOrder=c(0,0), include.mean=TRUE), distribution.model="
    std")

```

```r
##### PROGRESS + ERROR COUNTERS #####
start_time <- Sys.time()
total_iter <- n - window
fail_UBS   <- 0
fail_IWDC  <- 0
fail_any   <- 0

for(i in (window+1):n){

  iter <- i - window

  if(iter %% 50 == 0 || i == n){
    now <- Sys.time()
    elapsed_min <- as.numeric(difftime(now, start_time, units="mins"))
    progress <- iter / total_iter
    eta_min <- if(progress > 0) elapsed_min * (1 - progress) / progress else NA
    cat(sprintf("i = %d | %4.1f %% done | elapsed %4.1f min | ETA %4.1f min | fails UBS %d IWDC %d any %d\n", i, 100*progress, elapsed_min, eta_min, fail_UBS, fail_IWDC, fail_any))
  }

  if(((i-(window+1)) %% rebalance_every) != 0){ w_UBS[i] <- w_UBS[i-1]; w_IWDC[i] <- w_IWDC[i-1]; next }

  UBS_window  <- as.numeric(UBS_r[(i-window):(i-1)])
  IWDC_window <- as.numeric(IWDC_r[(i-window):(i-1)])

  fit_UBS <- tryCatch(ugarchfit(spec=spec, data=UBS_window, solver="hybrid"), error=function(e){ fail_UBS <<- fail_UBS + 1; NULL })
  fit_IWDC <- tryCatch(ugarchfit(spec=spec, data=IWDC_window, solver="hybrid"), error=function(e){ fail_IWDC <<- fail_IWDC + 1; NULL })

  if(is.null(fit_UBS) || is.null(fit_IWDC)){ fail_any <- fail_any + 1; w_UBS[i] <- w_UBS[i-1]; w_IWDC[i] <- w_IWDC[i-1]; next }

  fc_UBS  <- ugarchforecast(fit_UBS, n.ahead=1)
  fc_IWDC <- ugarchforecast(fit_IWDC, n.ahead=1)

  mu_UBS     <- as.numeric(fitted(fc_UBS))
  sigma_UBS  <- as.numeric(sigma(fc_UBS))
  mu_IWDC    <- as.numeric(fitted(fc_IWDC))
  sigma_IWDC <- as.numeric(sigma(fc_IWDC))

  nu_UBS  <- coef(fit_UBS)["shape"]
  nu_IWDC <- coef(fit_IWDC)["shape"]

  q_UBS  <- qt(alpha, df=nu_UBS)
  q_IWDC <- qt(alpha, df=nu_IWDC)

  VaR_UBS[i]  <- max(-(mu_UBS + sigma_UBS*q_UBS), 1e-6)
  VaR_IWDC[i] <- max(-(mu_IWDC + sigma_IWDC*q_IWDC), 1e-6)

  inv1 <- 1/VaR_UBS[i]
  inv2 <- 1/VaR_IWDC[i]

  w_UBS[i]  <- inv1/(inv1+inv2)
  w_IWDC[i] <- inv2/(inv1+inv2)
}

w_UBS[is.na(w_UBS)] <- 0.5
w_IWDC[is.na(w_IWDC)] <- 0.5
```

```
port_r <- w_UBS*UBS_r + w_IWDC*IWDC_r
port_r <- na.omit(port_r)
port_value <- exp(cumsum(port_r))

plot(port_value, col="darkgreen", lwd=2, main="GARCH VaR Based Portfolio Value", ylab="Portfolio Value (Start = 1)", xlab="Date")
grid()

#####Portfolio vs single assets comparisons #####
##### PERFORMANCE COMPARISON PLOT #####

#check if its still ok
port_r <- na.omit(port_r)

port_value <- xts(exp(cumsum(as.numeric(port_r))), order.by=index(port_r))
colnames(port_value) <- "PORT"

# dates alinged
UBS_plot  <- UBS[index(port_value)]
IWDC_plot <- IWDC[index(port_value)]

# normalize
UBS_norm  <- UBS_plot/as.numeric(first(UBS_plot))
IWDC_norm <- IWDC_plot/as.numeric(first(IWDC_plot))
PORT_norm <- port_value/as.numeric(first(port_value))

# y lim on all series because i cant see more
ymin <- min(UBS_norm, IWDC_norm, PORT_norm, na.rm=TRUE)
ymax <- max(UBS_norm, IWDC_norm, PORT_norm, na.rm=TRUE)

#  plot
par(mar=c(5,5,4,2))
plot(PORT_norm, col="darkgreen", lwd=2, ylim=c(ymin, ymax), main="Performance Comparison: UBS vs IWDC vs GARCH-VaR Portfolio", ylab="Normalized Value (Start = 1)", xlab="Date")
grid()
lines(UBS_norm, col="red", lwd=2)
lines(IWDC_norm, col="blue", lwd=2)
```