Politecnico di Torino

Master's Degree in Mathematical Engineering

A.a 2023/2024

March 2024

Vine copulas for capital requirements: a probability equivalent level analysis

Supervisor
Prof. Franco Pellerey

Candidate
Filippo Grobbo

. . . . . .

. . . . . .

Summary

The study of capital requirements is an extremely important topic for the stability of the financial system. Recent geopolitical shocks including the war in Ukraine, the Israeli-Palestinian conflict, tensions on supply chains in the Red Sea, in parallel with the Quantitative Tightening monetary policies of central banks are putting its robustness under pressure. Being able to give an estimate of risk and interdependencies within investment portfolios allows one to have a clear vision of costs-benefits in terms of risk-rewards. This work proposes the use of copula theory both to address the problem of studying assets dependencies and predicting capital requirements in the form of risk measures. The class of Vine copulas is presented, which aims to overcome the limits linked to the adoption of a particular family of copulas in addressing the cited topics. This class in fact allows modeling the problem without the need to rely on a precise family but incorporating them all, within the computational limits of the R package used. A Monte Carlo simulation is developed and the validity of presented models is confirmed. Finally, an analysis of probability equivalent levels is carried out, a tool that could be very useful for risk management, since it enables the comparison of risk measures under different conditions of observability of financial markets.

Acknowledgements

I would like to thank my family for allowing me to embark on this university career and those who stood by me during difficult decisions, especially in the last year. I would also like to thank Professor Pellerey who was able to create the conditions to allow me to work at my best. I really appreciated the freedom I was given in choosing the topics I wanted to explore, as well as the willingness to discuss on Saturday afternoons and the sympathetic attitude toward my personal work situation, which did nothing to create a proactive environment for my thesis.

Contents

List of Tables

List of Figures

Chapter 1

Financial Background

1.1 The purpose of Finance

Finance is the study and discipline of money and capital assets. On one side, money is any item accepted as payment for goods, services and repayment of debts. On the other side, capital assets are properties of any kind, movable or immovable, tangible or intangible, fixed or circulating, that are held by an assessee.

Behind these two trivial concepts, finance is a multidisciplinary field that can involve insights from economics, mathematics, statistics and computer science. According to my risk management professor, the purposes of finance are mainly two:

- Shifting consumption over time: if I have excess income in some period I might delay consumption to a later time period. By a similar token, I might wish to anticipate consumption to an earlier time period. This is possible thanks to the financial system, which tries to match savers and borrowers, so that all of them may improve their consumption timing. It is needless to specify how everything has a cost, which depends on lending and borrowing rates.

- Risk transfer: an investor may shape the probability distribution of his wealth according to his taste and appetite for risk. Clearly, for any player hedging a risk exposure away, there must be another participant willing to assume that risk or part of it. Adding also the uncertainty that circumscribes this environment, the financial system plays the role of risk transfer mechanism.

These two points allow to introduce the so called time value of the money. The principle is based on the premise that money has the potential to earn interest or other returns over time, depending on economic and geopolitical conditions. ECB's negative interest rates policy is an example of monetary tool that can influence negatively future value of the money, in a period of low inflation and economic stagnation.

Actually, as can be seen from the second point, time is intertwined with another fundamental dimension in finance, namely, uncertainty or risk. Risks can be of different types and are almost coutless in the finacial context, to name a few in simple words:

- Market Risk: it is the risk of financial losses due to fluctuations in market prices or the value of financial instruments.
- Credit Risk: it is the risk that an unexpected change in a counterparty's creditworthiness, collateral value, or use of available margin in the event of default could generate an unexpected change in the value of banks' exposures.
- Operational Risk: the risk of financial losses resulting from inadequate or failed internal processes, systems, people or external events.
- Liquidity Risk: the risk of being unable to buy or sell financial instruments quickly without significantly affecting their prices. A crucial element that contributed to banks' inability to handle 2008 financial crisis was the lack of liquidity.
- Model Risk: the risk of financial losses resulting from errors or inadequacies in financial models used for decision-making.
- Systemic Risk: the risk that events affecting the financial system as a whole could lead to widespread financial instability.

Quantitative Risk Management is the area of finance focused on studying how to control risk and balance possibility of gains. It includes the process of measuring risk and developing strategies to manage it. The adjective Quantitative should refer to the use of mathematical models. In my humble opinion, which comes from university experience and brief work experience, I believe that quantitative risk management should not be an exclusively data-driven process. In other words, the massive use of data is useless if there is no underlying modeling of the financial system. Models should not only be data-driven, but also algorithm-driven, which means that they should not only seek a good fit to the data, but also be algebraically stated and proved according to a theoretical formalism.

1.2 Capital Requirements in Risk Management

Throughout its history, the financial landscape has operated in absence of a uniform global regulatory framework, establishing common criteria for risk management and financial robustness internationally. This implied that each country or financial entity have adopted its own protocols and capital requirements without following a universally accepted standardized approach. The lack of international agreement resulted in greater diversity in risk management practices among nations and financial institutions. This diversification has generated inequalities in the financial soundness of banks over the years, increasing systemic risk in the event of a crisis.

Over time, unified regulations and principles for the banking sector were introduced, with the aim of filling gaps in international supervision coverage, ensuring that no banking institution escaped supervision. Limitations of these regulations particularly emerged during the severe 2008 financial crisis, where banks failed to cope with the crisis precisely because of insufficient liquidity. It was during those years that the Basel Committee took action, issuing a set of provisions that are known as the Basel III Accords. These are global

treaties, constituting a regulatory framework that establishes guidelines for managing risk and strengthening financial soundness in the banking sector. The agreement aims to ensure that banks maintain an appropriate level of capital, sufficient to cover potential losses.

Although most of the troubled institutions were banks, a number of insurers were also affected by the crisis, due to poor investment decisions by insurers that led to significant losses, interconnectedness with banks, or general evidence of inadequate governance, says Insurance and Authority [2018]. The crisis demonstrated the importance of an harmonized understanding of risks by all actors of the financial system and the need to consider broader implications for financial stability. For this reason, Solvency II Directive was implemented, which also includes the calculation of capital requirements for different categories of assets held by insurance and reinsurance companies.

Key points that connect these topics are how to measure risk, quantify the probability of loss and how to account for the interdependence that can be created among different financial institutions. Hence the need to define what risk measures and copulas are, the cornerstone topics of this work.

1.3 Univariate Time Series Analysis

Before going in details of what risk measures and copulas are, it is important to define the building blocks that enable the study of these two topics, i.e. univariate time series. In a nutshell, they are a set of observations $\{r_t : t = 1,2,...\}$ taken over time, where there is assumed to be a temporal dependence. Each time serie can contain a trend, i.e. a general direction or tendency exhibited by the data, and seasonal effects, i.e. recurring patterns or fluctuations that follow a consistent and predictable cycle over specific time intervals. Dependence between close observations is translated into the concepts of homoschedasticity and heteroschedasticity. The first one represents the condition where the variance of observations remains constant, indicating consistent level of dispersion. The second one instead is the contrary, i.e. a scenario where variance varies, indicating unequal levels of dispersion between observartions.

From a mathematical point of view, given a probability space $(\Omega, \mathcal{F}, \mathcal{P})$ with sample space Ω, σ-Algebra $\mathcal{F}$ and probability measure $\mathcal{P}$, a time serie can be considered as a realization of a stochastic process. The latter is a collection of random variables $(R_t)_{t\in\mathcal{T}}$ defined on $(\Omega, \mathcal{F}, \mathcal{P})$, with values in $(\mathbb{R}, \mathcal{B}(\mathbb{R}))$, where $\mathcal{B}(\mathbb{R})$ is the σ-Algebra of Borel. In our case $\mathcal{T}$ is the set of all time istants, while the random variable R_t can be the log return of a particular asset at time t, which is obviously uncertain. Important properties that allow to distinguish different time series are:

- Strong stationarity: it means that by traslating a serie for a time h its distribution does not change. In formulas, $\forall h \in \mathbb{R}, \forall t_1, ..., t_n \in \mathcal{T}, \forall n \in \mathbb{N}, \forall r = (r_1, ..., r_n) \in \mathbb{R}^n$

$$f_{R_{t_1},...,R_{t_n}}(r_1, ..., r_n) = f_{R_{t_1+h},...,R_{t_n+h}}(r_1, ..., r_n) \tag{1.1}$$

  where f is the multivariate density function.

- Weak stationarity of order k: it is the property of a stochastic process where the first k moments exist and are time invariant. In formulas, $\forall t_1, ..., t_n \in \mathcal{T}, \forall h \in \mathbb{R}, \forall k_1, ..., k_n \in \mathbb{N}$ such that $k_1 + k_2 + ... + k_n = k$

$$E[R_{t_1}^{k_1} R_{t_2}^{k_2} ... R_{t_n}^{k_n}] = E[R_{t_1+h}^{k_1} R_{t_2+h}^{k_2} ... R_{t_n+h}^{k_n}] \tag{1.2}$$

- Markov property: the law of a process at a given time t_i does not depend on its entire past, but only on its most recent past. In formulas, given any set of temporal indexes $t_1 < t_2 < ... < t_i$

$$P(R_{t_i} = r_i | R_{t_i-1} = r_{i-1}, R_{t_i-2} = r_{i-2}, ..., R_{t_1} = r_1) = P(R_{t_i} = r_i | R_{t_i-1} = r_{i-1}) \tag{1.3}$$

- Markov property of order h: it corresponds to a more restrictive case than the general law. In fact, the distribution at a given time t_i depends only on the previous h times, in particular

$$\begin{aligned} &P(R_{t_i} = r_i | R_{t_i-1} = r_{i-1}, R_{t_i-2} = r_{i-2}, ..., R_{t_2} = r_2, R_{t_1} = r_1) = \\ &= P(R_{t_i} = r_i | R_{t_i-1} = r_{i-1}, R_{t_i-2} = r_{i-2}, R_{t_i-h} = r_{i-h}) \end{aligned} \tag{1.4}$$

In addition, two other elements that will be discussed throughout the work are the autocovariance and autocorrelation functions. These are nothing more than the covariance e correlation calculated between two different time instants t_i and t_{i+l}. For weak stationary processes of order at least two, these fuctions depends only on the lagged time difference and are defined as:

$$\gamma(l) := Cov(R_{t_i}, R_{t_i+l}) = E(R_{t_i} R_{t_i+l}) - E(R_{t_i}) E(R_{t_i+l}) \tag{1.5}$$

$$\rho(l) := \frac{Cov(R_{t_i}, R_{t_i+l})}{Var(R_{t_i})} = \frac{\gamma(l)}{\gamma(0)} \tag{1.6}$$

Univariate time series models are the tools used to characterize, for example, the individual assets that make up a portfolio, evaluating their mean, variance and understanding the autocorrelation of log returns as the time interval changes. Henceforth, weak stationarity of order at least two will be assumed, and the term will often be referred to simply as stationarity. For non-stationary time series, the concepts presented are fully replicable, although some adjustments would be needed, such as detrending series. The main models used are presented below, along with the technical specifications chosen specifically for this work. Closed formulas for mean, variance, autocovariance and autocorrelation are available for all models, but they are complicated and their addition would not add any value.

1.3.1 Autoregressive Models

Autoregressive model (AR) of order p express the mathematical concept that present value r_t of a time serie can be explained by a function involving its previous p values, namely $r_{t-1}, ..., r_{t-p}$. In the conventional AR model, this functional relationship is linear:

$$r_t = \sum_{l=1}^{p} \phi_l r_{t-l} + w_t \tag{1.7}$$

where ϕ_l are parameters to be estimated while w_t are i.i.d random variables, called innovation processes and follow a skewed-t distribution with parameters $(\xi, \omega, \alpha, \nu)$. This distribution was proposed for the first time by Hansen [1994] and basically is the extension of a student-t distribution including also a parameter to shape skewness. Formally:

$$\begin{aligned} w_t &= \xi + V^{-1/2} Z \\ Z &\sim SN(\xi, \omega, \alpha) \\ V &\sim \chi^2(\nu)/\nu \end{aligned} \tag{1.8}$$

where V is a chi-squared distribution with ν degress of freedom and SN(ξ, ω, α) is a skewed normal distribution, which is a random variable such that:

$$\begin{aligned} f_Z(x) &= 2\varphi(z-\xi)\Phi(\alpha\omega^{-1}(z-\xi)) \\ \varphi(z) &= (2\pi)^{-1/2} exp(-\frac{z^2}{2}) \\ \Phi(z) &= \int_{-\infty}^{z} \varphi(x)\, dx \end{aligned} \tag{1.9}$$

ξ and ω are called respectively position and scale parameter, as they are nothing but the mean and variance of a classic normal distribution. α instead is a parameter that refers to skewness, which can be either positive or negative.

Intuitively, AR(p) models are Markov processes of order p. By defining the linear operator $B^p r_t := r_{t-p}$ and calling $\Theta_p(B) := (1 - \phi_1 B - \phi_2 B^2 - ... - \phi_p B^p)$, it is possible to rewrite an AR(p) model as:

$$\Theta_p(B) r_t = w_t \tag{1.10}$$

It can be proved that AR(p) models are strong stationary if and only if all the roots of the characteristic polynomial $\Theta_p(B)$ are greater than one in absolute value. This translate for example in having $|\phi_l|<1$ $\forall$ l=1,...,p in case of AR(1) with normally distributed innovation.

1.3.2 Moving Average Models

In a completely similar way it is possbile to define moving average models (MA) of order q, where the present value r_t of a time serie is a function of last q innovations terms. The functional relation is again linear:

$$r_t = \sum_{l=1}^{q} \theta_l w_{t-l} + w_t \tag{1.11}$$

where θ_l are parameters to be estimated and $w_i \overset{\text{i.i.d}}{\sim}$ skewed-t$(\xi, \omega, \alpha, \nu)$ $\forall$ i=t-q,...,t. Also in this case, by defining the polynomial $\phi_q(B) := (1 + \theta_1 B + \theta_2 B^2 + ... + \theta_q B^q)$, it is possible to write a MA(q) model as:

$$r_t = \phi_q(B) w_t \tag{1.12}$$

It can be proved that MA(q) processes are strong stationary if and only if all the roots of $\phi_q(B)$ are greater than one in absolute value.

1.3.3 Autoregressive Moving Average Models

AR models are effective for capturing trend-like patterns in the data. They represent how past values influence the current state. The term "autoregressive" in fact indicates self-regression, meaning that the serie is regressed on its own past values.

On the other hand, the focus of MA models is considering random fluctuations. These models are effective when the emphasis is on filtering out short-term irregularities, by highlighting the impact of recent shocks or unexpected events and providing insights into the short-term volatility of time series.

Combinations of AR and MA models can enhance the overall modeling capacity, creating the so called Autoregressive Moving Average models (ARMA). Stationarity is critical when working with AR models, as these models assume that relationships over time are constant. If series are not weakly stationary, the parameters of AR models may vary and make difficult model estimation. This problem does not occur for MA models, since they are finite linear combination of a random variables for which the first two moments are time invariant, so weak stationarity of order 2 always holds. From what has been said regarding AR models, it is clear to require that time series under consideration should be at least weakly stationary of order at least 2.

Mathematically, ARMA(p,q) models can be written as:

$$r_t = \phi_0 + \sum_{l=1}^{p} \phi_l r_{t-l} + w_t + \sum_{l=1}^{q} \theta_l w_{t-l} \tag{1.13}$$

with $w_i \overset{\text{i.i.d}}{\sim}$ skewed-t$(\xi, \omega, \alpha, \nu)$ $\forall$ i=t-q,...,t. Parameters estimation and strong stationarity conditions follow from AR(p) and MA(q) models.

1.3.4 Generalized Autoregressive Conditional Heteroschedasticity Models

Up to now, models with costant variance over time were presented. However, different situations may arise in the reality of stressed and correlated markets. As reported by Jondeau et al. [2006], volatility of returns can cluster in some cases, so that large variation of prices (positive or negative) are expected after large variation of prices (of either sign). This suggest that assets volatility tend to revert to some mean rather than remaining constant or moving in monotonic fashion over time. In addition, according to Ait-Sahalia et al. [2013], rising asset prices are accompanied by declining volatility and vice versa. In fact, when asset prices fall, companies mechanically become more leveraged as the relative value of their debt increases relative to that of their equity. As a result, it is natural to expect their stocks to become riskier and thus more volatile. The term leverage effect refers to this phenomenon, and it has been documented that it is generally asymmetrical, in the sense that declines in stock prices are accompanied by greater increases in volatility than the decline in volatility that accompanies the increase in prices.

While asymmetric behaviors and fat tail can be partially captured by skewed-t innovation processes, the same cannot be said for volatility clustering and leverage effect.

This is the reason why Generalized Autoregressive Conditional Heteroschedasticity models (GARCH) and their variations were introduced firstly by Bollerslev [1986]. To put it simply, GARCH models can be decribed as ARMA models for the volatility. Mathematically, as done by Sommer [2022], we can decompose the return at time t r_t in a mean term and a volatility term:

$$\begin{aligned} r_t &= \mu_t + \sigma_t \\ \mu_t &= \phi_0 + \sum_{l=1}^{p} \phi_l r_{t-l} + w_t + \sum_{l=1}^{q} \theta_l w_{t-l} \\ \sigma_t &= w_t \epsilon_t \\ w_t^2 &= \alpha_0 + \sum_{l=1}^{m} \alpha_i \sigma_{t-i}^2 + \sum_{j=1}^{s} \beta_j w_{t-j}^2 \end{aligned} \tag{1.14}$$

with $\epsilon_t \sim$ skewed-t$(\xi, \omega, \alpha, \nu)$. With this notation we can say that the mean term follows an ARMA(p,q) model, while the volatility term follows a GARCH(m,s) model. Putting together we obtain the so called ARMA(p,q)-GARCH(m,s) model. Some conditions that must hold are: $\alpha_i \geq 0$, $\beta_i \geq 0 \; \forall i, j$ and $\sum_{i=1}^{max(m,s)} (\alpha_i + \beta_j) < 1$. The last one is a sufficient but not necessary condition to guarantee a finite variance of innovations and so weakly stationary of time series, as reported by Jondeau et al. [2006].

It is important to observe that the analogy with ARMA models shows why volatility clustering can be explained with GARCH models, as trend patterns, random fluctuations, and short-term irregularities in volatility can be explained with this type of models. Regarding leverage effect, there are variants of GARCH models, such as the threshold GARCH by Zakoian [1991], that allow to account for this phenomenon as well, but this is beyond the scope of the following work.

1.3.5 Estimation, forecast and model quality assesment

Having presented ARMA and GARCH models, it is necessary to give some background about their estimation, prediction and model quality assessment. As we have noted in previous sections, these models involve the estimation of various parameters, each one playing a key role in characterizing time dynamics and volatility patterns of financial time series. Understanding the estimation process is important both for underlying the structure of data but also to establish the effectiveness of these models in capturing and predicting market behavior.

In this work, parameters are determined via maximum likelihood estimation. By fixing skewed-t distribution as innovations and through transformations of random variables, it is possible to obtain analytically the density of returns $f(r_t, r_{t-1}, ..., r_1; \theta)$, where $\theta = (\phi_0, ..., \phi_p, \theta_1, ..., \theta_q, \alpha_0, ..., \alpha_m, \beta_1, ..., \beta_s, \xi, \omega, \alpha, \nu)$ is the vector of parameters. After this, given a vector of observed returns $r_1, ..., r_t$, the likelihood is defined as:

$$L(\theta) := f(\theta; r_t, ..., r_1) = \prod_{i=1}^{t} f(r_t; r_{t-1}, ..., r_1) \tag{1.15}$$

where the last equality holds thanks to factorisation of the joint distribution $f(\theta; r_t, ..., r_1)$. Maximization of this function returns an estimator $\hat{\theta}$ of the parameters θ. The process is performed first for the mean term, then on the series of squared residuals for the volatility term, since w_t^2 must be modeled according to an ARMA model, and not σ_t. All this is done in a rolling window fashion for the following analysis. More details about the rolling window approach are given below but basically it is a window of fixed size moving through the time series data, where, at each step, the model parameters are recalculated based on the observations within the window. This dynamic process allows the model to capture evolving patterns over time, providing a real-time and adaptive estimation of parameters.

Time series forecasting, on the other hand, can be done recursively, iterating a one-step ahead forecast for the desired number of times. In fact, given a series of returns and the associated estimated parameters of an ARMA-GARCH model up to time t, it is possible to obtain a return at time t+1, assuming that the model structure is the same, by simply simulating an observation from the known distribution.

The last step, and maybe the most importan one, is model quality assesment. Actually, accuracy and reliability of presented models is an essential element, because without that the whole discussion would lose meaning. Graphical approaches based on properties of ARMA-GARCH autocorrelation function exist, but they become useless when considering portfolios with substantial number of assets. For this reason, the procedure adopted in this work is to study standardized residuals, defined as:

$$z_t = \frac{r_t - \hat{\mu}_t}{\hat{\sigma}_t} \tag{1.16}$$

Theoretically, since they are by definition equivalent to the innovation processes, they should be independent and identically distributed according to a skewed-t distribution with common parameters. In addition, by the hypothesis of weakly stationarity, they should not exibit any type of autocorrelation and this can be verified by the Ljung-Box test of Ljung and Box [1978]. The null hypothesis is that residuals up to lag H are independently distributed and the test examines the autocorrelation function by means of the statistic:

$$\mathcal{Q} = T(T+2) \sum_{h=1}^{H} \frac{\hat{\rho}(h)^2}{T-h} \overset{\text{under } H_0}{\sim} \chi^2(H) \tag{1.17}$$

Operatively, good fit happens with high p-values i.e. one does not want to reject the null hypothesis. Often, multiple values of H are tested.

In conclusion, performance evaluation between different models to choose the most suitable one is done with the Akaike Information Criterion (AIC) of Akaike [1973] and the Bayesian Information Criterion (BIC) of Schwarz [1978], defined as:

$$\begin{aligned} AIC &:= -2\ln(L(\theta)) + 2k \\ BIC &:= -2\ln(L(\theta)) + k\ln(n) \end{aligned} \tag{1.18}$$

where k is the number of model parameters and n is the sampel size. These are widely utilized tools for model selection because they offer selection criteria that balance model efficiency, measured through likelihood, with a penalty for model complexity in terms of the number of parameters. This helps prevent overfitting, ensuring the choice of models that are both efficient and parsimonious.

1.4 Risk Measures

According to Brandimarte [2018] a risk measure ξ is a functional that maps a continuous random variable R_t^P to a real number:

$$\xi : R_t^P \to \mathbb{R} \tag{1.19}$$

where t ≥ 0 is an instant of time and R_t^P is a continuous measurable function that can represent, for example, the value of a portfolio P at time t. More generally, a risk measure is an objective assessment of risk and, to be such, is usually characterized by certain properties of coherence in the context of quantitative risk management. The following properties define a coherent risk measure:

- Normalization: if a random variable is identically zero ($R^P \equiv 0$), it is reasonable to set $\xi(0) = 0$. In simple words, if we do not hold any portfolio, we are not exposed to any risk.
- Monotonicity: if $R_1^P \leq R_2^P$, meaning that the inequality holds almost surely, i.e., for all possible outcomes, with exception of a set of measure zero, then $\xi(R_1^P) \geq \xi(R_2^P)$. In simple words, if the value of portfolio 1 is never larger than the value of portfolio 2, then portfolio 1 is at least as risky as portfolio 2.
- Translation invariance: if we add a fixed amount a to the portfolio, the risk measure is affected: $\xi(R^P + a) = \xi(R^P) - a$. If a > 0, then we are adding a positive value to our portfolio for each possible scenario, so risk is reduced.
- Positive homogeneity: if we double the amount invested in a portfolio, we double risk. Formally: $\xi(bR^P) = b\xi(R^P)$, for b ≥ 0. This condition does not encapsulate the more general case where the liquidity effect is taken into account, i.e., the larger a position the more difficult is to disassemble it, since large sales contribute to further depreciation in reality.
- Subadditivity: the risk of the sum of two random variables cannot exceed the sum of their respective risks: $\xi(R_1^P + R_2^P) \leq \xi(R_1^P) + \xi(R_2^P)$. In simple words diversification is expected to decrease risk and cannot increase it.

A direct interpretation of these properties is that $\xi(R^P + \xi(R^P)) = \xi(R^P) - \xi(R^P) = 0$, which means that the risk measure of a portfolio can be considered as the minimum amount of additional capital needed to make portfolio acceptable in some sense, i.e. with risk equal to 0. In addition, thanks to homogeneity and subadditivity, every coherent risk measure is also convex, since $\xi(\lambda R_1^P + (1-\lambda)R_2^P) \leq \lambda\xi(R_1^P) + (1-\lambda)\xi(R_2^P)\ \forall\lambda \in [0,1]$.

1.4.1 Value-at-risk

In this work two particular risk measures are taken into consideration. The first one is the value-at-risk (VaR), which can be considered as the worst expected loss over a given time horizon for a fixed confidence level, knowing the market underlying distribution. To understand the importance of this risk measure, it is sufficient to note that according

to Basel III, Pillar I, there is a risk coverage indicator, known as Stressed VaR, used to assess risk under stress conditions, which could result in increased capital for some trading activities and derivative instruments, especially in the case of complex transactions. In a complementary way, as reported by Insurance and Authority [2022], the Solvency Capital Requirements should at least cover non-life underwriting risk, life underwriting risk, health underwriting risk, market risk and counterparty default risk, which should all be calibrated using a value-at-risk approach. Mathematically speaking, value-at-risk at confidence level $\alpha \in (0,1)$ is nothing but the α quantile of portfolio return distribution R_t^P at a specific time t:

$$VaR_\alpha^{P,t} := sup\left\{r|F_{R_t^P}(r) \leq \alpha\right\} = \mathcal{Q}_{R_t^P}(\alpha) \tag{1.20}$$

It can be proved that value-at-risk satisfies all properties of coherent risk measures except Subadditivity, as shown in Brandimarte [2018].

1.4.2 Expected Shortfall

The other risk measure adopted in this work tries to overcome value-at-risk limits from a mathematical point of view and is called expected shortfall (ES). Roughly speaking it is the expected value of a portfolio, conditional on having a value lower than value-at-risk. References about this measure are both present in Insurance and Authority [2016] and on Banking Supervision [2019], where is used to derive shocks for stress test and capital requirements for internal rating models respectively. From a mathematical point of view, it is defined as:

$$ES_\alpha^{P,t} := \mathbb{E}\left[R_t^P | R_t^P \leq VaR_\alpha^{P,t}\right] = \frac{1}{\alpha}\int_0^\alpha r f_{R_t^P}(r)\, dr = \frac{1}{\alpha}\int_0^\alpha VaR_u^{P,t}\, du \tag{1.21}$$

where the last equality holds thanks to the change of variable u=F(r), which implies that r=$F_{R_t^P}^{-1}(u) = VaR_u^{P,t}(r)$. The peculiarity of ES is that it is a coherent risk measure, which therefore also includes the property of subadditivity.

1.4.3 Backtesting

Backtesting aims to assess model accuracy of predicted risk measures by employing both unconditional and conditional coverage hypothesis tests to study the behavior of the number of exceedances. Given portfolio observed returs over a period of time h, an exceedance occurs when the value of the portfolio is worse than the predicted risk measure. The tests adopted in this work are presented below and depend on the risk measure chosen:

- Value-at-risk test of Kupiec (Kupiec [1995]): unconditional tests to evaluate if the observed frequency of exceedances aligns with the chosen confidence level within the selected time interval. The null hypothesis H_0 states that the expected violation rate is equal to the theoretical confidence level α. The test statistic is defined as:

$$LR_{uc} = -2ln[\alpha^n(1-\alpha)^h - n] + 2ln(n/h)^n(1-n/h)h - n] \overset{\text{under } H_0}{\sim} \chi^2(1) \tag{1.22}$$

  where n and n/h are respectively the total number and observed proportion of exceedances. Also in this case, good fit happens with high p-values, i.e. one does not

want to reject the null hypothesis. As noted by Tòfoli et al. [2019], Kupiec's test may be rather questionable, since it does not check whether exceedances are clustered with respect to the time at which they appear, but rather assumes they arise independently.

- Value-at-risk test of Christoffersen (Christoffersen [1998]): to account for indepence of exceedances, Christoffersen considered a binary first-order Markov chain for the exceedances defined as follows:

$$\begin{aligned} &I_t := \mathbb{1}_{\{r_t < \hat{VaR}_\alpha^t\}} \; t = 1, ..h \\ &S = \{1 := VaR \; violation, 0 := No \; VaR \; violation\} \\ &\pi_{i,j} = P(I_t = j | I_{t-1} = i) = (n_{i,j} / \sum_j n_{i,j}) \; \forall i, j = 0{,}1 \end{aligned} \tag{1.23}$$

where I is the stochastic process representing the indicator sequence of violations and S is the state space. Under H_0: $\pi_{0,1} = \pi_{1,1} = \alpha$, i.e. the estimated violation rate is equal to α and exceedances are independet. The test statistic is:

$$LR_{cc} = -2ln[\alpha^n(1-\alpha)^h - n] + 2ln[(\pi_{0,1})^{n_{0,1}}(1-\pi_{0,1})^{n_{0,0}}\pi_{1,1}^{n_{1,1}} - (1-\pi_{1,1})^{n_{1,0}}] \overset{\text{under } H_0}{\sim} \chi^2(2) \tag{1.24}$$

- Expected Shortfall Test of McNeil and Frey (McNeil et al. [2000]): the null hypothesis asserts that the excess conditional shortfall, i.e. the excess of the actual series when VaR is violated, is independent and identically distributed (i.i.d.) with zero mean. A one-sided t-test is employed against the alternative hypothesis that the mean of the excess shortfall is greater than zero. The rationale behind this is to investigate whether the conditional shortfall is systematically underestimated by the model under consideration. The t-test statistic is:

$$t = \frac{\bar{X} - \mu_0}{\sigma / \sqrt{n}} \tag{1.25}$$

where $\bar{X}$ is the sample mean of the excess shortfall, μ_0 is the assumed mean (zero under H_0), σ is the sample standard deviation and n is the sample size. To obtain p-values associated with the test statistic, the distribution of the excess shortfall is crucial. However, making assumptions about the distribution can introduce bias. To solve this problem, the bootstrap resampling method is used to obtain empirical estimates of the shortfall excess distribution. Bootstrap consists in repeatedly sampling with replacement from the empirical distribution of observed data to generate the so called multiple bootstrap samples. For each sample, the t-test statistic is calculated, resulting in a distribution of test statistics. The p-value is then calculated as the proportion of bootstrap statistics that are more extreme than the observed test statistic. This nonparametric approach allows for a more robust evaluation of the hypothesis test, mitigating potential biases associated with assumptions about the distribution of excess shortfall.

Chapter 2

Copula Theory

Having outlined how to measure risk in the univariate case, the next problem that arises is the construction of multivariate distribution models, which can be complicated if one wants to go beyond the simple gaussian case. In general, there are two main approaches to define the structure of n variables functions:

- On the one hand there are parametric families of distribution depending on limited number of parameters, such as the multivariate normal distribution previously mentioned.

- On the other hand, the approach can start by characterizing individual random variables and then putting pieces together, as copula theory does.

To make concepts more clear, the example presented by Nelsen [2005] can come to aid. Without giving too much technicalitites, consider a pair of random variables X and Y, with cumulative distribution fuction (CDF) $F(x) = P(X \leq x)$ and $G(Y) = P(Y \leq y)$, respectively, and joint distribution F(x,y)=$P(X \leq x, Y \leq y)$. For each pair of real numbers (x,y) we can associate three numbers that lies all in [0,1], namely: $F_X(x)$, $F_y(y)$ and $F_{X,Y}(x, y)$. In other words, each pair (x,y) leads to a point $(F_X(x), F_y(y))$ in the unit square [0,1]x[0,1], and this ordered pair correspond in turn to a number $F_{X,Y}(x, y)$ in [0,1]. The correspondance which assigns the value of joint distribution to each ordered pair of values of individual CDFs is called copula.

Another important aspect having to do with the multivariate world is the study of correlation and dependencies between individual constituents of multivariate distributions. Two well known measures to capture these are: Person's correlation coefficient and Kendall's tau. The first one, however, is only able to capture linear associations, while the second one is more related to the concepts of concordance and discordance between one random variable and another identically distributed one. Copulas, on the other hand, are a way of representing the dependence between random variables independently of marginal distributions, but instead capturing the association between ranks, which are invariant under monotonic transformations of the initial random variables. More details about the structure and definition of all these objects are explained below.

2.1 Copulas Fundamentals

The idea behind copulas is to factorise an n-dimensional multivariate cumulative distribution $F_X(x_1, ..., x_n)$ into two componets:

- a set of univariate marginal CDFs, namely $F_1(x_1), ..., F_n(x_n)$
- a function C called copula that maps the n univariate marginals from the hypercube $[0,1]^n$ into the unit interval [0,1].

C should be interpreted as a sensible joint CDF, and to make sense, certain properties must be satisfied. Mathematically, a copula C, is a function defined on $[0,1]^n$, such that:

- C is grounded: in simple terms, below the support of any univariate marginal, the CDF must be 0. Formally, C$(u_1, ..., u_n)$=0 whenever at least one u_k is equal to the lowest value of the corresponding univariate marginal support.
- C is n-increasing: it is the natural extension in the n-dimensional case of the so called rectangular inequality. To make the concept clear, in the bidimensional case, $\forall\ S = [a_1, b_1] \times [a_2, b_2]$, it should hold that $C(b_1, b_2) - C(b_1, a_2) - C(a_1, b_2) + C(a_1, a_2) \geq 0$. This relationship can be easily extended to the multidimensional case.
- C has margins C_k, k=1,...,n satisfying $C_k(u) = u\ \forall\ u \in [0,1]$, i.e. uniform margins.

Under these requirements, thanks to the famous Sklar' theorem, it is possibile to say that, given any n-dimensional distribution with continuous marginal CDFs $F_1, ..., F_n$, there exist a unique copula C such that:

$$F_X(x_1, ..., x_n) = C(F_1(x_1), ..., F_n(x_n)\ \forall x = (x_1, ..., x_n) \tag{2.1}$$

Assuming invertibility of the univariate marginal CDFs and by taking the transformation $u_i = F_i(x_i)\ \forall i$, we can rewrite the equation as C$(u_1, ..., u_n) = F_X(F_1^{-1}(u_1), ..., F_n^{-1}(u_n))$. From now on we will refer to copula scale as data denoted by u_i and to original/real scale when dealing with the initial data x_i. In addition, we will often fall back on the two-dimensional case to simplify the discussion.

After defining copulas, several families of them can be derived. On the one hand, some are obtained from the joint CDF of multivariate distributions, such as gaussian and t copulas. On the other hand, some have an explicit representation and can be, for example, one-parameter copulas, such as certain archimedeans, or two-parameter copulas, such as some members of the biparameter bivariate (BB) family. In the following sections, we will introduce a specific type of copula that eliminates the need to analyze each individual family to determine which one is most suitable for describing a generic data set. However, it is crucial to grasp the distinctive element among copulas, i.e. copula density, as this factor will guide the search for the most suitable one. In the bivariate case, since C is a joint sensible CDF, it is possible to write:

$$\begin{aligned} f_{X_1,X_2}(x_1, x_2) &= \frac{\partial^2 F_{X_1,X_2}(x_1, x_2)}{\partial x_1 \partial x_2} = \frac{\partial^2 C(F_{X_1}(x_1), F_{X_2}(x_2))}{\partial x_1 \partial x_2} = c(F_{X_1}(x_1), F_{X_2}(x_2)) f_{X_1}(x_1) f_{X_2}(x_2) \\ \implies c(u_1, u_2) &\overset{u_i = F(x_i)}{=} c(F_{X_1}(x_1), F_{X_2}(x_2)) = \frac{f_{X_1,X_2}(x_1, x_2)}{f_{X_1}(x_1) f_{X_2}(x_2)} \end{aligned} \tag{2.2}$$

where c is copula density. Same can be done for the multidimensional case. Copula families are characterized by different densities, which in turn allow to examine different joint behavior of random variables. The extent of tail thickness and its symmetry properties are for example reflected by the density and enable the modeling of various degrees of multivariate dependence. In addition, rotations of copulas, such as 90-degree counterclockwise rotation given by $C_{90}(u1, u2) = C(1 - u_1, 1 - u_2)$, can be used, for example, to model negative dependence with copulas that have positive tail dependence. Figure 2.1, taken from Sommer [2022], show marginal normal contour plots for some bivariate copula families. Inverse normal scale distribution is adopted for better visualization and comparison of tails. Each copula, is displayed with strong and weak dependence. These graphs show plausible patterns of tail dependence that can be obtained with different copula families.

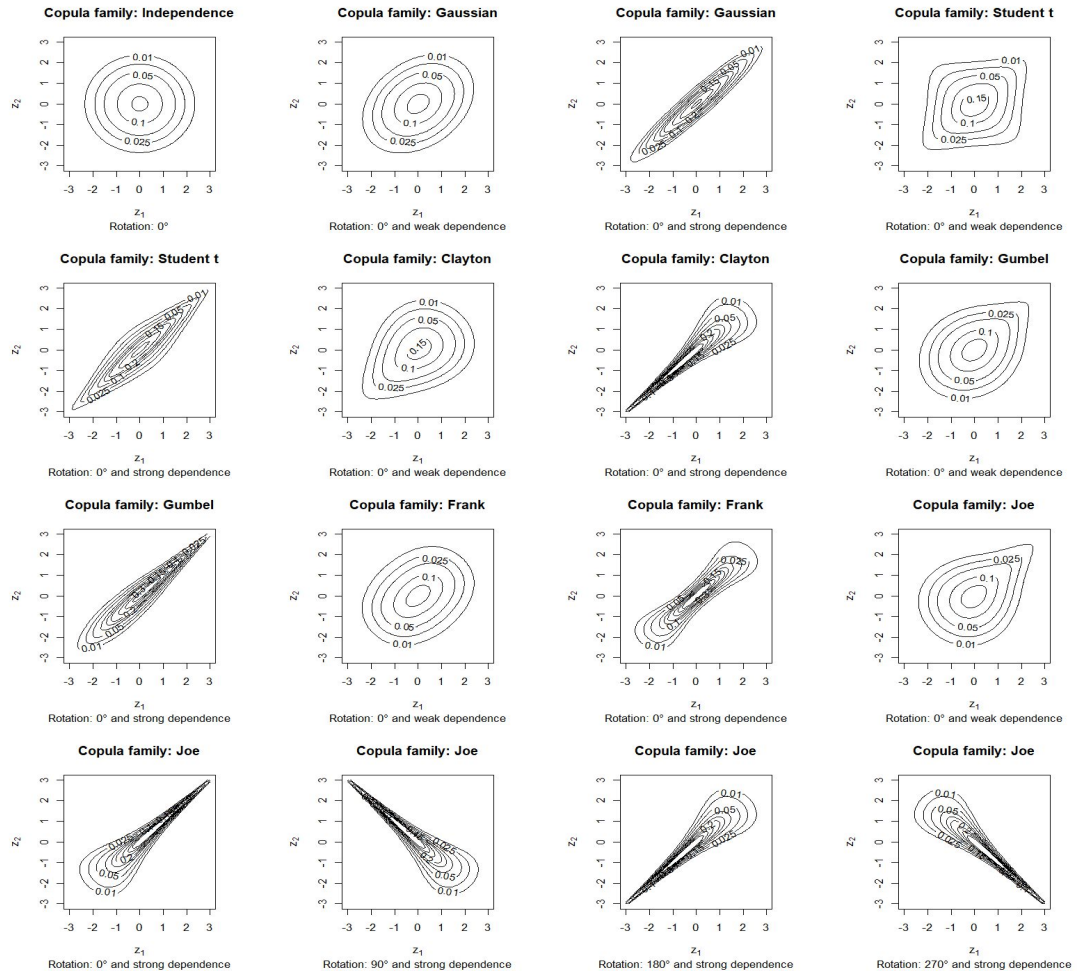

Figure 2.1. Marginal normal contour plots of some bivariate copula families.

2.2 Vine Copulas

Despite the large number of bivariate copulas, the multivariate case has been little studied until recently. As already said, the most popular model is multivariate gaussian distribution, even if it accounts only for symmetric and non-heavy tails. There have been efforts to extend the class of Archimedean copulas in the multivariate case, but these models require additional restrictions on parameters. On the other side, firstly Joe [1996], then Bedford and Cooke [2001], proposed a probabilistic construction of multivariate distributions based on bivariate copulas as building blocks. This approach is the so called pair-copula construction and was formally outlined by Aas et al. [2009].

The main idea is to decompose the dependence among n variables by studying the dependence among the $\frac{n(n-1)}{2}$ possible pairings of all variables. However, since to define the weight of an asset within a portfolio it is necessary to consider the dependence it has with all other assets, pair dependence is studied factorising the multivariate density. Consider for example a portfolio X=(X_1, X_2, X_3), where X_i represent the i-th asset. Its density can be written in two ways:

$$\begin{aligned} f_{123}(x_1, x_2, x_3) &= f_3(x_3) f_{2|3}(x_2|x_3) f_{1|23}(x_1|x_2, x_3) \\ f_{123}(x_1, x_2, x_3) &= c_{123}(F_1(x_1), F_2(x_2), F_3(x_3)) f_1(x_1) f_2(x_2) f_3(x_3) \end{aligned} \tag{2.3}$$

where $c_{i,j,k}$ is the copula density associated to the vector (X_i, X_j, X_k). The first equality holds as a consequence of the product formula, while the second follows from equation 2.2. By reiterating 2.3 for the elements present in the right hand side of the first equation it is possible to write:

$$\begin{aligned} f_{2|3}(x_2|x_3) &= \frac{f_{23}(x_2, x_3)}{f_3(x_3)} = \frac{c_{23}(F_2(x_2), F_3(x_3)) f_3(x_3) f_2(x_2)}{f_3(x_3)} = c_{23}(F_2(x_2), F_3(x_3)) f_2(x_2) \\ f_{1|23}(x_1|x_2, x_3) &= \frac{f_{123}(x_1, x_2, x_3)}{f_{23}(x_2, x_3)} = \frac{f_{13|2}(x_1, x_3|x_2) f_2(x_2)}{f_{23}(x_2, x_3)} = \\ &= \frac{c_{13|2}(F_{1|2}(x_1|x_2), F_{3|2}(x_3|x_2)) f_{1|2}(x_1|x_2) f_{3|2}(x_3|x_2) f_2(x_2)}{f_{23}(x_2, x_3)} = \\ &= c_{13|2}(F_{1|2}(x_1|x_2), F_{3|2}(x_3|x_2)) f_{1|2}(x_1|x_2) \\ &= c_{13|2}(F_{1|2}(x_1|x_2), F_{3|2}(x_3|x_2)) c_{12}(F_1(x_1), F_2(x_2)) f_1(x_1) \\ f_{123}(x_1, x_2, x_3) &= f_3(x_3) f_{2|3}(x_2|x_3) f_{1|23}(x_1|x_2, x_3) \\ &= f_3(x_3) c_{23}(F_2(x_2), F_3(x_3)) f_2(x_2) \cdot \\ &\cdot c_{13|2}(F_{1|2}(x_1|x_2), F_{3|2}(x_3|x_2)) c_{12}(F_1(x_1), F_2(x_2)) f_1(x_1) \end{aligned} \tag{2.4}$$

What we have done is writing a joint multivariate density as the product of univariate marginal densities and bivariate copulas, called pair-copulas. This is the approach behind the definition of vine copulas, which allows to evaluate families and parameters for each pair-copula independently, consequently obtaining a wide variety of dependency structures. The concept can be extended in the n-dimensional case and in order to do this we have to refer to the structures defined in Dißmann et al. [2013].

2.2.1 R-vine copulas definition

The first problem that arises from what said until now is the identification of the needed pairs of variables and their corresponding set of conditioning variables. In plain english, what is done for the portfolio X=(X_1, X_2, X_3) can be repeated in a completely similar way for the portfolio $\hat{X}$=(X_3, X_2, X_1). It is evident that the two portfolios are identical, however, the multivariate joint density factorisation will result different by applying equation 2.4. This problem is overcome by defining a sequence of trees called regular vine (R-vine), whose multivariate distribution will be the R-vine copula.

An object $\mathcal{V} = (T_1, ..., T_{n-1})$ defined over n elements is called a regular vine if:

- T_1 is a tree with nodes $N_1 = 1, ..., n$ and a set of edges denoted by E_1.
- For i=2,...,n-1, T_i is a tree with nodes N_i=E_{i-1} and edge set E_i.
- For i=2,...,n-1 and $\{a, b\} \in E_i$ with a=$\{a_1, a_2\}$ and b=$\{b_1, b_2\}$ it must hold that #(a $\cap$ b)=1 (proximity condition), where # denotes the cardinality of a set.

In simple words, as reported by Dißmann et al. [2013], an R-vine on n elements is a nested set of n-1 trees such that the edges of tree j become the nodes of tree j+1. Proximity condition ensures that two nodes in tree j+1 are connected if these nodes share a common node in tree j. An example of R-vine on seven variables can be seen in Figure 2.2, which will be used as reference to make all concepts clear from now on.

In order to define an R-vine copula we must first introduce some additional elements, which are:

- Complete union of an edge $e_i \in E_i$ $\forall i$: it is the set of all indexes that an edge countains. In formulas, $U_{e_i} = \{n_1 \in N_1 | \exists e_j \in E_j, j = 1, ..., i-1, \; with \; n_1 \in e_1 \in e_2 \in ... \in e_{i-1} \in e_i\} \in N_1$. For example, the complete union of the edge a between (1,2) and (2,3) in T_2 of Figure 2.2 is {1,2,3}. Similarly, the complete union of the edge b between (2,3) and (3,6) in T_2 of Figure 2.2 is {2,3,6}.
- Conditioning set of an edge $e_i = \{a, b\} \in E_i$ with a,b $\in N_i$ $\forall i$: it is the intersection of complete unions of edges a and b. Note that the meanings of node and edge are interchangeable, given the structure of R-vines where each node of a tree corresponds to an edge of the previous tree. In formulas, $D_{e_i} = U_a \cap U_b$ and as an example one can see that the conditioning set between edges a and b previously defined is {2,3}.
- Conditioned set of an edge $e_i = \{a, b\} \in E_i$ with a,b $\in N_i$ $\forall i$: it is the symmetric difference of complete unions of edges a and b. In formulas, $C_{e_i} = (U_a \setminus U_b) \cup (U_b \setminus U_a) = C_{e_i,a} \cup C_{e_i,b}$, where $C_{e_i,(\cdot)} := U_{(\cdot)} \setminus D_{e_i}$ can be thought as the conditioned set of a generic edge e_i with respect to a generic node/edge $(\cdot)$. For example, the conditioning set between edges a and b previously defined is {1,6}.
- Constrain set of $\mathcal{V}$: it is defined as $\mathcal{CV} = \{(\{C_{e,a}, C_{e,b}\}, D_e) | e \in E_i, e = \{a, b\}, i = 1, ..., n-1\}$. Essentially, it is a structure that combines the conditioning and conditioned sets for each edge. In Figure 2.2, for every edge of the R-vine, the conditioned set is represented by what is written before "|", while the conditioning set is represented by what is written after "|".

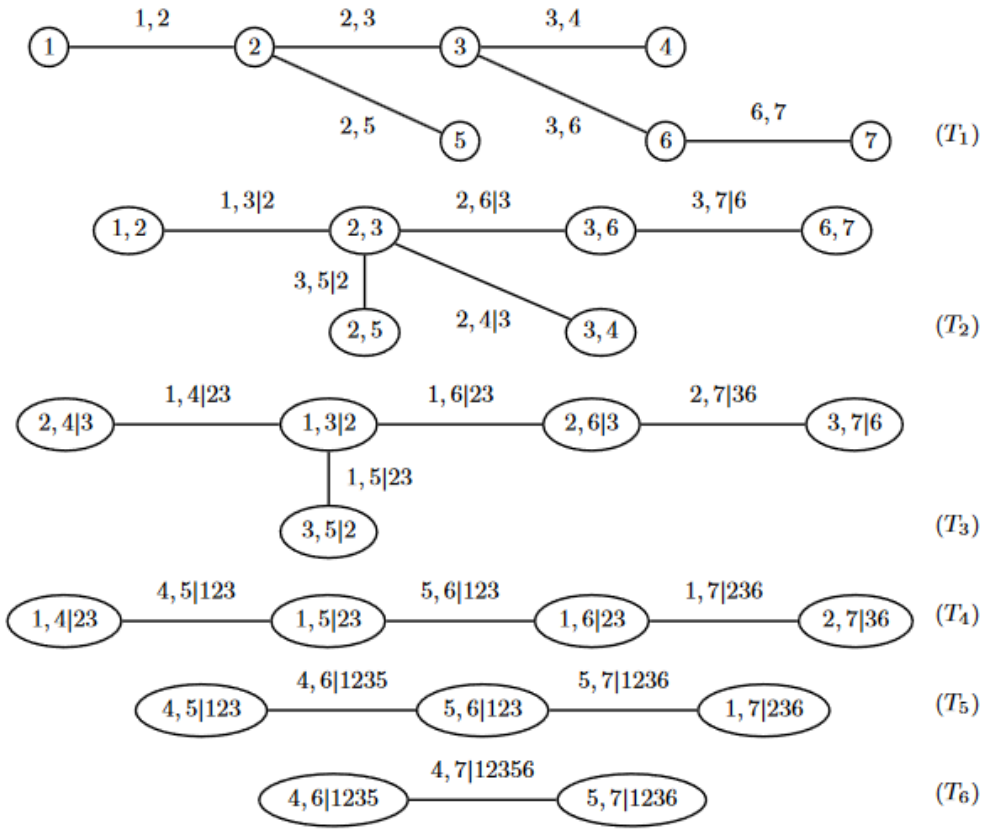

Figure 2.2. Example of R-vine on seven variables, taken from Dißmann et al. [2013].

Having defined all the possible details that characterize an R-vine object, we can now characterize R-vine copulas: the triple (F,$\mathcal{V}$,B) is called R-vine copula specification if F=$(F_1, ..., F_n)$ is a vector of continuous invertible distribution functions, $\mathcal{V}$ is a n-dimensional R-vine and B=$\{B_e | e \in E_i, i = 1, ..., n-1\}$ is the set of bivariate copulas families associated to each edge of the different trees, also called pair-copulas. It is important to immediately highlight the distinction between R-vine copula specification and R-vine copula distribution. As reported by Dißmann et al. [2013] a joint distribution F of a random vector $(X_1, ..., X_n)$ is said to realize an R-vine copula specification (F,$\mathcal{V}$,B) if for each $e \in E_i,\ i = 1, ..., n-1,\ e = \{a, b\},\ B_e$ is the bivariate copula of $X_{C_{e,a}}$ and $X_{C_{e,b}}$ given $X_{D_e} = \{X_i | i \in D_e\}$, where it is assumed that this conditional copula is independent of the conditioning variables X_{D_e}. The distribution followed by the copula instead is called R-vine copula distribution. In plain english, R-vine copula specification refers to data in real scale, whose copula associated with the multivariate CDF is an R-vine copula. On the other side, R-vine copula distribution refers to data on copula scale. Using the same notation of Dißmann et al. [2013], we will refer to the copula density with family B_e, associated to the edge e=$\{a, b\}$, as $c_{C_{e,a},C_{e,b}|D_e}$. One thing that can be observed

immediately concerns the possibility of adopting different copulas families independently from each other. Nonetheless, as the specified copulas in a tree impact the conditioned variables utilized in subsequent trees, the selection of different copulas will exert mutual influence. It therefore seems reasonable that portfolio dependences can be modeled by means of R-vine copulas.

2.2.2 Density of an R-vine specifications

We are now able to extend what is reported in equation 2.4 for the multivariate case with n>3. In fact, given a R-vine copula, thanks to the Sklar theorem and the notation adopted, there is a unique multivariate distribution F that realizes this copula, and its density is:

$$f_{1...n}(x) = \prod_{k=1}^{n} f_k(x_k) \prod_{i=1}^{n-1} \prod_{e \in E_i} c_{C_{e,a},C_{e,b}|D_e}(F_{Ce,a|D_e}(x_{C_{e,a}}|x_{D_e})F_{Ce,b|D_e}(x_{C_{e,b}}|x_{D_e})) \quad (2.5)$$

where x=$(x_1, ..., x_n)$, e=$\{a, b\}$, $x_{D_e} = \{x_i | i \in D_e\}$ and f_i is the marginal density $\forall i = 1, ..., n$.
It is quite obvious that the calculation of this function can be very complicated as the dimensionality increases. For this reason, Dißmann et al. [2013] defines a way to store R-vines as arrays. The concept involves organizing the constrain set of an R-vine within the columns of an n-dimensional lower triangular matrix. This structural arrangement will be used also in the next sections for likelihood calculation and sampling. The two main elements needed to encode R-vines in arrays are:

- Array constrain set: given a lower triangular matrix M=$(m_{i,j})_{i,j=1,...,n}$, it is defined as $\mathcal{CM} := C_M(1) \cup ... \cup C_M(n-1)$ where $C_M(i) = \{(\{m_{i,i}, m_{k,i}\}, D) | k = i+1, ..., n, D = \{m_{k+1,i}, ..., m_{n,i}\}\} \forall i = 1, ..., n-1$. Similarly to above, $\{m_{i,i}, m_{k,i}\}$ is called conditioned set, while D is the conditioning set.

- R-vine array: a lower triangular matrix M=$(m_{i,j})_{i,j=1,...,n}$ is a R-vine array if for i=1,...,n-1 and k=i+1,...,n-1, there is a j in i+1,...,n-1 such that $(m_{k,i}, \{m_{k+1,i}, ..., m_{n,i}\}) \in B_M(j)$ or $\in \hat{B}_M(j)$, where:

$$\begin{aligned} B_M(j) &:= \{(m_{j,j}, D) | k = j+1, ..., n; D = \{m_{k,j}, ..., m_{n,j}\}\} \\ \hat{B}_M(j) &:= \{(m_{k,j}, D) | k = j+1, ..., n; D = \{m_{j,j}\} \cup \{m_{k+1,j}, ..., m_{n,j}\}\} \end{aligned} \quad (2.6)$$

It can be proved that the constraint set $\mathcal{V}$ of an R-vine is equal to the array constraint set $\mathcal{CM}$ of the corresponding R-vine array M. However, matrix M associated to a R-vine structure is in general not unique, as outlined by Dißmann et al. [2013]. Using definition 2.6, it is possible to rewrite equation 2.5 as:

$$f_{1...n} = \prod_{j=1}^{n} f_j \prod_{k=n-1}^{1} \prod_{i=1}^{k+1} c_{m_{k,k},m_{i,k}|m_{i+1,k},...,m_{n,k}}(F_{m_{k,k}|m_{i+1,k},...,m_{n,k}}, F_{m_{i,k}|m_{i+1,k},...,m_{n,k}}) \quad (2.7)$$

where arguments of the fuctions are omitted to simplify notation.

Algorithm 2.1 Density of an R-vine specification.

Input: R-vine specification in array form, i.e., M, T, P, where $m_{k,k} = n - k + 1$, $k = 1, ..., n$.

Output: Density of the R-vine distribution at $(x_1, \dots x_n)$ for the given R-vine specification.

Set $f = 1$.
Allocate $V^{\text{direct}} = (v_{i,k}^{\text{direct}} | i, k = 1, \dots, n)$.
Allocate $V^{\text{indirect}} = (v_{i,k}^{\text{indirect}} | i, k = 1, \dots, n)$.
Set $(v_{n,1}^{\text{direct}}, v_{n,2}^{\text{direct}}, \dots, v_{n,n}^{\text{direct}}) = (F_n(x_n), F_{n-1}(x_{n-1}), \dots, F_1(x_1))$.
Let $\mathbf{M} = (\mathbf{m}_{i,k} | i, k = 1, \dots, n)$ with $\mathbf{m}_{i,k} = \max\{m_{i,k}, \dots, m_{n,k}\}$ for all $k = 1, \dots, n$ and $i = k, \dots, n$.
for $k = n - 1, \dots, 1$ **do** {Iteration over the columns of M}
  for $i = n, \dots, k + 1$ **do** {Iteration over the rows of M}
    Set $z_{i,k}^{(1)} = v_{i,k}^{\text{direct}}$.
    if $\mathbf{m}_{i,k} = m_{i,k}$ **then**
      Set $z_{i,k}^{(2)} = v_{i,(n-\mathbf{m}_{i,k}+1)}^{\text{direct}}$.
    else
      Set $z_{i,k}^{(2)} = v_{i,(n-\mathbf{m}_{i,k}+1)}^{\text{indirect}}$.
    end if
    Set $f = f \cdot c(z_{i,k}^{(1)}, z_{i,k}^{(2)} | t_{i,k}, p_{i,k})$.
    Set $v_{i-1,k}^{\text{direct}} = h(z_{i,k}^{(1)}, z_{i,k}^{(2)} | t_{i,k}, p_{i,k})$ and $v_{i-1,k}^{\text{indirect}} = h(z_{i,k}^{(2)}, z_{i,k}^{(1)} | t_{i,k}, p_{i,k})$, where h is the conditional distribution function as defined in (2).
  end for
end for
return Return the joint density f.

Figure 2.3. R-vine specification density estimation algorithm.

Figure 2.3 shows the procedure defined by Dißmann et al. [2013] to evaluate equation 2.7. Structures T and P are the matrices that contain information about types and parameters of each bivariate copula. In particular, for all $j = 1, \dots, n-1$, $i = j+1, \dots, n$, the entry $m_{i,j}$ of matrix M encodes the copula of variables indexed by $m_{j,j}$ and $m_{i,j}$ conditional on the variables indexed by $\{m_{i+1,j}, \dots, m_{n,j}\}$. $t_{i,j}$ denotes the copula family, while $p_{i,j}$ contains the parameters of this copula. M is assumed with all the diagonal entries ordered from n to 1, i.e., $m_{k,k} = n - k + 1 \forall k$. Ordering the array differently will lead to the same R-vine but with relabeled indexes. $\mathbb{M}$ is called the maximum array and is defined as:

$$\mathbb{M} = (\mathbf{m_{i,k}})_{i,k=1,\dots,n} \; with \; \mathbf{m_{i,k}} = max\{m_{i,k}, ..., m_{n,k} \; \forall k = 1, ..., n \; and \; i = k, ..., n\} \quad (2.8)$$

In words, $\mathbf{m_{i,k}}$ is the maximum of all entries in the k-th column of M from the bottom up to the i-th element. The definition of $\mathbb{M}$ is used to associate the correct indexes with the different entries of the matrix M. Knowing the parameters and families that characterize

each density c of equation 2.7, it is possible to evaluate the conditional distributions $F_{(\cdot)|(\cdot)}$ inside the brackets of c. This can be done using the so called h-functions. According to the notation of equation 2.5, Joe [1996] proved that the following relation holds:

$$\begin{aligned} F_{Ce,a|D_e}(x_{C_{e,a}}|x_{D_e}) &= \frac{\partial C_{Ca|D_a}(F_{C_{a,a_1}|D_a}(x_{C_{a,a_1}}|x_{D_a}), F_{C_{a,a_2}|D_a}(x_{C_{a,a_2}}|x_{D_a}))}{\partial F_{C_{a,a_2}|D_a}(x_{C_{a,a_2}}|x_{D_a})} \\ &:= h(F_{C_{a,a_1}|D_a}(x_{C_{a,a_1}}|x_{D_a}), F_{C_{a,a_2}|D_a}(x_{C_{a,a_2}}|x_{D_a})) \end{aligned} \tag{2.9}$$

where $F_{C_{a,a_1}|D_a}(x_{C_{a,a_1}}|x_{D_a})$ and $F_{C_{a,a_2}|D_a}(x_{C_{a,a_2}}|x_{D_a})$ are obtained in the same way by recursion. With all these instruments the algorithm basically performs an outer for-loop to iterate over the columns of M from right to left, an inner for-loop to iterate over the rows of M from bottom up to the diagonal and computes the h-function$_{i,k}$ for copula type $t_{i,k}$ with parameters $p_{i,k}$ in a recursively manner. The h-functions are stored in the arrays V^{direct} and $V^{indirect}$. Dißmann et al. [2013] proved that $z^{(1)}_{i,k} = F_{m_{k,k}|m_{i+1,k},...,m_{n,k}}(x_{k,k}|x_{i+1,k}, ..., x_{n,k})$ and $z^{(2)}_{i,k} = F_{m_{i,k}|m_{i+1,k},...,m_{n,k}}(x_{i,k}|x_{i+1,k}, ..., x_{n,k})$ for k=n-1,...,1 and i=n,...,k+1. This allows to evaluate efficiently the multivariate density of an R-vine copula specification.

2.3 Structure Selection of R-vine Copula specifications

Up to now we have seen how to evaluate the density of an R-vine copula specification given a vector of observed values. However, copulas parameters were supposed to be known. In a context where these arguments are not known a maximum log-likelihood estimation approach can be used to infer them. Log likelihood expression L can be obtained repeating algorithm shown is Figure 2.3 and sobstituting L=0 in line 0 and $L = L + log\ c(z^{(1)}_{i,k}, z^{(2)}_{i,k}|t_{i,k}, p_{i,k})$ in line 14. A more operational and less computationally expensive approach than what has just been said is the sequential estimation, where tree structure of R-vines is exploited to estimate the parameters of each pair-copula of each tree separately. Essentially, the parameters of pair copulas associated with the first tree are first estimated, then the variables are transformed into the copula scale and used to estimate the parameters of pair copulas in the subsequent tree. This process is repeated until all the trees are finished and usually represents a good approximation for the joint maximum likelihood estimation.

It is therefore clear that the problem underlying this sequential approach is the characterization of R-vine structures, which from a mathematical point of view translates into the problem of determining the correct order of the trees. In fact, sorting tree nodes differently from an initial R-vine copula specification will result in the same R-vine but with relabeled indexes. From a practical point of view, since these indexes could, for example, represent assets, it is important to link them correctly and meaningfully. Wrong assets connections could have drastic consequences on the estimation of portfolios internal dependencies, which in turn could lead to incorrect prediction of risk measures in stressed market conditions. This point is highlighted by Dißmann et al. [2013] to be important for real-world applications, where R-vine copulas must be fitted from a generic data set.

An additional reason for adopting a sequential estimation strategy, instead of for example testing all possible copula specifications, lies in the very rapid increase in the number of possible index orderings, and so of R-vine tree structures, as the total number of nodes increase. It is quite intuitive how the global optimum is not guaranteed with a sequential approach, which is why a possible future extension could be instead to consider a dynamic programming approach for structure selection. Despite this, with the model under consideration, trees are selected so that the chosen pairs model the strongest pairwise dependencies. This is a desiderable feature for real world applications, as usually it is important to firstly model correctly the dependence structure between random variables that have high dependence. In addition, it is also natural to assume that randomness is driven by the dependence of only some variables and not all. All these concepts are translated into formulas by algorithm in Figure 2.4. Kendall tau is chosen as a measure

Algorithm 3.1 Sequential method to select an R-vine model based on Kendall's tau.

Input: Data $(x_{\ell 1}, \dots x_{\ell n})$, $\ell = 1, ..., N$ (realizations of i.i.d. random vectors).
Output: R-vine copula specification, i.e., $\mathcal{V}$, B.

1: Calculate the empirical Kendall's tau $\hat{\tau}_{j,k}$ for all possible variable pairs $\{j,k\}, 1 \leq j < k \leq n$.
2: Select the spanning tree that maximizes the sum of absolute empirical Kendall's taus, i.e.,

$$\max \sum_{e=\{j,k\} \text{ in spanning tree}} |\hat{\tau}_{j,k}|.$$

3: For each edge $\{j,k\}$ in the selected spanning tree, select a copula and estimate the corresponding parameter(s). Then transform $\widehat{F}_{j|k}(x_{\ell j}|x_{\ell k})$ and $\widehat{F}_{k|j}(x_{\ell k}|x_{\ell j})$, $\ell = 1, ..., N$, using the fitted copula $\widehat{C}_{jk}$ (see (2)).
4: **for** $i = 2, \dots, n-1$ **do** {Iteration over the trees}
5: Calculate the empirical Kendall's tau $\hat{\tau}_{j,k|D}$ for all conditional variable pairs $\{j,k|D\}$ that can be part of tree T_i, i.e., all edges fulfilling the proximity condition (see Definition 2.1).
6: Among these edges, select the spanning tree that maximizes the sum of absolute empirical Kendall's taus, i.e.,

$$\max \sum_{e=\{j,k|D\} \text{ in spanning tree}} |\hat{\tau}_{j,k|D}|.$$

7: For each edge $\{j,k|D\}$ in the selected spanning tree, select a conditional copula and estimate the corresponding parameter(s). Then transform $\widehat{F}_{j|k\cup D}(x_{\ell j}|x_{\ell k}, \boldsymbol{x}_{\ell D})$ and $\widehat{F}_{k|j\cup D}(x_{\ell k}|x_{\ell j}, \boldsymbol{x}_{\ell D})$, $\ell = 1, ..., N$, using the fitted copula $\widehat{C}_{jk|D}$ (see (2)).
8: **end for**

Figure 2.4. R-vine structure selection algorithm.

of dependece, since it is at least able to capture correlations that are not necessarily linear. Given two random vectors $X = [X_1, X_2]$, $\hat{X} = [\hat{X}_1, \hat{X}_2]$ independent and identically distributed, it is defined as:

$$\begin{aligned}\rho_\tau(X, \hat{X}) &= P\{(X_1 - \hat{X}_1)(X_2 - \hat{X}_2) > 0\} - P\{(X_1 - \hat{X}_1)(X_2 - \hat{X}_2) < 0\} \\ &= \mathbb{E}[sign((X_1 - \hat{X}_1)(X_2 - \hat{X}_2))]\end{aligned} \tag{2.10}$$

Trees are constructed maximizing the sum of absolute empirical Kendall taus, which are the weights of edges connecting nodes in the trees. This is done using a maximum spanning tree algorithm (MST), such as Prim's algorithm, where trees are constructed starting with a vertex and incrementally adding the edge with the largest ρ_τ. Proximity condition of R-vines ensure that every graph constructed is also connected, so that for sure MST will lead finally to a tree. Once the structure is defined, pair-copula families and the respective parameters should be chosen and estimated. The selection process involves calculating the AIC for each potential copula family and then choosing the copula with lowest AIC. Independence copulas are incorporated into the selection procedure, since going deeper and deeper into the trees one observes that Kendall tau decreases, as shown by Dißmann et al. [2013] in the multivariate normal case. It is precisely at this point that the strength of R-vine copulas can be observed, since the choice of the best copula family with related parameters will be guided by the degree of dependence that exists between the variables under examination, rather than by the adoption of always the same family with just different parameters. Dependence that can be positive or negative in the tails and distributed symmetrically or asymmetrically depending on the chosen family.

Chapter 3

Monte Carlo Simulation

Monte Carlo simulations play a critical role in refining and validating models. These simulations, rooted in probabilistic methodologies, allow to evaluate the robustness of models under varying conditions, providing valuable insights into their behavior. This chapter is dedicated to assess the plausibility of R-vines structure selection model. In particular, the approach under examination regards the sequential selection and estimation method presented in the previous section.

3.1 Inference of R-vine Copula specifications

Besides parameter estimation, there is also another aspect to take into consideration in statistical inference, namely simulation, which allows to numerically explore the distribution of statistical results that usually have analytical closed formulas complex to evaluate. For R-vine copula specifications, simulation can be done using the inverse probability integral transformation. Starting from the base, given a continuous random variable X with cumulative distribution F_X and a realization x, the probability integral trasform (PIT) of x is nothing but u:=F_X(x). More in general, the random variable U:=$F_X(X) \sim$ U[0,1], since it holds that:

$$F_U(u) = P(U \leq u) = P(F_X(X) \leq u) = P(X \leq F_X^{-1}(u)) = P(X \leq x) = F_X(x) \quad (3.1)$$

This means that the PIT is the transformation which maps data from original scale to copula scale. However, since R-vine copula specifications are in terms of real scale, our objective is to simulate a sample on the real scale, given that its copula follows an R-vine distribution. Therefore, the endpoint is represented by x rather than u. That is why simulation is done using the inverse probability integral transform: given a uniform random variable U[0,1], i.e. in copula scale, $F_X^{-1}(U)$ is a random variable with distribution equal to F_X, since it holds that:

$$F_{F_X^{-1}(U)}(x) = P(F_X^{-1}(U) \leq x) = P(F_X(F_X^{-1}(U)) \leq F_X(x)) = P(U \leq F_X(x)) = F_X(x) \quad (3.2)$$

where the last equality is true because the CDF of uniform random variables is the identity function. The reasoning can be extended also in terms of h-functions for equation 2.9, since

they are defined as conditional cumulative distribution functions. This enables to obtain a sample from an R-vine copula specification by performing the following operations:

$$\begin{aligned} x_1 &= u_1 \\ x_2 &= F^{-1}_{2|1}(u_2|x_1) \\ x_3 &= F^{-1}_{3|12}(u_3|x_1, x_2) \\ &... \\ x_n &= F^{-1}_{n|12...n-1}(u_n|x_1, ..., x_{n-1}) \end{aligned} \tag{3.3}$$

where $u_1, ..., u_n$ are independent, uniform samples on [0,1] while $F^{-1}_{j|12...j-1}(u_j|x_1, ..., x_{j-1})$ $j = 1, ...n$ is computed using the inverse of h functions. Rosenblatt [1952] established the validity of the relationships outlined in equations 3.3 through an inductive approach. All these concepts are summarized in Figure 3.1, where the algorithm is exactly the same of Figure 2.3, with the only differences related to the need to apply an inverse PIT instead of evaluating the density.

3.2 Empirical Application

The simulation study explained below confirms the validity of the presented model. The analysis is conducted in four different scenarios. For each one of these, R-vine tree structure chosen is the same of Figure 2.2. Dißmann et al. [2013] have shown that the corresponding R-vine array is:

$$M = \begin{bmatrix} 7 & & & & & & \\ 4 & 4 & & & & & \\ 5 & 6 & 6 & & & & \\ 1 & 5 & 5 & 5 & & & \\ 2 & 1 & 1 & 1 & 1 & & \\ 3 & 2 & 2 & 3 & 3 & 3 & \\ 6 & 3 & 3 & 2 & 2 & 2 & 2 \end{bmatrix}$$

The process consists in simulating data knowing parameters and families of each pair-copula. After this, structure selection algorithm of Figure 2.4 is applied to derive a plausible R-vine copula specification. To evaluate the accuracy of the model in reconstructing the initial structure, three quantitites are defined, as done by Dißmann et al. [2013]:

- General τ difference: this meausure aims to capture the ability of the proposed model to correctly derive the dependency between observations. Having available the structure of the fitted and real models, data from the fitted model are simulated and the empirical Kendall tau of the latter is calculated. Similarly, the empirical Kendall tau is calculated from the initial (real) data, which had already been simulated from the known R-vine structure. These two quantities are then subtracted obtaining an error matrix where each entry belongs to the interval [-2,2], since $\rho_\tau \in [-1,1]$ is a measure of dependence between two variables. Subsequently, the average over the error matrix is calculated to obtain a closed value as error. The procedure is iterated

Algorithm 2.2 Simulation of an R-vine specification.

Input: R-vine specification in array form, i.e., M, T, P, where $m_{k,k} = n - k + 1,\ k = 1, ..., n$.

Output: Random observations $(x_1, \ldots, x_n)$ from the R-vine specification.

Let $u_1, \ldots, u_n$ be independent uniform samples.
Allocate $V^{\text{direct}} = (v_{i,k}^{\text{direct}} | i, k = 1, \ldots, n)$.
Allocate $V^{\text{indirect}} = (v_{i,k}^{\text{indirect}} | i, k = 1, \ldots, n)$.
Set $(v_{n,1}^{\text{direct}}, v_{n,2}^{\text{direct}}, \ldots, v_{n,n}^{\text{direct}}) = (u_1, u_2, \ldots u_n)$.
Let $\mathbf{M} = (\mathbf{m}_{i,k} | i, k = 1, \ldots, n)$ with $\mathbf{m}_{i,k} = \max\{m_{i,k}, \ldots, m_{n,k}\}$ for all $k = 1, \ldots n-1$ and $i = k, \ldots, n$.
$x_1 = v_{n,n}^{\text{direct}}$
for $k = n-1, \ldots, 1$ **do** {Iteration over the columns of M}
  for $i = k+1, \ldots, n$ **do** {Iteration over the rows of M}
    if $\mathbf{m}_{i,k} = m_{i,k}$ **then**
      Set $z_{i,k}^{(2)} = v_{i,(n-\mathbf{m}_{i,k}+1)}^{\text{direct}}$.
    else
      Set $z_{i,k}^{(2)} = v_{i,(n-\mathbf{m}_{i,k}+1)}^{\text{indirect}}$.
    end if
    Set $v_{n,k}^{\text{direct}} = h^{-1}(v_{n,k}^{\text{direct}}, z_{i,k}^{(2)} | t_{i,k}, p_{i,k})$
  end for
  $x_{n-k+1} = v_{n,k}^{\text{direct}}$
  for $i = n, \ldots, k+1$ **do** {Iteration over the rows of M}
    Set $z_{i,k}^{(1)} = v_{i,k}^{\text{direct}}$
    Set $v_{i-1,k}^{\text{direct}} = h(z_{i,k}^{(1)}, z_{i,k}^{(2)} | t_{i,k}, p_{i,k})$ and $v_{i-1,k}^{\text{indirect}} = h(z_{i,k}^{(2)}, z_{i,k}^{(1)} | t_{i,k}, p_{i,k})$.
  end for
end for
return Return sample $(x_1, \ldots, x_n)$.

Figure 3.1. R-vine specification simulation algorithm.

a specific number of times, averaging the errors obtained across different repetitions to mitigate estimate variability and prevent the occurrence of biased results.

- Lower and Upper τ difference: these errors are a natural extension of the previous one to measure the strenght of joint tail behavior. Same procedure of above in fact is applied to the so called coefficient of upper and lower exceedance Kendall tau, defined in the bidimensional case as:

$$\begin{aligned} \tau^{lower}(U_1, U_2) &:= \tau(U_1, U_2 | U_1 \leq 0.2, U_2 \leq 0.2) \\ \tau^{upper}(U_1, U_2) &:= \tau(U_1, U_2 | U_1 \geq 0.8, U_2 \geq 0.8) \end{aligned} \tag{3.4}$$

where U_1 and U_2 represents two variables in the copula scale.
In simple words, having simulated data by the true and real model, for each possible pairing of variables only the observations that are lower or higher than the thresholds

are retained. The idea is therefore to have a measure of dependence for what can be considered as extreme events, given that it depends on whether the variables take on very small or large values. Also in this case a lower triangular empirical Kendall tau matrices are calculated, subtracted, averaged over their entries and then over all repetitions.

All the empirical versions of Kendall tau are calculated using the algorithm developed by Knight [1966], which is a computer method for calculating Kendall tau for ungrouped data. The scenarios considered are:

1. Fixed BB1 copula family: each pair-copula family contained in the matrix T is of the same type, i.e. BB1 copula with no rotation. This is a two-parameter copula used to capture more than one type of dependence on tails. In particular, as reported by Joe [1997], copula density can be written as:

$$C(u, v, \theta, \sigma) = \{1 + [(u^{-\theta} - 1)^{\sigma} + (v^{-\theta} - 1)^{\sigma}]^{1/\sigma}\}^{-1/\theta} \ \theta > 0, \ \sigma \geq 1. \tag{3.5}$$

   where θ and σ are the two parameters. Upper and lower tail dependece coefficients are respectively equal to $\lambda_u = 2 - 2^{1/\sigma}$, $\lambda_l = 2^{-1/(\theta\sigma)}$. For numerical reasons, the first parameter can be only in the interval [0,7], while the second one can be only in [1,7].

2. BB1 copulas with random rotations: this scenario is a more general case of the previous one. Copula family is still fixed but rotations are allowed. The entries of matrix T in fact are sampled randomly choosing between BB1 copula without rotation or with rotations of 90, 180 or 270 degrees.

3. Mixed copulas as done by Dißmann et al. [2013]: this approach is slightly different from those presented above. Parameters are not chosen randomly but according to a prespecified Kendall tau and family matrix T:

$$\tau = \begin{bmatrix} & & & & & \\ 0.05 & & & & & \\ 0.10 & 0.10 & & & & \\ 0.15 & 0.15 & 0.15 & & & \\ 0.20 & 0.20 & 0.20 & 0.20 & & \\ 0.25 & 0.30 & 0.35 & 0.40 & 0.45 & \\ 0.50 & 0.55 & 0.60 & 0.65 & 0.70 & 0.75 \end{bmatrix}$$

$$T = \begin{bmatrix} & & & & & \\ N & & & & & \\ F & N & & & & \\ N & F & N & & & \\ G & SG & G & SG & & \\ F & N & F & N & t & \\ SG & G & SG & G & t & t \end{bmatrix}$$

   where N=Gaussian, t=Student-t, G=Gumbel, SG=180 degrees rotated Gumbel, F=Frank. The choice to determine parameters starting from a setting of Kendall

taus is possible because only families of one parameter copulas have been chosen, with the exception of Student-t. Closed formulas of parameters are available on the user guide of *VineCopula* package in R. Regarding Student-t, the first parameter is derived from Kendall tau values while the second one, i.e. the degree of freedom, is linearly increased by 1 for each pair-copula terms in higher order trees and start with 3 in the first tree, as done by Dißmann et al. [2013].

4. Mixed copulas choosing every possible familiy: this scenario represents the most general case possible that can be implemented based on all the functions defined within the *VineCopula* package. In fact, copula families are selected randomly from all the existing families, which are included below for the sake of completeness in the discussion: Gaussian, Student-t, Clayton, Gumbel, Frank, Joe, BB1, BB6, BB7, BB8, Tawn type 1, Tawn type 2 and all the possibile rotation of 90, 180 or 270 degrees. Differently from the mixed scenario of Dißmann et al. [2013], parameters are sampled randomly and not starting from a Kendall tau setting. For numerical reasons, each copula has a limited range of allowed parameter values. The final family matrix obtained is:

$$T = \begin{bmatrix} 90°\text{Joe} & & & & & \\ 180°\text{BB8} & 180°\text{BB7} & & & & \\ \text{N} & \text{t} & 90°\text{Joe} & & & \\ 180°\text{Tawn type 1} & 270°\text{BB8} & 180°\text{G} & 180°\text{G} & & \\ \text{N} & 270°\text{BB1} & 180°\text{Tawn type 1} & 90°\text{BB7} & \text{Independence} & \\ \text{G} & 90°\text{Tawn type 1} & 180°\text{BB6} & 90°\text{G} & 180°\text{BB6} & 180°\text{BB1} \end{bmatrix}$$

where the abbreviations are the same adopted above and the notation $x°Y$ stands for copula family Y rotated of x degrees.

The whole study is repeated considering different sizes in terms of number of simulated samples N_1 and number of repetition N_2. Table 3.1 summarizes results obtained. Some considerations can be made looking at the table. In fact, the scenarios have been numbered in ascending order of complexity. Imagining, for example, that the data generated represent observations of portfolio returns: in the first scenario the dependence between assets can be considered fixed, while in the last scenario the interdependence is quite random, within the limits of copula families defined in *VineCopula* package. In general, errors are very low and decrease as the number of observations and repetitions increase. This is a desirable feature, since variability due to simulations can affect what are the final estimates. Two slightly anomalous cases are represented by scenarios 2 and 4. In the case of BB1 copulas with random rotations what is observed is that the generic τ difference remains fairly stable as dimensionality increases while the tail dependence errors do not have a linear behavior. Regarding scenario 4, increasing number of repetitions seems to help decrease in the error. However, increasing the sample size also adds noise, particularly for the upper τ difference. These behaviors might indicate greater difficulty in modeling asymmetric dependencies between different assets, for example, if some are positively and others negatively correlated. Errors remain fairly circumscribed,

Scenario	Dimensions	General τ diff.	Lower τ diff.	Upper τ diff.
1	$\{N_1 = 500, N_2 = 10\}$	0.014	0.045	0.072
1	$\{N_1 = 500, N_2 = 100\}$	0.011	0.039	0.057
1	$\{N_1 = 500, N_2 = 1000\}$	0.012	0.040	0.061
1	$\{N_1 = 1000, N_2 = 1000\}$	0.009	0.033	0.056
2	$\{N_1 = 500, N_2 = 10\}$	0.010	0.021	0.058
2	$\{N_1 = 500, N_2 = 100\}$	0.012	0.020	0.031
2	$\{N_1 = 500, N_2 = 1000\}$	0.013	0.029	0.023
2	$\{N_1 = 1000, N_2 = 1000\}$	0.012	0.030	0.050
3	$\{N_1 = 500, N_2 = 10\}$	0.024	0.064	0.062
3	$\{N_1 = 500, N_2 = 100\}$	0.022	0.057	0.063
3	$\{N_1 = 500, N_2 = 1000\}$	0.020	0.058	0.067
3	$\{N_1 = 1000, N_2 = 1000\}$	0.015	0.046	0.051
4	$\{N_1 = 500, N_2 = 10\}$	0.024	0.034	0.041
4	$\{N_1 = 500, N_2 = 100\}$	0.081	0.076	0.080
4	$\{N_1 = 500, N_2 = 1000\}$	0.036	0.043	0.057
4	$\{N_1 = 1000, N_2 = 1000\}$	0.017	0.029	0.116

Table 3.1. Simulation study results.

although the problem of modeling tail dependencies is an important issue, as highlighted earlier when discussing the leverage effect. Indeed, it clearly emerges that the upper tau exceedance is more variable than the two other estimates. The positive side of the coin is that the issue we aim to tackle primarily concerns modeling dependence amidst market stresses, particularly focusing on downward assets returns, i.e. low values on copula scale, which are well captured according to the small values of lower τ exceedance. However, it remains beneficial for financial institutions to establish a clear correlation between assets during for example bull markets.

The interested reader might question the methodology used to validate the model. In fact, algorithm of Figure 3.4 shows the process of selecting the structure and families associated with R-vine copulas. Thus, one might wonder how well the method presented is able to reconstruct the exact structure and families given simulations. This verification was observed to come up short in general when comparing the estimated families and structures with the original ones. However, it should be pointed out that the initial problem was modeling dependencies between assets in the multivariate case and not modeling portfolios. Since these tools will later be used to estimate measures associated with capital requirements, the structure comes up short given the need to be able to asses correlation in stressed conditions.

Chapter 4

Probability Equivalent Level Analysis

The final section of this work focuses on integrating the elements discussed so far, i.e. risk measures and vine copulas. Sommer [2022] has already provided a framework and the *portvine* package to tackle this problem. However, what remains unexplored is a recent research area, called the study of probability equivalent levels. The following paragraphs aim to provide an empirical and non-formal demonstration regarding the characterization of these levels in the multidimensional case and for different risk measures. The exploration is made possible thanks to the flexibility of R-vine copulas. Additionally, a different, and hopefully more operational, interpretation of the topic is also presented.

Before going into details, it may be useful to give a brief idea of what will be covered. Figure 4.1 and 4.2, taken from Sommer [2022], explain simply the context.

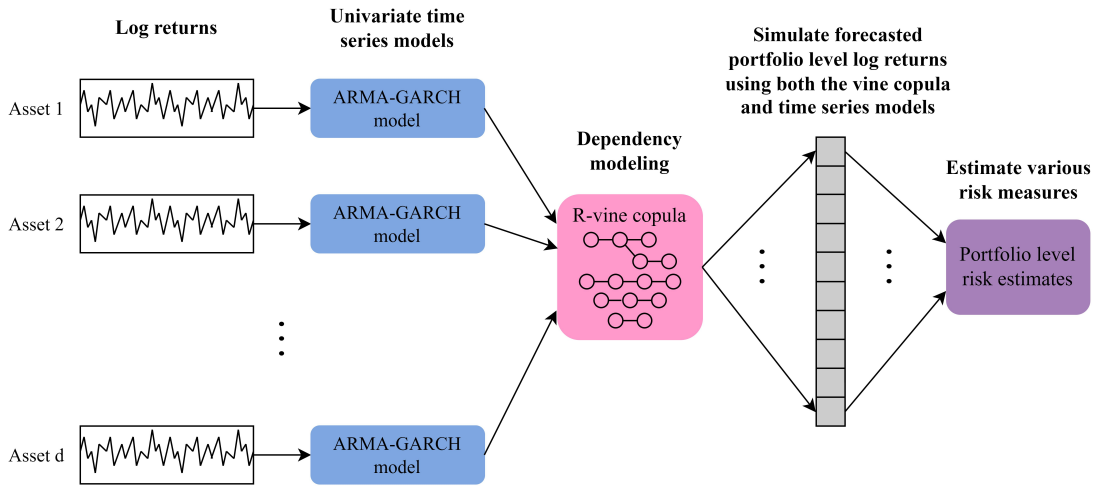

Figure 4.1. Unconditional risk estimation approach.

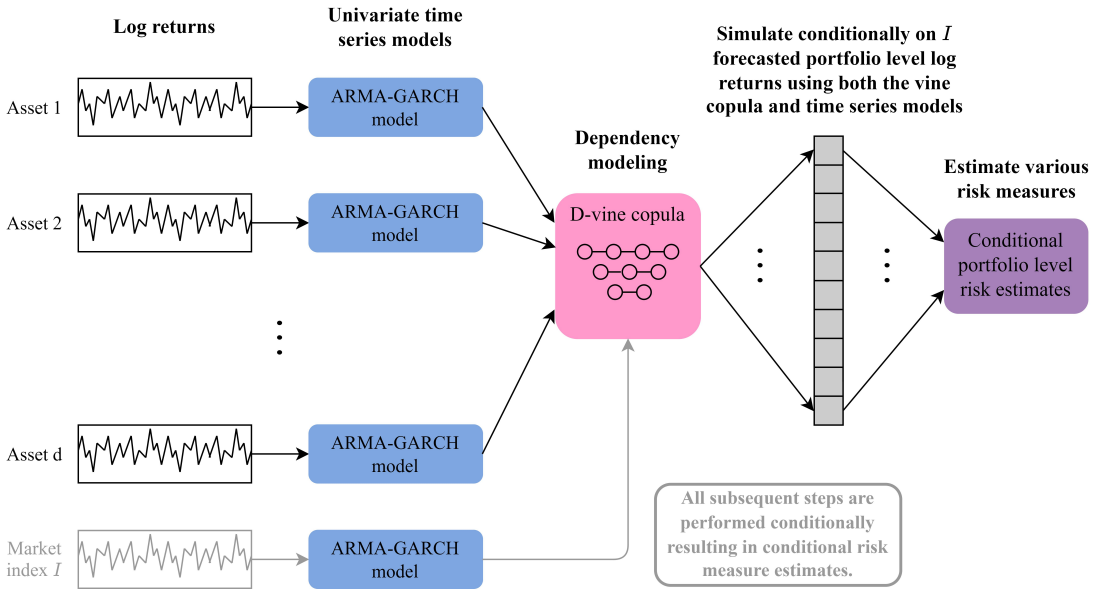

Figure 4.2. Conditional risk estimation approach.

Given a portfolio composed of several assets, each one is firstly modeled singularly with ARMA-GARCH processes, then their dependece is modeled thanks to R-vine copulas and finally risk measure estimates are obtained starting from simulated portfolio log returns. Figure 4.2 introduce a further element with respect to first one, i.e. the conditioning on the value of other assets to obtain the so called conditional risk measures. This will be the starting point for probability equivalent level analysis.

4.1 Unconditional vine copula based risk measure estimation

Figure 4.1 gives already a general idea of how a risk measure is inferred, but going deeper in the analysis, Sommer [2022] has developed a method based on vine copulas and a rolling window. As already said, a rolling window is basically a window of fixed size moving through a time series data. The process repeated in each window is called one step ahead unconditional risk measure estimation, and is summarized in Figure 4.5.

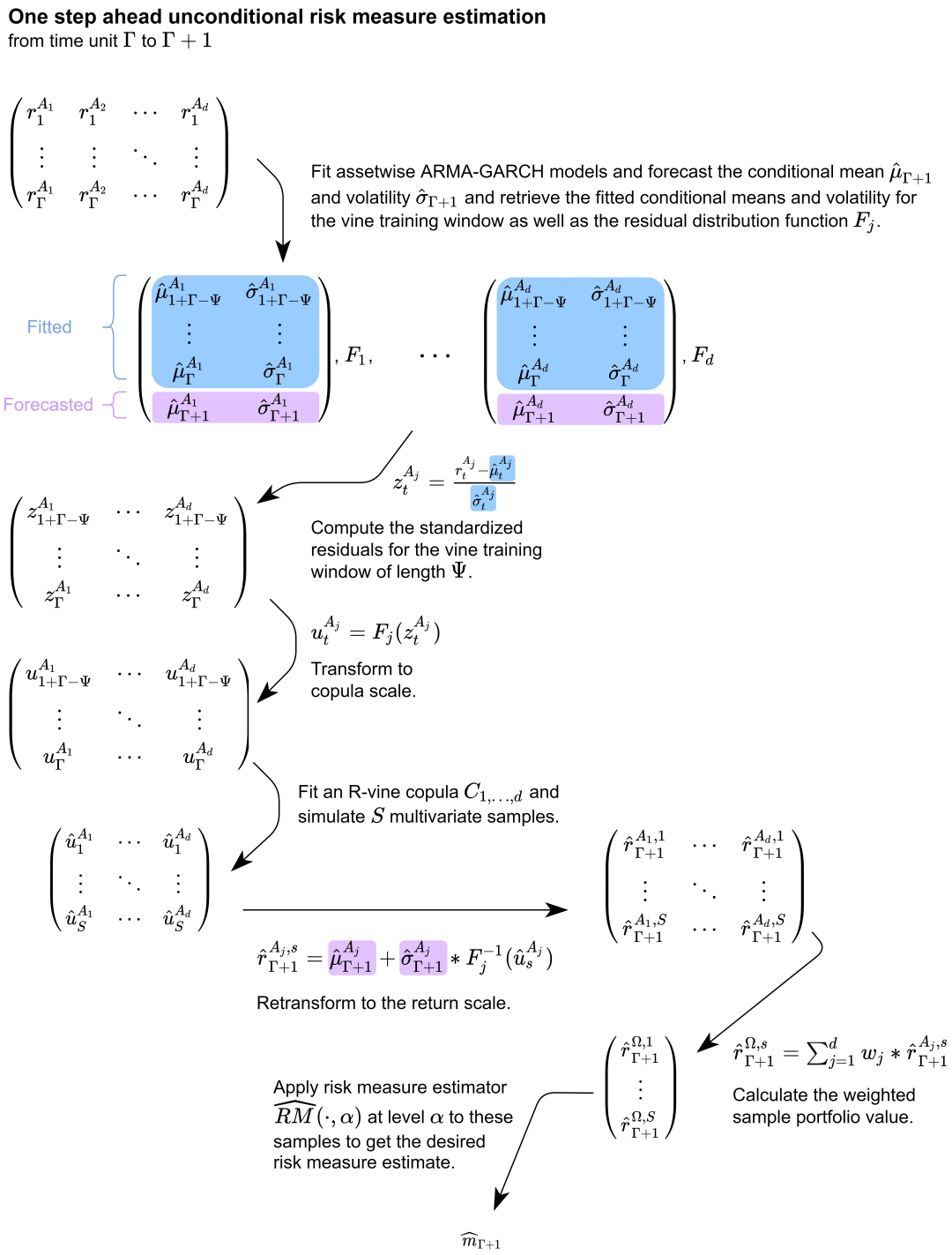

Figure 4.3. One step ahead unconditional risk measure estimation approach.

The notation adopted by Sommer [2022] is mandatory to understand the concept:

- $\Omega = \{w_j, r_t^{A_j} | t = 1, ..., T : j = 1, ...d\}$ is a portfolio with d $\in \mathbb{N}$ assets denoted as $A_1, ..., A_d$. $r_t^{(\cdot)}$ represents the log return of $(\cdot)$ at time t, while w_j are assets weights within the portfolio.
- $\Gamma < T$ is the length of fitting window for marginal models, which moves forward as time goes on.
- $\Psi \leq \Gamma$ is the length of fitting window for vine copula models, which moves forward as time goes on.
- S $\in \mathbb{N}$ is the number of simulated log return for risk measure estimation.

Knowing this, the procedure for one step ahead forecast can be summarized as:

1. Fit an ARMA-GARCH model for each log return series for the time frame of the fitting window Γ. Subsequently, forecast mean and volatility at time $\Gamma + 1$. A grid search to find best model orders has been set up. However, given the large computational costs also deriving from the other parts that will be described later, ARMA(1,1)-GARCH(1,1) models were adopted for each series, being recognized in literature as good starting points for financial applications.
2. Compute standardized residuals for each univariate model, transform them in copula scale using a probability integral transformation and fit a R-vine copula along the time frame of the fitting window Ψ.
3. Simulate S samples on copula scale from the R-vine copula just fitted, transform them in the log returns scale applying an inverse probability integral transformation and use the weight of each asset to compute portfolio log return samples at time $\Gamma + 1$.
4. From the vector of dimension S a Monte Carlo method is used to derive risk measures estimates. The focus of this work will be in VaR and ES. Regarding the first one, the standard empirical quantile function based on the set of S samples can be evaluated at the confidence level α chosen, to obtain the forecast $\widehat{VaR}_{\Gamma+1}$. Regarding ES instead, the mean of simulated samples that fall under the corresponding $\widehat{VaR}_{\Gamma+a+1}$ is calculated to obtain the forecast $\widehat{ES}_{\Gamma+1}$.

A computational burden underlying the model just presented is that marginals and dependency structures are refitted at each step, from Γ+1 up to T, which is considered the final time horizon. For this reason, the definition of parameters γ and κ was introduced. The main simplifications they introduce concern the possibility of making forecast for time interval grater than 1 and refit models after a certain period of use, in order to capture evolutions in the dependency structure but maintaining a balance in terms of computational costs. In particular, $\gamma \leq (T - \Gamma) \in \mathbb{N}$ is the length of forecasting window for marginal models, while $\kappa \leq \gamma \in \mathbb{N}$ is the length of usage window for vine copula models. Their utility can be better understood looking at Figure 4.4.

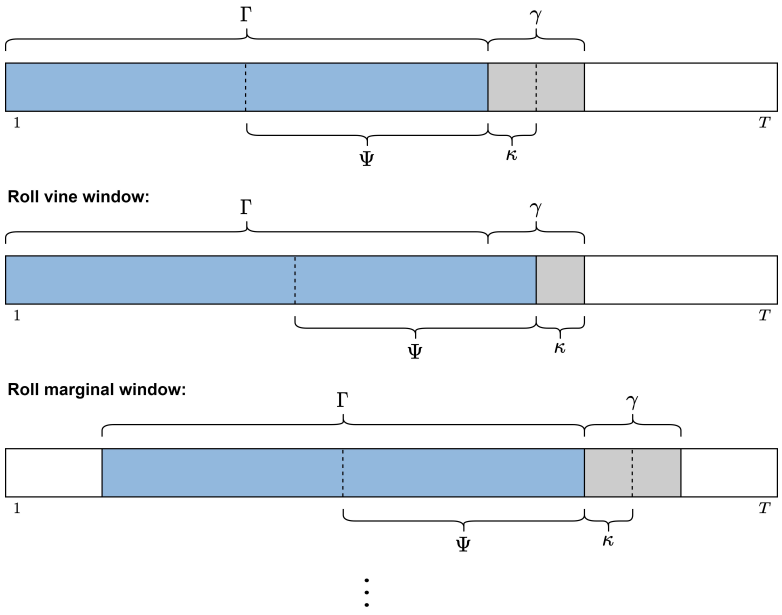

Figure 4.4. Example of a rolling window shifting over time.

Figure 4.5 instead reports pseudo code of the entire algorithm implemented in *portvine* package.

Algorithm 6: The rolling window unconditional risk measure estimation approach

Input : Portfolio Ω, parameters $\Gamma, \gamma, \Psi, \kappa, S$ and risk measure estimator $RM(\cdot|\alpha)$ at level α

Output: Risk measure estimates $\hat{m}_t$ for $t \in \{\Gamma+1, \ldots, T\}$

First estimate the marginal models in a rolling window fashion.

for $marginal\_window \leftarrow 1$ **to** $\lceil (T-\Gamma)/\gamma \rceil$ **do**

  for $j \leftarrow 1$ **to** d **do**

    Fit an appropriate marginal model i.e. here an ARMA-GARCH model on $r^{A_j}_{1+\gamma*(marginal\_window-1)}, \ldots, r^{A_j}_{\Gamma+\gamma*(marginal\_window-1)}$

    Estimate from the model the conditional means and standard deviations given below.

    $\hat{\mu}^{A_j}_{1+\Gamma-\Psi+\gamma*(marginal\_window-1)}, \ldots, \hat{\mu}^{A_j}_{min[T,\Gamma+\gamma*(marginal\_window)]}$

    $\hat{\sigma}^{A_j}_{1+\Gamma-\Psi+\gamma*(marginal\_window-1)}, \ldots, \hat{\sigma}^{A_j}_{min[T,\Gamma+\gamma*(marginal\_window)]}$

    Calculate the standardized residuals using the formula $z^{A_j}_t = \frac{r^{A_j}_t - \hat{\mu}^{A_j}_t}{\hat{\sigma}^{A_j}_t}$ for the window in order to get $z^{A_j}_{1+\Gamma-\Psi+\gamma*(marginal\_window-1)}, \ldots, z^{A_j}_{min[T,\Gamma+\gamma*(marginal\_window)]}$

    They should, under the assumption of a correct marginal model, be approximate i.i.d. with marginal distribution function F_j. Convert those values on the residual scale to the copula scale by applying F_j to each of the values resulting in $u^{A_j}_{1+\Gamma-\Psi+\gamma*(marginal\_window-1)}, \ldots, u^{A_j}_{min[T,\Gamma+\gamma*(marginal\_window)]}$

  end

end

Estimate the dependence structure and the risk measures by simulation in a rolling window fashion.

for $vine\_window \leftarrow 1$ **to** $\lceil (T-\Gamma)/\kappa \rceil$ **do**

  Use only $u^{A_j}_t, \hat{\mu}^{A_j}_t, \hat{\sigma}^{A_j}_t$ from the marginal window $\lceil \kappa * vine\_window/\gamma \rceil$.

  Fit a regular vine copula $C_{1,\ldots,d}$ using the copula data $u^{A_j}_{1+\Gamma-\Psi+\kappa*(vine\_window-1)}, \ldots, u^{A_j}_{\Gamma+\kappa*(vine\_window-1)}$, for $j \in \{1, \ldots, d\}$.

  for $k \leftarrow 1$ **to** κ **if** $vine\_window < \lceil (T-\Gamma)/\kappa \rceil \vee (T-\Gamma) \bmod \kappa \equiv 0$ **else to** $(T-\Gamma) \bmod \kappa$ **do**

    Simulate from $C_{1,\ldots,d}$ multivariate samples $\hat{u}_s = (\hat{u}^{A_1}_s, \ldots, \hat{u}^{A_d}_s)$, for $s \in 1, \ldots, S$

    Then transform the samples from copula $\rightarrow$ residual $\rightarrow$ return scale.

    $\hat{r}^{A_j,s}_{\Gamma+\kappa*(vine\_window-1)+k} = \hat{\mu}^{A_j}_{\Gamma+\kappa*(vine\_window-1)+k} + \hat{\sigma}^{A_i}_{\Gamma+\kappa*(vine\_window-1)+k} * F_j^{-1}(\hat{u}^{A_j}_s)$, for $j \in \{1, \ldots, d\}$ and $s \in \{1, \ldots, S\}$

    Get the weighted full portfolio return by a simple weighted sum.

    $\hat{r}^{\Omega,s}_{\Gamma+\kappa*(vine\_window-1)+k} = \sum_{j=1}^{d} w_j * \hat{r}^{A_j,s}_{\Gamma+\kappa*(vine\_window-1)+k}$, for $s \in \{1, \ldots, S\}$

    Having all these samples one can estimate various risk measures.

    $\hat{m}_{\Gamma+\kappa*(vine\_window-1)+k} = RM(\{\hat{r}^{\Omega,s}_{\Gamma+\kappa*(vine\_window-1)+k} | s \in \{1, \ldots, S\}\}|\alpha)$

  end

end

Figure 4.5. Rolling window unconditional risk measure estimation algorithm.

4.2 Conditional vine copula based risk measure estimation

Regarding prediction of conditional risk measures, the previously introduced model remains similar but some adjustments need to be made. In particular, the framework developed by Sommer [2022] for this case, is focused on a subclass of R-vine copulas, called Drawable vine (D-vine). These are nothing but R-vine copulas with the additional constraint that each tree is an ordered connected path with no loops. In formulas, $|\{e \in E_j | n \in e\}| \leq 2 \ \forall n \in N_j, j = 1, ..., d-1$. An example can be seen in Figure 4.6, whose copula density is:

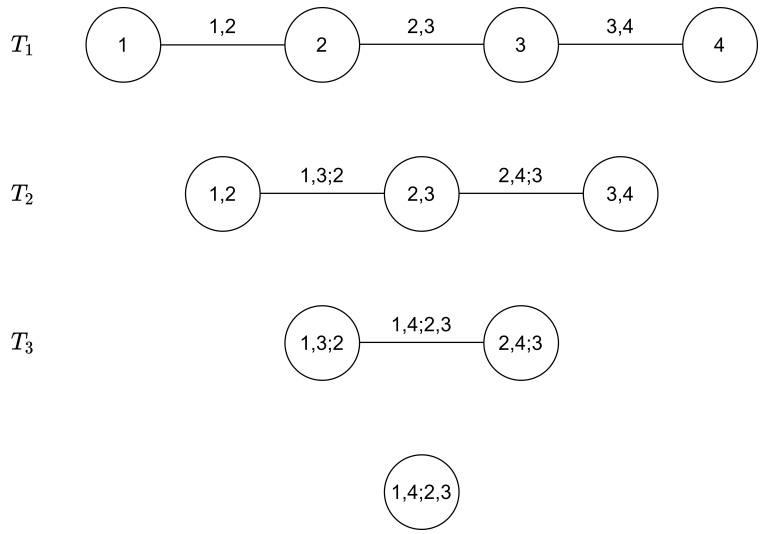

Figure 4.6. Example of D-vine tree structure.

$$
\begin{aligned}
c_{1234}(u_1, u_2, u_3, u_4) &\overset{eq.2.2}{=} \frac{f_{1234}(x_1, x_2, x_3, x_4)}{f_1(x_1) f_2(x_2) f_3(x_3) f_4(x_4)} \\
&\overset{eq.2.4}{=} c_{12}(u_1, u_2) \times c_{23}(u_2, u_3) \times c_{34}(u_3, u_4) \\
&\quad \times c_{13|2}(C_{1|2}(u_1|u_2), C_{3|2}(u_3|u_2)) \\
&\quad \times c_{24|3}(C_{2|3}(u_2|u_3), C_{4|3}(u_4|u_3)) \\
&\quad \times c_{14|23}(C_{1|23}(u_1|u_2, u_3), C_{4|23}(u_4|u_2, u_3))
\end{aligned}
\tag{4.1}
$$

where argumets inside brackets of last equality are written according to the notation $u_i := F(x_i) \ \forall i$.

The reasons why it is necessary to introduce this subclass of vines are mainly two:

- The need of easily sampling observations conditional on having observed other values. Conditioning values from now on are assumed to be market indexes, named I, and can be at maximum two. In addition, they are supposed to be the rightmost leaves nodes, i.e. the nodes at the end of the first path in the tree structure.

- By fixing the order of the first tree, the structure and order of all the other trees is automatically derived for D-vines, as outlined by Dißmann et al. [2013].

In particular, the algorithm depicted in Figure 3.1 can be rephrased more conveniently by leveraging the well-established Rosenblatt transform and defining the three matrices:

$$V = \begin{bmatrix} u_1 & u_2 & u_3 & u_4 & ... \\ - & C_{2|1}(u_2|u_1) & C_{3|2}(u_3|u_2) & C_{4|3}(u_4|u_3) & ... \\ - & - & C_{3|21}(u_3|u_2,u_1) & C_{4|32}(u_4|u_3,u_2) & ... \\ - & - & - & C_{4|123}(u_4|u_1,u_2,u_3) & \\ - & - & - & - & ... \end{bmatrix}$$

$$V^2 = \begin{bmatrix} u_1 & u_2 & u_3 & u_4 & ... \\ - & C_{1|2}(u_1|u_2) & C_{2|3}(u_2|u_3) & C_{3|4}(u_3|u_4) & ... \\ - & - & C_{1|23}(u_1|u_2,u_3) & C_{2|34}(u_2|u_3,u_4) & ... \\ - & - & - & C_{1|234}(u_1|u_2,u_3,u_4) & \\ - & - & - & - & ... \end{bmatrix}$$

$$\Theta = \begin{bmatrix} - & \theta_{12} & \theta_{23} & \theta_{34} & ... \\ - & - & \theta_{31|2} & \theta_{42|3} & ... \\ - & - & - & \theta_{41|32} & ... \\ - & - & - & - & ... \end{bmatrix}$$

These are the equivalent versions of $V^{direct}, V^{indirect}$ and T of Figure 3.1, but the final algorithm is greatly simplified, as can be seen in Figure 4.7. Again the pseudo code is reported on copula scale, since the framework depicted by Sommer [2022] involves simulating data from D-vine copulas to subsequently obtain risk measures estimates. In the case of conditional sampling the algorithm is the same with the exception that u_1 and u_2 are fixed, as they represent market indexes values on the copula scale. In addition to this, the formulation of the final algorithm involves a redefinition of index ordering, given that market indexes should be the rightmost leaves nodes in the first path.

So far it has been observed how simulation of D-vines is simplified compared to R-vines. The further element of simplification is the determination of D-vines structures. Consider the d assets previously introduced: given the path structure of D-vines trees, what we want to determine is a permutation $j_1, ..., j_d$ of indexes 1,...,d that maximize the overall likelihood. Sommer [2022] proposes a methodology where the order:

$$A_{j_d} - A_{j_{d-1}} - ... - A_{j_1} - I \tag{4.2}$$

is fixed sucht that A_{j_1} is the most dependent asset on the market index I, A_{j_2} is the most dependent on the asset A_{j_1}, after accounting for the effect of market index I, and so on. The idea is to assign weights to the edges corresponding to the bivariate copula densities using 2 measures:

- Pearson correlation coefficient: used to find most correlated assets in the first tree. Given $A_{j_{(\cdot)}}$ and I, it is defined as:

$$\rho := \rho(A_{j_{(\cdot)}}, I) := Corr(A_{j_{(\cdot)}}, I) = \frac{Cov(A_{j_{(\cdot)}}, I)}{\sqrt{Var(A_{j_{(\cdot)}})}\sqrt{Var(I)}} \tag{4.3}$$

Algorithm 1: Sampling from D-vine copulas (Taken from Czado (2019))

Input : Parameter matrix Θ for the d-dimensional D-vine $C_{1,\dots,d}(\cdot)$

Output: A d-dimensional sample $u_1, \dots, u_d$ from $C_{1,\dots,d}(\cdot)$

Sample $w_j \overset{i.i.d.}{\sim} \mathcal{U}(0,1)$, $j \in \{1, \dots, d\}$

$v_{1,1} = w_1$, $v_{1,1}^2 = w_1$

for $j \leftarrow 2$ **to** d **do**

  $v_{j,j} = w_j$

  for $k \leftarrow j-1$ **to** 1 **do**

    $v_{k,j} = h^{-1}_{j|j-k;(j-k+1):(j-1)}(v_{k+1,j}|v^2_{k,j-1}, \theta_{j|j-k;(j-k+1):(j-1)})$

    if $j < d$ **then**

      $v^2_{k+1,j} = h_{j-k|j;(j-k+1):(j-1)}(v^2_{k,j-1}|v_{k,j}, \theta_{j-k|j;(j-k+1):(j-1)})$

  end

  $v^2_{1,j} = v_{1,j}$

end

$u_i = v_{1,j}$ for $j \in \{1, \dots, d\}$

Figure 4.7. D-vine copula simulation algorithm.

where Var and Cov are the variance and covariance respectively. A_{j_1} is the assets which maximize equation 4.3 and the same reason is applied for the other nodes in the first tree.

- Partial correlation coefficient: used from the second tree onwards to measures the dependence between two random variables, with the effect of a set of controlling random variables removed. Given d assets $A_1, ..., A_d$ and the reduced index set $I^d_{-(i,j)} = \{1, ..., d\} \setminus \{i, j\}$ with $i \neq j$, it is defined as:

$$\rho_{i,j|I^d_{-(i,j)}} = sgn(b_{i,j|I^d_{-(i,j)}}) \times \sqrt{b_{i,j|I^d_{-(i,j)}} \times b_{j,i|I^d_{-(i,j)}}} \tag{4.4}$$

where $b_{i,j|I^d_{-(i,j)}}$ are the coefficients that minimize $\mathbb{E}[(A_{j_i} - \sum_{j=2, j\neq i}^{d} a_{i,j|I^d_{-(i,j)}} A_{j_i})^2]$ with respect to $a_{i,j|I^d_{-(i,j)}}$. The notation $i, j|I^d_{-(i,j)}$ is adopted to account for the conditioning sets of D-vines.

The final algorithm is explained in Figure 4.8, while Figure 4.9 shows an example of how the algorithm is applied in a four dimensional case. It is crucial to point out that the notation of equation 4.3 can be misleading, because in reality Pearson's coefficient is not calculated among real scale data, but between copula scale data that are transformed to normalized scale, defined as:

$$z_{i,j} = \phi^{-1}(u_{i,j}) \ \forall i, j \tag{4.5}$$

where $u_{i,j}$ are data in copula scale and ϕ is the density function of a N(0,1). This means that Pearson correlation is nothing but Spearman ρ correlation with an additional linear

Algorithm 2: Determine D-vine ordering including a market index as the rightmost leaf node

Input : $A_1, \ldots, A_d$ and a market index I with corresponding i.i.d. copula data $u_1, \ldots, u_d, u^I$ as well as a cutoff depth $c_{\text{depth}} \in \{1, \ldots, d\}$ defaulting to d

Output: $j_1, \ldots, j_d$ permutation for the indices $1, \ldots, d$ specifying an ordering as in (4.2)

Transform the copula data to the normalized scale using Equation (4.5) . The data on the normalized scale is denoted by $z_1, \ldots, z_d, z^I$ with the convention $z_{j_0} = z^I$ for simplicity in the algorithm below.

$j_1 \leftarrow \delta \in \{1, \ldots, d\}$ s.t. $|\rho_{\delta,I}| \geq |\rho_{\delta',I}|, \forall \delta' \in \{1, \ldots, d\}$

for $l \leftarrow 2$ **to** d **do**

$\quad j_l = \text{argmax}_{\delta \in \{1,\ldots,d\} \setminus \{j_1,\ldots,j_{l-1}\}} \sum_{k=max(l-c_{\text{depth}},0)}^{l-1} |\rho_{\delta,j_k;j_{k+1}:j_{l-1}}|$

end

Here $\rho_{a,b;D}$ denotes the Pearson (if D is the empty set) or partial correlation coefficient between z_a and z_b with conditioning set D and it is estimated always using the data on the normalized scale.

Figure 4.8. D-vine structure selection algorithm based on one conditional market index.

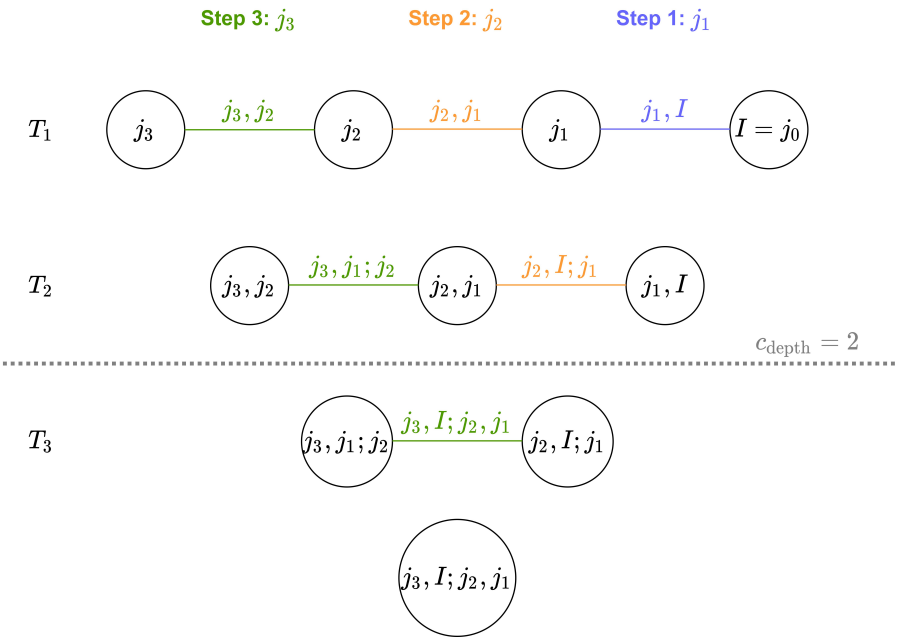

Figure 4.9. Example of application of algorithm in Figure 4.8.

trasnformation between ranks given by ϕ^{-1}. Again, inverse normal scale is adopted by Sommer [2022] for better rescaling of tails. Finally, the introduction of the cutoff depth parameter c_{depth} implies that Pearson and Partial correlation coefficients can be calculated up to the specified depth of D-vine edges, so as to reduce the computational cost in case of high dimensions.

The study with two market indexes is analogous, with the only need to shift the indexes by one, obtaining the scheme:

$$A_{j_d} - A_{j_{d-1}} - ... - A_{j_1} - I_{\tilde{j}_1} - I_{\tilde{j}_2} \tag{4.6}$$

Ordering and sampling procedure are the same as before, since the only difference is that the value of another index is further fixed.

We are now able to extend the procedure presented in the previous section but for the conditional case:

1. First ARMA-GARCH processes are fitted starting from the log return scale, as done in Figure 4.5. At the same time, forecasting of conditional mean $\hat{\mu}^I_{\Gamma+1}$ and volatility $\hat{\sigma}^I_{\Gamma+1}$ for the next time instant can be done, obtaining the forecasted conditioning value for market indexes on the log return scale by performing $\hat{\mu}^I_{\Gamma+1} + \hat{\sigma}^I_{\Gamma+1} \times F_I^{-1}(\alpha^I)$ (inverse PIT).

2. Second, standardized residuals are calculated, transformed to copula scale, and fed into the algorithms of Figure 4.7, 4.8 to determine the order and simulate from D-vine copulas conditionally on observed values of the market indexes.

3. Finally, simulations are transformed to the log return scale to compute conditional sample portfolio values and derive conditional risk measure estimates.

Regardless of whether conditioning occurs with one or two market indexes, the key novelty of this approach lies in the first point of the preceding numbered list. Specifically, $\alpha^I \in (0,1)$ represents the confidence level of the estimated quantile obtained from the marginal market index distribution. This confidence level can be used as conditioning value of the final risk measure estimate **on the copula scale**. Since all the marginals on the copula scale are uniform, the quantile corresponds to the confidence level α^I itself. Consequently, the retransformation to the return scale is also performed on the conditioning value α^I as well as for all assets. This approach is referred to as the quantile strategy for estimating conditional risk measures.

In contrast, Sommer [2022] introduces an alternative approach known as residual strategy for estimating conditional risk measures. Instead of conditioning on a quantile level, this method conditions **on the copula scale residual** of index I from the previous time unit, denoted as u^I_{t-1}. The resulting estimated risk measure aims to mimic the behavior of conditioning on the predicted market index log return series and can be compared with the first one. It's important to note that conditioning series based on fitted residuals from the preceding time unit may tend to amplify sudden high volatility situations. For instance, if the univariate marginal time series model fails to anticipate a sudden price drop of the conditioning asset at time $t-1$, this could result in a very small copula scale residual u^I_{t-1}. Consequently, conditioning at time t on this sharp drop from the prior time unit may often exaggerate the decline of the estimated risk measures. Therefore, it is crucial to assess in practice the extent to which this volatility exaggeration might manifest. Figure 4.10 reports the complete algorithm implemented in *portvine* package for conditional risk measures estimate given a single market index as conditioning variable. The case with two market indexes is nothing but an extension of the presented algorithm, with the ordering given by equation 4.6.

Algorithm 7: The rolling window conditional risk measure estimation approach using D-vines

Input : Portfolio Ω, market index I, parameters $\Gamma, \gamma, \Psi, \kappa, S, \alpha^I$ and risk measure estimator $RM(\cdot|\alpha)$ at level α

Output: Conditional risk measure estimates $\hat{m}_{t|I}$ for $t \in \{\Gamma+1, \ldots, T\}$

First estimate the marginal models in a rolling window fashion.

for $marginal\_window \leftarrow 1$ **to** $\lceil (T-\Gamma)/\gamma \rceil$ **do**

  for $j \leftarrow 1$ **to** d **do**

    Fit an appropriate marginal model i.e. here an ARMA-GARCH model on $r^{A_j}_{1+\gamma*(marginal\_window-1)}, \ldots, r^{A_j}_{\Gamma+\gamma*(marginal\_window-1)}$

    Estimate from the model the conditional means and standard deviations given below.

    $\hat{\mu}^{A_j}_{1+\Gamma-\Psi+\gamma*(marginal\_window-1)}, \ldots, \hat{\mu}^{A_j}_{min[T,\Gamma+\gamma*(marginal\_window)]}$

    $\hat{\sigma}^{A_j}_{1+\Gamma-\Psi+\gamma*(marginal\_window-1)}, \ldots, \hat{\sigma}^{A_j}_{min[T,\Gamma+\gamma*(marginal\_window)]}$

    Calculate the standardized residuals using the formula $z^{A_j}_t = \frac{r^{A_j}_t - \hat{\mu}^{A_j}_t}{\hat{\sigma}^{A_j}_t}$ for the window in order to get $z^{A_j}_{1+\Gamma-\Psi+\gamma*(marginal\_window-1)}, \ldots, z^{A_j}_{min[T,\Gamma+\gamma*(marginal\_window)]}$

    As outlined in Algorithm 6 transform from residual to the copula scale by applying F_j to each of the values resulting in $u^{A_j}_{1+\Gamma-\Psi+\gamma*(marginal\_window-1)}, \ldots, u^{A_j}_{min[T,\Gamma+\gamma*(marginal\_window)]}$

  end

  Apply the same modeling procedure to the index returns r^I_t in order to estimate the u^I_t, $\hat{\mu}^I$ and $\hat{\sigma}^I$

end

Estimate the dependence structure and the risk measures by simulation in a rolling window fashion.

for $vine\_window \leftarrow 1$ **to** $\lceil (T-\Gamma)/\kappa \rceil$ **do**

  Use only $u_t, \hat{\mu}_t, \hat{\sigma}_t$ from the marginal window $\lceil \kappa * vine\_window/\gamma \rceil$.

  The copula data for the vine window is given by

  $u^A_{1+\Gamma-\Psi+\kappa*(vine\_window-1)}, \ldots, u^A_{\Gamma+\kappa*(vine\_window-1)}$, for $A \in \{A_1, \ldots, A_d, I\}$.

  Determine the ordering $A_{j_d} - A_{j_{d-1}} - \cdots - A_{j_1} - I$ i.e. the permutation of indices $j_1, \ldots, j_d$ as outlined in Algorithm 2 in order to fit the D-vine.

  Fit the corresponding D-vine copula $C_{j_1,\ldots,j_d,I}$ using the copula data and the ordering.

  for $k \leftarrow 1$ **to** κ **if** $vine\_window < \lceil (T-\Gamma)/\kappa \rceil \vee (T-\Gamma) \bmod \kappa \equiv 0$ **else to** $(T-\Gamma) \bmod \kappa$ **do**

    Simulate from $C_{j_1,\ldots,j_d|I}(\cdot|\alpha^I)$ multivariate samples $\hat{u}_s = (\hat{u}^{A_1}_s, \ldots, \hat{u}^{A_d}_s)$, for $s \in 1, \ldots, S$

    Then transform the samples and the conditioning quantile α^I from copula $\rightarrow$ residual $\rightarrow$ return scale.

    $\hat{r}^{A_j,s}_{\Gamma+\kappa*(vine\_window-1)+k} = \hat{\mu}^{A_j}_{\Gamma+\kappa*(vine\_window-1)+k} + \hat{\sigma}^{A_j}_{\Gamma+\kappa*(vine\_window-1)+k} * F^{-1}_j(\hat{u}^{A_j}_s)$, for $j \in \{1, \ldots, d\}$ and $s \in \{1, \ldots, S\}$

    $\hat{r}^{I,\alpha^I}_{\Gamma+\kappa*(vine\_window-1)+k} = \hat{\mu}^I_{\Gamma+\kappa*(vine\_window-1)+k} + \hat{\sigma}^I_{\Gamma+\kappa*(vine\_window-1)+k} * F^{-1}_I(\alpha^I)$

    Get the weighted full portfolio return by a simple weighted sum.

    $\hat{r}^{\Omega,s}_{\Gamma+\kappa*(vine\_window-1)+k} = \sum_{j=1}^d w_j * \hat{r}^{A_j,s}_{\Gamma+\kappa*(vine\_window-1)+k}$, for $s \in \{1, \ldots, S\}$

    Having all these samples one can estimate various conditional risk measures.

    $\hat{m}_{\Gamma+\kappa*(vine\_window-1)+k|I=\hat{r}^{I,\alpha^I}_{\Gamma+\kappa*(vine\_window-1)+k}} = RM(\{\hat{r}^{\Omega,s}_{\Gamma+\kappa*(vine\_window-1)+k} | s \in \{1, \ldots, S\}\}|\alpha)$

  end

end

Figure 4.10. Rolling window conditional risk measure estimation algorithm.

4.3 Probability equivalent levels definitions

The introduction of conditional risk measures, in particular the ones associated with the quantilie strategy, allows to introduce a more generic problem studied very recently by Ortega-Jiménez et al. [2024]. To be more clear, we can fix the concepts for the two risk measures treated up to now, i.e. VaR and ES. Algorithm of Figure 4.10 returns what can be defined as conditional-value-at-risk (CoVaR) and conditional-expected-shortfall (CoES). Formally, they are defined as:

- $CoVaR_{v,u}[\Omega|I] = VaR_v[\Omega|I = VaR_u[I]] = F^{-1}_{\Omega|I=VaR_u(X)}(v)$, where u $\in$ (0,1) is the confidence level for the market index I while v $\in$ (0,1) is the confidence level of portfolio Ω. This measure has been already defined by Ortega-Jiménez et al. [2024], but differently from them, here Ω is a multivariate random vector and not a single asset.
- $CoES_{v,u}[\Omega|I] = ES_v[\Omega|I = VaR_u[I]] = \mathbb{E}[\Omega|\Omega \leq VaR_v(\Omega), I = VaR_u[I]]$, i.e. the expected shorfall given that market index is at level u. It is important to note that we are always talking about a single market index, but from what has just been presented in the previous sections, all the reasonings can be extended to two market indexes.

The importance of these two risk measures is quite intuitive. From now on, all the arguments will be presented for VaR and CoVaR, but can be applied in the same way for ES and CoES. As pointed out by Acharya [2009], "the goal of prudential regulation should be to ensure the financial stability of the system as a whole, i.e., of an institution not only individually , but also as a part of the overall financial system". With respect to this, conditional risk measures could be very useful for making adjustments in capital requirements based on the performance of certain market indexes. In this sense, it can be said that conditional risk measures seem to better capture the dimension of systemic risk. In addition, given certain stressed market situation, it can be useful to identify if VaR is more or less conservative than CoVaR and so a better or worse risk measure in terms of capital requirements. The example made by Ortega-Jiménez et al. [2024] and translated with respect to our notation enables a deeper understanding of what just said. Suppose that, initially, the risk capital calculation is based on VaR at level v for a portfolio Ω. If, given a risk I and u $\in$ (0,1), $CoVaR_{v,u}[\Omega|I] > VaR_v[\Omega]$, then replacing VaR by CoVaR does not make sense, since CoVaR is more optimistic. If however $CoVaR_{v,u}[\Omega|I] < VaR_v[\Omega]$, then it is possible that VaR is underestimating the spillover effect and it can be more prudent to replace it with CoVaR. What has just been reported may seem the exact opposite of what was said in Ortega-Jiménez et al. [2024], however, it is important to remember that the definition adopted in this work for VaR is portfolio's return in the 5% of worst-case scenarios, if v=0.05, and not portfolio's loss in the 5% of worst-case scenarios. When returns are low typically losses are high. Thus, increasing conditioning level of market indexes implies better financial conditions and so higher returns, which may result in risk measures increasing. In this terms, it could be stated that if the estimate of returns increases in the 5% of unfavorable cases, i.e. risk measures increase, then risk estimates may be more optimistic.

As might be inferred from above, an interesting inquiry concerns the circumstances under which the two measures are equivalent. This leads to the definition of probability equivalent level: for a portfolio Ω, a market index I and a confidence level $v \in (0,1)$, the value u_v is defined as a probability equivalent level of CoVaR-VaR at the risk level v ($PELCoV_v$) for Ω if $CoVaR_{v,u_v}[\Omega|I] := VaR_v[\Omega|I = VaR_{u_v}[I]] \equiv VaR_v[\Omega]$. In plain english it is the conditional quantile u such that CoVaR and VaR, evealuated both at confidence level v, are equivalent. The same can be done for CoES and ES bringing to the definition of PELCoES. Ortega-Jiménez et al. [2024] have provided the hypothesis under which existence and uniqueness of $PELCoV_v$ is guaranteed, but limited to the bivariate case. The goal of the following analysis is to prove empirically the existence of these levels also in the multivariate case and extend the discussion also for PELCoES.

4.4 Empirical Application

The last section of this work represents the core analysis of this thesis. The adopted data set is available in *rugarch* library of R. It contains Dow Jones Industrial Average (DJIA) 30 Constituents closing value log returns for working days from 1987-03-16 to 2009-02-03, taken from Yahoo Finance. As reported in the guide, AIG has been replaced by KFT (Kraft Foods) on September 22, 2008. This is not reflected in the data set as that would bring the starting date of the data to 2001. For our purposes this lack is not essential but for completeness it is fair to specify it. In particular, the focus period for us is from 2007-01-01 to 2009-02-03, corresponding to the United States Bear Market and the Global Financial Crisis periods, mainly due to the housing bubble created by subprime mortgages, excessive risk-taking by global financial institutions and continuous buildup of toxic assets within banks. Figure 4.11 reports the performance of the index and its constituents in the reference period.

Looking at plots it is possible to say that time series are weakly stationary of order 2, since they have constant mean and finite variance. This implies that the first two moments are preserved. Loess smoothing line further confirms what just said. Unlike traditional regression techniques that assume a specific functional form for the relationship between variables, loess regression adapts to the local structure of the data by fitting multiple linear regressions over small subsets of the data. This adaptability allows the loess line to capture complex patterns and nonlinear relationships that may exist in the data, making it particularly useful in detecting trends or patterns that may not be apparent from the raw data alone. Having a fairly constant loess smoothing line reassures that time series mean is preserved over time.

Before going deep in PELCOVs analysis, a demonstration of conditional risk measures effectiveness is done. For training set, 1000 observations before 2008-04-18 are taken, both for marginals and vine copulas. The test set, used for the rolling window conditional/unconditional risk measure estimation approach, is the remaining part up to 2009-02-03 and consists of 200 observations. The choice to use these numbers of observations and April 2008 as threshold was driven by the need to have enough data to test models and not to increase computational costs excessively. However, this date makes sense from a macroeconomic point of view. In fact, by that time DJIA had already reached its historical

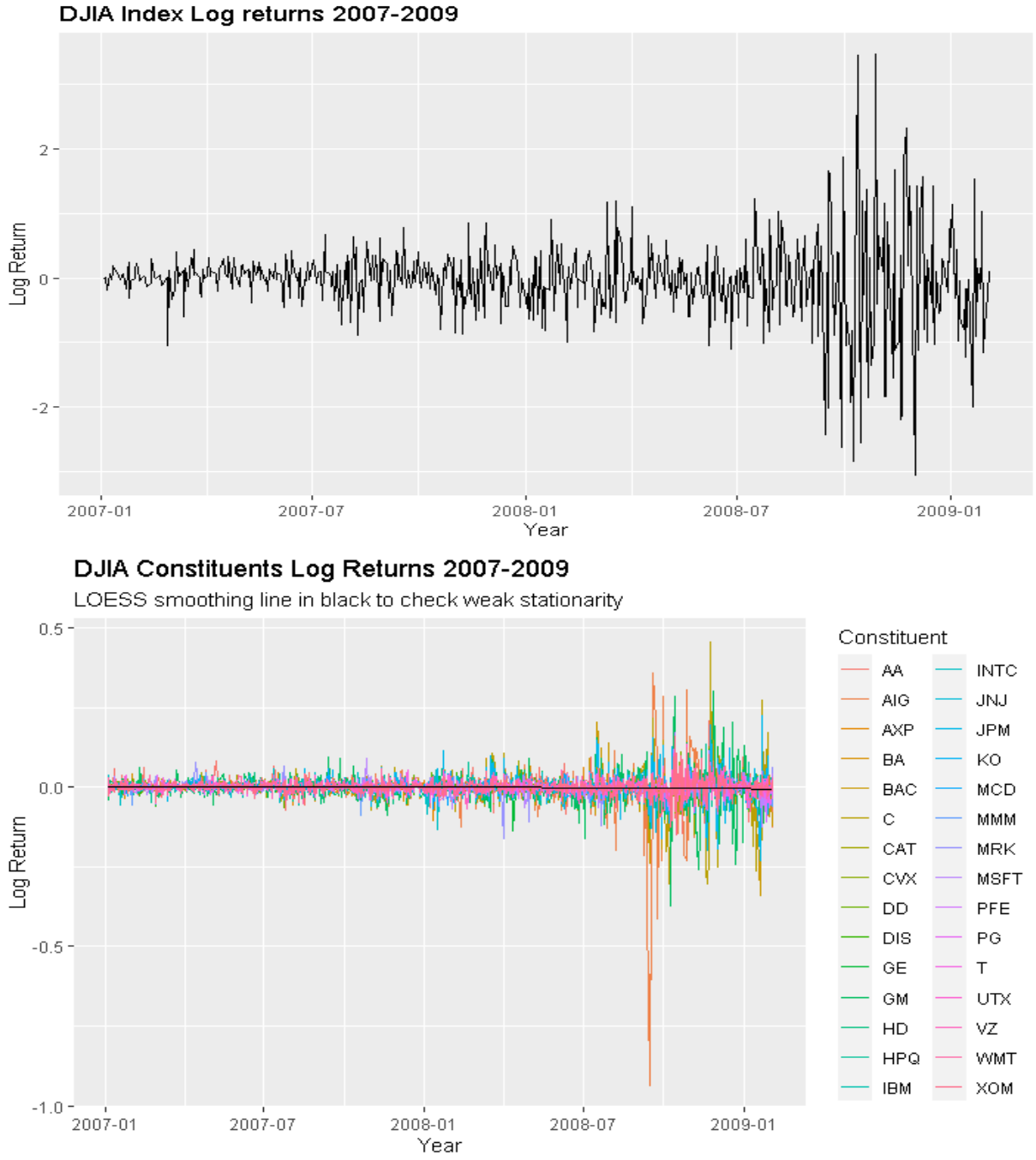

Figure 4.11. Dow Jones Industrial Average (DJIA) log returns in the reference period.

peaks, twice overshooting the 14000 mark, and also alternated moments of decline, such as the 8.3% drop recorded after July 2007 peak. Moreover, the first signs of economic weakness were already showing up, such as the near bankrupcy of Bear Sterns on March 2008, that was addressed thanks to the guarantees offered by the Federal Reserve on bad loans and their acquisition by JPMorgan Chase. Being able to capture systemic risk and possible downside spillovers effects was critical at that time but essential to be able to meet liquidity requirements that failed a few months later, leading to a global crisis.
Other parameters and quantities fixed are:

- $\gamma = 50$ so that 4 rolling window are computed at the end.
- $\kappa = 50$ so that at the beginning of each rolling window vine copulas are fitted again.
- $\Omega_1 = \{$Alcoa Corporation (AA), American Express Company (AXP), The Boeing Company (BA), Bank of America Corporation (BAC))$\}$ each weighing $\frac{1}{4}$.
- $I_1 = \{$Citigroup Inc. (C), Caterpillar Inc. (CAT)$\}$.
- $S = 500$.
- Confidence level for risk measures estimation equal to 5%, while confidence level of the estimated quantile obtained from the marginal market index distribution α^I is assumed to be 0.1 on copula scale.
- ARMA(1,1)-GARCH(1,1) models are chosen for each univariate time serie.

Ljung-box test p-values for serial autocorrelation at different times are reported below and show that marginals do not present any type of autocorrelation, since null hypothesis is never rejected:

- First rolling window p-values for standardized residuals:

	$t=1$	$t=5$	$t=10$	$t=15$	$t=20$
AA	0.6380067	0.5463938	0.6029006	0.6344385	0.6718554
AXP	0.8014358	0.9053650	0.1593253	0.3227831	0.5922897
BA	0.4676530	0.9880981	0.9515401	0.7557235	0.9220219
BAC	0.5154959	0.9828692	0.4915308	0.7299531	0.8291133

- Second rolling window p-values for standardized residuals:

	$t=1$	$t=5$	$t=10$	$t=15$	$t=20$
AA	0.7574107	0.4250166	0.5197523	0.5792903	0.5595016
AXP	0.9649823	0.7189497	0.1273333	0.3255091	0.6412178
BA	0.3584866	0.9509725	0.7782875	0.6425933	0.8159217
BAC	0.5889626	0.9750401	0.2606740	0.5787592	0.7132500

- Third rolling window p-values for standardized residuals:

	$t=1$	$t=5$	$t=10$	$t=15$	$t=20$
AA	0.7303668	0.4845496	0.5032916	0.3530659	0.5161353
AXP	0.9758730	0.6804913	0.2132273	0.3709773	0.6907675
BA	0.3565942	0.9610094	0.7203465	0.6127089	0.8014826
BAC	0.6870309	0.9555553	0.4242265	0.6489335	0.8104965

- Fourth rolling window p-values for standardized residuals:

	$t=1$	$t=5$	$t=10$	$t=15$	$t=20$
AA	0.4785026	0.8102419	0.6801353	0.6230811	0.5774870
AXP	0.7419728	0.7221892	0.5306022	0.7773300	0.9391385
BA	0.4223057	0.8533465	0.9064178	0.7290525	0.7063390
BAC	0.7907871	0.9558479	0.1554015	0.3116655	0.5337789

Same test is applied for standardized squared residuals, since in volatility term of equation 1.14 w_t^2 follows an ARMA model and not σ_t, obtaining the following results:

- First rolling window p-values for squared standardized residuals:

	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$
AA	0.3720645	0.3588128	0.5520004	0.7587870	0.8277244
AXP	0.1763380	0.5868463	0.3012886	0.4141761	0.6076664
BA	0.3100602	0.7714612	0.9323075	0.9715605	0.9816345
BAC	0.9376997	0.9005088	0.6384118	0.5807796	0.7340830

- Second rolling window p-values for squared standardized residuals:

	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$
AA	0.2901060	0.4123180	0.6495486	0.8579638	0.9231744
AXP	0.1825294	0.6219297	0.3051888	0.4206254	0.6002521
BA	0.2898876	0.8078642	0.9686650	0.9793846	0.9912506
BAC	0.8328646	0.8739790	0.7051593	0.6253267	0.7566121

- Third rolling window p-values for squared standardized residuals:

	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$
AA	0.2781621	0.3542505	0.4614083	0.7665455	0.9266283
AXP	0.1737278	0.6032993	0.2955229	0.3834031	0.5551994
BA	0.3166856	0.7722314	0.8726943	0.9458312	0.9755234
BAC	0.8350609	0.9535959	0.9202004	0.8726341	0.9193350

- Fourth rolling window p-values for squared standardized residuals:

	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$
AA	0.3840283	0.5761943	0.2772587	0.5221912	0.7443184
AXP	0.1481619	0.6964143	0.0617093	0.0970664	0.1774252
BA	0.5043815	0.9213580	0.7772781	0.7562458	0.6348034
BAC	0.8154017	0.7972667	0.8175921	0.8799484	0.8821706

Null hypothesis is again rejected zero times, demonstrating the validity of adopted marginal models. These models come into play only in the first part of algorithms shown in Figures 4.5, 4.10. The remaining part instead contains dependencies modeling and risk measures

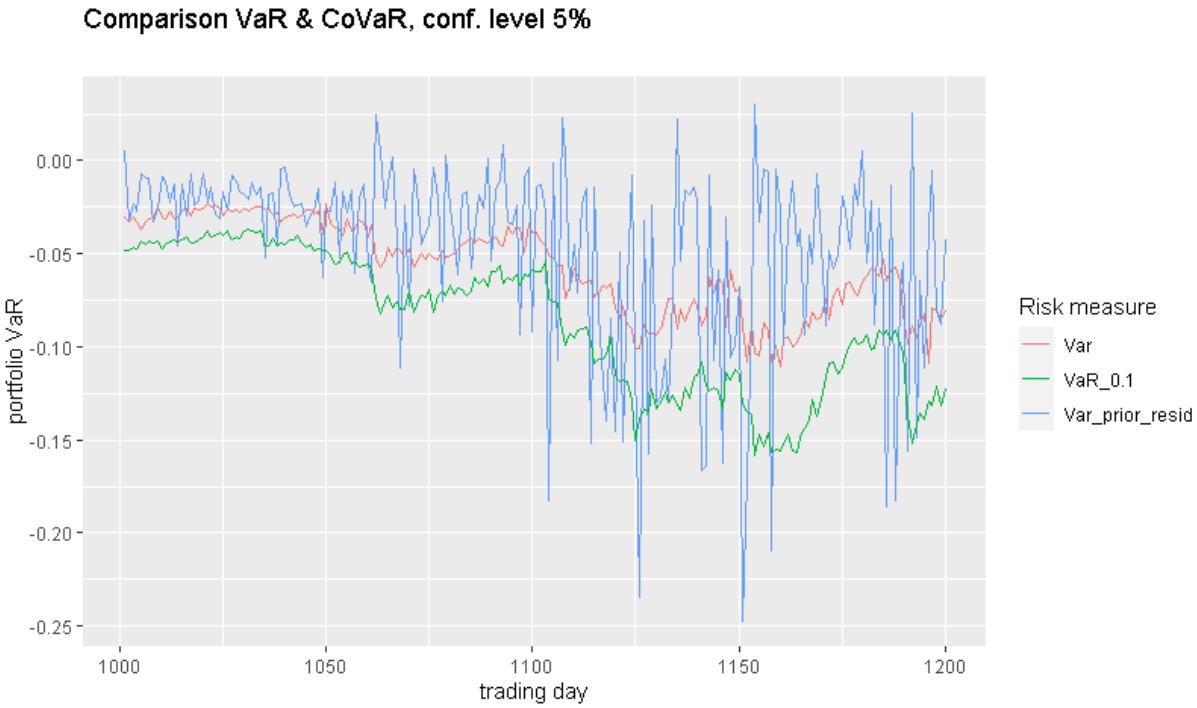

Figure 4.12. Unconditional and conditional rolling window estimation approaches for VaR.

forecast, both performed with a rolling window approach. Figures 4.12, 4.13 display all risk measures computed on the test set. Quantile strategy quantities are indicated with the suffix "_0.1", while residual ones are shortened as "_prior_resid".
As mentioned above, it is clear how conditional series forecast based on residuals of the time unit before, i.e. residual strategy, exaggerate high volatility situations. Having to assess which of the estimated risk measures are the best, the Kupiec, Christoffersen and McNeil & Frey tests are adopted. Before seeing numerical results, it may be useful to have graphical representations of how exceedances behave with respect to the different methods adopted. Figures 4.14, 4.15 show this behavior, hinting that residual strategy do not achieve formidable results, since it is quite obvious that forecasted risk measures have a number of exceedances well above the theoretical 5% of cases that they should actually represent.

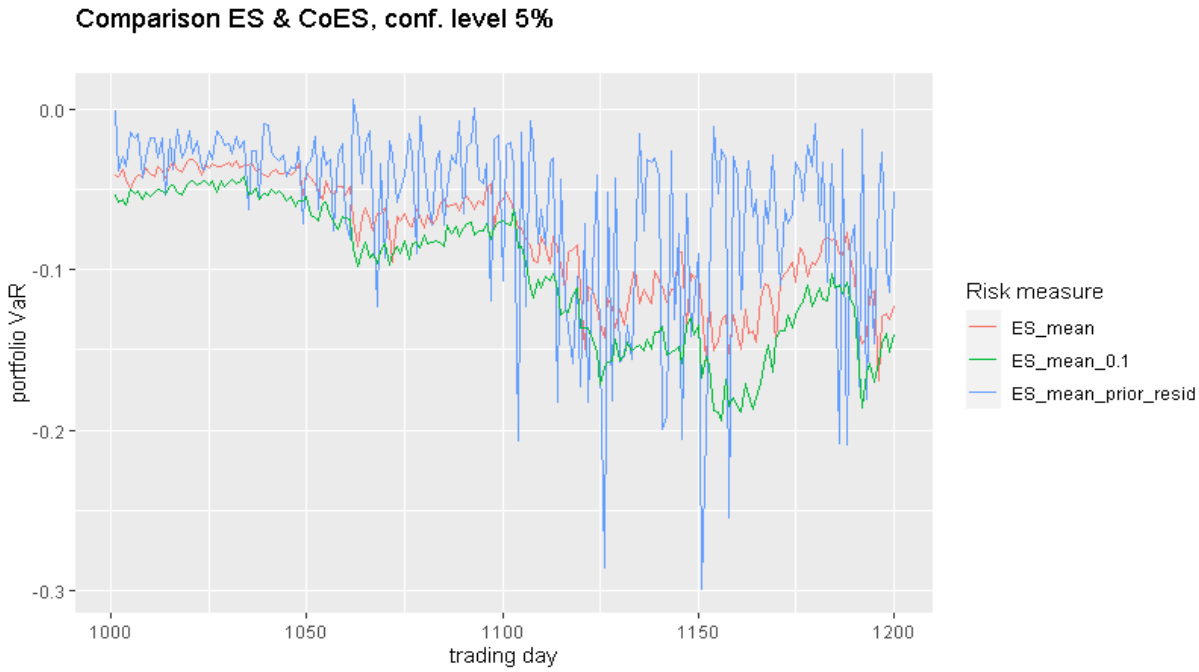

Figure 4.13. Unconditional and conditional rolling window estimation approaches for ES.

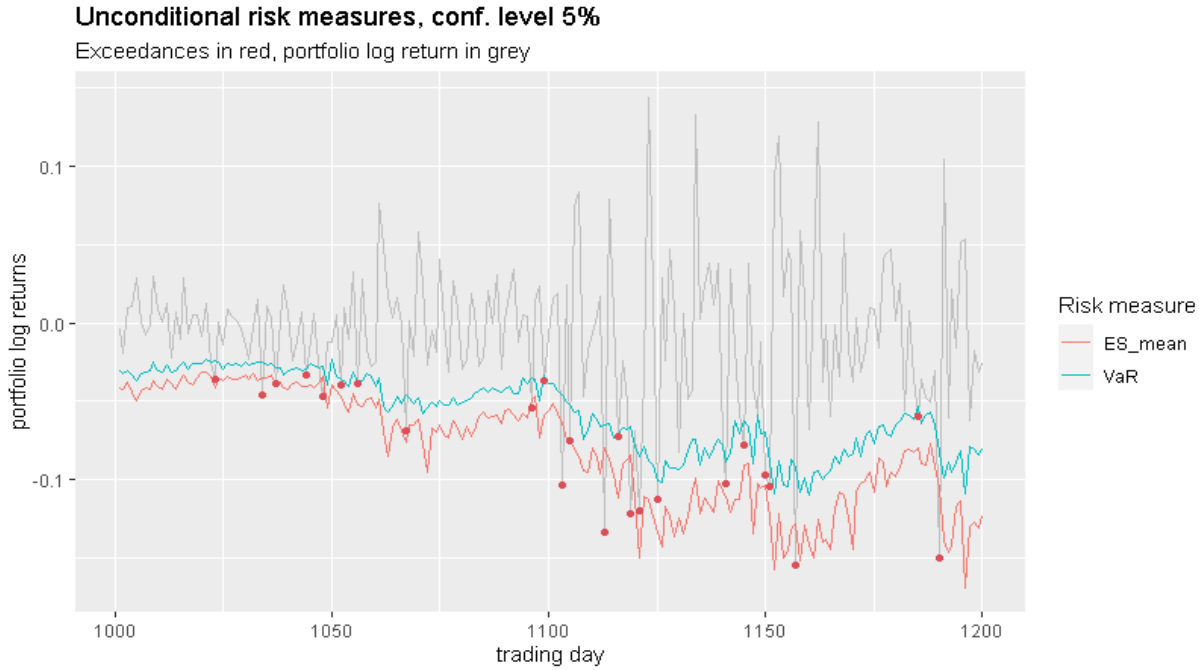

Figure 4.14. Unconditional strategy portfolio exceedances.

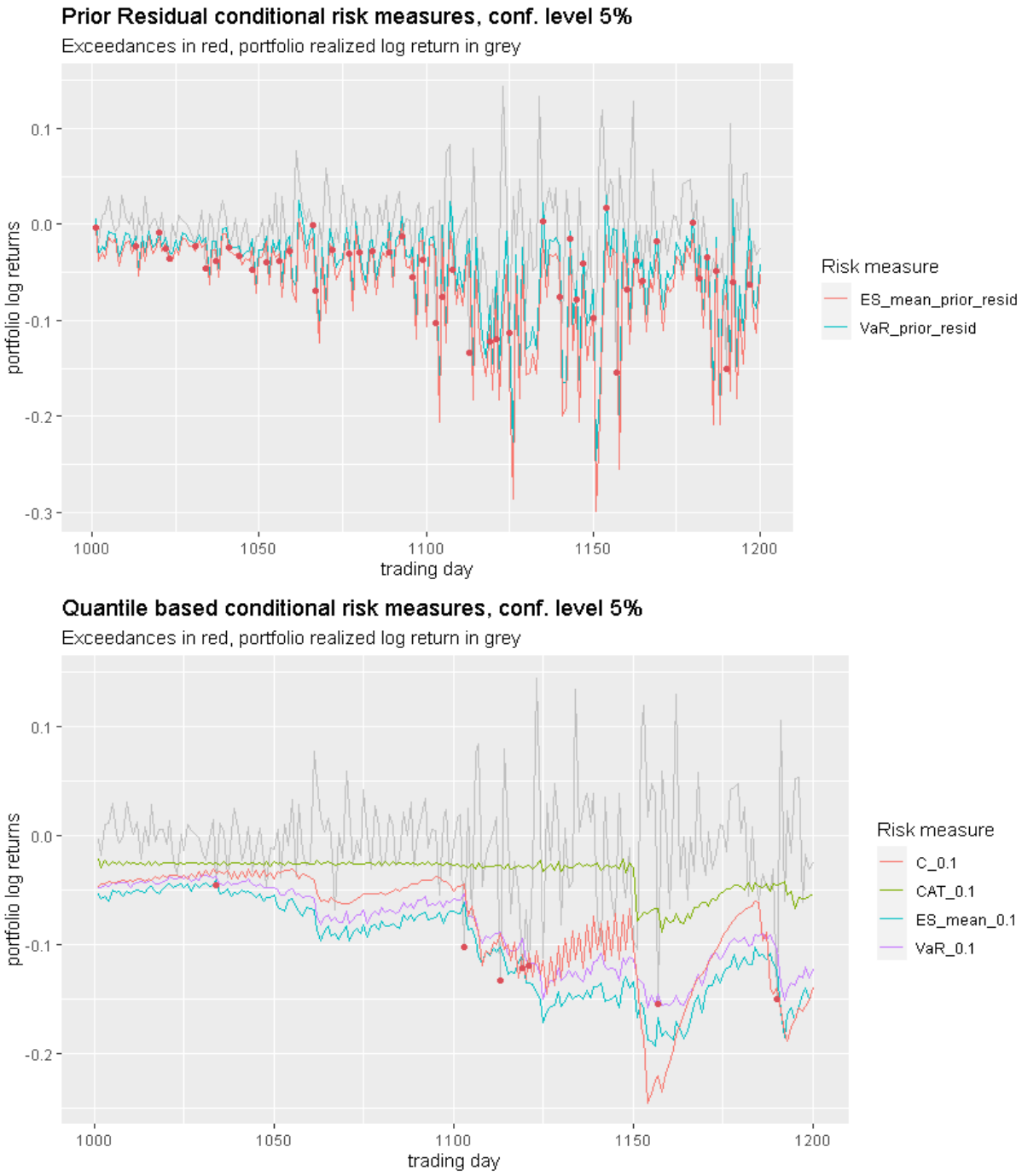

Figure 4.15. Conditional residual and quantile strategies portfolio exceedances.

Citigroup (C) and Catarpillar (CAT) are considered as market indexes for the quantile strategy and assumed to performing bad, since $\alpha^I = 0.1$ for both. Even if they cannot be considered true market indexes, conditioning on the value of other companies can be very useful if, for example, there are good relationships in terms of interbank deposits or other activities between two or more financial institutions. In addition, as reported later by Richard M. Bowen III, Business Chief Underwriter for Correspondent Lending in Citigroup's Consumer Lending Group, 60% of the mortgages purchased by Citigroup from some 1,600 mortgage companies were "defective" (were not underwritten according to the policy, or did not contain all of the documents required by the policy), this despite the fact that each of these 1,600 originators were contractually responsible (certified through representations and warranties) that their mortgages met Citigroup's standards. Basically, what we want to say in simple words is that these stress conditions could really have been realized, except that the problem had become of such a large magnitude that it forced government takeovers, which in the case of Citigroup reached 36% of stake.
Overall, conditional quantile strategy seems to be a good approximation of what should be the theoretical VaR at confidence level 5%. Results of the adopted hypothesis tests are reported below and confirm the validity of the conditional approach:

	VaR test of Kupiec	VaR test of Christoffersen
Null Hypothesis (H_0)	Correct Exceedances	Correct Exceedances & Independent
Actual exceedances	24	24
Expected exceedances	10	10
LR	15.1	17.1
LR critical value ($\alpha = 0.05$)	3.84	5.99
P-value	1.03×10^{-4}	1.94×10^{-4}
Decision	Reject H_0	Reject H_0

Table 4.1. Kupiec and Christoffersen VaR tests for unconditional rolling window approach.

	VaR test of Kupiec	VaR test of Christoffersen
Null Hypothesis (H_0)	Correct Exceedances	Correct Exceedances & Independent
Actual exceedances	50	50
Expected exceedances	10	10
LR	90	110
LR critical value ($\alpha = 0.05$)	3.84	5.99
P-value	0	0
Decision	Reject H_0	Reject H_0

Table 4.2. Kupiec and Christoffersen VaR tests for conditional residual rolling window approach.

	VaR test of Kupiec	VaR test of Christoffersen
Null Hypothesis (H_0)	Correct Exceedances	Correct Exceedances & Independent
Actual exceedances	7	7
Expected exceedances	10	10
LR	1.05	1.56
LR critical value ($\alpha = 0.05$)	3.84	5.99
P-value	0.305	0.457
Decision	Fail to Reject H_0	Fail to Reject H_0

Table 4.3. Kupiec and Christoffersen VaR tests for conditional quantile rolling window approach.

	ES Test of McNeil and Frey
Null Hypothesis (H_0)	Mean of Excess Violations of VaR is equal to zero
Actual Excess Violations	24
Expected Excess Violations	10
Bootstrap P-value	0.49
Decision	Fail to Reject H_0

Table 4.4. McNeil and Frey ES test for unconditional rolling window approach.

	ES Test of McNeil and Frey
Null Hypothesis (H_0)	Mean of Excess Violations of VaR is equal to zero
Actual Excess Violations	50
Expected Excess Violations	10
Bootstrap P-value	0.00208
Decision	Reject H_0

Table 4.5. McNeil and Frey ES test for conditional residual rolling window approach.

	ES Test of McNeil and Frey
Null Hypothesis (H_0)	Mean of Excess Violations of VaR is equal to zero
Actual Excess Violations	7
Expected Excess Violations	10
Bootstrap P-value	0.156
Decision	Fail to Reject H_0

Table 4.6. McNeil and Frey ES test for conditional quantile rolling window approach.

In fact, the message from tests results is that, as far as VaR is concerned and in this specific example, the best approximation of what should be the theoretical VaR at 5% confidence level is obtained by the conditional VaR with quantile strategy, since Kupiec and Christoffersen tests fail to reject only in this case. For ES, on the other hand, McNeil and Frey's test fail to reject both in case of conditional quantile strategy and for the unconditional approach, thus showing how the conditional approach is not always better.

Another specific case study, useful to show the strength of conditional approach and vine copulas is the following. Parameters and quantities set previously are unchanged except for the portfolio and training and test sets. Specifically, the portfolio under consideration is: $\Omega_3 = \{$International Business Machines Corporation (IBM), Intel Corporation (INTC), & Johnson & Johnson (JNJ), JPMorgan Chase & Co. (JPM)$\}$ each weighing $\frac{1}{4}$, $I_3 = \{$American International Group Inc. (AIG), The Coca-Cola Company (KO)$\}$. As training set, 1000 observations are taken between 2003-09-24 and 2007-09-13, while the period from 2008-04-21 to 2009-02-03 is taken as test set. This means that model training phase capture the two DJIA peaks previously mentioned while the Bearn Sterns failure is excluded. In theory, this nearly 7 months time hole should negatively impact the rolling window approach for estimating risk measures, since the purchase of not-so-healthy Bearn Sterns shares by JPMorgan Chase & Co. could have greatly impacted dependencies estimates for Ω_3 in those months. Yet, what can be observed from the results of Kupiec, Christoffersen and McNeil & Frey tests is that quantile strategy conditional risk measures fail to reject tests, unlike the others, thus showing their superior ability in approximating theoretical risk measures with respect to a given confidence level. In addition to this, this example also highlights the ability in correctly modeling and adapting through the different rolling windows of the dependence between assets, despite not having complete information.

What has been just said allows us to introduce the last part of this work, which is the study of probability equivalent levels. This tools should enable risk management and not just its measurement, as done so far. In particular, they allow to understand under what conditions one strategy is more or less conservative than others. In fact, as explained in the previous section, PELCoV, and similarly PELCoES, are the levels u_v of conditioning market indexes such that the conditional and unconditional approaches for estimating risk measures at a given portfolio confidence level v are equal. Higher conditioning levels may imply that CoVaR is less conservative than VaR, while lower levels may imply instead that VaR could underestimate spillover effects. In order to find PELCoV and PELCoES at a given confidence level v, the quantile strategy just presented is reiterated conditioning on market index values that are different from 0.1 and store inside the vector u_v. The study is repeated both for one conditional asset and for two conditional assets. For the latter case, only conditioning on the same value of α^I of both market indexes is allowed by *portvine* package. This simplification is due to both computational reasons but also for a theoretical nature. In fact, in the way just explained, probability equivalent levels can be represented through lines, given that each confidence level v of portfolio Ω is associated with a single conditioning level u_v of indexes. If it were possible to condition on two indexes with different values, then the result would be a surface of probability equivalent levels, a fact that greatly complicates the treatment and notation adopted up to this point. Possible future areas of study could concern this topic, as it is evident in reality that in

stressful situations not all market players behave in the same way. Parameters previously presented are rescaled according to table 4.7 to avoid increase in computational costs.

Parameters	**Values**
Training set	750 observations before 2008-07-01
Test set	150 observations after 2008-07-01
$\mathbf{\Gamma}$	750
$\mathbf{\Psi}$	750
$\boldsymbol{\gamma}$	50
$\boldsymbol{\kappa}$	25
$\mathbf{S}$	500
$\mathbf{v}$	{0.01, 0.015, 0.02, 0.025, 0.03, 0.035, 0.04, 0.045, 0.05, 0.055, 0.06, 0.065, 0.07}
$\boldsymbol{\Omega_1}$	{Alcoa Corporation (AA), American Express Company (AXP), The Boeing Company (BA), Bank of America Corporation (BAC)} each weighing $\frac{1}{4}$
$\boldsymbol{I_1}$	{Citigroup Inc. (C), Caterpillar Inc. (CAT)}
$\boldsymbol{\Omega_2}$	{DuPont de Nemours Inc. (DD), The Walt Disney Company (DIS), General Electric Company (GE), General Motors Company (GM)} each weighing $\frac{1}{4}$
$\boldsymbol{I_2}$	{The Home Depot Inc. (HD), HP Inc. (HPQ)}
$\boldsymbol{\Omega_3}$	{International Business Machines Corporation (IBM), Intel Corporation (INTC), Johnson & Johnson (JNJ), JPMorgan Chase & Co. (JPM)} each weighing $\frac{1}{4}$
$\boldsymbol{I_3}$	{American International Group Inc. (AIG), The Coca-Cola Company (KO)}

Table 4.7. Settings for probability equivalent level analysis.

Going into details, for a fixed portfolio confidence level v, what is done can be seen in Figures 4.16, 4.17, 4.18. All conditional strategies with different conditioning values are plotted together with unconditional risk measures. Conditioning values u_v are chosen according to the portfolio under consideration and by deductive reasoning. In fact, the analysis is initially made with the dummy vector $u_v = \{0.01,0.11,0.21,0.31,0.41,0.51, 0.61,0.71,0.81,0.91\}$, where each single value represent a plausible probability equivalent level. Next, the elements of the vector u_v such that above and below those there are no intersections between the conditional and unconditional approaches are identified. At this point, the process of Figures 4.16, 4.17, 4.18 is done for a new vector u_v with 10 points equally spaced between the previously identified thresholds. In this way, the search for the exact value of u_v among the elements contained in the vector is done in a more detailed and precise manner than just adopting the same u_v vector for all portofolios. Note that thresholds are set for all risk measures together, and not for VaR and ES separately, to avoid an excessive increase in computational costs. In addition, the reasoning is made first for u_v vector corresponding to probability equivalent levels analysis associated with one conditional asset risk measures and then repeated again for u_v vector in the case of probability equivalent level analysis with two conditional assets risk measures.

Figures 4.16, 4.17, 4.18 offer a dual interpretation of the concept of probability equivalent level. On the one hand there is the definition presented up to this moment, i.e. u_v are all intersection points between the black line and the colored lines. Actually, with few data available, lines do not intersect at points that have x axis in common. However, intersections clearly exist, as can be seen from plots. Having to determine the value of u_v in closed form, the strategy adopted for this work consists in fixing a particular colored line (corresponding to a particular conditioning level) and then calculating the number of upcrossing and downcrossing between the line just mentioned and the black one so as to obtain an approximation of number of times the two lines intersect. This reasoning is done for each different conditioning level, i.e. for all colored lines, obtaining for each element of the vector u_v the number of intersections with the unconditional approach. Subsequently, a weighted average of the different u_v is carried out, where the weight is given by the number of intersections, to obtain an approximation of PELCoV and PELCoES final values.

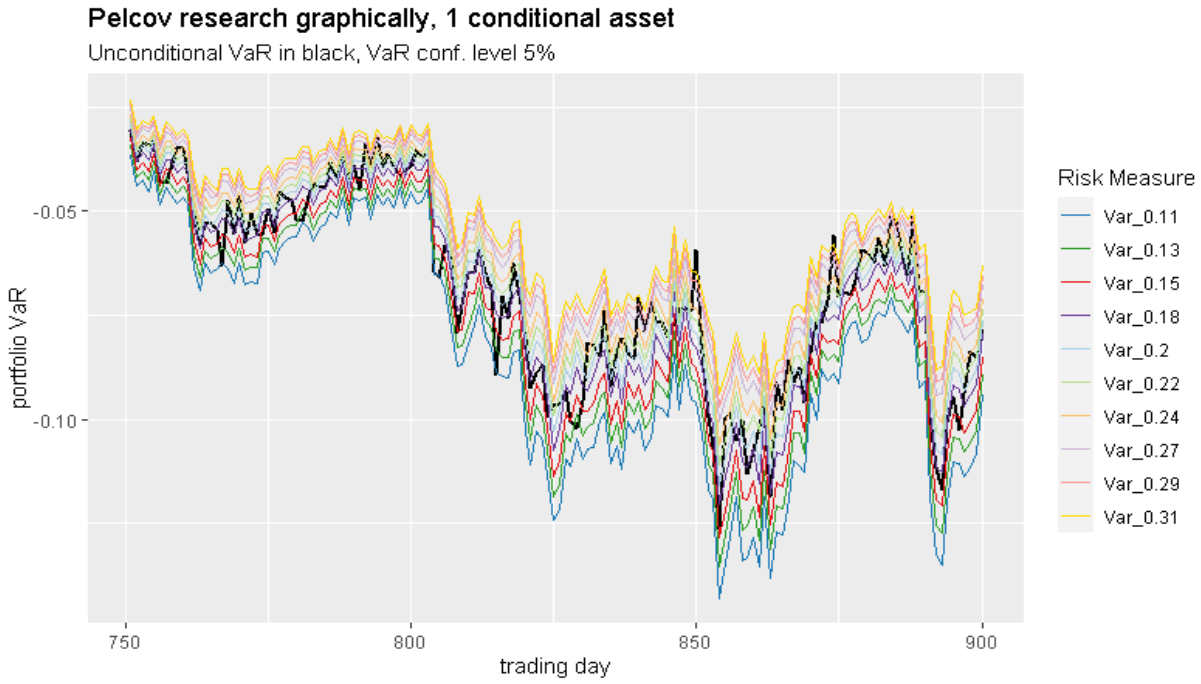

Figure 4.16. One conditional asset PELCoV analysis, portfolio confidence level v=0.05.

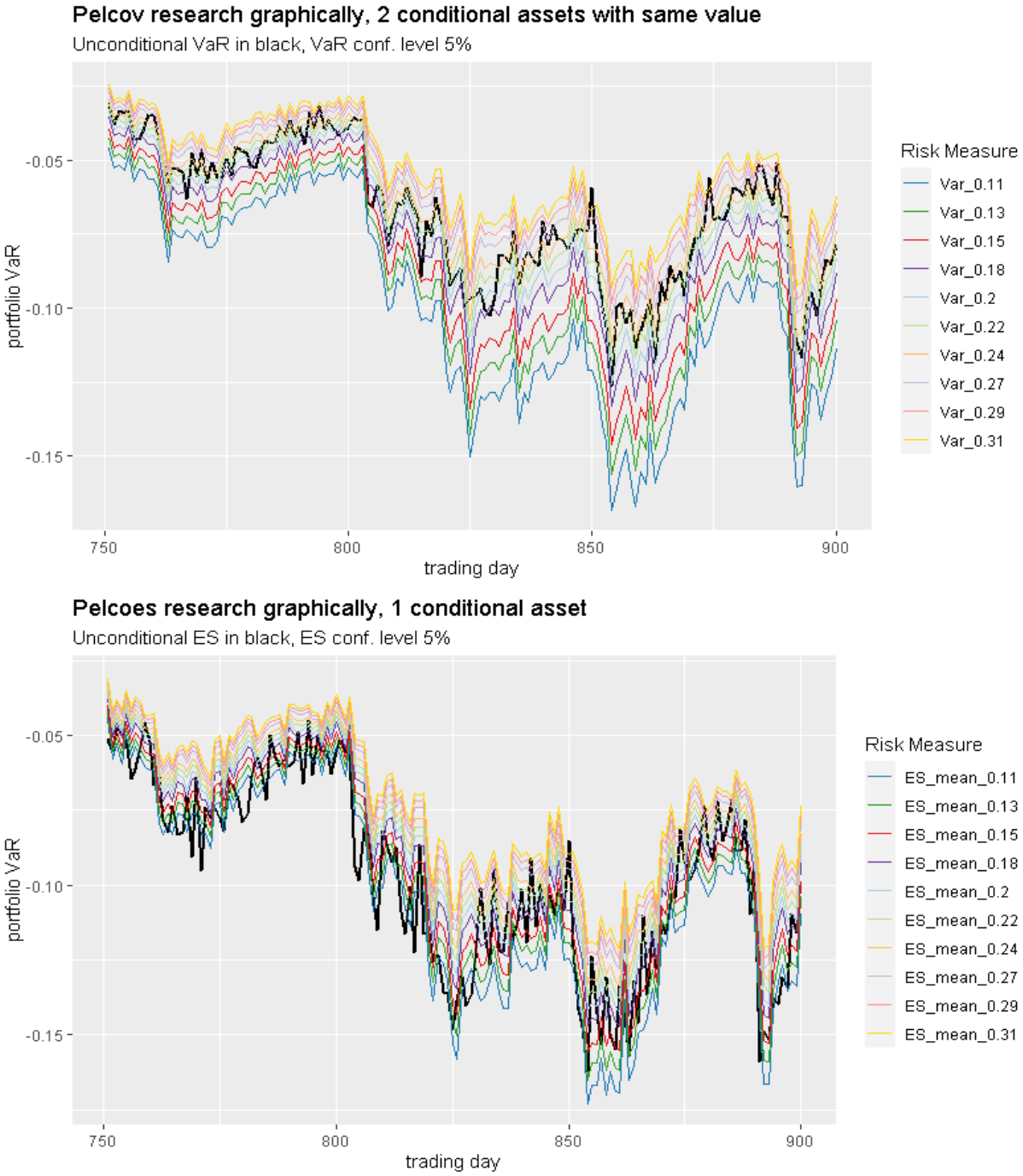

Figure 4.17. Two conditional assets PELCoV and one conditional PELCoES analyses, portfolio confidence level v=0.05.

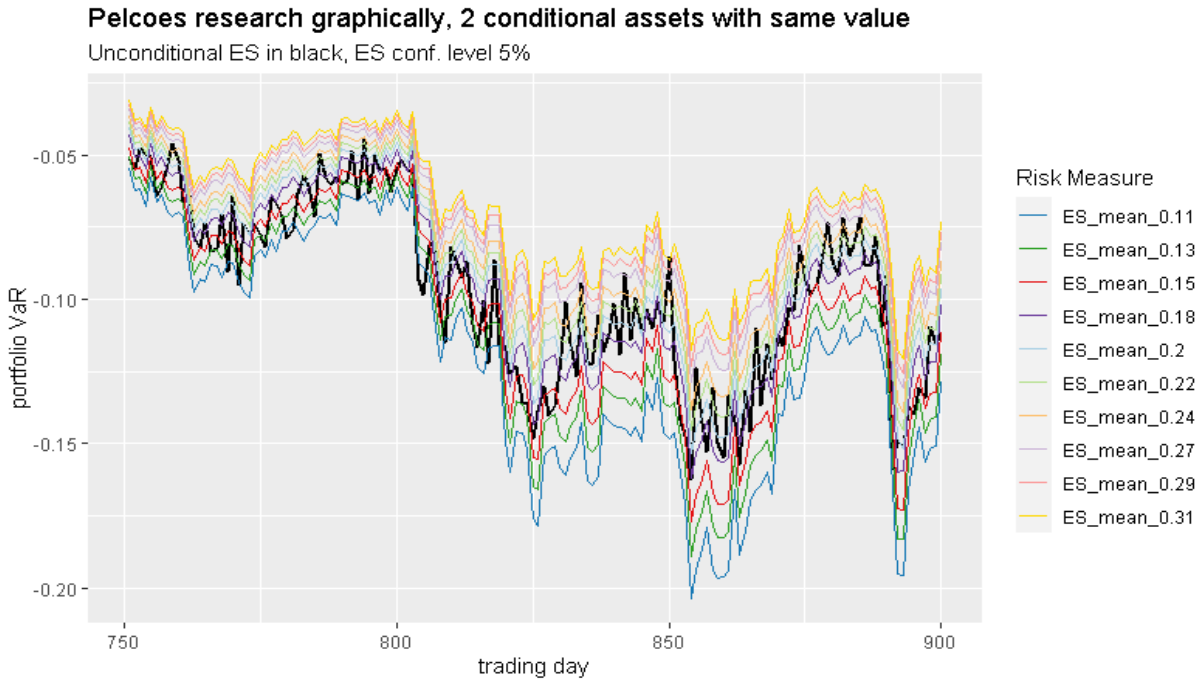

Figure 4.18. PELCoES analysis, portfolio confidence level v=0.05.

The whole procedure is repeated for each portfolio confidence level v and the final performance of PELCoV and PELCoES, whether conditioned on one or two assets, are shown from Figure 4.19 to Figure 4.30. Although on the y-axis it can be seen that sometimes values assumed by probability equivalent levels are very close and the differences are on the order of the second decimal place, in general the behavior of probability equivalent levels is varied. Figures 4.19, 4.20, 4.21, 4.22, 4.24 show how in certain cases, as v increases the same happens for u_v, revealing a kind of linear behavior. The fact may be quite intuitive, if one considers that with the increase of confidence level v, the returns in the v% worst cases increase, and consequently the conditioning levels for the conditional risk measures should also increase in order to obtain PELCoV and PELCoES. Despite this, as already pointed out by Ortega-Jiménez et al. [2024], there are conditions that guarantee monotonicity and the same should be investigated for the multivariate case. Looking at all plots together in fact we can observe that in general the behavior of probability equivalent levels seem to be either monotonic increasing or constant fluctuating around a certain value. A further element of study might be why the same probability equivalent level conditioned on different numbers of assets result in opposite trends of u_v values, as in Figures 4.21, 4.22.

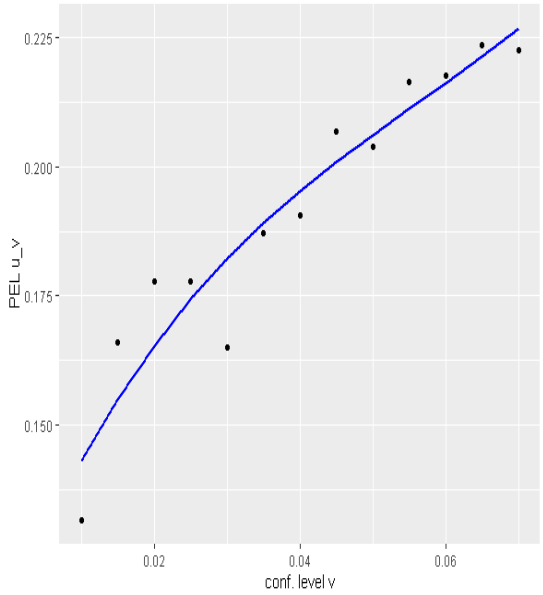

Figure 4.19. One conditional asset PELCoV trend for Ω_1, I_1.

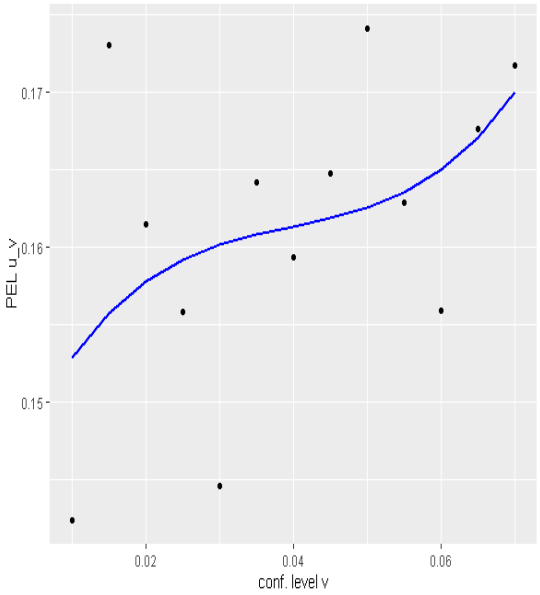

Figure 4.20. Two conditional assets PELCoV trend for Ω_1, I_1.

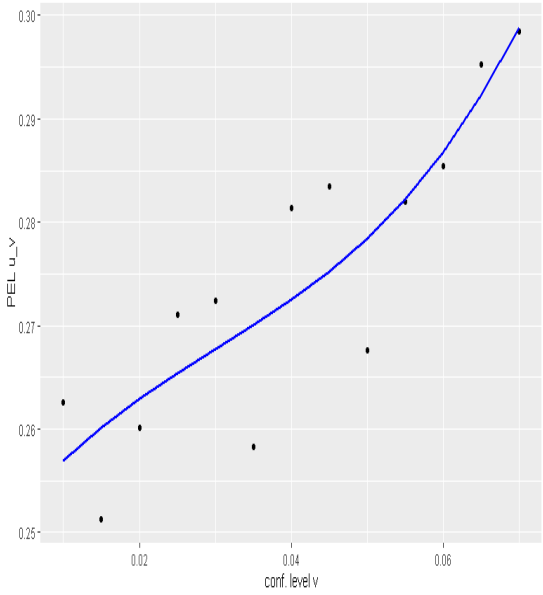

Figure 4.21. One conditional asset PELCoV trend for Ω_2, I_2.

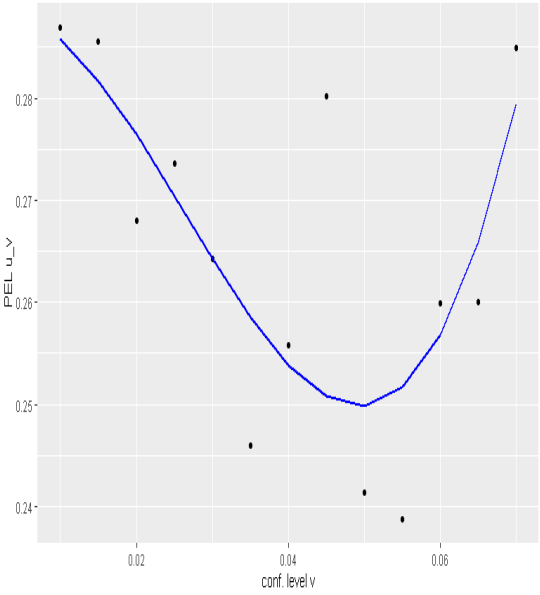

Figure 4.22. Two conditional assets PELCoV trend for Ω_2, I_2.

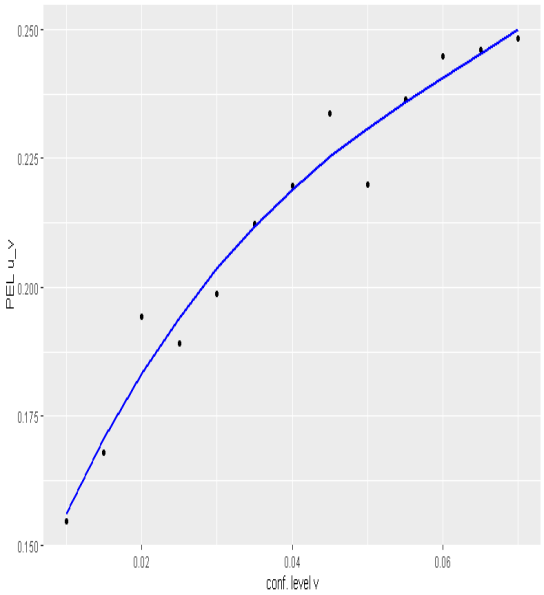

Figure 4.23. One conditional asset PELCoV trend for Ω_3, I_3.

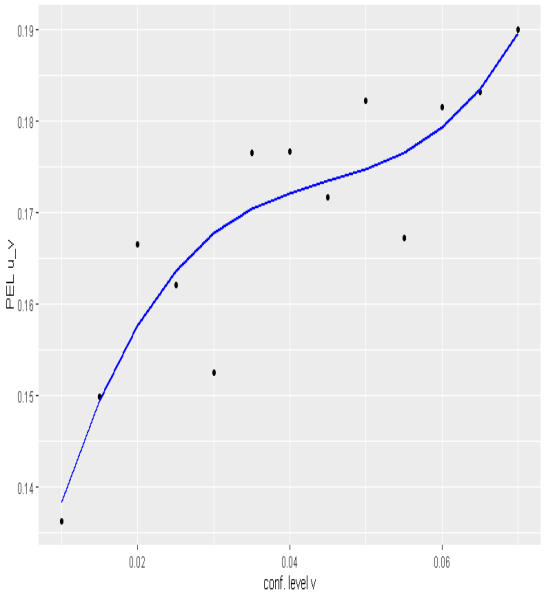

Figure 4.24. Two conditional assets PELCoV trend for Ω_3, I_3.

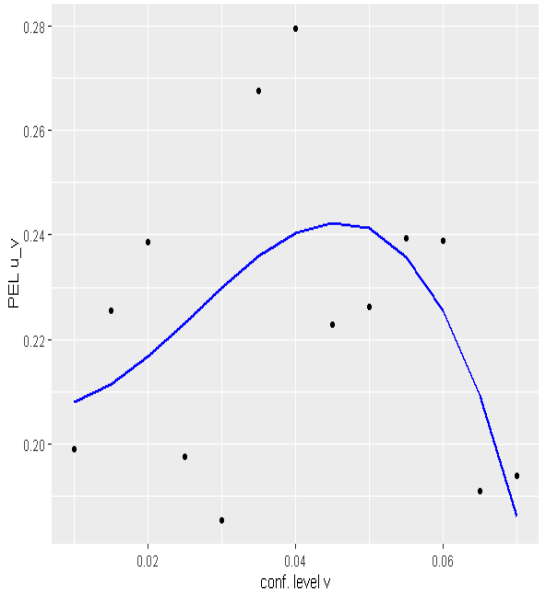

Figure 4.25. One conditional asset PELCoES trend for Ω_1, I_1.

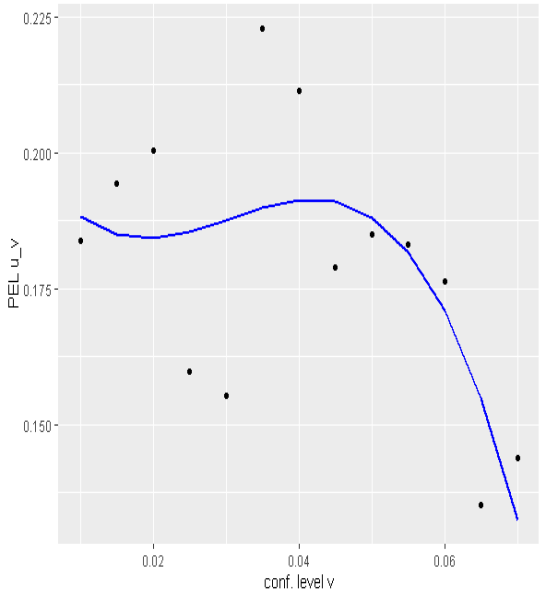

Figure 4.26. Two conditional assets PELCoES trend for Ω_1, I_1.

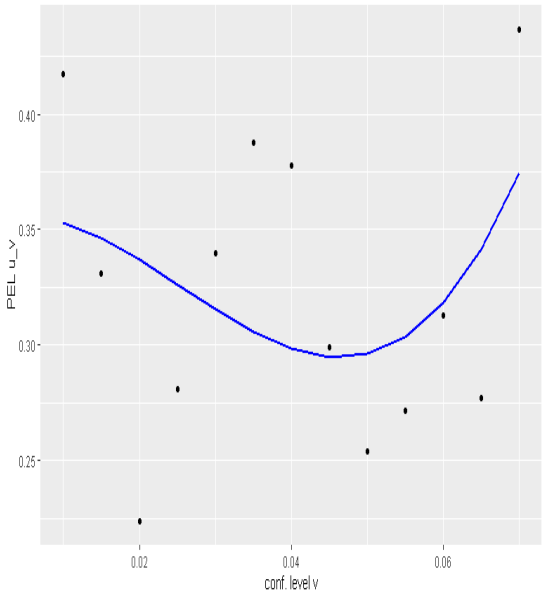

Figure 4.27. One conditional asset PELCoES trend for Ω_2, I_2.

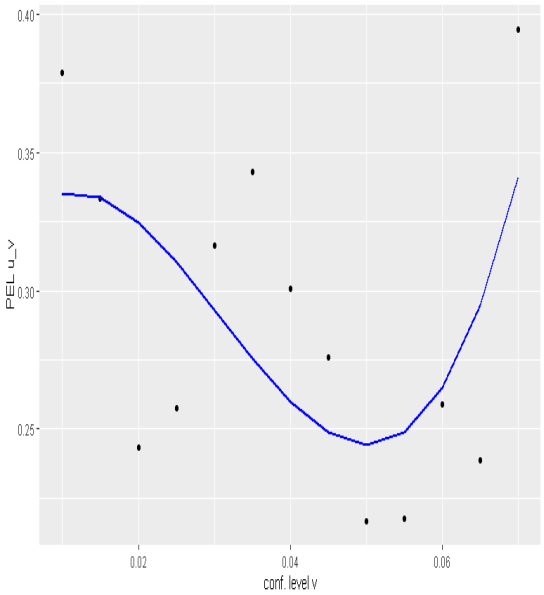

Figure 4.28. Two conditional assets PELCoES trend for Ω_2, I_2.

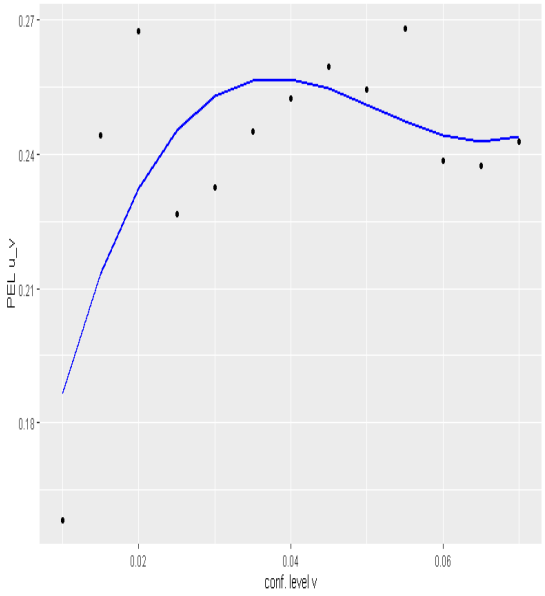

Figure 4.29. One conditional asset PELCoES trend for Ω_3, I_3.

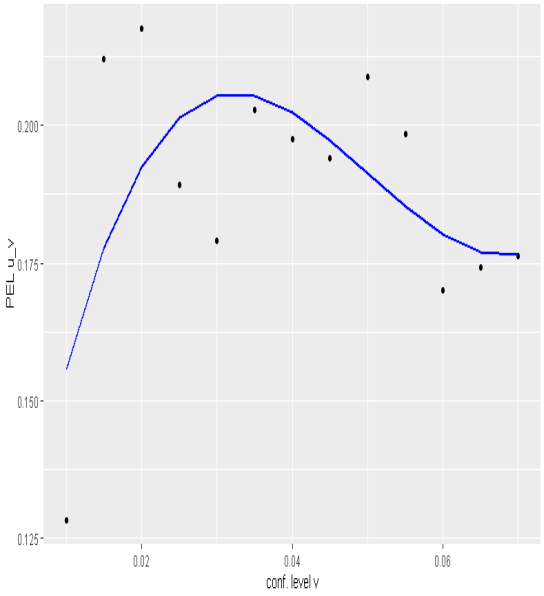

Figure 4.30. Two conditional assets PELCoES trend for Ω_3, I_3.

On the other hand, another interpretation of probability equivalent levels, which hopefully is more operational, is based on graphs 4.16, 4.17, 4.18. In fact, without the need to calculate the exact value of u_v for each portfolio confidence level v, we have already observed that:

- There are levels that can be interpreted as upper or lower bounds, in the sense that conditional risk measures never intersect with the respective unconditional one either from above or below. For example, in the first PELCoV analysis for Ω_1 with two conditional assets and the dummy vector u_v, these values were found to be 0.11 and 0.31.
- There exists a market index conditioning level u_v where it is evident that most of the overlap occurs for that value. For example, in the first PELCoV analysis for Ω_1 with two conditional assets and the dummy vector u_v, this value can be guessed to be 0.21.

These three quantities could be of great use for example in defining indicators for early warning systems based on historical data under stressed conditions. Their purpose could be to identify critical warning thresholds for repayment risks within a credit risk framework. This would enable banks to take corrective actions, such as increasing RWAs, to meet liquidity requirements during times of market stress.

Having reached this point, the careful reader will notice that all reasonings about monotonicity of probability equivalent levels have been made with conditional verbs. In particular, two observations were made in the preceding paragraphs:

- As the market indexes conditioning level α^I increases, we should expect conditional risk measures to increase as well.
- As the portfolio confidence level v for estimating risk measures increases, we should also expect probability equivalent levels final value u_v to increase as well.

Ortega-Jiménez et al. [2024] have already investigated the monotonic conditions under which the second point is true, even if only in the bidimensional case. What they have not completely outlined, however, concerns the first point. Intuitively one would be inclined to think that what is written is right. However, this is a false intuition, and to prove it the analysis presented so far has been repeated on a fourth portfolio $\Omega_4 = \{$ 1329, ETFMIB, GDAXIEX, SPY$\}$, with conditioning assets $I_4 = \{$GCJ4, CCK4$\}$ taken from Investing. Asset names are nothing but the symbols found in the financial markets of four ETFs that track Nikkei 225, FTSE MIB, DAX and S&P 500 indexes respectively. The conditioning indexes, on the other hand, are the prices of gold and cocoa futures respectively. The training data refer to 689 observations before 2007-09-4, in which all assets increase and then peak, while the test ones are the later 324 realizations, corresponding to the decline due to the Global Financial Crisis. Time series and copula parameters are chosen as in table 4.7 with the exceptions of:

- Γ =689
- Ψ=689

- γ=50
- κ=50

The uniqueness of this dataset, compared to the previous ones, can be seen in Figure 4.31 and Table 4.8, which represent univariate asset prices, Spearman's ρ and Kendall Tau matrices between training, test set and the conditioning assets respectively. Portfolio assets are positively correlated with each other and with the conditioning indexes in the training set. However, in the test set the behavior reverses, and this challenges the algorithms presented so far for their ability to correctly model the dependence and know how to adjust it according to market conditions.

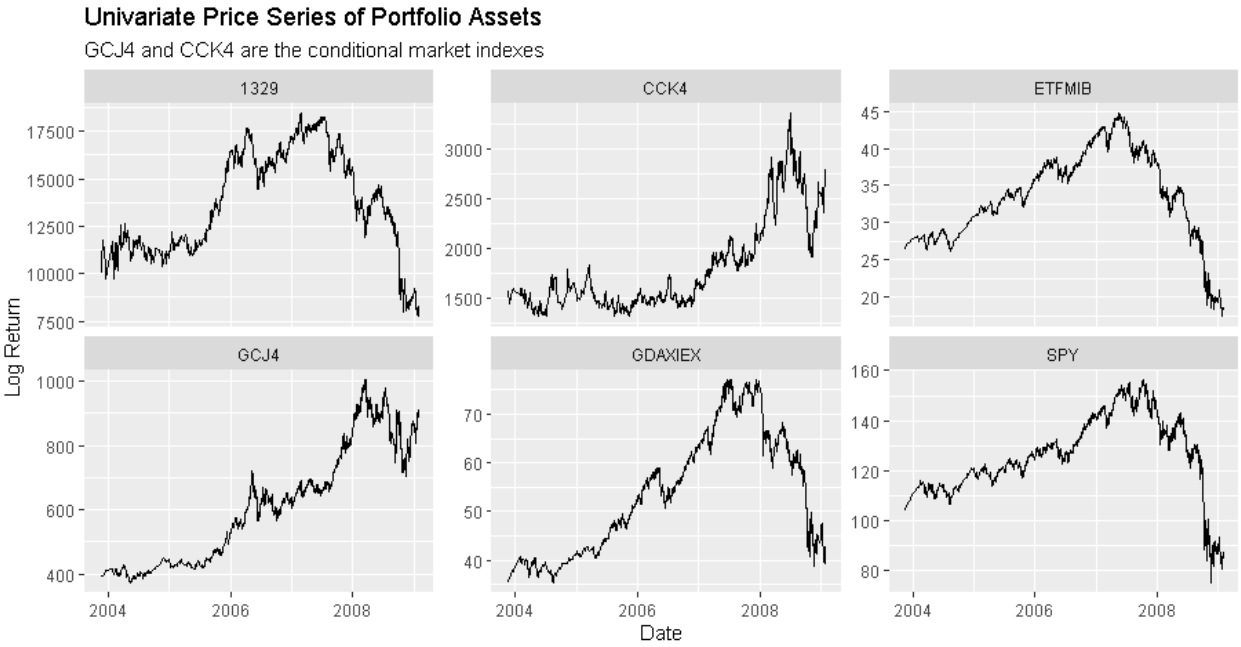

Figure 4.31. Asset prices for Ω_4 and I_4.

Spearman's ρ	GCJ4	CCK4
Ω_4 training set	0.8845225	0.5328657
Ω_4 test set	-0.1222775	-0.3032364

Kendall Tau	GCJ4	CCK4
Ω_4 training set	0.6932501	0.3852222
Ω_4 test set	-0.1098750	-0.1762041

Table 4.8. Correlation measures between Ω_4 and I_4 asset prices. GCJ4 and CCK4 prices are taken only in the respective set of Ω_4.

Moreover, Figure 4.31 is presented at price level, in order to highlight more the correlation between different assets. However, the procedure for estimating risk measures is done at the log return level, in order to ensure weak stationarity of time series under consideration. The logarithmic scale poses an additional problem for modeling dependencies, since the correlations seen in table 4.8 will not be as pronounced as for prices. In addition to this, two questions that can arise are: How will the estimated unconditional risk measures perform? Regarding conditional risk measures, is it fair to expect them to perform better when conditioning on positive value in the copula scales (i.e. $\alpha^I = 0.9$) or if conditioning on bad values (i.e. $\alpha^I = 0.1$)? The second question may seem meaningless, as one would expect that only by conditioning on the true value assumed by gold and cocoa, i.e positive values as can be seen in the test set for this particular case, the conditional risk measures would perform well. In fact, Ω_4 and I_4 are negatively correlated in the test set, which in turn implies that positive values of gold and cocoa in the copula scale would be matched by portfolio bad values in the copula scale. The results of Kupiec, Christoffersen and McNeil & Frey tests only partially confirm what just said, showing indeed that even conditioning on bad values of gold and cocoa the rolling window risk measure estimation algorithm still succeeds in correctly modeling dependencies and exceedances:

	VaR test of Kupiec	VaR test of Christoffersen
Null Hypothesis (H_0)	Correct Exceedances	Correct Exceedances & Independent
Actual exceedances	15	15
Expected exceedances	16	16
LR	0.0958	0.225
LR critical value ($\alpha = 0.05$)	3.84	5.99
P-value	0.757	0.893
Decision	Fail to Reject H_0	Fail to Reject H_0

Table 4.9. Kupiec and Christoffersen VaR tests, unconditional rolling window approach for Ω_4, I_4.

	VaR test of Kupiec	VaR test of Christoffersen
Null Hypothesis (H_0)	Correct Exceedances	Correct Exceedances & Independent
Actual exceedances	11	11
Expected exceedances	16	16
LR	1.97	2.75
LR critical value ($\alpha = 0.05$)	3.84	5.99
P-value	0.16	0.253
Decision	Fail to Reject H_0	Fail to Reject H_0

Table 4.10. Kupiec and Christoffersen VaR tests, conditional quantile rolling window approach with $\alpha^I = 0.1$ for Ω_4, I_4.

	VaR test of Kupiec	VaR test of Christoffersen
Null Hypothesis (H_0)	Correct Exceedances	Correct Exceedances & Independent
Actual exceedances	24	24
Expected exceedances	16	16
LR	3.47	3.5
LR critical value ($\alpha = 0.05$)	3.84	5.99
P-value	0.0627	0.174
Decision	Fail to Reject H_0	Fail to Reject H_0

Table 4.11. Kupiec and Christoffersen VaR tests, conditional quantile rolling window approach with $\alpha^I = 0.9$ for Ω_4, I_4.

	ES Test of McNeil and Frey
Null Hypothesis (H_0)	Mean of Excess Violations of VaR is equal to zero
Actual Excess Violations	15
Expected Excess Violations	16
Bootstrap P-value	0.309
Decision	Fail to Reject H_0

Table 4.12. McNeil and Frey ES test, unconditional rolling window approach for Ω_4, I_4.

	ES Test of McNeil and Frey
Null Hypothesis (H_0)	Mean of Excess Violations of VaR is equal to zero
Actual Excess Violations	11
Expected Excess Violations	16
Bootstrap P-value	0.275
Decision	Reject H_0

Table 4.13. McNeil and Frey ES test, conditional quantile rolling window approach with $\alpha^I = 0.1$ for Ω_4, I_4.

	ES Test of McNeil and Frey
Null Hypothesis (H_0)	Mean of Excess Violations of VaR is equal to zero
Actual Excess Violations	24
Expected Excess Violations	16
Bootstrap P-value	0.0825
Decision	Fail to Reject H_0

Table 4.14. McNeil and Frey ES test, conditional quantile rolling window approach with $\alpha^I = 0.9$ for Ω_4, I_4.

In order to give a meaningful explanation for these seemingly counterintuitive results, it is necessary to study the structures of R-vines and D-vines constructed when applying the rolling window risk measure estimation algorithms. This structures are shown in Figures 4.32 and 4.33. Looking in particular at figure 4.32 we can answer the first of the two questions presented earlier: unconditional risk measures perform well even in the case of datasets with negatively correlated assets due to the greater flexibility of R-vine structures, which are not simply paths but can potentially include more articulated trees. On the other hand, with respect to Figure 4.33, it can be observed that the weights of the edges connecting assets in the portfolio to the conditioning indexes are "lightly weighted," i.e they have low Kendall Tau values. In other words the algortims are focusing more on modeling internally portfolio dependencies, since they are positive and strong, than modeling poor negative dependencies with market indexes. This partially justifies the fact that, regardless of the conditioning level, tests are passed correctly in each case.

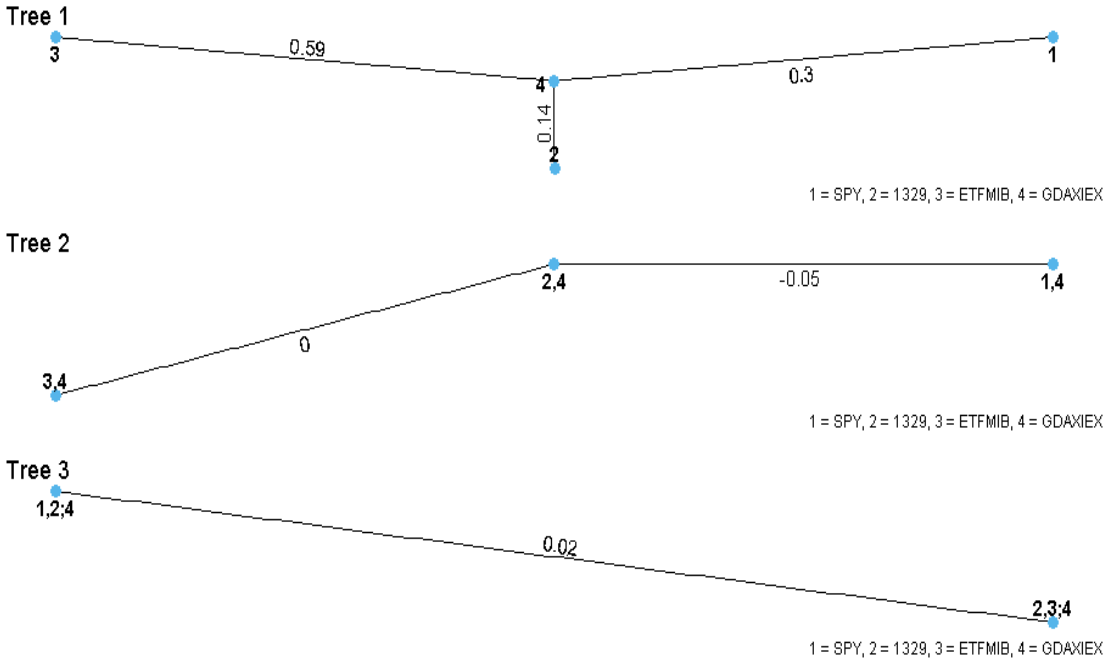

Figure 4.32. Example of R-vine structure for a particular rolling window.

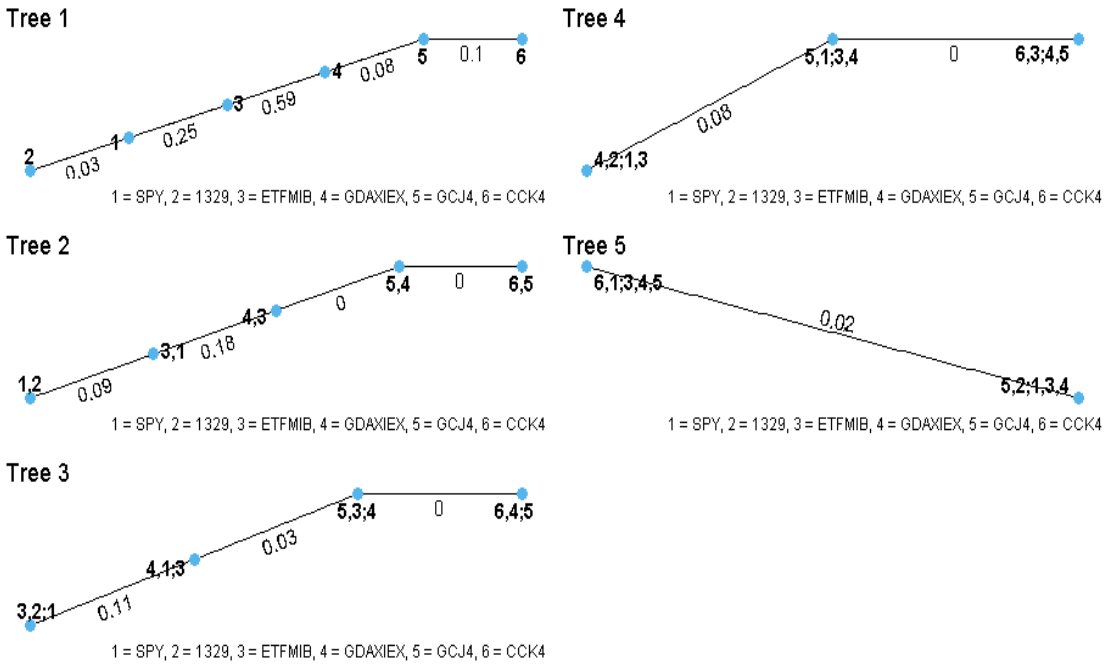

Figure 4.33. Example of D-vine structure for a particular rolling window.

Going finally to replicate the probability equivalent level analysis, an interesting fact emerges, closely related to the first issue of monotonicity mentioned few pages ago. Indeed, it has been said that as the conditioning level of market indexes α^I increases, it is false to expect that the conditional risk measures always increase. This is confirmed by Figure 4.34, where we can observe PELCoES analysis at 2.5% confidence level with one conditioning asset. The graph presents what might be called an "inverted structure", that is, positive values on which we condition in the copula scales correspond to lower risk measures, i.e. more conservative risk measures on the log return scale. In formulas, $CoES_{0.025,0.9}[\Omega_4|I_4] \leq CoES_{0.025,0.1}[\Omega_4|I_4]$. Thinking about it more closely, it makes some sense what happens in the figure if we look at the graph in parallel with those in Figure 4.31. In fact, by conditioning on positive values in the copula scales for gold and cocoa we are assuming that these assets are performing well in the market. Since they are negatively correlated with portfolio Ω_4, it is fair to expect accordingly that Ω_4 performs very poorly. This would explain why the conditional quantile risk measure estimation with $\alpha^I = 0.9$ turns out to be more conservative than the case with $\alpha^I = 0.1$.

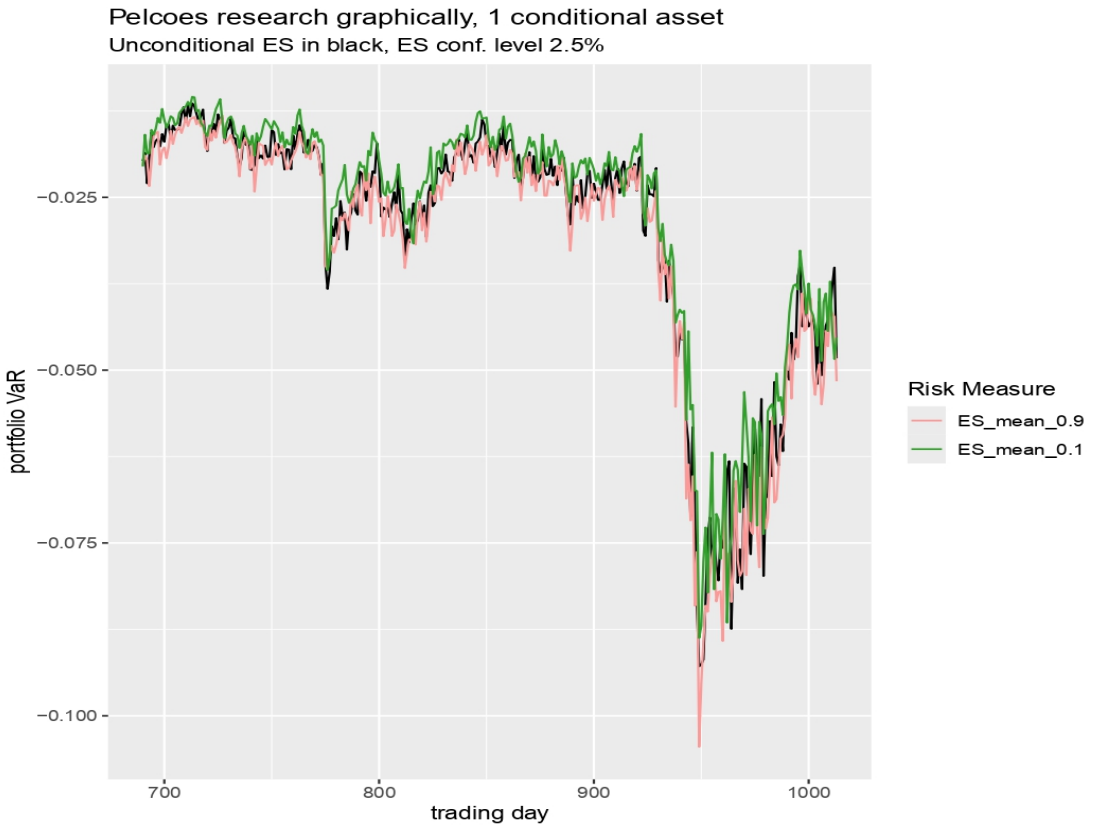

Figure 4.34. Example of inverted structure for conditional risk measures.

However, what has just been observed is only a special case, and going to analyze all the graphs such as those in Figures 4.16, 4.17, 4.18 but for every possible value of v, it can be

seen that there is no clear and definitive correspondence to say under what circumstances of dependence an "inverted structure" occurs or not. Looking for example at the number of exceedances in tables 4.10, 4.11, one can easily see that VaR and CoVar do not exhibit an "inverted structure" in that case, since the number of exceedances conditioning on $\alpha^I = 0.9$ is greater than the number of exceedances conditioning on $\alpha^I = 0.1$, thus $CoVaR_{0.05,0.9}[\Omega_4|I_4] \geq CoVaR_{0.05,0.1}[\Omega_4|I_4]$. What is certain at a more operational level, on the other hand, is that while the interpretation of probability equivalent levels is unique, the same cannot be said for the assesment of which risk measures are more or less conservative than others but a case-by-case study must be analyzed, studying the behaviors of the conditional risk measures for different values of the vector u_v and trying to extrapolate some sort of correspondence that determines the presence of an inverted structure or not.
For completeness, the results of probability equivalent level analysis for Ω_4 and I_4 are also shown below, similar to what can be seen from Figure 4.19 to Figure 4.30. Similar considerations to those made earlier can be repeated in part, with the difference that when conditioning on two assets it seems that PELCoV and PELCoES have a decreasing trend, which might be due to the presence of negatively correlated assets or "inverted structures" between conditional risk measures.

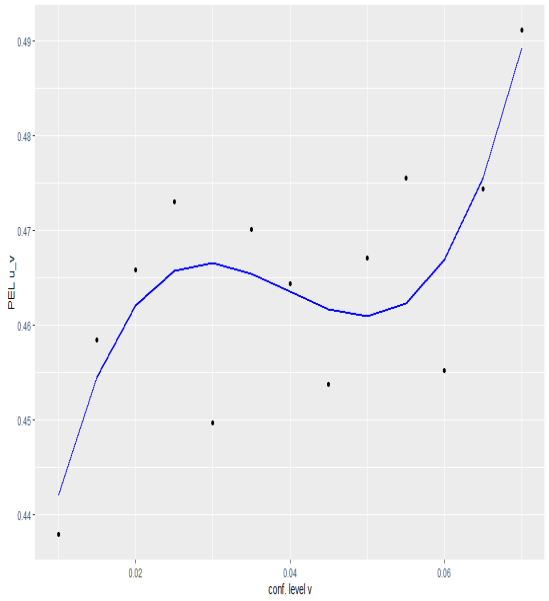

Figure 4.35. One conditional asset PELCoES trend for Ω_4, I_4.

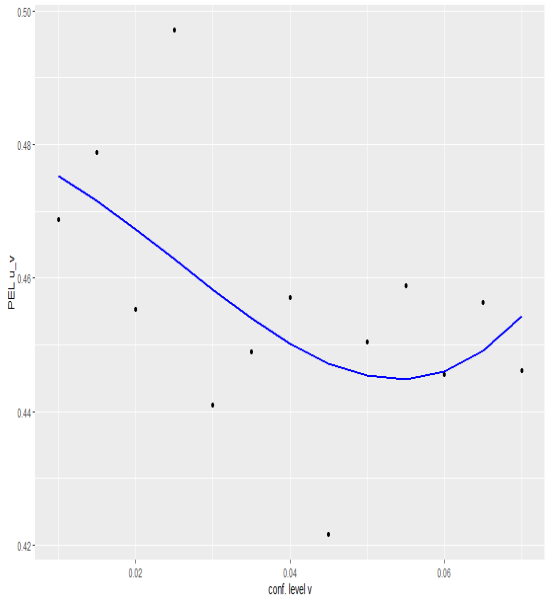

Figure 4.36. Two conditional assets PELCoES trend for Ω_4, I_4.

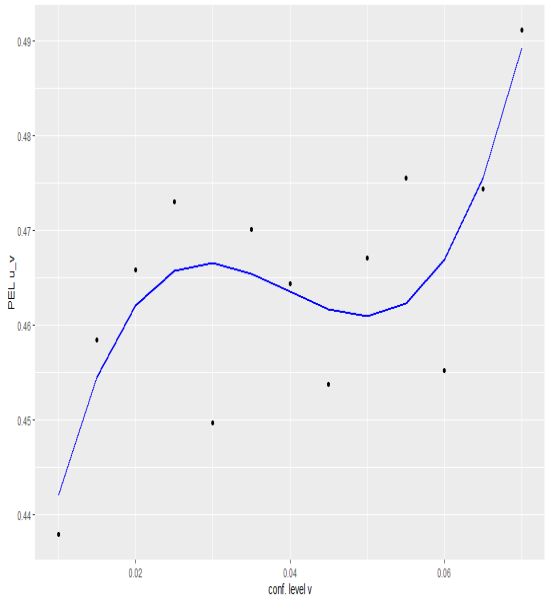

Figure 4.37. One conditional asset PELCoES trend for Ω_4, I_4.

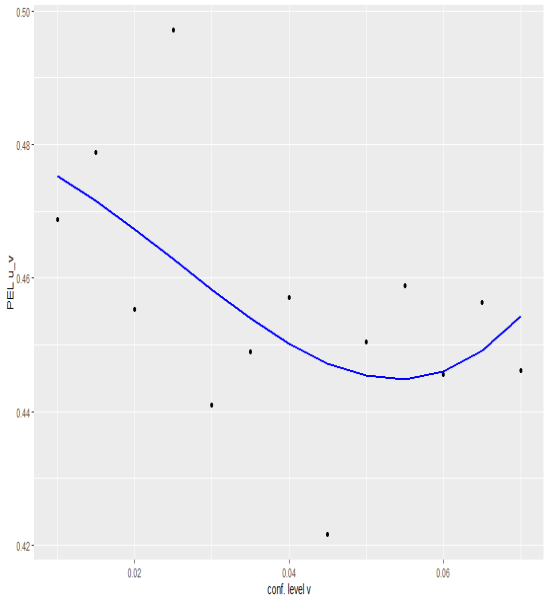

Figure 4.38. Two conditional assets PELCoES trend for Ω_4, I_4.

Chapter 5

Conclusion

The use of Vine copula presents various opportunities in the financial context, regardless of the specific area of application. In general, these tools provide a total view of what can be defined as systemic risk, being able to measure, predict and manage it. Thanks to univariate time series models, it is possible to feed Vine copula models to capture dependencies between different assets within an investment portfolio. Once the dependence structure is known, this work have addressed the prediction of risk measures, both with a conditional and unconditional approaches. The study of conditional risk measures, along with a demonstration of their validity, allows for a greater understanding of the different facets of systemic risk and brings to light another topic of equal importance, namely the study of probability equivalent levels. Their existence is proved with the following work, but a characterization from a more formal point of view for the multivariate case could be a future research topic, leading to the definition of multidimensional surfaces of probability equivalent levels. Beyond this, further extensions could be to consider an approach closer to dynamic programming, and thus less myopic than the current one, for evaluating R-Vine structures, as well as exploiting more complex univariate time series models, which would allow a more precise and detailed characterization of residuals.

Appendix A

R code

A.1 Monte Carlo Simulation

```
#libraries
rm(list = ls())
set.seed(129)
library(stats)
library(VineCopula)
library(skewt)
#global parameters
T=1000
d = 7
N=1000
dd = d*(d-1)/2
mat=matrix(c(7,4,5,1,2,3,6, 0,4,6,5,1,2,3, 0,0,6,5,1,2,3,
             0,0,0,5,1,3,2, 0,0,0,0,1,3,2, 0,0,0,0,0,3,2,
             0,0,0,0,0,0,2),7,7)
```

```
#MONTE CARLO SIMULATION
#SCENARIO WITH FIXED BB1 COPULA

#initializations
k=runif(7, 0, 7)
gamma=runif(7, 1, 7)
par1_mat=array(0,dim=c(d,d))
par1_mat[lower.tri(par1_mat)]=k
par2_mat=array(0,dim=c(d,d))
par2_mat[lower.tri(par2_mat)]=gamma
fam_mat=matrix(0,nrow=d,ncol=d)
fam_mat[lower.tri(fam_mat,diag=FALSE)]=7
rvm_r=RVineMatrix(Matrix=mat,family=fam_mat,par=par1_mat,par2=par2_mat)
rvine=array(NA, dim = c(T, d, N))
rvine_est=array(NA, dim=c(T, d, N))
tau=rvm_r$tau
gen_tau_diff=array(NA, dim = c(d, d, N))
```

```
lower_tau_diff=array(NA, dim = c(d, d, N))
upper_tau_diff=array(NA, dim = c(d, d, N))

#simulation
for (i in 1:N){
  rvine[,,i]=RVineSim(T,rvm_r)
}
#check missing values
sum(is.na(rvine))

#estimation
for (i in 1:N){
  rvm_est=RVineStructureSelect(rvine[,,i],progress=FALSE) #RVM = RVINE MATRIX

  #general tau difference
  rvine_est[,,i]=RVineSim(T,rvm_est) #RVINE EST=SAMPLE FROM RVM
  gen_tau_diff[,,i]=abs(TauMatrix(rvine[,,i])-TauMatrix(rvine_est[,,i]))

  #lower tau difference
  rvine_lower=matrix(nrow = 0, ncol = d)
  rvine_est_lower=matrix(nrow = 0, ncol = d)
  for (j in 1:(d-1)){
    for (k in (j+1):d){
      flg=rvine[,j,i]<=0.2 & rvine[,k,i]<=0.2
      rvine_lower=rbind(rvine_lower,rvine[flg,,i])
      flg_est=rvine_est[,j,i]<=0.2 & rvine_est[,k,i]<=0.2
      rvine_est_lower=rbind(rvine_est_lower,rvine_est[flg_est,,i])
    }
    if (dim(rvine_lower)[1]>1 & dim(rvine_est_lower)[1]>1){
      rvine_lower=unique(rvine_lower)
      rvine_est_lower=unique(rvine_est_lower)
      lower_tau_diff[,,i]=abs(TauMatrix(rvine_lower)-TauMatrix(rvine_est_lower))
    }
  }

  #upper tau difference
  rvine_upper=matrix(nrow = 0, ncol = d)
  rvine_est_upper=matrix(nrow = 0, ncol = d)
  for (j in 1:(d-1)){
    for (k in (j+1):d){
      flg=rvine[,j,i]>0.8 & rvine[,k,i]>0.8
      rvine_upper=rbind(rvine_upper,rvine[flg,,i])
      flg_est=rvine_est[,j,i]>0.8 & rvine_est[,k,i]>0.8
      rvine_est_upper=rbind(rvine_est_upper,rvine_est[flg_est,,i])
    }
    if (dim(rvine_upper)[1]>1 & dim(rvine_est_upper)[1]>1){
      rvine_lower=unique(rvine_lower)
      rvine_est_lower=unique(rvine_est_lower)
      upper_tau_diff[,,i]=abs(TauMatrix(rvine_upper)-TauMatrix(rvine_est_upper))
    }
  }
}

#display error
mean_tau_diff=apply(gen_tau_diff, c(1, 2), mean)
print(paste("mean generic tau difference 1",mean(mean_tau_diff[lower.tri(mean_tau_
    diff)])))
#T=500, N=10: 0.01361244393549
#T=500, N=100: 0.0111957247828991
#T=500, N=1000: 0.0115575364061456
#T=1000, N=1000: 0.00931771161637828

mean_lower_tau_diff=apply(lower_tau_diff, c(1, 2), mean)
print(paste("mean lower tau difference",mean(lower_tau_diff[lower.tri(lower_tau_
    diff)])))
#T=500, N=10: 0.0450569702002633
#T=500, N=100: 0.0393313219607121
```

```
#T=500, N=1000: 0.040562476569338
#T=1000, N=1000: 0.0329022128240668

mean_upper_tau_diff=apply(upper_tau_diff, c(1, 2), mean)
print(paste("mean upper tau difference",mean(upper_tau_diff[lower.tri(upper_tau_
    diff)])))
#T=500, N=10: 0.0721471873130148
#T=500, N=100: 0.0567947864616883
#T=500, N=1000: 0.06080246985279 86
#T=1000, N=1000: 0.055513015028374
```

```
#SCENARIO WITH BB1 COPULA RANDOM ROTATION

#initializations
par1_mat=array(0,dim=c(d,d))
par2_mat=array(0,dim=c(d,d))
fam_mat=matrix(0,nrow=d,ncol=d)
random_fam=sample(c(7,17,27,37), 7, replace = TRUE)
fam_mat[lower.tri(fam_mat,diag=FALSE)]=random_fam
par1_mat=array(0,dim=c(d,d))
par2_mat=array(0,dim=c(d,d))
for (i in 1:d){
  for (j in 1:d){
    if (fam_mat[i,j]==7 || fam_mat[i,j]==17){
      par1_mat[i,j]=runif(1, 0, 7)
      par2_mat[i,j]=runif(1, 1, 7)
    }
    else if (fam_mat[i,j]==27 || fam_mat[i,j]==37){
      par1_mat[i,j]=runif(1, -7, 0)
      par2_mat[i,j]=runif(1, -7, -1)
    }
  }
}
rvm_r=RVineMatrix(Matrix=mat,family=fam_mat,par=par1_mat,par2=par2_mat)
rvine=array(NA, dim = c(T, d, N))
rvine_est=array(NA, dim=c(T, d, N))
tau=rvm_r$tau
gen_tau_diff=array(NA, dim = c(d, d, N))
lower_tau_diff=array(NA, dim = c(d, d, N))
upper_tau_diff=array(NA, dim = c(d, d, N))

#simulation
for (i in 1:N){
  rvine[,,i]=RVineSim(T,rvm_r)
}
#check missing values
sum(is.na(rvine))

#estimation
for (i in 1:N){
  rvm_est=RVineStructureSelect(rvine[,,i],progress=FALSE) #RVM = RVINE MATRIX

  #general tau difference
```

```
  rvine_est[,,i]=RVineSim(T,rvm_est) #RVINE EST=SAMPLE FROM RVM
  gen_tau_diff[,,i]=abs(TauMatrix(rvine[,,i])-TauMatrix(rvine_est[,,i]))

  #lower tau difference
  rvine_lower=matrix(nrow = 0, ncol = d)
  rvine_est_lower=matrix(nrow = 0, ncol = d)
  for (j in 1:(d-1)){
    for (k in (j+1):d){
      flg=rvine[,j,i]<=0.2 & rvine[,k,i]<=0.2
      rvine_lower=rbind(rvine_lower,rvine[flg,,i])
      flg_est=rvine_est[,j,i]<=0.2 & rvine_est[,k,i]<=0.2
      rvine_est_lower=rbind(rvine_est_lower,rvine_est[flg_est,,i])
    }
    if (dim(rvine_lower)[1]>1 & dim(rvine_est_lower)[1]>1){
      rvine_lower=unique(rvine_lower)
      rvine_est_lower=unique(rvine_est_lower)
      lower_tau_diff[,,i]=abs(TauMatrix(rvine_lower)-TauMatrix(rvine_est_lower))
    }
  }

  #upper tau difference
  rvine_upper=matrix(nrow = 0, ncol = d)
  rvine_est_upper=matrix(nrow = 0, ncol = d)
  for (j in 1:(d-1)){
    for (k in (j+1):d){
      flg=rvine[,j,i]>0.8 & rvine[,k,i]>0.8
      rvine_upper=rbind(rvine_upper,rvine[flg,,i])
      flg_est=rvine_est[,j,i]>0.8 & rvine_est[,k,i]>0.8
      rvine_est_upper=rbind(rvine_est_upper,rvine_est[flg_est,,i])
    }
    if (dim(rvine_upper)[1]>1 & dim(rvine_est_upper)[1]>1){
      rvine_lower=unique(rvine_lower)
      rvine_est_lower=unique(rvine_est_lower)
      upper_tau_diff[,,i]=abs(TauMatrix(rvine_upper)-TauMatrix(rvine_est_upper))
    }
  }
}

#display error
mean_tau_diff=apply(gen_tau_diff, c(1, 2), mean)
print(paste("mean generic tau difference 1",mean(mean_tau_diff[lower.tri(mean_tau_
    diff)])))
#T=500, N=10: 0.0104550434201737
#T=500, N=100: 0.0121030327321309
#T=500, N=1000: 0.01346073967774
#T=1000, N=1000: 0.0116576354653324

mean_lower_tau_diff=apply(lower_tau_diff, c(1, 2), mean)
print(paste("mean lower tau difference",mean(lower_tau_diff[lower.tri(lower_tau_
    diff)])))
#T=500, N=10: 0.02140831181761
#T=500, N=100: 0.020322939800837
#T=500, N=1000: 0.028608124109949
#T=1000, N=1000: 0.03018801994080 99

mean_upper_tau_diff=apply(upper_tau_diff, c(1, 2), mean)
print(paste("mean upper tau difference",mean(upper_tau_diff[lower.tri(upper_tau_
    diff)])))
#T=500, N=10: 0.05756924390847 21
#T=500, N=100: 0.03142717383300 79
#T=500, N=1000: 0.02308405214304 07
#T=1000, N=1000: 0.05004248864053 96
```

```
#SCENARIO WITH MIXED COPULAS MIXED TAUS TAKEN FROM CZADO

#initializations
par1_mat=array(0,dim=c(d,d))
par2_mat=array(0,dim=c(d,d))
fam_mat=matrix(0,nrow=d,ncol=d)
fam_mat=matrix(c(0,1,5,1,4,5,14, 0,0,1,5,14,1,4, 0,0,0,1,4,5,14,
                 0,0,0,0,14,1,4, 0,0,0,0,0,2,2, 0,0,0,0,0,0,2,
                 0,0,0,0,0,0,0),7,7)
tau=matrix(c(0,0.05,0.10,0.15,0.20,0.25,0.50, 0,0,0.10,0.15,0.20,0.30,0.55,
             0,0,0,0.15,0.20,0.35,0.60, 0,0,0,0,0.20,0.40,0.65,
             0,0,0,0,0,0.45,0.70, 0,0,0,0,0,0,0.70,
             0,0,0,0,0,0,0),7,7)
deg_free_t=3
for (i in 1:7){
  for (j in 1:7){
    if (i>j){
      par1_mat[i,j]=BiCopTau2Par(family=fam_mat[i,j],tau=tau[i,j])
      if (fam_mat[i,j]==2){
        par2_mat[i,j]=deg_free_t
        deg_free_t=deg_free_t+1
      }
    }
  }
}
rvm_r=RVineMatrix(Matrix=mat,family=fam_mat,par=par1_mat,par2=par2_mat)
rvine=array(NA, dim = c(T, d, N))
rvine_est=array(NA, dim=c(T, d, N))
gen_tau_diff=array(NA, dim = c(d, d, N))
lower_tau_diff=array(NA, dim = c(d, d, N))
upper_tau_diff=array(NA, dim = c(d, d, N))

#simulation and estimation
for (i in 1:N){
  rvine[,,i]=RVineSim(T,rvm_r)
  rvm_est=RVineStructureSelect(rvine[,,i],progress=FALSE) #RVM = RVINE MATRIX

  #general tau difference:
  #simulate data with true model, compute empirical tau from simulated data (A),
  #estimate the model, simulate data from the estimated model,
  #compute empirical tau on simulated data from estimated model (B),
  #compute A-B
  rvine_est[,,i]=RVineSim(T,rvm_est) #RVINE EST=SAMPLE FROM RVM
  gen_tau_diff[,,i]=abs(TauMatrix(rvine[,,i])-TauMatrix(rvine_est[,,i]))

  #lower tau difference
  rvine_lower=matrix(nrow = 0, ncol = d)
  rvine_est_lower=matrix(nrow = 0, ncol = d)
  for (j in 1:(d-1)){
    for (k in (j+1):d){
      flg=rvine[,j,i]<=0.2 & rvine[,k,i]<=0.2
      rvine_lower=rbind(rvine_lower,rvine[flg,,i])
      flg_est=rvine_est[,j,i]<=0.2 & rvine_est[,k,i]<=0.2
      rvine_est_lower=rbind(rvine_est_lower,rvine_est[flg_est,,i])
    }
```

```
    if (dim(rvine_lower)[1]>1 & dim(rvine_est_lower)[1]>1){
      rvine_lower=unique(rvine_lower)
      rvine_est_lower=unique(rvine_est_lower)
      lower_tau_diff[,,i]=abs(TauMatrix(rvine_lower)-TauMatrix(rvine_est_lower))
    }
  }

  #upper tau difference
  rvine_upper=matrix(nrow = 0, ncol = d)
  rvine_est_upper=matrix(nrow = 0, ncol = d)
  for (j in 1:(d-1)){
    for (k in (j+1):d){
      flg=rvine[,j,i]>0.8 & rvine[,k,i]>0.8
      rvine_upper=rbind(rvine_upper,rvine[flg,,i])
      flg_est=rvine_est[,j,i]>0.8 & rvine_est[,k,i]>0.8
      rvine_est_upper=rbind(rvine_est_upper,rvine_est[flg_est,,i])
    }
    if (dim(rvine_upper)[1]>1 & dim(rvine_est_upper)[1]>1){
      rvine_lower=unique(rvine_lower)
      rvine_est_lower=unique(rvine_est_lower)
      upper_tau_diff[,,i]=abs(TauMatrix(rvine_upper)-TauMatrix(rvine_est_upper))
    }
  }
}

#display error
mean_tau_diff=apply(gen_tau_diff, c(1, 2), mean)
print(paste("mean generic tau difference 1",mean(mean_tau_diff[lower.tri(mean_tau_
    diff)])))
#T=500, N=10: 0.0240433629163088
#T=500, N=100: 0.0217010478099055
#T=500, N=1000: 0.0202402946846073
#T=1000, N=1000: 0.0151380412793746

mean_lower_tau_diff=apply(lower_tau_diff, c(1, 2), mean)
print(paste("mean lower tau difference",mean(lower_tau_diff[lower.tri(lower_tau_
    diff)])))
#T=500, N=10: 0.0643451872350793
#T=500, N=100: 0.0574120432934252
#T=500, N=1000: 0.0584803049953999
#T=1000, N=1000: 0.0460839739232319

mean_upper_tau_diff=apply(upper_tau_diff, c(1, 2), mean)
print(paste("mean upper tau difference",mean(upper_tau_diff[lower.tri(upper_tau_
    diff)])))
#T=500, N=10: 0.0623787921797875
#T=500, N=100: 0.0632510552615448
#T=500, N=1000: 0.0668592147271203
#T=1000, N=1000: 0.0509377528813044

#study of anomalous cases
#desktop_path<- file.path(Sys.getenv("USERPROFILE"), "Desktop", "rvine_save.csv")
#rvine_save<- as.data.frame(as.table(rvine[,,c(292,514,533,736,745,989)]))
#write.csv(rvine_save, file =desktop_path, row.names = FALSE)
#read_df <- read.csv("rvine_save.csv")
#rvine_na<- array(read_df$Freq, dim = c(500, 7, 6))
#rvm_est of rvine[,,745] has got a NA in position (3,2) of tau matrix
#(same for rvine[,,533]), this implies that mean_tau_diff2 has got also NA,
#in particular in position (2,1) (3,2) (4,2) (5,2) (6,1))
```

```
#SCENARIO WITH EVERY POSSIBLE FAMILY RANDOMLY

#initializations
par1_mat=array(0,dim=c(d,d))
par2_mat=array(0,dim=c(d,d))
rvine=array(NA, dim = c(T, d, N))
rvine_est=array(NA, dim=c(T, d, N))
fam_mat=matrix(0,nrow=d,ncol=d)
gen_tau_diff=array(NA, dim = c(d, d, N))
lower_tau_diff=array(NA, dim = c(d, d, N))
upper_tau_diff=array(NA, dim = c(d, d, N))

#choose copula family and parameters
copula_families=c(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 13,
                  14, 16, 17, 18, 19, 20, 23, 24, 26, 27,
                  28, 29, 30, 33, 34, 36, 37, 38, 39, 40,
                  104, 114, 124, 134, 204, 214, 224, 234)
random_fam=sample(copula_families, dd, replace=TRUE)
fam_mat[lower.tri(fam_mat,diag=FALSE)]=random_fam
for (i in 1:d) {
  for (j in 1:d) {
    if (fam_mat[i, j] %in% c(1, 2)) {
      par1_mat[i, j] = runif(1, -1, 1)
      if (fam_mat[i, j] == 2) par2_mat[i, j] = runif(1, 2, 1e3)
    } else if (fam_mat[i, j] %in% c(3, 13)) {
      par1_mat[i, j] = runif(1, 0, 28)
    } else if (fam_mat[i, j] %in% c(4, 14)) {
      par1_mat[i, j] = runif(1, 1, 17)
    } else if (fam_mat[i, j] == 5) {
      prova=runif(1, -35, 35)
      while (prova==0){
        prova=runif(1, -35, 35)
      }
      par1_mat[i, j] = prova
    } else if (fam_mat[i, j] %in% c(6, 16)) {
      par1_mat[i, j] = runif(1, 1, 30)
    } else if (fam_mat[i, j] %in% c(7, 17)) {
      par1_mat[i, j] = runif(1, 0, 7)
      par2_mat[i, j] = runif(1, 1, 7)
    } else if (fam_mat[i, j] %in% c(8, 18)) {
      par1_mat[i, j] = runif(1, 1, 6)
      par2_mat[i, j] = runif(1, 1, 8)
    } else if (fam_mat[i, j] %in% c(9, 19)) {
      par1_mat[i, j] = runif(1, 1, 6)
      par2_mat[i, j] = runif(1, 0, 75)
    } else if (fam_mat[i, j] %in% c(10, 20)) {
      par1_mat[i, j] = runif(1, 1, 8)
      par2_mat[i, j] = runif(1, 1e-4, 1)
    } else if (fam_mat[i, j] %in% c(23, 33)) {
      par1_mat[i, j] = runif(1, -28, 0)
    } else if (fam_mat[i, j] %in% c(24, 34)) {
      par1_mat[i, j] = runif(1, -17, -1)
    } else if (fam_mat[i, j] %in% c(26, 36)) {
      par1_mat[i, j] = runif(1, -30, -1)
    } else if (fam_mat[i, j] %in% c(27, 37)) {
      par1_mat[i, j] = runif(1, -7, 0)
      par2_mat[i, j] = runif(1, -7, -1)
```

```
    } else if (fam_mat[i, j] %in% c(28, 38)) {
      par1_mat[i, j] = runif(1, -6, -1)
      par2_mat[i, j] = runif(1, -8, -1)
    } else if (fam_mat[i, j] %in% c(29, 39)) {
      par1_mat[i, j] = runif(1, -6, -1)
      par2_mat[i, j] = runif(1, -75, 0)
    } else if (fam_mat[i, j] %in% c(30, 40)) {
      par1_mat[i, j] = runif(1, -8, -1)
      par2_mat[i, j] = runif(1, -1, -1e-4)
    } else if (fam_mat[i, j] %in% c(41, 51)) {
      par1_mat[i, j] = runif(1, 0, 1e10)
    } else if (fam_mat[i, j] %in% c(61, 71)) {
      par1_mat[i, j] = runif(1, -1e3, 0)
    } else if (fam_mat[i, j] == 42) {
      b <- runif(1, -1, 1)
      limA <- (b - 3 - sqrt(9 + 6 * b - 3 * b^2))/2
      while (limA>1) {
        b <- runif(1, -1, 1)
        limA <- (b - 3 - sqrt(9 + 6 * b - 3 * b^2))/2
      }
      par1_mat[i, j] = runif(1,limA,1)
      par2_mat[i, j] = b
    } else if (fam_mat[i, j] %in% c(104, 114, 204, 214)) {
      par1_mat[i, j] = runif(1, 1, 1e3)
      par2_mat[i, j] = runif(1, 0, 1)
    } else if (fam_mat[i, j] %in% c(124, 134, 224, 234)) {
      par1_mat[i, j] = runif(1, -1e3, -1)
      par2_mat[i, j] = runif(1, 0, 1)
    }
  }
}
rvm_r=RVineMatrix(Matrix=mat,family=fam_mat,par=par1_mat,par2=par2_mat)
tau=rvm_r$tau

#simulation
for (i in 1:N){
  rvine[,,i]=RVineSim(T,rvm_r)
}
#check missing values
sum(is.na(rvine))

#estimation
for (i in 1:N){
  rvm_est=RVineStructureSelect(rvine[,,i],progress=FALSE) #RVM = RVINE MATRIX

  #general tau difference
  rvine_est[,,i]=RVineSim(T,rvm_est) #RVINE_EST = SAMPLE FROM RVM
  gen_tau_diff[,,i]=abs(TauMatrix(rvine[,,i])-TauMatrix(rvine_est[,,i]))

  #lower tau difference
  rvine_lower=matrix(nrow = 0, ncol = d)
  rvine_est_lower=matrix(nrow = 0, ncol = d)
  for (j in 1:(d-1)){
    for (k in (j+1):d){
      flg=rvine[,j,i]<=0.2 & rvine[,k,i]<=0.2
      rvine_lower=rbind(rvine_lower,rvine[flg,,i])
      flg_est=rvine_est[,j,i]<=0.2 & rvine_est[,k,i]<=0.2
      rvine_est_lower=rbind(rvine_est_lower,rvine_est[flg_est,,i])
    }
    if (dim(rvine_lower)[1]>1 & dim(rvine_est_lower)[1]>1){
      rvine_lower=unique(rvine_lower)
      rvine_est_lower=unique(rvine_est_lower)
      lower_tau_diff[,,i]=abs(TauMatrix(rvine_lower)-TauMatrix(rvine_est_lower))
    }
  }

  #upper tau difference
```

```
  rvine_upper=matrix(nrow = 0, ncol = d)
  rvine_est_upper=matrix(nrow = 0, ncol = d)
  for (j in 1:(d-1)){
    for (k in (j+1):d){
      flg=rvine[,j,i]>0.8 & rvine[,k,i]>0.8
      rvine_upper=rbind(rvine_upper,rvine[flg,,i])
      flg_est=rvine_est[,j,i]>0.8 & rvine_est[,k,i]>0.8
      rvine_est_upper=rbind(rvine_est_upper,rvine_est[flg_est,,i])
    }
    if (dim(rvine_upper)[1]>1 & dim(rvine_est_upper)[1]>1){
      rvine_lower=unique(rvine_lower)
      rvine_est_lower=unique(rvine_est_lower)
      upper_tau_diff[,,i]=abs(TauMatrix(rvine_upper)-TauMatrix(rvine_est_upper))
    }
  }
}

#display error
mean_tau_diff=apply(gen_tau_diff, c(1, 2), mean)
print(paste("mean generic tau difference",mean(mean_tau_diff[lower.tri(mean_tau_
    diff)])))
#T=500, N=10: 0.0238954098673538
#T=500, N=100: 0.0806560380616617
#T=500, N=1000: 0.0357822972386742
#T=1000, N=1000: 0.0174836106895621

mean_lower_tau_diff=apply(lower_tau_diff, c(1, 2), mean)
print(paste("mean lower tau difference",mean(lower_tau_diff[lower.tri(lower_tau_
    diff)])))
#T=500, N=10: 0.0341544492487925
#T=500, N=100: 0.0756173075316744
#T=500, N=1000: 0.0433710712192429
#T=1000, N=1000: 0.0297517949215619

mean_upper_tau_diff=apply(upper_tau_diff, c(1, 2), mean)
print(paste("mean upper tau difference",mean(upper_tau_diff[lower.tri(upper_tau_
    diff)])))
#T=500, N=10: 0.0413958695116491
#T=500, N=100: 0.0795217912376646
#T=500, N=1000: 0.0571722129818986
#T=1000, N=1000: 0.115660439599446
```

A.2 DJIA Vine Copula based Risk Measures forecast

```
#libraries
rm(list = ls())
set.seed(129)
library(VineCopula)
library(rugarch)
library(mvtsplot)
library(portvine)
library(stats)
library(rvinecopulib)
library(skewt)
library(magrittr)
library(ggplot2)
library(tidyverse)
data("dji30ret")
```

```
#RISK MEASURES COMPUTATION AND BACKTESTING
#plot djia constituents and index
data_inizio_crisi <- as.Date("2007-01-01")
dates=as.Date(row.names(dji30ret))
dji30ret_07_09 <- dji30ret[dates > data_inizio_crisi, ]

djia <- dji30ret_07_09 %>%
  rownames_to_column(var = "date") %>%
  mutate(date = as.Date(date)) %>%
  gather(key = "Constituent", value = "log_return", -date)
ggplot(data = djia, aes(x = date, y = log_return, color = Constituent)) +
  geom_line() +
  geom_smooth(method = "loess", se = FALSE, color = "black", linewidth = 0.5) +
  labs(title = "DJIA Constituents Log Returns 2007-2009",
       subtitle = "LOESS smoothing line in black to check weak stationarity",
       x = "Year",
       y = "Log Return")

index <- data.frame(data = rowSums(dji30ret_07_09)) %>%
  rownames_to_column(var = "date") %>%
  mutate(date = as.Date(date))
ggplot(data = index, aes(x = date, y = data)) +
  geom_line() +
  labs(title = "DJIA Index Log returns 2007-2009",
       x = "Year",
       y = "Log Return")



#train/test set construction
train <- dji30ret %>%
  rownames_to_column(var = "date") %>%
  mutate(date = as.Date(date)) %>%
  filter(date <= as.Date("2008-04-18")) %>% #2008-07-01***
  tail(1000) #1200***
#train=train[1:1000,1:length(train)] #***
test <- dji30ret %>%
  rownames_to_column(var = "date") %>%
  mutate(date = as.Date(date)) %>%
  tail(200) #200***
train_test=rbind(train,test)
cat("Dimensione train set:", nrow(train))
cat("Dimensione test set:", nrow(test))
cat("Dimensione dataset:", nrow(train_test))


# #gridsearch for arma(p,q)-garch(x,y) orders selection (commented due to high
    computational cost)
# max_order <- 3 #possible values of p, q, x, y (from 1 to 3)
# combinaz <- expand.grid(p=1:max_order,q=1:max_order,x=1:max_order,y=1:max_order)
# aic_mat <- array(NA, dim=c(dim(dji30ret)[2],dim(combinaz)[1]))
# bic_mat <- array(NA, dim=c(dim(dji30ret)[2],dim(combinaz)[1]))
# best_aic=array(NA,dim=dim(dji30ret)[2])
# best_bic=array(NA,dim=dim(dji30ret)[2])
# best_combinaz=array(NA,dim=dim(dji30ret)[2])
# for (i in 2:dim(train)[2]){
#   print(i)
#   for (j in 1:dim(combinaz)[1]) {
```

```
#     model <- ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(
    combinaz[j,3], combinaz[j,4])),
#                         mean.model = list(armaOrder = c(combinaz[j,1], combinaz[j
    ,2])),
#                         distribution.model="sstd")
#     fit_model <- ugarchfit(spec = model, data = train[,i], solver = "hybrid")
#     aic_mat[i,j] <- infocriteria(fit_model)[1]
#     bic_mat[i,j] <- infocriteria(fit_model)[2]
#   }
# }
#
# print(sum(is.na(aic_mat)))
# print(sum(is.na(bic_mat)))
# for (i in 1:dim(dji30ret)[2]){
#   best_aic[i]=which.min(aic_mat[i,])
#   best_bic[i]=which.min(bic_mat[i,])
#   if (best_aic[i]<best_bic[i]){
#     best_combinaz[i]=best_aic[i]
#   }
#   else {
#     best_combinaz[i]=best_bic[i]
#   }
# }
# print(best_aic)
# print(best_bic)
# print(best_combinaz)
# #ARMA-GARCH specification given the best orders
# spec_list <- lapply(1:30, function(i) {
#   aic_values <- combinaz[best_bic[i],]
#   spec <- default_garch_spec(ar = aic_values$p, ma = aic_values$q, arch = aic_
    values$x, garch = aic_values$y)
#   return(spec)
#   })
# names(spec_list)=names(dji30ret)
```

```
#unconditional var,es estimation using a rolling window
#marginal settings
marg_settings <- marginal_settings(
  train_size = 1000,
  refit_size = 50, #length of forecasting window of marginal models
  #individual_spec = spec_list #uncomment this line if ARMA-GARCH gridsearch is
      done
  default_spec = default_garch_spec()
)

#vine settings
uncond_vine_settings <- vine_settings(
  train_size = 1000,
  refit_size = 50, #how many times use the same copula
)
```

```
col_index <- setdiff(2:ncol(train_test), 8)
col_sampled <- c(2,3,4,5,6,7)
#col_sampled <- c(15,16,17,18,19,20) #***
asset_names=names(train_test)[col_sampled]
weights_portaf=setNames(c(rep(1/(length(asset_names)-2), length(asset_names) - 2),
    0, 0), asset_names)
realized=rowSums(test[,col_sampled[1:4]]/4)

#rolling estimation approach: fit ARMA-GARCH, take residuals, apply PIT, fit vine
uncond_risk_roll <- estimate_risk_roll(
  data = train_test[,col_sampled[1:4]],
  weights=weights_portaf[1:4],
  marginal_settings = marg_settings,
  vine_settings = uncond_vine_settings,
  alpha = c(0.05),
  risk_measures = c("VaR", "ES_mean"),
  n_samples = 500,
  trace = TRUE
)
df_risk=risk_estimates(uncond_risk_roll,exceeded = TRUE)

#ljung-box test for serial autocorrelation at different lags (H0: no
    autocorrelation)
marginals <- fitted_marginals(uncond_risk_roll)
vines=fitted_vines(uncond_risk_roll)
lag=c(1,5,10,15,20)
ljung_mat_std <- array(NA, dim=c(4,5,4)) #asset x lag x rolling_window
ljung_mat_sqr_std <- array(NA, dim=c(4,5,4)) #asset x lag x rolling_window
for (i in 1:4){
  for (j in 1:4){ #portfolio with 4 asset
    std_resid <- roll_residuals(marginals[[names(train_test)[col_sampled[j]]]],
        roll_num = i)
    sqr_std_resid <- roll_residuals(marginals[[names(train_test)[col_sampled[j]]]],
        roll_num = i)**2
    for (t in 1:5){
      ljung_mat_std[j,t,i]=Box.test(std_resid, lag = lag[t], type = "Lju")$p.value
      ljung_mat_sqr_std[j,t,i]=Box.test(sqr_std_resid, lag = lag[t], type = "Lju")$
          p.value
    }
  }
}
cat("Ljung-Box test p-values on residuals\n")
ljung_mat_std
cat("Null hypothesis rejected",sum(ljung_mat_std < 0.05),"times")
#roll window 1
#0.6380067 0.5463938 0.6029006 0.6344385 0.6718554
#0.8014358 0.9053650 0.1593253 0.3227831 0.5922897
#0.4676530 0.9880981 0.9515401 0.7557235 0.9220219
#0.5154959 0.9828692 0.4915308 0.7299531 0.8291133
#roll window 2
#0.7574107 0.4250166 0.5197523 0.5792903 0.5595016
#0.9649823 0.7189497 0.1273333 0.3255091 0.6412178
#0.3584866 0.9509725 0.7782875 0.6425933 0.8159217
#0.5889626 0.9750401 0.2606740 0.5787592 0.7132500
#roll window 3
#0.7303668 0.4845496 0.5032916 0.3530659 0.5161353
#0.9758730 0.6804913 0.2132273 0.3709773 0.6907675
#0.3565942 0.9610094 0.7203465 0.6127089 0.8014826
#0.6870309 0.9555553 0.4242265 0.6489335 0.8104965
#roll window 4
#0.4785026 0.8102419 0.6801353 0.6230811 0.5774870
#0.7419728 0.7221892 0.5306022 0.7773300 0.9391385
#0.4223057 0.8533465 0.9064178 0.7290525 0.7063390
#0.7907871 0.9558479 0.1554015 0.3116655 0.5337789
#Null hypothesis rejected 0 times
```

```
cat("Ljung-Box test pvalues on squared residuals\n")
ljung_mat_sqr_std
cat("Null hypothesis rejected",sum(ljung_mat_sqr_std < 0.05),"times")
#roll window 1
#0.3720645 0.3588128 0.5520004 0.7587870 0.8277244
#0.1763380 0.5868463 0.3012886 0.4141761 0.6076664
#0.3100602 0.7714612 0.9323075 0.9715605 0.9816345
#0.9376997 0.9005088 0.6384118 0.5807796 0.7340830
#roll window 2
#0.2901060 0.4123180 0.6495486 0.8579638 0.9231744
#0.1825294 0.6219297 0.3051888 0.4206254 0.6002521
#0.2898876 0.8078642 0.9686650 0.9793846 0.9912506
#0.8328646 0.8739790 0.7051593 0.6253267 0.7566121
#roll window 3
#0.2781621 0.3542505 0.4614083 0.7665455 0.9266283
#0.1737278 0.6032993 0.2955229 0.3834031 0.5551994
#0.3166856 0.7722314 0.8726943 0.9458312 0.9755234
#0.8350609 0.9535959 0.9202004 0.8726341 0.9193350
#roll window 4
#0.3840283 0.5761943 0.27725871 0.52219124 0.7443184
#0.1481619 0.6964143 0.06170925 0.09706642 0.1774252
#0.5043815 0.9213580 0.77727810 0.75624577 0.6348034
#0.8154017 0.7972667 0.81759214 0.87994837 0.8821706
#Null hypothesis rejected 0 times

#plot of results
x_geom_point=df_risk$row_num[df_risk$exceeded]
y_geom_point=df_risk$realized[df_risk$exceeded]
df_geom_point=data.frame(df_risk$realized[df_risk$exceeded])
df_risk %>%
  ggplot() +
  geom_line(aes(x = row_num, y = realized), col = "grey") +
  geom_line(aes(x = row_num, y = risk_est, col = factor(risk_measure))) +
  scale_fill_manual() +
  geom_point(aes(x = x_geom_point, y = y_geom_point),
             data = df_geom_point,
             col = "#db4f59")+
  labs(x = "trading day",
       y = "portfolio log returns",
       col = "Risk measure",
       title = "Unconditional risk measures, conf. level 5%",
       subtitle = "Exceedances in red, portfolio log return in grey")
```

```
#conditional var,es estimation using a rolling window
#vine settings
cond_vine_settings <- vine_settings(
  train_size = 1000,
  refit_size = 50, #how many times use the same copula
  family_set = c("parametric"),
  vine_type = "dvine")
```

```
#confidence level of the estimated quantile from the marginal distribution
pelcov=0.1
pelcov_str=as.character(pelcov)
#rolling estimation approach: fit ARMA-GARCH, take residuals, apply PIT, fit vine
cond_risk_roll <- estimate_risk_roll(
  data = train_test[col_sampled[1:6]],
  weights =weights_portaf ,
  marginal_settings = marg_settings,
  vine_settings = cond_vine_settings,
  alpha = c(0.05),
  risk_measures = c("VaR", "ES_mean"),
  n_samples = 500,
  cond_vars = c(names(train_test)[col_sampled[5]],names(train_test)[col_sampled
      [6]]),
  cond_u=pelcov, #value inside pelcov is associated to both conditioning variables
  ##QUANTILE STRATEGY=marginal market index on copula scale
  #(marginals for copulas are unif-->quantile is the confidence level itself),
  prior_resid_strategy = TRUE,
  #RESIDUAL STRATEGY=conditioning values for the forecast at time t
  #are the PIT of index I residual at time t-1
  trace = TRUE
)

#PLOT OF ALL STRATEGIES
df_cond=risk_estimates(cond_risk_roll,exceeded = TRUE)
ggplot(df_cond) +
  geom_line(data=df_cond, aes(x = row_num, y = realized), col = "grey") +
  geom_line(data = subset(df_cond, risk_measure == "VaR" & cond_u == "prior_resid")
      ,
            aes(x = row_num, y = risk_est,col="Var_prior_resid")) +
  geom_line(data = subset(df_cond, risk_measure == "VaR" & cond_u == pelcov_str),
            aes(x = row_num, y = risk_est,col=paste0("VaR_",pelcov_str))) +
  geom_line(data = subset(df_cond, risk_measure == "ES_mean" & cond_u == "prior_
      resid"),
            aes(x = row_num, y = risk_est,col="ES_mean_prior_resid")) +
  geom_line(data = subset(df_cond, risk_measure == "ES_mean" & cond_u == pelcov_str
      ),
            aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_str))) +
  labs(x = "trading day",
       y = "portfolio log returns",
       col = "Risk measure",
       title = "All conditional risk measures, conf. level 5%",
       subtitle = "Exceedances in red, portfolio realized log return in grey")+
  scale_fill_manual(
    name = "Risk Measure",
    labels = c(paste0("ES_mean_",pelcov_str),"ES_mean_prior_resid",paste0("VaR_",
        pelcov_str),"Var_prior_resid")
  )

#QUANTILE STRATEGY PLOT
ggplot(df_cond) +
  geom_line(aes(x = row_num, y = realized), col = "grey") +
  geom_line(data = subset(df_cond, risk_measure == "VaR" & cond_u == pelcov_str),
            aes(x = row_num, y = risk_est,col=paste0("VaR_",pelcov_str))) +
  geom_line(data = subset(df_cond, risk_measure == "ES_mean" & cond_u == pelcov_str
      ),
            aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_str))) +
  geom_line(data = subset(df_cond, cond_u == pelcov_str),
            aes(x = row_num, y =!!as.name(names(train_test)[col_sampled[5]]),
                col=paste0(names(train_test)[col_sampled[5]],"_",pelcov))) +
  geom_line(data = subset(df_cond, cond_u == pelcov_str),
            aes(x = row_num, y =!!as.name(names(train_test)[col_sampled[6]]),
            col=paste0(names(train_test)[col_sampled[6]],"_",pelcov))) +
  geom_point(data=subset(df_cond,exceeded== TRUE & cond_u == pelcov_str),
             aes(x = row_num, y=realized),col="#db4f59")+
  scale_fill_manual(
    name = "Risk Measure",
```

```
    labels = c(paste0(names(train_test)[col_sampled[6]],"_",pelcov),
               paste0(names(train_test)[col_sampled[5]],"_",pelcov),
               paste0("ES_mean_",pelcov_str),paste0("VaR_",pelcov_str)))+
  labs(x = "trading day",
       y = "portfolio log returns",
       col = "Risk measure",
       title = "Quantile based conditional risk measures, conf. level 5%",
       subtitle = "Exceedances in red, portfolio realized log return in grey")

#PRIOR RESIDUAL COPULA SCALE STRATEGY PLOT
#It should be noted that the conditional series based on the fitted
#residuals of the time unit before will most likely exaggerate
#sudden high volatility situations
ggplot(df_cond, aes(x = row_num)) +
  geom_line(aes(y = realized), col = "grey") +
  geom_line(data = subset(df_cond, risk_measure == "VaR" & cond_u == "prior_resid")
      , aes(y = risk_est,col="VaR_prior_resid")) +
  geom_line(data = subset(df_cond, risk_measure == "ES_mean" & cond_u == "prior_
      resid"), aes(y = risk_est,col="ES_mean_prior_resid")) +
  geom_point(data=subset(df_cond,exceeded== TRUE & cond_u == "prior_resid"),aes(y=
      realized),col="#db4f59")+
  labs(x = "trading day",
       y = "portfolio log returns",
       col = "Risk measure",
       title = "Prior Residual conditional risk measures, conf. level 5%",
       subtitle = "Exceedances in red, portfolio realized log return in grey")+
  scale_fill_manual(
    name = "Risk Measure",
    labels = c("ES_mean_prior_resid","VaR_prior_resid")
  )

#var, covar comparison
lowest_serie=df_cond$risk_est[df_cond$risk_measure=="VaR" & df_cond$cond_u==pelcov_
    str]
middle_serie=df_risk$risk_est[df_risk$risk_measure=="VaR"]
upper_serie=df_cond$risk_est[df_cond$risk_measure=="VaR" & df_cond$cond_u=="prior_
    resid"]
ggplot() +
  geom_line(data = subset(df_risk, risk_measure == "VaR"),
            aes(x = row_num, y = risk_est,col="Var")) +
  geom_line(data = subset(df_cond, risk_measure == "VaR" & cond_u == "prior_resid")
      ,
            aes(x = row_num, y = risk_est,col="Var_prior_resid")) +
  geom_line(data = subset(df_cond, risk_measure == "VaR" & cond_u == pelcov_str),
            aes(x = row_num, y = risk_est,col=paste0("VaR_",pelcov_str))) +
  labs(x = "trading day",
       y = "portfolio VaR",
       col = "Risk measure",
       title = "Comparison VaR & CoVaR, conf. level 5%")+
  scale_fill_manual(
    name = "Risk Measure",
    labels = c("VaR",paste0("VaR_",pelcov_str),"Var_prior_resid")
  )

#es, coes comparison
ggplot() +
  #geom_line(data=df_risk, aes(x = row_num, y = realized), col = "black") +
  #geom_line(data=df_cond, aes(x = row_num, y = realized), col = "yellow") +
  #geom_line(data=test, aes(x = (501:650), y =rowSums(test[,col_sampled[1:4]]/4)),
      col = "red") +
  geom_line(data = subset(df_risk, risk_measure == "ES_mean"),
            aes(x = row_num, y = risk_est,col="ES_mean")) +
  geom_line(data = subset(df_cond, risk_measure == "ES_mean" & cond_u == "prior_
      resid"),
            aes(x = row_num, y = risk_est,col="ES_mean_prior_resid")) +
  geom_line(data = subset(df_cond, risk_measure == "ES_mean" & cond_u == pelcov_str
      ),
```

```
            aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_str))) +
  #geom_point(data=subset(df_cond, exceeded== TRUE),
  #           aes(x = row_num,y=realized),col="#db4f59")+
  #geom_point(data=subset(df_cond,exceeded== TRUE & risk_measure == "VaR" & cond_u
      == "prior_resid" & realized>middle_serie),
  #           aes(x = row_num,y=realized),col="green3")+
  #geom_point(data=subset(df_risk,exceeded== TRUE & risk_measure == "VaR" &
      realized>=lowest_serie),
  #           aes(x = row_num,y=realized),col="yellow")+
  labs(x = "trading day",
       y = "portfolio VaR",
       col = "Risk measure",
       title = "Comparison ES & CoES, conf. level 5%")+
  scale_fill_manual(
    name = "Risk Measure",
    labels = c("ES_mean",paste0("ES_mean_",pelcov_str),"ES_mean_prior_resid")
  )
```

```
#backtesting
uncond_var=df_risk[df_risk$risk_measure=="VaR",2]
cond_var_quantile=df_cond[df_cond$risk_measure=="VaR" & df_cond$cond_u==pelcov_str
    ,2]
cond_var_residual=df_cond[df_cond$risk_measure=="VaR" & df_cond$cond_u=="prior_
    resid",2]
bckt_var_uncond=VaRTest(alpha = 0.05, actual=realized,
                        VaR=uncond_var, conf.level = 0.95) #0.05)#
bckt_var_quantile=VaRTest(alpha = 0.05, actual=realized,
                          VaR=cond_var_quantile, conf.level = 0.95)
bckt_var_residual=VaRTest(alpha = 0.05, actual=realized,
                          VaR=cond_var_residual, conf.level = 0.95)
cat("unconditional strategy for VaR estimation\n", str(bckt_var_uncond))
# $ expected.exceed: num 10
# $ actual.exceed  : num 24
# $ uc.H0          : chr "Correct Exceedances"
# $ uc.LRstat      : num 15.1
# $ uc.critical    : num 3.84
# $ uc.LRp         : num 0.000103
# $ uc.Decision    : chr "Reject H0"
# $ cc.H0          : chr "Correct Exceedances & Independent"
# $ cc.LRstat      : num 17.1
# $ cc.critical    : num 5.99
# $ cc.LRp         : num 0.000194
# $ cc.Decision    : chr "Reject H0"
cat("conditional quantile strategy for VaR estimation\n", str(bckt_var_quantile))
# $ expected.exceed: num 10
# $ actual.exceed  : num 7
# $ uc.H0          : chr "Correct Exceedances"
# $ uc.LRstat      : num 1.05
# $ uc.critical    : num 3.84
# $ uc.LRp         : num 0.305
# $ uc.Decision    : chr "Fail to Reject H0"
```

```
# $ cc.H0           : chr "Correct Exceedances & Independent"
# $ cc.LRstat       : num 1.56
# $ cc.critical     : num 5.99
# $ cc.LRp          : num 0.457
# $ cc.Decision     : chr "Fail to Reject H0"
cat("conditional prior residual strategy for VaR estimation\n", str(bckt_var_
    residual))
# $ expected.exceed: num 10
# $ actual.exceed  : num 50
# $ uc.H0           : chr "Correct Exceedances"
# $ uc.LRstat       : num 90
# $ uc.critical     : num 3.84
# $ uc.LRp          : num 0
# $ uc.Decision     : chr "Reject H0"
# $ cc.H0           : chr "Correct Exceedances & Independent"
# $ cc.LRstat       : num 110
# $ cc.critical     : num 5.99
# $ cc.LRp          : num 0
# $ cc.Decision     : chr "Reject H0"

uncond_es=df_risk[df_risk$risk_measure=="ES_mean",2]
cond_es_quantile=df_cond[df_cond$risk_measure=="ES_mean" & df_cond$cond_u==pelcov_
    str ,2]
cond_es_residual=df_cond[df_cond$risk_measure=="ES_mean" & df_cond$cond_u=="prior_
    resid",2]
bckt_es_uncond=ESTest(alpha = 0.05, actual=realized, ES=uncond_es,
                      VaR=uncond_var, conf.level = 0.95,boot = TRUE, n.boot = 1000)
bckt_es_quantile=ESTest(alpha = 0.05, realized, cond_es_quantile,
                        cond_var_quantile, conf.level = 0.95, boot = TRUE, n.boot =
                            1000)
bckt_es_residual=ESTest(alpha = 0.05, realized,cond_es_residual,
                        cond_var_residual, conf.level = 0.95, boot = TRUE, n.boot =
                            1000)
cat("unconditional strategy for ES estimation\n", str(bckt_es_uncond))
# $ expected.exceed: num 10
# $ actual.exceed  : int 24
# $ H1              : chr "Mean of Excess Violations of VaR is greater than zero"
# $ boot.p.value    : num 0.49
# $ p.value         : num 0.452
# $ Decision        : chr "Fail to Reject H0"
cat("conditional quantile strategy for ES estimation\n", str(bckt_es_quantile))
# $ expected.exceed: num 10
# $ actual.exceed  : int 7
# $ H1              : chr "Mean of Excess Violations of VaR is greater than zero"
# $ boot.p.value    : num 0.156
# $ p.value         : num 0.0762
# $ Decision        : chr "Fail to Reject H0"
cat("conditional prior residual strategy for ES estimation\n", str(bckt_es_residual
    ))
# $ expected.exceed: num 10
# $ actual.exceed  : int 50
# $ H1              : chr "Mean of Excess Violations of VaR is greater than zero"
# $ boot.p.value    : num 0.00208
# $ p.value         : num 0.000193
# $ Decision        : chr "Reject H0"



#***:
#these are the code-level changes that must be made to obtain the
#results that refer to the specific example presented in section 4.4
#(Omega_3={International Business Machines Corporation (IBM),
#Intel Corporation (INTC), Johnson & Johnson (JNJ), JPMorgan Chase & Co. (JPM)}
#each weighing 1/4, I_3={American International Group Inc. (AIG),
#The Coca-Cola Company (KO)}, training set with 1000 observations
#between 2003-09-24 and 2007-09-13, test set from 2008-04-21 to 2009-02-03)
```

A.3 DJIA Probability Equivalent Level Analysis

```
#libraries
rm(list = ls())
set.seed(129)
library(VineCopula)
library(rugarch)
library(mvtsplot)
library(portvine)
library(stats)
library(rvinecopulib)
library(skewt)
library(magrittr)
library(ggplot2)
library(tidyverse)
data("dji30ret")
```

```
#PROBABILITY EQUIVALENT LEVEL ANALYSIS
#initializations
train <- dji30ret %>%
  rownames_to_column(var = "date") %>%
  mutate(date = as.Date(date)) %>%
  filter(date < as.Date("2008-07-01")) %>%
  tail(750)
test <- dji30ret %>%
  rownames_to_column(var = "date") %>%
  mutate(date = as.Date(date)) %>%
  tail(150)
train_test=rbind(train,test)
cat("Dimensione train set:", nrow(train))
cat("Dimensione test set:", nrow(test))
cat("Dimensione dataset:", nrow(train_test))

marg_settings <- marginal_settings(
  train_size = 750,
  refit_size = 50, #length of forecasting window of marginal models
  default_spec = default_garch_spec()
)

uncond_vine_settings <- vine_settings(
  train_size = 750,
  refit_size =25, #how many times use the same copula
)

cond_vine_settings <- vine_settings(
  train_size = 750,
  refit_size = 25, #how many times use the same copula
  family_set = c("parametric"),
  vine_type = "dvine")

#definition of risk levels at which pelcov should be detected
risk_levels_v=c
    (0.01,0.015,0.02,0.025,0.03,0.035,0.04,0.045,0.05,0.055,0.06,0.065,0.07)
final_pel_uv=array(NA,dim=c(4,length(risk_levels_v)))
```

```
col_index <- setdiff(2:ncol(train_test), 8)
col_sampled <- c(2,3,4,5,6,7) #portfolio 1)
#col_sampled <- c(9,10,11,12,13,14) #portfolio 6)
#col_sampled <- c(15,16,17,18,19,20) #portfolio 7)
asset_names=names(train_test)[col_sampled]
weights_portaf=setNames(c(rep(1/(length(asset_names)-2), length(asset_names) - 2),
    0, 0), asset_names)
realized=rowSums(test[,col_sampled[1:4]]/4)
pel_plots <- vector("list", length(risk_levels_v)) #vector containing all plots to
    see graphically PELs

for (v in 1:length(risk_levels_v)){
  #unconditional rolling window estimation approach: fit ARMA-GARCH, take residuals
      , apply PIT, fit vine
  uncond_risk_roll <- estimate_risk_roll(
    data = train_test[,col_sampled[1:4]],
    weights=weights_portaf[1:4],
    marginal_settings = marg_settings,
    vine_settings = uncond_vine_settings,
    alpha = c(risk_levels_v[v]),
    risk_measures = c("VaR", "ES_mean"),
    n_samples = 500,
    trace = TRUE
  )
  df_risk=risk_estimates(uncond_risk_roll,exceeded = TRUE)
  uncond_var=df_risk[df_risk$risk_measure=="VaR",2]
  uncond_es=df_risk[df_risk$risk_measure=="ES_mean",2]

  #ONE CONDITIONAL ASSET
  #find the correct interval to search for pel
  pel_trial= seq(0.01, 0.91, by = 0.1)
  intersections_trial=array(0,dim=c(4,length(pel_trial)))
  pel_trial_str=as.character(pel_trial)
  df_cond_list_trial=vector("list", length = length(pel_trial))

  #one conditional asset rolling window estimation approach: fit ARMA-GARCH, take
      residuals, apply PIT, fit vine
  for (i in 1:length(pel_trial)){
    cond_risk_roll <- estimate_risk_roll(
      data = train_test[col_sampled[1:5]],
      weights =weights_portaf[1:5],
      marginal_settings = marg_settings,
      vine_settings = cond_vine_settings,
      alpha = c(risk_levels_v[v]),
      risk_measures = c("VaR", "ES_mean"),
      n_samples = 500,
      cond_vars = c(names(train_test)[col_sampled[5]]),
      cond_u=pel_trial[i],
      prior_resid_strategy = TRUE,
      trace = TRUE
    )
    df_cond_list_trial[[i]]=risk_estimates(cond_risk_roll,exceeded = TRUE)
  }

  #search intersections for pelcov e peloces
  dfs_var_trial<- list()
  dfs_es_trial <- list()
  for (i in 1:10) {
    dfs_var_trial[[i]] <- subset(df_cond_list_trial[[i]], risk_measure == "VaR" &
        cond_u == pel_trial_str[i])
    dfs_es_trial[[i]] <- subset(df_cond_list_trial[[i]], risk_measure == "ES_mean"
        & cond_u == pel_trial_str[i])
    diff_series1 <- dfs_var_trial[[i]][,2] - uncond_var
    diff_series2 <- dfs_es_trial[[i]][,2] - uncond_es
    for (j in 2:length(diff_series1)) {
      if (diff_series1[j - 1] * diff_series1[j] < 0) {
        intersections_trial[1,i] <- intersections_trial[1,i] + 1
```

```
    }
    if (diff_series2[j - 1] * diff_series2[j] < 0) {
      intersections_trial[2,i] <- intersections_trial[2,i] + 1
    }
  }
}

#take 10 equally spaced points in the interval where intersections with pelcov
#and pelcoes are !=0 (minimum index associated with the first non-zero value
#for both pelcov and pelcoes is taken so that the for loop only has to be
#executed once in subsequent rows and does not increase too much computational
    cost)
min_intersect=min(head(which(intersections_trial[1,]!=0),1), head(which(
    intersections_trial[2,]!=0),1))
max_intersect=max(tail(which(intersections_trial[1,]!=0),1), tail(which(
    intersections_trial[2,]!=0),1))
pelcov_1d= round(seq(pel_trial[min_intersect], pel_trial[max_intersect], length =
     10),2) #0.1, 0.5, by = 0.05***
pelcov_1d_str=as.character(pelcov_1d)
df_cond_list_1d=vector("list", length = length(pelcov_1d))
intersections=array(0,dim=c(4,10)) #dim=c(4,9)***

#one conditional asset rolling window estimation approach: fit ARMA-GARCH, take
    residuals, apply PIT, fit vine
for (i in 1:length(pelcov_1d)){
  cond_risk_roll <- estimate_risk_roll(
    data = train_test[col_sampled[1:5]],
    weights =weights_portaf[1:5],
    marginal_settings = marg_settings,
    vine_settings = cond_vine_settings,
    alpha = c(risk_levels_v[v]),
    risk_measures = c("VaR", "ES_mean"),
    n_samples = 500,
    cond_vars = c(names(train_test)[col_sampled[5]]),
    cond_u=pelcov_1d[i],
    prior_resid_strategy = TRUE,
    trace = TRUE
  )
  df_cond_list_1d[[i]]=risk_estimates(cond_risk_roll,exceeded = TRUE)
}

#search intersections for pelcov e peloces
dfs_var_1d<- list()
dfs_es_1d <- list()
for (i in 1:10) { #1:9***
  dfs_var_1d[[i]] <- subset(df_cond_list_1d[[i]], risk_measure == "VaR" & cond_u
      == pelcov_1d_str[i])
  dfs_es_1d[[i]] <- subset(df_cond_list_1d[[i]], risk_measure == "ES_mean" & cond
      _u == pelcov_1d_str[i])
  diff_series1 <- dfs_var_1d[[i]][,2] - uncond_var
  diff_series2 <- dfs_es_1d[[i]][,2] - uncond_es
  for (j in 2:length(diff_series1)) {
    if (diff_series1[j - 1] * diff_series1[j] < 0) {
      intersections[1,i] <- intersections[1,i] + 1
    }
    if (diff_series2[j - 1] * diff_series2[j] < 0) {
      intersections[2,i] <- intersections[2,i] + 1
    }

  }
}

#stores graphical results
#pelcov
first_elements_var_1d <- sapply(dfs_var_1d, function(df) df[1, "risk_est"])
order_index_var_1d <- order(first_elements_var_1d)
```

```
colors <- c("#1f78b4", "#33a02c", "#e31a1c", "#6a3d9a", "#a6cee3", "#b2df8a", "#
    fdbf6f", "#cab2d6", "#fb9a99", "gold") #remove gold***
legend_data_var_1d <- data.frame(
  labels = c(paste0("Var_", pelcov_1d_str[order_index_var_1d])),
  colors = colors[order_index_var_1d]
)
plot<- ggplot() +
  geom_line(data = subset(df_risk, risk_measure == "VaR"),
            aes(x = row_num, y = risk_est), color="black", linewidth=0.75)+
  geom_line(data = subset(df_cond_list_1d[[1]], risk_measure == "VaR" & cond_u ==
      pelcov_1d_str[1]),
            aes(x = row_num, y = risk_est,col=paste0("Var_",pelcov_1d_str[1])))+
  geom_line(data = subset(df_cond_list_1d[[2]], risk_measure == "VaR" & cond_u ==
      pelcov_1d_str[2]),
            aes(x = row_num, y = risk_est,col=paste0("Var_",pelcov_1d_str[2])))+
  geom_line(data = subset(df_cond_list_1d[[3]], risk_measure == "VaR" & cond_u ==
      pelcov_1d_str[3]),
            aes(x = row_num, y = risk_est,col=paste0("Var_",pelcov_1d_str[3])))+
  geom_line(data = subset(df_cond_list_1d[[4]], risk_measure == "VaR" & cond_u ==
      pelcov_1d_str[4]),
            aes(x = row_num, y = risk_est,col=paste0("Var_",pelcov_1d_str[4])))+
  geom_line(data = subset(df_cond_list_1d[[5]], risk_measure == "VaR" & cond_u ==
      pelcov_1d_str[5]),
            aes(x = row_num, y = risk_est,col=paste0("Var_",pelcov_1d_str[5])))+
  geom_line(data = subset(df_cond_list_1d[[6]], risk_measure == "VaR" & cond_u ==
      pelcov_1d_str[6]),
            aes(x = row_num, y = risk_est,col=paste0("Var_",pelcov_1d_str[6])))+
  geom_line(data = subset(df_cond_list_1d[[7]], risk_measure == "VaR" & cond_u ==
      pelcov_1d_str[7]),
            aes(x = row_num, y = risk_est,col=paste0("Var_",pelcov_1d_str[7])))+
  geom_line(data = subset(df_cond_list_1d[[8]], risk_measure == "VaR" & cond_u ==
      pelcov_1d_str[8]),
            aes(x = row_num, y = risk_est,col=paste0("Var_",pelcov_1d_str[8])))+
  geom_line(data = subset(df_cond_list_1d[[9]], risk_measure == "VaR" & cond_u ==
      pelcov_1d_str[9]),
            aes(x = row_num, y = risk_est,col=paste0("Var_",pelcov_1d_str[9])))+
  geom_line(data = subset(df_cond_list_1d[[10]], risk_measure == "VaR" & cond_u
      == pelcov_1d_str[10]),
            aes(x = row_num, y = risk_est,col=paste0("Var_",pelcov_1d_str[10])))+
              #comment these 2 lines***
  labs(x = "trading␣day",
       y = "portfolio␣VaR",
       col = "Risk␣measure",
       title = "Pelcov␣research␣graphically,␣1␣conditional␣asset",
       subtitle = paste0("Unconditional␣VaR␣in␣black,␣VaR␣conf.␣level␣",risk_
           levels_v[v]*100, "%"))+
  scale_color_manual(name = "Risk␣Measure", values = legend_data_var_1d$colors,
      labels = legend_data_var_1d$labels)
pel_plots[[v]][[1]] <- plot
#pelcoes
first_elements_es_1d <- sapply(dfs_es_1d, function(df) df[1, "risk_est"])
order_index_es_1d <- order(first_elements_es_1d)
legend_data_es_1d <- data.frame(
  labels = c(paste0("ES_mean_", pelcov_1d_str[order_index_es_1d])),
  colors = colors[order_index_es_1d]
)
plot<- ggplot() +
  geom_line(data = subset(df_risk, risk_measure == "ES_mean"),
            aes(x = row_num, y = risk_est), color="black", linewidth=0.75)+
  geom_line(data = subset(df_cond_list_1d[[1]], risk_measure == "ES_mean" & cond_
      u == pelcov_1d_str[1]),
            aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_1d_str[1])
                ))+
  geom_line(data = subset(df_cond_list_1d[[2]], risk_measure == "ES_mean" & cond_
      u == pelcov_1d_str[2]),
            aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_1d_str[2])
                ))+
```

```
  geom_line(data = subset(df_cond_list_1d[[3]], risk_measure == "ES_mean" & cond_
      u == pelcov_1d_str[3]),
            aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_1d_str[3])
                ))+
  geom_line(data = subset(df_cond_list_1d[[4]], risk_measure == "ES_mean" & cond_
      u == pelcov_1d_str[4]),
            aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_1d_str[4])
                ))+
  geom_line(data = subset(df_cond_list_1d[[5]], risk_measure == "ES_mean" & cond_
      u == pelcov_1d_str[5]),
            aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_1d_str[5])
                ))+
  geom_line(data = subset(df_cond_list_1d[[6]], risk_measure == "ES_mean" & cond_
      u == pelcov_1d_str[6]),
            aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_1d_str[6])
                ))+
  geom_line(data = subset(df_cond_list_1d[[7]], risk_measure == "ES_mean" & cond_
      u == pelcov_1d_str[7]),
            aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_1d_str[7])
                ))+
  geom_line(data = subset(df_cond_list_1d[[8]], risk_measure == "ES_mean" & cond_
      u == pelcov_1d_str[8]),
            aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_1d_str[8])
                ))+
  geom_line(data = subset(df_cond_list_1d[[9]], risk_measure == "ES_mean" & cond_
      u == pelcov_1d_str[9]),
            aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_1d_str[9])
                ))+
  geom_line(data = subset(df_cond_list_1d[[10]], risk_measure == "ES_mean" & cond
      _u == pelcov_1d_str[10]),
            aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_1d_str
                [10])))+ #comment these 2 lines***
  labs(x = "trading␣day",
       y = "portfolio␣VaR",
       col = "Risk␣measure",
       title = "Pelcoes␣research␣graphically,␣1␣conditional␣asset",
       subtitle = paste0("Unconditional␣ES␣in␣black,␣ES␣conf.␣level␣",risk_levels
           _v[v]*100, "%"))+
  scale_color_manual(name = "Risk␣Measure", values = legend_data_es_1d$colors,
      labels = legend_data_es_1d$labels)
pel_plots[[v]][[2]] <- plot



#TWO CONDITIONAL ASSETS
#find the correct interval to search for pel
#two conditional assets rolling window estimation approach: fit ARMA-GARCH, take
    residuals, apply PIT, fit vine
for (i in 1:length(pel_trial)){
  cond_risk_roll <- estimate_risk_roll(
    data = train_test[col_sampled],
    weights =weights_portaf,
    marginal_settings = marg_settings,
    vine_settings = cond_vine_settings,
    alpha = c(risk_levels_v[v]),
    risk_measures = c("VaR", "ES_mean"),
    n_samples = 500,
    cond_vars = c(names(train_test)[col_sampled[5]],names(train_test)[col_sampled
        [6]]),
    cond_u=pel_trial[i],
    prior_resid_strategy = TRUE,
    trace = TRUE
  )
  df_cond_list_trial[[i]]=risk_estimates(cond_risk_roll,exceeded = TRUE)
}

#search intersections for pelcov e pelcoes
```

```
for (i in 1:10) {
  dfs_var_trial[[i]] <- subset(df_cond_list_trial[[i]], risk_measure == "VaR" &
      cond_u == pel_trial_str[i])
  dfs_es_trial[[i]] <- subset(df_cond_list_trial[[i]], risk_measure == "ES_mean"
      & cond_u == pel_trial_str[i])
  diff_series1 <- dfs_var_trial[[i]][,2] - uncond_var
  diff_series2 <- dfs_es_trial[[i]][,2] - uncond_es
  for (j in 2:length(diff_series1)) {
    if (diff_series1[j - 1] * diff_series1[j] < 0) {
      intersections_trial[3,i] <- intersections_trial[3,i] + 1
    }
    if (diff_series2[j - 1] * diff_series2[j] < 0) {
      intersections_trial[4,i] <- intersections_trial[4,i] + 1
    }
  }
}

#take 10 equally spaced points in the interval where intersections with pelcov
#and pelcoes are !=0 (minimum index associated with the first non-zero value
#for both pelcov and pelcoes is taken so that the for loop only has to be
#executed once in subsequent rows and does not increase too much computational
    cost)
min_intersect=min(head(which(intersections_trial[3,]!=0),1), head(which(
    intersections_trial[4,]!=0),1))
max_intersect=max(tail(which(intersections_trial[3,]!=0),1), tail(which(
    intersections_trial[4,]!=0),1))
pelcov_2d_a= round(seq(pel_trial[min_intersect], pel_trial[max_intersect], length
    = 10),2) #0.1, 0.5, by = 0.05***
pelcov_2d_a_str=as.character(pelcov_2d_a)
df_cond_list_2d_a=vector("list", length = length(pelcov_2d_a))

#two conditional assets rolling window estimation approach: fit ARMA-GARCH, take
    residuals, apply PIT, fit vine
for (i in 1:length(pelcov_2d_a)){
  cond_risk_roll <- estimate_risk_roll(
    data = train_test[col_sampled],
    weights =weights_portaf,
    marginal_settings = marg_settings,
    vine_settings = cond_vine_settings,
    alpha = c(risk_levels_v[v]),
    risk_measures = c("VaR", "ES_mean"),
    n_samples = 500,
    cond_vars = c(names(train_test)[col_sampled[5]],names(train_test)[col_sampled
        [6]]),
    cond_u=pelcov_2d_a[i],
    prior_resid_strategy = TRUE,
    trace = TRUE
  )
  df_cond_list_2d_a[[i]]=risk_estimates(cond_risk_roll,exceeded = TRUE)
}

#search intersections for pelcov e pelcoes
dfs_var_2d_a<- list()
dfs_es_2d_a <- list()
for (i in 1:10) { #1:9***
  dfs_var_2d_a[[i]] <- subset(df_cond_list_2d_a[[i]], risk_measure == "VaR" &
      cond_u == pelcov_2d_a_str[i])
  dfs_es_2d_a[[i]] <- subset(df_cond_list_2d_a[[i]], risk_measure == "ES_mean" &
      cond_u == pelcov_2d_a_str[i])
  diff_series1 <- dfs_var_2d_a[[i]][,2] - uncond_var
  diff_series2 <- dfs_es_2d_a[[i]][,2] - uncond_es
  for (j in 2:length(diff_series1)) {
    if (diff_series1[j - 1] * diff_series1[j] < 0) {
      intersections[3,i] <- intersections[3,i] + 1
    }
    if (diff_series2[j - 1] * diff_series2[j] < 0) {
      intersections[4,i] <- intersections[4,i] + 1
```

```
    }
  }
}

#stores graphical results
#pelcov
first_elements_var_2d_a <- sapply(dfs_var_2d_a, function(df) df[1, "risk_est"])
order_index_var_2d_a <- order(first_elements_var_2d_a)
legend_data_var_2d_a <- data.frame(
  labels = c(paste0("Var_", pelcov_2d_a_str[order_index_var_2d_a])),
  colors = colors[order_index_var_2d_a]
)
plot<- ggplot() +
  geom_line(data = subset(df_risk, risk_measure == "VaR"),
            aes(x = row_num, y = risk_est), color="black", linewidth=0.75)+
  geom_line(data = subset(df_cond_list_2d_a[[1]], risk_measure == "VaR" & cond_u
      == pelcov_2d_a_str[1]),
            aes(x = row_num, y = risk_est,col=paste0("Var_",pelcov_2d_a_str[1])))
              +
  geom_line(data = subset(df_cond_list_2d_a[[2]], risk_measure == "VaR" & cond_u
      == pelcov_2d_a_str[2]),
            aes(x = row_num, y = risk_est,col=paste0("Var_",pelcov_2d_a_str[2])))
              +
  geom_line(data = subset(df_cond_list_2d_a[[3]], risk_measure == "VaR" & cond_u
      == pelcov_2d_a_str[3]),
            aes(x = row_num, y = risk_est,col=paste0("Var_",pelcov_2d_a_str[3])))
              +
  geom_line(data = subset(df_cond_list_2d_a[[4]], risk_measure == "VaR" & cond_u
      == pelcov_2d_a_str[4]),
            aes(x = row_num, y = risk_est,col=paste0("Var_",pelcov_2d_a_str[4])))
              +
  geom_line(data = subset(df_cond_list_2d_a[[5]], risk_measure == "VaR" & cond_u
      == pelcov_2d_a_str[5]),
            aes(x = row_num, y = risk_est,col=paste0("Var_",pelcov_2d_a_str[5])))
              +
  geom_line(data = subset(df_cond_list_2d_a[[6]], risk_measure == "VaR" & cond_u
      == pelcov_2d_a_str[6]),
            aes(x = row_num, y = risk_est,col=paste0("Var_",pelcov_2d_a_str[6])))
              +
  geom_line(data = subset(df_cond_list_2d_a[[7]], risk_measure == "VaR" & cond_u
      == pelcov_2d_a_str[7]),
            aes(x = row_num, y = risk_est,col=paste0("Var_",pelcov_2d_a_str[7])))
              +
  geom_line(data = subset(df_cond_list_2d_a[[8]], risk_measure == "VaR" & cond_u
      == pelcov_2d_a_str[8]),
            aes(x = row_num, y = risk_est,col=paste0("Var_",pelcov_2d_a_str[8])))
              +
  geom_line(data = subset(df_cond_list_2d_a[[9]], risk_measure == "VaR" & cond_u
      == pelcov_2d_a_str[9]),
            aes(x = row_num, y = risk_est,col=paste0("Var_",pelcov_2d_a_str[9])))
              +
  geom_line(data = subset(df_cond_list_2d_a[[10]], risk_measure == "VaR" & cond_u
       == pelcov_2d_a_str[10]),
            aes(x = row_num, y = risk_est,col=paste0("Var_",pelcov_2d_a_str[10]))
              )+ #comment these 2 lines***
  labs(x = "trading␣day",
       y = "portfolio␣VaR",
       col = "Risk␣measure",
       title = "Pelcov␣research␣graphically,␣2␣conditional␣assets␣with␣same␣value
           ",
       subtitle = paste0("Unconditional␣VaR␣in␣black,␣VaR␣conf.␣level␣",risk_
           levels_v[v]*100, "%"))+
  scale_color_manual(name = "Risk␣Measure", values = legend_data_var_2d_a$colors,
       labels = legend_data_var_2d_a$labels)
pel_plots[[v]][[3]] <- plot
#pelcoes
first_elements_es_2d_a <- sapply(dfs_es_2d_a, function(df) df[1, "risk_est"])
```

```
  order_index_es_2d_a <- order(first_elements_es_2d_a)
  legend_data_es_2d_a <- data.frame(
    labels = c(paste0("ES_mean_", pelcov_2d_a_str[order_index_es_2d_a])),
    colors = colors[order_index_es_2d_a]
  )
  plot<- ggplot() +
    geom_line(data = subset(df_risk, risk_measure == "ES_mean"),
              aes(x = row_num, y = risk_est), color="black", linewidth=0.75)+
    geom_line(data = subset(df_cond_list_2d_a[[1]], risk_measure == "ES_mean" &
        cond_u == pelcov_2d_a_str[1]),
              aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_2d_a_str
                  [1])))+
    geom_line(data = subset(df_cond_list_2d_a[[2]], risk_measure == "ES_mean" &
        cond_u == pelcov_2d_a_str[2]),
              aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_2d_a_str
                  [2])))+
    geom_line(data = subset(df_cond_list_2d_a[[3]], risk_measure == "ES_mean" &
        cond_u == pelcov_2d_a_str[3]),
              aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_2d_a_str
                  [3])))+
    geom_line(data = subset(df_cond_list_2d_a[[4]], risk_measure == "ES_mean" &
        cond_u == pelcov_2d_a_str[4]),
              aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_2d_a_str
                  [4])))+
    geom_line(data = subset(df_cond_list_2d_a[[5]], risk_measure == "ES_mean" &
        cond_u == pelcov_2d_a_str[5]),
              aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_2d_a_str
                  [5])))+
    geom_line(data = subset(df_cond_list_2d_a[[6]], risk_measure == "ES_mean" &
        cond_u == pelcov_2d_a_str[6]),
              aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_2d_a_str
                  [6])))+
    geom_line(data = subset(df_cond_list_2d_a[[7]], risk_measure == "ES_mean" &
        cond_u == pelcov_2d_a_str[7]),
              aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_2d_a_str
                  [7])))+
    geom_line(data = subset(df_cond_list_2d_a[[8]], risk_measure == "ES_mean" &
        cond_u == pelcov_2d_a_str[8]),
              aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_2d_a_str
                  [8])))+
    geom_line(data = subset(df_cond_list_2d_a[[9]], risk_measure == "ES_mean" &
        cond_u == pelcov_2d_a_str[9]),
              aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_2d_a_str
                  [9])))+
    geom_line(data = subset(df_cond_list_2d_a[[10]], risk_measure == "ES_mean" &
        cond_u == pelcov_2d_a_str[10]),
              aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_2d_a_str
                  [10])))+ #comment these 2 lines***
    labs(x = "trading day",
         y = "portfolio VaR",
         col = "Risk measure",
         title = "Pelcoes research graphically, 2 conditional assets with same
             value",
         subtitle = paste0("Unconditional ES in black, ES conf. level ",risk_levels
             _v[v]*100, "%"))+
    scale_color_manual(name = "Risk Measure", values = legend_data_es_2d_a$colors,
        labels = legend_data_es_2d_a$labels)
  pel_plots[[v]][[4]] <- plot



  #pel computation at confidence level v
  for (u in 1:4){
    final_pel_uv[u,v]=sum(pelcov_1d*intersections[u,])/sum(intersections[u,])
  }
}
```

```
#display pelcov and pelcoes
#titles1 <- c("1D PELCoV", "2D PELCoV", "1D PELCoES", "2D PELCoES") #uncomment if
    you want more verbose graphs
titles2 <- lapply(1:4, function(i) {asset_names[1:4]})
titles3 <- lapply(1:4, function(i) {
  if (i%%2==1) {asset_names[5]}
  else {asset_names[5:6]}})
interp <- lapply(1:nrow(final_pel_uv), function(i) {
  #compute polynomial regression of degree 3
  df <- data.frame(risk_levels_v, final_pel_uv[i, ])
  fit <- lm(final_pel_uv[i, ] ~ poly(risk_levels_v, degree = 3), data = df)
  df$fit <- predict(fit, newdata = df)

  ggplot(df, aes(x = risk_levels_v, y = final_pel_uv[i, ])) +
    geom_point() +
    geom_line(aes(y = fit), color = "blue", linewidth= 0.8) +
    #uncomment if you want more verbose graphs
    labs(#title = bquote(Omega ~ "={" * .(paste(titles2[[i]], collapse = ", ")) *
        "}, I={" * .(paste(titles3[[i]], collapse = ", ")) *"}"), #.(titles1[i]) *
        ": " *
         #subtitle = "Polynomial regression line of degree 3 in blue",
         x = "conf. level v",
         y = "PEL u_v")
})
interp
final_pel_uv
# 1) AA AXP BA BAC C CAT (last 2 are conditional assets)
#0.1315929 0.1660231 0.1778372 0.1778481 0.1649654 0.1872483 0.1906231 0.2068702
    0.2038284 0.2165409 0.2177612 0.2236431 0.2227600
#0.1424242 0.1730055 0.1614634 0.1558475 0.1446237 0.1641935 0.1593125 0.1647778
    0.1740670 0.1628826 0.1559259 0.1676025 0.1716779
#0.1991707 0.2256502 0.2386801 0.1976724 0.1855499 0.2674615 0.2794323 0.2229091
    0.2261692 0.2394510 0.2390196 0.1909870 0.1939024
#0.1838813 0.1942647 0.2005169 0.1598492 0.1554198 0.2227960 0.2114919 0.1788485
    0.1850932 0.1830472 0.1764336 0.1351351 0.1438542
# 6) DD DIS GE GM HD HPQ (last 2 are conditional assets)
#0.2625907 0.2513230 0.2600990 0.2710265 0.2724731 0.2582703 0.2813208 0.2835163
    0.2676515 0.2819536 0.2853943 0.2952299 0.2984783
#0.2869123 0.2855908 0.2680608 0.2736190 0.2642260 0.2459919 0.2558029 0.2801931
    0.2413611 0.2387772 0.2598615 0.2600721 0.2849837
#0.4172696 0.3310995 0.2238636 0.2810654 0.3400000 0.3877723 0.3779633 0.2991948
    0.2538919 0.2716556 0.3130928 0.2772805 0.4365810
#0.3790141 0.3332824 0.2434351 0.2575058 0.3163871 0.3429487 0.3007859 0.2758795
    0.2167840 0.2176060 0.2589209 0.2385388 0.3943970
# 7) IBM INTC JNJ JPM AIG KO (last 2 are conditional assets)
#0.1547107 0.1679472 0.1942857 0.1891977 0.1987059 0.2121538 0.2196500 0.2337767
    0.2199415 0.2363700 0.2447399 0.2461094 0.2481570
#0.1363043 0.1498187 0.1664935 0.1620625 0.1524918 0.1765347 0.1767130 0.1716349
    0.1822581 0.1672222 0.1815167 0.1832440 0.1900000
#0.1583371 0.2443167 0.2674737 0.2266436 0.2326786 0.2451887 0.2524026 0.2596215
    0.2544361 0.2680000 0.2384028 0.2374672 0.2427226
#0.1284085 0.2119363 0.2175200 0.1891223 0.1790966 0.2027203 0.1973913 0.1940000
    0.2088222 0.1983946 0.1701660 0.1741622 0.1762500
# 8) SPY 1329 ETFMIB GDAXIEX GCJ4 CCK4 (last 2 are conditional assets)
#0.4330632 0.4506940 0.4584478 0.4824329 0.4600778 0.4607028 0.4611606 0.4433006
    0.4677973 0.4743519 0.4629703 0.4695897 0.4830257
#0.4662500 0.4890334 0.4480587 0.4923675 0.4370400 0.4294943 0.4650104 0.4129321
    0.4423387 0.4410939 0.4499849 0.4560852 0.4409484
#0.4379221 0.4585175 0.4658549 0.4730480 0.4496754 0.4700928 0.4643601 0.4537870
    0.4671429 0.4755140 0.4552888 0.4743959 0.4911050
#0.4688369 0.4788561 0.4554041 0.4971166 0.4409750 0.4489094 0.4571217 0.4215634
    0.4504274 0.4589161 0.4456067 0.4564160 0.4462035



#save plots in pdfs
```

```
for (i in 1:length(risk_levels_v)) {
  pdf_name <- paste("risk_level_", risk_levels_v[i], ".pdf", sep = "")
  pdf(pdf_name)
  for (j in 1:length(pel_plots[[i]])) {
    print(pel_plots[[i]][[j]])
  }
  dev.off()
}



#all results below refers to the setting *** of the commented parts:
#change the code where you find *** in order to obtain same results.
# 1) AA AXP BA BAC C CAT (last 2 are conditional assets)
#final_pel_uv
#0.1771676 0.1828829 0.1597222 0.1973684 0.1896341 0.1829861 0.1892617 0.2022581
    0.2091837 0.2235849 0.2416084 0.2266055 0.2330882
#0.1645455 0.1747253 0.1557692 0.1576923 0.1650000 0.1503106 0.1433735 0.1576531
    0.1733831 0.1578125 0.1752874 0.1589744 0.1610465
#0.1787975 0.1993007 0.1891447 0.2049180 0.2178947 0.2099237 0.2117925 0.2223485
    0.2355670 0.2446429 0.2437500 0.2417647 0.2461165
#0.1401840 0.1626582 0.1635870 0.1641566 0.1671642 0.1588816 0.1624031 0.1745161
    0.1803922 0.1719697 0.1940299 0.1942308 0.1838028
# 6) DD DIS GE GM HD HPQ (last 2 are conditional assets)
#final_pel_uv
#0.2719585 0.2560976 0.2748188 0.2698606 0.2558712 0.2779116 0.2453917 0.2662996
    0.2694656 0.2699134 0.3027108 0.2852941 0.2943182
#0.2688735 0.2742729 0.2592025 0.2698880 0.2458092 0.2678571 0.2431343 0.2339844
    0.2529304 0.2468750 0.2751938 0.2613043 0.2619792
#0.2943478 0.2841564 0.2770000 0.2770202 0.2895722 0.2898734 0.2838028 0.2983240
    0.3080537 0.3057325 0.3338415 0.3043333 0.3171429
#0.2624332 0.2893072 0.2823256 0.2746032 0.2718062 0.2834862 0.2798701 0.2730263
    0.2859694 0.2704969 0.3158854 0.2881910 0.2959239
# 7) IBM INTC JNJ JPM AIG KO (last 2 are conditional assets)
#final_pel_uv
#0.2035714 0.2084746 0.1991489 0.2051351 0.1965839 0.2185484 0.2271429 0.2494220
    0.2487805 0.2608974 0.2461765 0.2604938 0.2574830
#0.1821739 0.1916045 0.1632597 0.1842995 0.1596875 0.1761905 0.1783163 0.2109005
    0.2110169 0.2142292 0.1902778 0.2046763 0.2154255
#0.2034483 0.2241259 0.2136076 0.2392638 0.2378261 0.2536765 0.2588816 0.2677632
    0.2756000 0.2841667 0.2791262 0.3008621 0.3056818
#0.1708520 0.2002008 0.1769036 0.1829480 0.2000000 0.1964912 0.2046729 0.2206897
    0.2170854 0.2339623 0.2133333 0.2370968 0.2378378
```

A.4 $\{\Omega_4, I_4\}$ Risk Measures forecast and Probability Equivalent Level Analysis

```
#libraries
rm(list = ls())
set.seed(129)
library(VineCopula)
library(rugarch)
library(mvtsplot)
library(portvine)
library(stats)
library(rvinecopulib)
library(skewt)
library(magrittr)
library(ggplot2)
library(tidyverse)
library(readxl)
library(ggraph)
```

```
#RISK MEASURES COMPUTATION AND BACKTESTING
#import datasets
db <- read_excel("db.xlsx", sheet = "db2")
db=db[nrow(db):1,]
db <- column_to_rownames(db, var = "Date")

db_p <- read_excel("db.xlsx", sheet = "db2 price")
db_p=db_p[nrow(db_p):1,]
db_p <- column_to_rownames(db_p, var = "Date")
db_p_2 <- db_p %>%
  rownames_to_column(var = "date") %>%
  mutate(date = as.Date(date)) %>%
  gather(key = "Constituent", value = "log_return", -date)

#plot univariate price series
ggplot(data = db_p_2, aes(x = date, y = log_return)) +
  geom_line() +
  labs(x = "Date", y = "Log Return") +
  facet_wrap(~ Constituent, scales = "free_y", ncol = 3) +
  ggtitle("Univariate Price Series of Portfolio Assets") +
  labs(subtitle = "GCJ4 and CCK4 are the conditional market indexes")

#compute correlation matrices
threshold_test <- as.Date("2007-09-4")
dates=as.Date(row.names(db_p))
realized_train=as.numeric(rowSums(subset(db_p, row.names(db_p)<= threshold_test)
    [,1:4])/4)
realized_test=as.numeric(rowSums(subset(db_p, row.names(db_p) > threshold_test)
    [,1:4])/4)
gold_train=subset(db_p, row.names(db_p)<= threshold_test)[,5]
gold_test=subset(db_p, row.names(db_p)> threshold_test)[,5]
cocoa_train=subset(db_p, row.names(db_p)<= threshold_test)[,6]
cocoa_test=subset(db_p, row.names(db_p)> threshold_test)[,6]

cor_spearman <- matrix(NA, nrow = 2, ncol = 2)
cor_kendall <- matrix(NA, nrow = 2, ncol = 2)
cor_spearman[1, 1] <- cor.test(realized_train, gold_train, method = "spearman",
    exact = FALSE)$estimate
cor_kendall[1, 1] <- cor.test(realized_train, gold_train, method = "kendall",exact
    = FALSE)$estimate
cor_spearman[1, 2] <- cor.test(realized_train, cocoa_train, method = "spearman",
    exact = FALSE)$estimate
cor_kendall[1, 2] <- cor.test(realized_train, cocoa_train, method = "kendall",exact
    = FALSE)$estimate
cor_spearman[2, 1] <- cor.test(realized_test, gold_test, method = "spearman",exact
    = FALSE)$estimate
cor_kendall[2, 1] <- cor.test(realized_test, gold_test, method = "kendall",exact =
    FALSE)$estimate
cor_spearman[2, 2] <- cor.test(realized_test, cocoa_test, method = "spearman",exact
    = FALSE)$estimate
cor_kendall[2, 2] <- cor.test(realized_test, cocoa_test, method = "kendall",exact =
    FALSE)$estimate
rownames(cor_spearman) <- c("realized_train", "realized_test")
colnames(cor_spearman) <- c("gold_train", "cocoa_train")
```

```
rownames(cor_kendall) <- c("realized_train", "realized_test")
colnames(cor_kendall) <- c("gold_train", "cocoa_train")cor_spearman
#db2 price
#                gold_train cocoa_train
#realized_train  0.8845225   0.5328657
#realized_test  -0.1222775  -0.3032364
cor_kendall
#db2 price
#                gold_train cocoa_train
#realized_train  0.6932501   0.3852222
#realized_test  -0.1098750  -0.1762041
```

```
#train/test set construction
train <- db %>%
  rownames_to_column(var = "date") %>%
  mutate(date = as.Date(date)) %>%
  filter(date <= as.Date("2007-09-4"))
test <- db %>%
  rownames_to_column(var = "date") %>%
  mutate(date = as.Date(date)) %>%
  filter(date > as.Date("2007-09-4"))
train_test=rbind(train,test)
cat("Dimensione train set:", nrow(train))
cat("Dimensione test set:", nrow(test))
cat("Dimensione dataset:", nrow(train_test))

#unconditional var,es estimation using a rolling window
#marginal settings
marg_settings <- marginal_settings(
  train_size =nrow(train),
  refit_size = 50, #length of forecasting window of marginal models
  #individual_spec = spec_list #uncomment this line if ARMA-GARCH gridsearch is
      done
  default_spec = default_garch_spec()
)

#vine settings
uncond_vine_settings <- vine_settings(
  train_size = nrow(train),
  refit_size = 50, #how many times use the same copula
)

col_index <- setdiff(2:ncol(train_test), 8)
col_sampled <- c(2,3,4,5,6,7)
asset_names=names(train_test)[col_sampled]
weights_portaf=setNames(c(rep(1/(length(asset_names)-2), length(asset_names) - 2),
    0, 0), asset_names)
realized=rowSums(test[,col_sampled[1:4]]/4)

#rolling estimation approach: fit ARMA-GARCH, take residuals, apply PIT, fit vine
uncond_risk_roll <- estimate_risk_roll(
```

```
  data = train_test[,col_sampled[1:4]],
weights=weights_portaf[1:4],  marginal_settings = marg_settings,
  vine_settings = uncond_vine_settings,
  alpha = c(0.05),
  risk_measures = c("VaR", "ES_mean"),
  n_samples = 500,
  trace = TRUE
)
df_risk=risk_estimates(uncond_risk_roll,exceeded = TRUE)

#ljung-box test for serial autocorrelation at different lags (H0: no
    autocorrelation)
marginals <- fitted_marginals(uncond_risk_roll)
vines=fitted_vines(uncond_risk_roll)
lag=c(1,5,10,15,20)
ljung_mat_std <- array(NA, dim=c(4,5,4)) #asset x lag x rolling_window
ljung_mat_sqr_std <- array(NA, dim=c(4,5,4)) #asset x lag x rolling_window
for (i in 1:4){
  for (j in 1:4){ #portfolio with 4 asset
    std_resid <- roll_residuals(marginals[[names(train_test)[col_sampled[j]]]],
        roll_num = i)
    sqr_std_resid <- roll_residuals(marginals[[names(train_test)[col_sampled[j]]]],
         roll_num = i)**2
    for (t in 1:5){
      ljung_mat_std[j,t,i]=Box.test(std_resid, lag = lag[t], type = "Lju")$p.value
      ljung_mat_sqr_std[j,t,i]=Box.test(sqr_std_resid, lag = lag[t], type = "Lju")$
          p.value
    }
  }
}
cat("Ljung-Box test p-values on residuals\n")
ljung_mat_std
cat("Null hypothesis rejected",sum(ljung_mat_std < 0.05),"times")
#db2: Null hypothesis rejected 11 times

cat("Ljung-Box test pvalues on squared residuals\n")
ljung_mat_sqr_std
cat("Null hypothesis rejected",sum(ljung_mat_sqr_std < 0.05),"times")
#db2: Null hypothesis rejected 7 times

#plot of results
x_geom_point=df_risk$row_num[df_risk$exceeded]
y_geom_point=df_risk$realized[df_risk$exceeded]
df_geom_point=data.frame(df_risk$realized[df_risk$exceeded])
df_risk %>%
  ggplot() +
  geom_line(aes(x = row_num, y = realized), col = "grey") +
  geom_line(aes(x = row_num, y = risk_est, col = factor(risk_measure))) +
  scale_fill_manual() +
  geom_point(aes(x = x_geom_point, y = y_geom_point),
             data = df_geom_point,
             col = "#db4f59")+
  labs(x = "trading day",
       y = "portfolio log returns",
       col = "Risk measure",
       title = "Unconditional risk measures, conf. level 5%",
       subtitle = "Exceedances in red, portfolio log return in grey")

#conditional var,es estimation using a rolling window
#vine settings
cond_vine_settings <- vine_settings(
  train_size = nrow(train),
  refit_size = 50, #how many times use the same copula
  family_set = c("parametric"),
  vine_type = "dvine")

#confidence level of the estimated quantile from the marginal distribution
```

```
pelcov=0.9
pelcov_str=as.character(pelcov)#rolling estimation approach: fit ARMA-GARCH, take
    residuals, apply PIT, fit vine
cond_risk_roll <- estimate_risk_roll(
  data = train_test[col_sampled[1:6]],
  weights =weights_portaf ,
  marginal_settings = marg_settings,
  vine_settings = cond_vine_settings,
  alpha = c(0.05),
  risk_measures = c("VaR", "ES_mean"),
  n_samples = 500,
  cond_vars = c(names(train_test)[col_sampled[5]],names(train_test)[col_sampled
      [6]]),
  cond_u=pelcov, #value inside pelcov is associated to both conditioning variables
  ##QUANTILE STRATEGY=marginal market index on copula scale
  #(marginals for copulas are unif-->quantile is the confidence level itself),
  prior_resid_strategy = TRUE,
  #RESIDUAL STRATEGY=conditioning values for the forecast at time t
  #are the PIT of index I residual at time t-1
  trace = TRUE
)

#QUANTILE STRATEGY PLOT
df_cond=risk_estimates(cond_risk_roll,exceeded = TRUE)
ggplot(df_cond) +
  geom_line(aes(x = row_num, y = realized), col = "grey") +
  geom_line(data = subset(df_cond, risk_measure == "VaR" & cond_u == pelcov_str),
            aes(x = row_num, y = risk_est,col=paste0("VaR_",pelcov_str))) +
  geom_line(data = subset(df_cond, risk_measure == "ES_mean" & cond_u == pelcov_str
      ),
            aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_str))) +
  geom_line(data = subset(df_cond, cond_u == pelcov_str),
            aes(x = row_num, y =!!as.name(names(train_test)[col_sampled[5]]),
                col=paste0(names(train_test)[col_sampled[5]],"_",pelcov))) +
  geom_line(data = subset(df_cond, cond_u == pelcov_str),
            aes(x = row_num, y =!!as.name(names(train_test)[col_sampled[6]]),
                col=paste0(names(train_test)[col_sampled[6]],"_",pelcov))) +
  geom_point(data=subset(df_cond,exceeded== TRUE & cond_u == pelcov_str),
             aes(x = row_num, y=realized),col="#db4f59")+
  scale_fill_manual(
    name = "Risk Measure",
    labels = c(paste0(names(train_test)[col_sampled[6]],"_",pelcov),
               paste0(names(train_test)[col_sampled[5]],"_",pelcov),
               paste0("ES_mean_",pelcov_str),paste0("VaR_",pelcov_str)))+
  labs(x = "trading day",
       y = "portfolio log returns",
       col = "Risk measure",
       title = "Quantile based conditional risk measures, conf. level 5%",
       subtitle = "Exceedances in red, portfolio realized log return in grey")

#PRIOR RESIDUAL COPULA SCALE STRATEGY PLOT
#It should be noted that the conditional series based on the fitted
#residuals of the time unit before will most likely exaggerate
#sudden high volatility situations
ggplot(df_cond, aes(x = row_num)) +
  geom_line(aes(y = realized), col = "grey") +
  geom_line(data = subset(df_cond, risk_measure == "VaR" & cond_u == "prior_resid")
      , aes(y = risk_est,col="VaR_prior_resid")) +
  geom_line(data = subset(df_cond, risk_measure == "ES_mean" & cond_u == "prior_
      resid"), aes(y = risk_est,col="ES_mean_prior_resid")) +
  geom_point(data=subset(df_cond,exceeded== TRUE & cond_u == "prior_resid"),aes(y=
      realized),col="#db4f59")+
  labs(x = "trading day",
       y = "portfolio log returns",
       col = "Risk measure",
       title = "Prior Residual conditional risk measures, conf. level 5%",
       subtitle = "Exceedances in red, portfolio realized log return in grey")+
```

```
  scale_fill_manual(
name = "Risk Measure",     labels = c("ES_mean_prior_resid","VaR_prior_resid")
  )

#backtesting
uncond_var=df_risk[df_risk$risk_measure=="VaR",2]
cond_var_quantile=df_cond[df_cond$risk_measure=="VaR" & df_cond$cond_u==pelcov_str
    ,2]
cond_var_residual=df_cond[df_cond$risk_measure=="VaR" & df_cond$cond_u=="prior_
    resid",2]
bckt_var_uncond=VaRTest(alpha = 0.05, actual=realized,
                        VaR=uncond_var, conf.level = 0.95) #0.05)#
bckt_var_quantile=VaRTest(alpha = 0.05, actual=realized,
                          VaR=cond_var_quantile, conf.level = 0.95)
bckt_var_residual=VaRTest(alpha = 0.05, actual=realized,
                          VaR=cond_var_residual, conf.level = 0.95)
cat("unconditional strategy for VaR estimation\n", str(bckt_var_uncond))
# db2:
# $ expected.exceed: num 16
# $ actual.exceed  : num 15
# $ uc.H0          : chr "Correct Exceedances"
# $ uc.LRstat      : num 0.0958
# $ uc.critical    : num 3.84
# $ uc.LRp         : num 0.757
# $ uc.Decision    : chr "Fail to Reject H0"
# $ cc.H0          : chr "Correct Exceedances & Independent"
# $ cc.LRstat      : num 0.225
# $ cc.critical    : num 5.99
# $ cc.LRp         : num 0.893
# $ cc.Decision    : chr "Fail to Reject H0"
cat("conditional quantile strategy for VaR estimation\n", str(bckt_var_quantile))
# db2 0.1:
# $ expected.exceed: num 16
# $ actual.exceed  : num 11
# $ uc.H0          : chr "Correct Exceedances"
# $ uc.LRstat      : num 1.97
# $ uc.critical    : num 3.84
# $ uc.LRp         : num 0.16
# $ uc.Decision    : chr "Fail to Reject H0"
# $ cc.H0          : chr "Correct Exceedances & Independent"
# $ cc.LRstat      : num 2.75
# $ cc.critical    : num 5.99
# $ cc.LRp         : num 0.253
# $ cc.Decision    : chr "Fail to Reject H0"

# db2 0.9:
# $ expected.exceed: num 16
# $ actual.exceed  : num 24
# $ uc.H0          : chr "Correct Exceedances"
# $ uc.LRstat      : num 3.47
# $ uc.critical    : num 3.84
# $ uc.LRp         : num 0.0627
# $ uc.Decision    : chr "Fail to Reject H0"
# $ cc.H0          : chr "Correct Exceedances & Independent"
# $ cc.LRstat      : num 3.5
# $ cc.critical    : num 5.99
# $ cc.LRp         : num 0.174
# $ cc.Decision    : chr "Fail to Reject H0"
cat("conditional prior residual strategy for VaR estimation\n", str(bckt_var_
    residual))

uncond_es=df_risk[df_risk$risk_measure=="ES_mean",2]
cond_es_quantile=df_cond[df_cond$risk_measure=="ES_mean" & df_cond$cond_u==pelcov_
    str ,2]
cond_es_residual=df_cond[df_cond$risk_measure=="ES_mean" & df_cond$cond_u=="prior_
    resid",2]
bckt_es_uncond=ESTest(alpha = 0.05, actual=realized, ES=uncond_es,
```

```
                        VaR=uncond_var, conf.level = 0.95,boot = TRUE, n.boot = 1000)
bckt_es_quantile=ESTest(alpha = 0.05, realized, cond_es_quantile,
                                cond_var_quantile, conf.level = 0.95, boot = TRUE, n.
    boot = 1000)
bckt_es_residual=ESTest(alpha = 0.05, realized,cond_es_residual,
                        cond_var_residual, conf.level = 0.95, boot = TRUE, n.boot =
                             1000)
cat("unconditional strategy for ES estimation\n", str(bckt_es_uncond))
# db2:
# $ expected.exceed: num 16
# $ actual.exceed  : int 15
# $ H1             : chr "Mean of Excess Violations of VaR is greater than zero"
# $ boot.p.value   : num 0.309
# $ p.value        : num 0.227
# $ Decision       : chr "Fail to Reject H0"

cat("conditional quantile strategy for ES estimation\n", str(bckt_es_quantile))
# db2 0.1:
# $ expected.exceed: num 16
# $ actual.exceed  : int 11
# $ H1             : chr "Mean of Excess Violations of VaR is greater than zero"
# $ boot.p.value   : num 0.275
# $ p.value        : num 0.182
# $ Decision       : chr "Fail to Reject H0"

# db2 0.9:
# $ expected.exceed: num 16
# $ actual.exceed  : int 24
# $ H1             : chr "Mean of Excess Violations of VaR is greater than zero"
# $ boot.p.value   : num 0.144
# $ p.value        : num 0.0825
# $ Decision       : chr "Fail to Reject H0"
cat("conditional prior residual strategy for ES estimation\n", str(bckt_es_residual
    ))



#plot vine structures
uncond_vines=fitted_vines(uncond_risk_roll)
cond_vines=fitted_vines(cond_risk_roll)
plot(uncond_vines[[1]], tree="ALL", var_names="legend", edge_labels="tau")
plot(cond_vines[[1]], tree=1:5, var_names="legend", edge_labels="tau")



#PROBABILITY EQUIVALENT LEVEL ANALYSIS
#definition of risk levels at which pelcov should be detected
risk_levels_v=c
    (0.01,0.015,0.02,0.025,0.03,0.035,0.04,0.045,0.05,0.055,0.06,0.065,0.07)
final_pel_uv=array(NA,dim=c(4,length(risk_levels_v)))
pel_plots <- vector("list", length(risk_levels_v)) #vector containing all plots to
    see graphically PELs

for (v in 1:length(risk_levels_v)){
  #rolling estimation approach: fit ARMA-GARCH, take residuals, apply PIT, fit vine
  uncond_risk_roll <- estimate_risk_roll(
    data = train_test[,col_sampled[1:4]],
    weights=weights_portaf[1:4],
    marginal_settings = marg_settings,
    vine_settings = uncond_vine_settings,
    alpha = c(risk_levels_v[v]),
    risk_measures = c("VaR", "ES_mean"),
    n_samples = 500,
    trace = TRUE
  )
  df_risk=risk_estimates(uncond_risk_roll,exceeded = TRUE)
  uncond_var=df_risk[df_risk$risk_measure=="VaR",2]
```

```
  uncond_es=df_risk[df_risk$risk_measure=="ES_mean",2]
intersections=array(0,dim=c(4,9))


  #1D CASE
  pelcov_1d= c(0.1,0.9)
  pelcov_1d_str=as.character(pelcov_1d)
  df_cond_list_1d=vector("list", length = length(pelcov_1d))
  var_tests_1d=vector("list", length = length(pelcov_1d))
  es_tests_1d=vector("list", length = length(pelcov_1d))

  #rolling estimation approach: fit ARMA-GARCH, take residuals, apply PIT, fit vine
  for (i in 1:length(pelcov_1d)){
    cond_risk_roll <- estimate_risk_roll(
      data = train_test[col_sampled[1:5]],
      weights =weights_portaf[1:5],
      marginal_settings = marg_settings,
      vine_settings = cond_vine_settings,
      alpha = c(risk_levels_v[v]),
      risk_measures = c("VaR", "ES_mean"),
      n_samples = 500,
      cond_vars = c(names(train_test)[col_sampled[5]]),
      cond_u=pelcov_1d[i],
      prior_resid_strategy = TRUE,
      trace = TRUE
    )
    df_cond_list_1d[[i]]=risk_estimates(cond_risk_roll,exceeded = TRUE)
    cond_var_quantile=df_cond_list_1d[[i]][df_cond_list_1d[[i]]$risk_measure=="VaR"
        & df_cond_list_1d[[i]]$cond_u==pelcov_1d_str[i] ,2]
    var_tests_1d[[i]]=VaRTest(alpha = risk_levels_v[v], actual=realized,
                              VaR=cond_var_quantile, conf.level = 1-risk_levels_v[v
                                ])
    cond_es_quantile=df_cond_list_1d[[i]][df_cond_list_1d[[i]]$risk_measure=="ES_
        mean" & df_cond_list_1d[[i]]$cond_u==pelcov_1d_str[i] ,2]
    es_tests_1d[[i]]=ESTest(alpha = risk_levels_v[v], realized, cond_es_quantile,
                            cond_var_quantile, conf.level = 1-risk_levels_v[v],
                                boot = TRUE, n.boot = 1000)
  }

  #pelcov
  dfs_var_1d<- list()
  for (i in 1:2) {
    dfs_var_1d[[i]] <- subset(df_cond_list_1d[[i]], risk_measure == "VaR" & cond_u
        == pelcov_1d_str[i])
    diff_series <- dfs_var_1d[[i]][,2] - uncond_var
    for (j in 2:length(diff_series)) {
      if (diff_series[j - 1] * diff_series[j] < 0) {
        intersections[1,i] <- intersections[1,i] + 1
      }
    }
  }

  first_elements_var_1d <- sapply(dfs_var_1d, function(df) df[1, "risk_est"])
  order_index_var_1d <- order(first_elements_var_1d)
  colors <- c("#33a02c", "#fb9a99")
  legend_data_var_1d <- data.frame(
    labels = c(paste0("Var_", pelcov_1d_str[order_index_var_1d])),
    colors = colors[order_index_var_1d]
  )

  #stores graphical results
  plot<- ggplot() +
    geom_line(data = subset(df_risk, risk_measure == "VaR"),
              aes(x = row_num, y = risk_est), color="black")+
    geom_line(data = subset(df_cond_list_1d[[1]], risk_measure == "VaR" & cond_u ==
        pelcov_1d_str[1]),
              aes(x = row_num, y = risk_est,col=paste0("Var_",pelcov_1d_str[1])))+
```

```
    geom_line(data = subset(df_cond_list_1d[[2]], risk_measure == "VaR" & cond_u ==
        pelcov_1d_str[2]),
aes(x = row_num, y = risk_est,col=paste0("Var_",pelcov_1d_str[2])))+     labs(x = "
                trading day",
         y = "portfolio VaR",
         col = "Risk measure",
         title = "Pelcov research graphically, 1 conditional asset",
         subtitle = paste0("Unconditional VaR in black, VaR conf. level ",risk_
             levels_v[v]*100, "%"))+
    scale_color_manual(name = "Risk Measure", values = legend_data_var_1d$colors,
        labels = legend_data_var_1d$labels)
  pel_plots[[v]][[1]] <- plot

  #pelcoes
  dfs_es_1d <- list()
  for (i in 1:2) {
    dfs_es_1d[[i]] <- subset(df_cond_list_1d[[i]], risk_measure == "ES_mean" & cond
        _u == pelcov_1d_str[i])
    diff_series <- dfs_es_1d[[i]][,2] - uncond_es
    for (j in 2:length(diff_series)) {
      if (diff_series[j - 1] * diff_series[j] < 0) {
        intersections[2,i] <- intersections[2,i] + 1
      }
    }
  }

  first_elements_es_1d <- sapply(dfs_es_1d, function(df) df[1, "risk_est"])
  order_index_es_1d <- order(first_elements_es_1d)
  legend_data_es_1d <- data.frame(
    labels = c(paste0("ES_mean_", pelcov_1d_str[order_index_es_1d])),
    colors = colors[order_index_es_1d]
  )

  #stores graphical results
  plot<- ggplot() +
    geom_line(data = subset(df_risk, risk_measure == "ES_mean"),
              aes(x = row_num, y = risk_est), color="black")+
    geom_line(data = subset(df_cond_list_1d[[1]], risk_measure == "ES_mean" & cond_
        u == pelcov_1d_str[1]),
              aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_1d_str[1])
                  ))+
    geom_line(data = subset(df_cond_list_1d[[2]], risk_measure == "ES_mean" & cond_
        u == pelcov_1d_str[2]),
              aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_1d_str[2])
                  ))+
    labs(x = "trading day",
         y = "portfolio VaR",
         col = "Risk measure",
         title = "Pelcoes research graphically, 1 conditional asset",
         subtitle = paste0("Unconditional ES in black, ES conf. level ",risk_levels
             _v[v]*100, "%"))+
    scale_color_manual(name = "Risk Measure", values = legend_data_es_1d$colors,
        labels = legend_data_es_1d$labels)
  pel_plots[[v]][[2]] <- plot



  #2D CASE
  pelcov_2d_a= c(0.1,0.9)
  pelcov_2d_a_str=as.character(pelcov_2d_a)
  df_cond_list_2d_a=vector("list", length = length(pelcov_2d_a))
  var_tests_2d_a=vector("list", length = length(pelcov_2d_a))
  es_tests_2d_a=vector("list", length = length(pelcov_2d_a))

  #rolling estimation approach: fit ARMA-GARCH, take residuals, apply PIT, fit vine
  for (i in 1:length(pelcov_2d_a)){
    cond_risk_roll <- estimate_risk_roll(
```

```
      data = train_test[col_sampled],
weights =weights_portaf,       marginal_settings = marg_settings,
      vine_settings = cond_vine_settings,
      alpha = c(risk_levels_v[v]),
      risk_measures = c("VaR", "ES_mean"),
      n_samples = 500,
      cond_vars = c(names(train_test)[col_sampled[5]],names(train_test)[col_sampled
          [6]]),
      cond_u=pelcov_2d_a[i],
      prior_resid_strategy = TRUE,
      trace = TRUE
    )
    df_cond_list_2d_a[[i]]=risk_estimates(cond_risk_roll,exceeded = TRUE)
    cond_var_quantile=df_cond_list_2d_a[[i]][df_cond_list_2d_a[[i]]$risk_measure=="
        VaR" & df_cond_list_2d_a[[i]]$cond_u==pelcov_2d_a_str[i] ,2]
    var_tests_2d_a[[i]]=VaRTest(alpha = risk_levels_v[v], actual=realized,
                                VaR=cond_var_quantile, conf.level = 1-risk_levels_v
                                    [v])
    cond_es_quantile=df_cond_list_2d_a[[i]][df_cond_list_2d_a[[i]]$risk_measure=="
        ES_mean" & df_cond_list_2d_a[[i]]$cond_u==pelcov_2d_a_str[i] ,2]
    es_tests_2d_a[[i]]=ESTest(alpha = risk_levels_v[v], realized, cond_es_quantile,
                              cond_var_quantile, conf.level = 1-risk_levels_v[v],
                                  boot = TRUE, n.boot = 1000)
  }

  #pelcov
  dfs_var_2d_a<- list()
  for (i in 1:2) {
    dfs_var_2d_a[[i]] <- subset(df_cond_list_2d_a[[i]], risk_measure == "VaR" &
        cond_u == pelcov_2d_a_str[i])
    diff_series <- dfs_var_2d_a[[i]][,2] - uncond_var
    for (j in 2:length(diff_series)) {
      if (diff_series[j - 1] * diff_series[j] < 0) {
        intersections[3,i] <- intersections[3,i] + 1
      }
    }
  }

  first_elements_var_2d_a <- sapply(dfs_var_2d_a, function(df) df[1, "risk_est"])
  order_index_var_2d_a <- order(first_elements_var_2d_a)
  legend_data_var_2d_a <- data.frame(
    labels = c(paste0("Var_", pelcov_2d_a_str[order_index_var_2d_a])),
    colors = colors[order_index_var_2d_a]
  )

  #stores graphical results
  plot<- ggplot() +
    geom_line(data = subset(df_risk, risk_measure == "VaR"),
              aes(x = row_num, y = risk_est), color="black")+
    geom_line(data = subset(df_cond_list_2d_a[[1]], risk_measure == "VaR" & cond_u
        == pelcov_2d_a_str[1]),
              aes(x = row_num, y = risk_est,col=paste0("Var_",pelcov_2d_a_str[1])))
                  +
    geom_line(data = subset(df_cond_list_2d_a[[2]], risk_measure == "VaR" & cond_u
        == pelcov_2d_a_str[2]),
              aes(x = row_num, y = risk_est,col=paste0("Var_",pelcov_2d_a_str[2])))
                  +
    labs(x = "trading␣day",
         y = "portfolio␣VaR",
         col = "Risk␣measure",
         title = "Pelcov␣research␣graphically,␣2␣conditional␣assets␣with␣same␣value
             ",
         subtitle = paste0("Unconditional␣VaR␣in␣black,␣VaR␣conf.␣level␣",risk_
             levels_v[v]*100, "%"))+
    scale_color_manual(name = "Risk␣Measure", values = legend_data_var_2d_a$colors,
         labels = legend_data_var_2d_a$labels)
  pel_plots[[v]][[3]] <- plot
```

```
  #pelcoes
  dfs_es_2d_a <- list()
  for (i in 1:2) {
    dfs_es_2d_a[[i]] <- subset(df_cond_list_2d_a[[i]], risk_measure == "ES_mean" &
        cond_u == pelcov_2d_a_str[i])
    diff_series <- dfs_es_2d_a[[i]][,2] - uncond_es
    for (j in 2:length(diff_series)) {
      if (diff_series[j - 1] * diff_series[j] < 0) {
        intersections[4,i] <- intersections[4,i] + 1
      }
    }
  }

  first_elements_es_2d_a <- sapply(dfs_es_2d_a, function(df) df[1, "risk_est"])
  order_index_es_2d_a <- order(first_elements_es_2d_a)
  legend_data_es_2d_a <- data.frame(
    labels = c(paste0("ES_mean_", pelcov_2d_a_str[order_index_es_2d_a])),
    colors = colors[order_index_es_2d_a]
  )

  #stores graphical results
  plot<- ggplot() +
    geom_line(data = subset(df_risk, risk_measure == "ES_mean"),
              aes(x = row_num, y = risk_est), color="black")+
    geom_line(data = subset(df_cond_list_2d_a[[1]], risk_measure == "ES_mean" &
        cond_u == pelcov_2d_a_str[1]),
              aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_2d_a_str
                  [1])))+
    geom_line(data = subset(df_cond_list_2d_a[[2]], risk_measure == "ES_mean" &
        cond_u == pelcov_2d_a_str[2]),
              aes(x = row_num, y = risk_est,col=paste0("ES_mean_",pelcov_2d_a_str
                  [2])))+
    labs(x = "trading day",
         y = "portfolio VaR",
         col = "Risk measure",
         title = "Pelcoes research graphically, 2 conditional assets with same
             value",
         subtitle = paste0("Unconditional ES in black, ES conf. level ",risk_levels
             _v[v]*100, "%"))+
    scale_color_manual(name = "Risk Measure", values = legend_data_es_2d_a$colors,
        labels = legend_data_es_2d_a$labels)
  pel_plots[[v]][[4]] <- plot



  #pel computation at confidence level v
  for (u in 1:4){
    final_pel_uv[u,v]=sum(pelcov_1d*intersections[u,])/sum(intersections[u,])
  }
}

#save plots in pdfs
for (i in 1:length(risk_levels_v)) {
  pdf_name <- paste("risk_level_", risk_levels_v[i], ".pdf", sep = "")
  pdf(pdf_name)
  for (j in 1:length(pel_plots[[i]])) {
    print(pel_plots[[i]][[j]])
  }
  dev.off()
}
```

Bibliography

Kjersti Aas, Claudia Czado, Arnoldo Frigessi, and Henrik Bakken. Pair-copula constructions of multiple dependence. *Insurance: Mathematics and Economics*, 44(2):182–198, 2009.

V.V. Acharya. A theory of systemic risk and design of prudential bank regulation. *Journal of financial stability*, 5(3):224–255, 2009.

Yacine Ait-Sahalia, Jianqing Fan, and Yan Li. The leverage effect puzzle: Disentangling sources of bias at high frequency. *Princeton University, USA*, 2013.

H. Akaike. Information theory and an extension of the maximum likelihood principle. *Proceedings of the 2nd International Symposium on Information Theory*, pages 267–281, 1973.

Tim Bedford and Roger M. Cooke. Probability density decomposition for conditionally dependent random variables modeled by vines. *Annals of Mathematics and Artificial Intelligence*, 32:245–268, 2001.

Tim Bedford and Roger M. Cooke. Vines - a new graphical model for dependent random variables. *Annals of Statistics*, 30(4):1031–1068, 2002.

T. Bollerslev. Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, 1986.

P. Brandimarte. *An Introduction to Financial Markets: A Quantitative Approach*. John Wiley Sons, Hoboken, NJ, 2018.

P.F. Christoffersen. Evaluating interval forecasts. *International Economic Review*, 39(4), 1998. doi: 10.2307/2527349.

J. Dißmann, E.C. Brechmann, C. Czado, and D. Kurowicka. Selecting and estimating regular vine copulae and application to financial returns. *Computational Statistics Data Analysis*, 59:52–69, 2013. ISSN 0167-9473. doi: https://doi.org/10.1016/j.csda.2012.08.010. URL https://www.sciencedirect.com/science/article/pii/S0167947312003131.

B.E Hansen. Autoregressive conditional density estimation. *International Economic Review*, pages 705–730, 1994.

European Insurance and Occupational Pensions Authority. 2016 eiopa insurance stress test report. 2016. URL https://www.eiopa.europa.eu/browse/financial-stability/insurance-stress-test_en.

European Insurance and Occupational Pensions Authority. Failures and near misses in insurance: overview of the causes and early identification. 2018. URL https://www.eiopa.europa.eu/eiopa-examines-causes-insurers-failures-and-near-misses-2018-07-17_en.

European Insurance and Occupational Pensions Authority. General rules on the taking-up and pursuit of direct insurance and reinsurance activities. 2022. URL https://www.eiopa.europa.eu/rulebook/solvency-ii-single-rulebook/directive-1382009ec-solvency-ii-directive_en.

Harry Joe. Families of m-variate distributions with given margins and m(m1)/2 bivariate dependence parameters. In Ludger Ruschendorf, Berthold Schweizer, and Michael D. Taylor, editors, *Distributions with fixed marginals and related topics*, pages 120–141. Institute of Mathematical Statistics, Hayward, 1996.

Harry Joe. *Multivariate Models and Multivariate Dependence Concepts (1st ed.)*. Springer, 1997.

Eric Jondeau, Ser-Huang Poon, and Michael Rockinger. *Financial Modeling Under Non-Gaussian Distributions*. Springer, 2006.

W. R. Knight. A computer method for calculating kendall's tau with ungrouped data. *Journal of the American Statistical Association 61 (314)*, pages 436–439, 1966.

P.H. Kupiec. Techniques for verifying the accuracy of risk measurement models. *Journal of Derivatives*, 3(2): 73–84, 1995. doi: 10.3905/jod.1995.407677.

George Milton Ljung and George Edward Pelham Box. On a measure of lack of fit in time series models. *Biometrika*, 65(2):297–303, 1978. doi: 10.1093/biomet/65.2.297.

Alexander J. McNeil, Rudiger Frey, and Paul Embrechts. Estimation of tail-related risk measures for heteroscedastic financial time series: an extreme value approach. *Journal of Empirical Finance*, 7:271–300, 2000.

Roger B. Nelsen. *An introduction to Copulas*. Springer, 2005.

Basel Committee on Banking Supervision. Basel committee on banking supervision minimum capital requirements for market risk. *Journal of Machine Learning Research*, 2019. URL https://www.bis.org/bcbs/publ/d457.htm.

Patricia Ortega-Jiménez, Franco Pellerey, Miguel A. Sordo, and Alfonso Suárez-Llorens. Probability equivalent level for covar and var. *Insurance: Mathematics and Economics*, 115:22–35, 2024. ISSN 0167-6687. doi: https://doi.org/10.1016/j.insmatheco.2023.12.004. URL https://www.sciencedirect.com/science/article/pii/S0167668723001063.

Murray Rosenblatt. Remarks on a multivariate transformation. *The Annals of Mathematical Statistics*, 23(3): 470–472, 1952.

G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978. doi: 10.1214/aos/1176344136.

E. Sommer. Estimation and backtesting of the expected shortfall and value at risk using vine copulas - an unconditional and conditional rolling window approach. *Master's thesis, Technical University Munich*, 2022.

Pedro Vitor Tòfoli, Fernando Augusto Ziegelmann, Otavio Candido, and Patricia Landini Valls Pereira. Dynamic d-vine copula model with applications to value-at-risk (var). *Journal of TIme Series Econometrics*, 2019.

J.M Zakoian. Threshold heteroskedastic models. *Journal of Economic Dynamics and Control*, 1991.