Copula model proposal for interest rate and equity shocks aggregation under Solvency II

Asier Gil Sedano

Carlos III University of Madrid

This is submitted for the Master of Science

in Actuarial Science

Madrid 2023

Contents

1. Introduction

Insurance companies, due to their business nature, are exposed to multiple risks. These risks begin with the signing of an insurance contract, where the company commits to some uncertain future payments, which in some cases can be doubly uncertain in terms of frequency and severity.

However, claims risks are not the only risks that insurance companies must face. For example, there are additional risks due to the financial component needed to be provided in the insurance products, where guaranteed returns are sometimes promised, especially in long-term products such as lifetime annuities. To achieve these returns, companies must invest in the market by acquiring financial assets that provide the necessary profitability to meet their obligations, with assets varying from fixed income to equity values such as stocks or linked bonds.

The Solvency II regulation recently introduced and mandatory in many European countries since 2016 establishes a framework that considers all the risks to which an insurance company is exposed. This way all insurance companies can provide a more self-adapted solvency capital calculation in comparison to the calculation required by previous regulation. This way, the capital requirements criteria are also unified so all companies are considering the same risk scheme defined in the regulation.

The following illustration shows the risk map proposed by the regulation Solvency II:

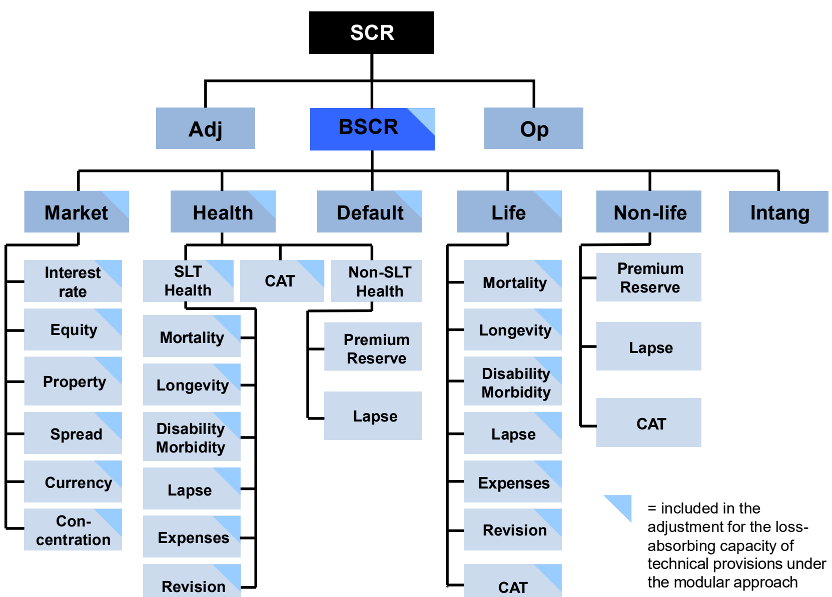

Figure 1. Solvency II risk map

When it comes to risk aggregation, the use of the standard formula provided by the regulation together with the risk correlations provided is the most common practice for insurance companies. However, there are other more complex alternatives that allow for a more appropriate aggregation of company risks, such as internal models that can use the same risk map or another if considered more appropriate.

The underlying assumptions under the standard model proposed by the regulation are very restrictive and mainly conditioned by the normality of the data. This can easily be proven to be true in some modules due to the central limit theorem, but in the case of the market module it is well known that financial series are not governed by normality. In these cases, the standard formula cannot properly reflect the real market behavior and missing the impact the market shocks could have in the solvency of the insurance companies.

The standard formula makes simplifications for the sake of simplicity so the results can be easily applied and interpreted by any risk manager. However, when trying to determine the real trade-off this simplification provides, there is hardly any literature, if not any, about a reliable comparison of this simplified approach and a statistically backed method.

Additionally, the recent high inflation scenarios have proven the weaknesses of the interest rate model and shocks provided by the regulation and a new model calibration that properly reflects the market behavior is required for the proper aggregation modelling.

2. Scope

The scope of this thesis is to validate the shocks proposed by EIOPA, especially in the current situation with the latest interest rates hikes due to inflation which had never occurred before. This approach is especially significant when coming from a very low interest rate period and there was no expectation of this type of hike. The combination of these two factors of low interest rates and the relative value shocks definition applied to a low interest rate, has meant to many insurance companies, and specially banks a complete reorganization of their assets portfolio and liabilities to withstand the market changes, which means this is no minor issue.

Additionally, after studying the Eiopa approach for the determination of the interest rate shocks it has been determined that the correlations between different factors involving this analysis may not be properly capturing the real dependence between the factors. This is why a more sophisticated analysis using copulas might be more interesting and could provide a better performance when measuring the risk. Another point is that the interest rate model considered by Eiopa does not consider any type of time dependency, which is well known to be present in any financial time series.

Additionally, the equity and the currency shocks pretend to be validated. Since these shocks are usually referring to general univariate variables like the European stock indexes or the euro indexes, the analysis is far easier than the case of the interest rates. Therefore, this analysis can relatively easily be validated so the overall performance and dependence can be studied.

As for other remaining variables, which are the property, spread and concentration, it has been concluded that they are not feasible. First the property analysis cannot be reliably performed since the property prices have proven to be mostly driven by political decisions and regulation. As for the spread, Eiopa has a well-defined methodology based in market data and the analysis itself could be the scope of an entire thesis. Because of this reason, it has been excluded from this work. The last variable, which is concentration, cannot easily be analyzed for a general case since this depends very much on each company portfolio and it is not considered to be relevant for the current analysis since the companies can easily hedge this risk.

After the definition of all these variables, the current regulatory requirement is fulfilled by the usage of the standard formula for all the market risk aggregation. The usage of this correlation matrix is based on a multivariate gaussian dependence, which in most cases is not fulfilled and there are not enough variables to justify that the underlying assumption is the central limit theorem. It is comprehensible that Eiopa needs to provide a feasible solution for this aggregation to be applied by the insurers, but it has been considered that the performance of the standard formula has not been validated in any way. In this work the usage of copulas for this aggregation is pretended to be used and compared so this performance can be properly checked.

For all the previously determined works, R programming language is used, and the code can be found in the annexes. As for the copula modelling the package “copula” by Hofert et al. is used and for the time series modelling the package “rugarch” by Alexis Galanos is used.

3. State of art

As mentioned before, there is a very few literature about the usage of copulas for market risk aggregation when considering interest rates, equity and currency.

When determining the interest rate model proposed by Eiopa by the usage of principal components it is well documented in the Solvency II Calibration Paper (EIOPA, 2010b). As for the calibration, in that text is mentioned that regression is obtained for the shock calibration. This method underlies the assumption of normality which as mentioned is usually not accurate for financial modelling. In any case the approach proposed properly captures the dependency between the principal components and their correlations. A deeper description of this documented methodology is provided later.

The ARIMA-GARCH models for financial risk modelling have been used for a long time and there are many papers showing good behavior for risk modelling. The most common models implemented in the "rugarch" package are explained.

As for the copulas, there is much information about the Elliptical and Archimedean copulas and the nesting processes for the construction of more complex dependence structures. While the usage of these elements in financial series is a trending research line.

In the current literature, the usage of the copula is mostly analyzed for the claims correlation in non-life business but there is not much analysis when it comes to market. Considering that all financial companies are exposed to the market and most of them are regulated by considering these aspects, it has been considered that there should be a more proper analysis.

The normal copula is one of the Eiopa underlying assumptions under the standard formula. Even if the Eiopa documentation refers to the data following an elliptical copula, there is not much literature validating this assumption, and it is almost nonexistent when it comes to determine the relationship between the interest rates and the equity and currency. This underlying assumption can be critical when determining the shocks since the student t copula and the normal copula provide very different results. In the Eiopa documentation it has not been clearly quantified the trade-off of simplicity for performance made by the regulator.

By this analysis, the internal models capital differences could also be established by determining the differences of the standard formula and a more advanced dependence structure estimation. This way, this work could be used by the companies as a proxy of the feasibility of developing a own internal model in terms of capital savings.

The theoretical concepts about copulas applied in this work will be explained in this part for the reader to be easier to understand the work.

3.1. Copulas

3.1.1. Definition

The copulas are multivariate distribution functions that determine the relationship between different univariate margins $U(0,1)$ of two or more random variables.

The copulas shall satisfy the following conditions:

- $C(1,\ldots,1,u_k;1,\ldots,1) = u_k$ for every $k \leq n$ and $\forall u_k \in [0,1]$
- $C(u_1,\ldots,u_{k-1},0_k,u_{k+1},\ldots,u_n) = 0$ for any $k \leq n$
- Shall be monotonic in all u random variables.

3.1.2. Sklar theorem

The Sklar theorem (Sklar, 1959) states that any multivariate distribution function shall comply with:

$$H(x_1,x_2,\ldots,x_n) = P[X_1 \leq x_1, X_2 \leq x_2, \ldots, X_n \leq X_n]$$

From a random vector $(X_1,X_2,\ldots,X_n)$ can be expressed by his marginal distribution functions and a copula so:

$$H(\boldsymbol{x}) = H(x_1,x_2,\ldots,x_n) = C\big(F_1(x_1),F_2(x_2),\ldots,F_n(x_n)\big) = C(u_1,u_2,\ldots,u_n)$$

The theorem also states that this copula is unique if the marginal distribution function of the random variables is unique.

Form the previous expression, know that if it exists, the density f of a function is determined by:

$$f(x_1,x_2,\ldots,x_n) = \frac{\partial^n}{\partial x_1 \partial x_2 \ldots \partial x_n} F(x_1,x_2,\ldots,x_n)$$

That in case H is continuous and both C and the marginal distribution functions are continuous, then the density function h from H satisfies:

$$h(x) = c\big(F_1(x_1),F_2(x_2),\ldots,F_n(x_n)\big)$$

And the density function will be given by:

$$c(\boldsymbol{u}) = \frac{h\left(F_1^{-1}(u_1),F_2^{-1}(u_2),\ldots,F_n^{-1}(u_n)\right)}{f_1\big(F_1^{-1}(u_1)\big)\cdot f_2\big(F_2^{-1}(u_2)\big)\cdot\ldots\cdot f_n\big(F_n^{-1}(u_n)\big)} \qquad \boldsymbol{u} \in (0,1)^n$$

This expression is particularly important since it is the density function used for some estimation methods as the maximum likelihood method or the pseudo maximum likelihood method.

The simplest case of a copula is the independence copula, that would be determined by the following expression:

$$\Pi(\boldsymbol{u}) = \prod_{j=1}^{n} u_j \qquad \boldsymbol{u} \in [0,1]^n$$

3.1.3. Fréchet-Hoeffding bounds theorem

The Fréchet-Hoeffding theorem (Fréchet, 1935), can be generalized (Rüschendorf, 1981) establishing that any copula of dimension n shall have a lower bound W and an upper bound M that satisfy:

$$W(\boldsymbol{u}) = \max\left\{\sum_{j=1}^{n} u_j - n + 1, 0\right\} \qquad \text{and} \qquad M(\boldsymbol{u}) = \min_{1 \le h \le n}\{u_j\} \qquad \boldsymbol{u} \in [0,1]^n$$

Once defined the upper and lower bounds, it will satisfy:

$$W(\boldsymbol{u}) \le C(\boldsymbol{u}) \le M(\boldsymbol{u}), \qquad \boldsymbol{u} \in [0,1]^n$$

Additionally for the case in which $U \sim U(0,1)$ it will satisfy:

$$(U, 1-U) \sim W \qquad \text{and} \qquad (U, \dots, U) \sim M$$

Being the copula W an antimonotone copula and M copula a monotone copula.

3.1.4. Other correlations measures

Even if the most generalized correlation measure is the Pearson correlation defined by the following equation:

$$cor(X_1, X_2) = \frac{E[(X_1 - E[X_1]) \cdot (X_2 - E[X_2])}{\sigma_{X_1}\sigma_{X_2}}$$

This correlation measure has two main problems that can arise in practice. The first one is the one related to the existence of the measure for distributions of long tails where it can be infinite. The second problem is the no invariability under X_1 and X_2 transformations.

These problems are especially relevant in some fields like the actuarial where long tail distributions are frequent which have not any finite mean or where the distribution of the random variables are transformations.

Because of this reason, and especially when working with copulas there are some rank correlation coefficients that can solve these problems. Amongst others, the two most common correlation measures are the Spearman Rho and the Kendall Tau.

3.1.4.1. Spearman Rho

The Spearman Rho is numerically defined as follows:

$$\rho_s = \rho_s(X_1, X_2) = cor\big(F_1(X_1), F_2(X_2)\big) = \frac{E[(F_1(X_1) - E[F_1(X_1)])(F_2(X_2) - E[F_2(X_2)])]}{\sigma_{F_1(X_1)}\sigma_{F_2(X_2)}}$$

This measure is not subject to any of the limitations defined for the Pearson correlation and will uniquely depend on the underlying copula that relates the random variables X_1 and X_2.

If the previous expression is developed by the definition $F_n(X_n) = U_n$, then the following expression can be developed:

$$\rho_s = \frac{cov(U_1, U_2)}{\sigma_{U_1}\sigma_{U_2}}$$

Then, by stablishing the relationship between U_1 and U_2 with the underlying copula C it can be directly obtained the correlation measure of the copula as:

$$\rho_s(C) = \frac{\int_0^1 \int_0^2 u_1 u_2 dC(u_1, u_2) - \left(\frac{1}{2}\right)^2}{\sqrt{\frac{1}{12}}\sqrt{\frac{1}{12}}} = 12 \int_0^1 \int_0^1 C(u_1, u_2) du_1 du_2 - 3$$

This way correlations could be determined as the copula moments and therefore used for the parameter's estimation.

3.1.4.2. Kendall Tau

Being (X_1', X_2') and independent copy of (X_1, X_2), Kendall Tau is a measure that measures the concordance between the two random variables. Two variables can be defined as concordant when $(x_1 - x_1')(x_2 - x_2') > 0$ and discordant when being less than 0.

This way, when two variables are fully concordant this measure will be equal to 1 and when they are fully discordant will be -1.

Therefore, Kendall Tau is defined as:

$$\tau(X_1, X_2) = E\big[sign\big((X_1 - X_1')(X_2 - X_2')\big)\big]$$

And considering that the probability of X being over and under X' are complementary, the previous expression can be rewritten as:

$$\tau(C) = 4 \int_0^1 \int_0^1 C(u_1, u_2)\, dC(u_1, u_2) - 1$$

The previous expression allows Kendall Tau to be related with the Spearman Rho (Durbin & Stuart, 1951).

$$-1 \leq 3\tau - 2\rho \leq 1$$

The previous expression and the following allow to establish the full bound of dependencies between these two correlation measures.

$$\frac{1+\rho}{2} \geq \left(\frac{1+\tau}{2}\right)^2$$

$$\frac{1+\rho}{2} \geq \left(\frac{1+\tau}{2}\right)^2$$

$$\frac{3\tau-1}{2} \leq \rho \leq \frac{1+2\tau-\tau^2}{2}\ , \qquad \tau \geq 0$$

$$\frac{\tau^2+2\tau-1}{2} \leq \rho \leq \frac{1+3\tau}{2}\ , \qquad \tau \leq 0$$

These equations lead to the following plotted dependency:

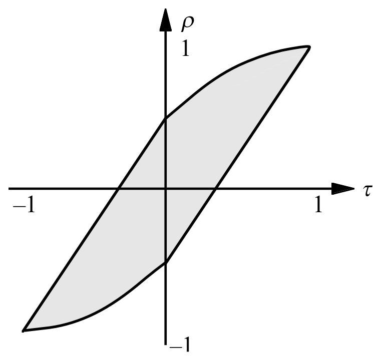

Figure 2. Spearman Rho and Kendall Tau bounds

3.1.5. Most common copulas

The elliptical copulas are those that have a position vector know as μ and a scale matrix $\Sigma = AA^T$. The A matrix is the matrix obtained by the Cholesky decomposition that relate an independent vector U so that:

$$\boldsymbol{X} = \mu + A\boldsymbol{U}$$

Also, there exist the so-called Archimedean Copulas, which are of the form:

$$C(u) = \psi\left(\psi^{-1}(u_1) + \psi^{-1}(u_2) + \cdots + \psi^{-1}(u_n)\right)$$

Being the generator function ψ a function that shall satisfy:

- Must be a continuous increasing equation in the rank $[0, \infty)$
- $\psi(0) = 1$ and $\psi(\infty) = 0$ therefore the inverse shall satisfy $\psi[0,1] \to [0, \infty)$
- It will be monotone Ψ if it satisfies with all the derivatives $\psi^{k)}$ satisfy $(-1)^k \psi^{k)}(t) \geq 0$ for every $k \in \{0,1, \dots, n2\}$ and it will be completely monotone if it satisfies in $k \in \{0,1, \dots, n\}$

Once all these conditions are satisfied, it will be derived the density function that is defined as:

$$c(u) = \frac{\psi^{n)}(\psi^{-1}(u_1) + \psi^{-1}(u_2) + \cdots + \psi^{-1}(u_n)}{\prod_{j=1}^{n} \psi^{1)}\left(\psi^{-1}(u_j)\right)}$$

3.1.5.1. Normal elliptical copula

The normal elliptical copula or multivariate normal copula is defined by the following distribution function.

$$C(\boldsymbol{u_1}, \boldsymbol{u_2}, \dots, \boldsymbol{u_n}) = \Phi_n\left(\Phi^{-1}(\boldsymbol{u_1}), \Phi^{-1}(\boldsymbol{u_2}), \dots, \Phi^{-1}(\boldsymbol{u_n})\right)$$

Where Φ_n has not a closed formula. However, there is a formula for the density function $h(u)$, which would be determined for a null position vector and a determined Σ scale matrix as:

$$\phi_n(x) = \frac{e^{-\frac{1}{2}x^T\Sigma^{-1}x}}{(2\pi)^{\frac{n}{2}}\sqrt{\det(\Sigma)}}$$

Being $x = \left(\Phi^{-1}(u_1), \Phi^{-1}(u_2), \dots, \Phi^{-1}(u_n)\right)$.

And by numerical methods, integrating in u^n it could be obtained by the following expression:

$$C(u_1, u_2, \dots, u_3) = \int_0^{u_n} \cdots \int_0^{u_2} \int_0^{u_1} \phi_n\left(\Phi^{-1}(u_1), \Phi^{-1}(u_2), \dots, \Phi^{-1}(u_n)\right) du_1 du_2 \dots du_n$$

As an illustrative example, it is shown a sample of a random variable governed by a three-dimensional normal copula with a scale matrix:

$$\Sigma = \begin{pmatrix} 1 & 0.8 & -0.7 \\ 0.8 & 1 & 0.6 \\ -0.7 & 0.6 & 1 \end{pmatrix}$$

And the plot points. Note that there are plotted the U^n and not the X^n which could or not be normal.

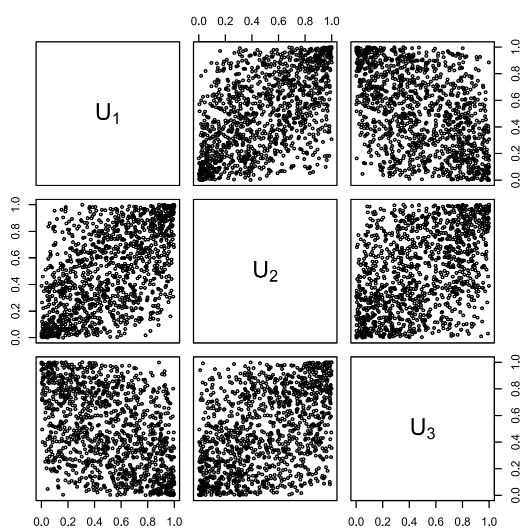

Figure 3. Normal Copula Example

3.1.5.2. t-Student elliptical copula

The multivariate t-Student has a density function that is determined as:

$$t_{n,\nu}(x) = \frac{\Gamma\left(\frac{\nu+n}{2}\right)}{\Gamma\left(\frac{\nu}{2}\right)(\pi\nu)^{\frac{n}{2}}\sqrt{det\,\Sigma}}\left(1+\frac{x^T\Sigma^{-1}x}{\nu}\right)^{-\frac{\nu+n}{2}}$$

Being $x = \left(t_\nu^{-1}(u_1), t_\nu^{-1}(u_2), \ldots, t_\nu^{-1}(u_n)\right)$.

Therefore, it can be integrated by the following expression:

$$C_\nu^t(u_1, u_2, \ldots, u_3) = \int_0^{u_n} \cdots \int_0^{u_2}\int_0^{u_1} t_{n,\nu}\left(t_\nu^{-1}(u_1), t_\nu^{-1}(u_2), \ldots, t_\nu^{-1}(u_n)\right) du_1 du_2 \ldots du_n$$

That can be solved obtaining:

$$c_\nu^t(u) = \frac{\Gamma\left(\frac{\nu+n}{2}\right)}{\Gamma\left(\frac{\nu}{2}\right)\sqrt{det\,\Sigma}}\left(\frac{\Gamma\left(\frac{\nu}{2}\right)}{\Gamma\left(\frac{\nu+1}{2}\right)}\right)^n \frac{\left(1+\frac{u'\Sigma u}{\nu}\right)^{-\frac{\nu+n}{2}}}{\prod_{j=1}^n\left(1+\frac{x_j^2}{\nu}\right)^{-\frac{\nu+1}{2}}}$$

As an illustrative example, the same scale matrix and same sample size as for the normal is provided but for a t-Student distribution function with 4 degrees of freedom, obtaining the following plot:

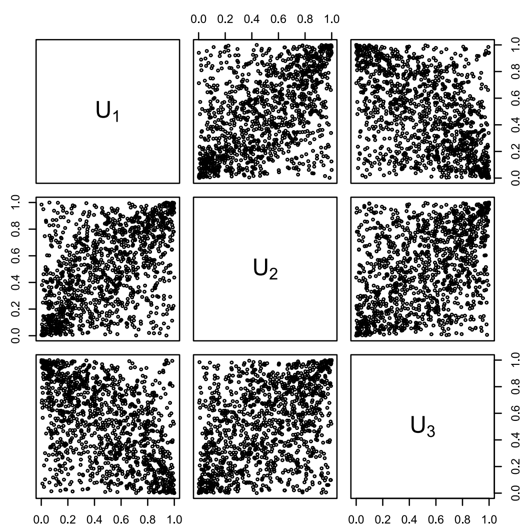

Figure 4. t-Student Copula example (df = **4**)

It is observed the tails are much greater than they are for the normal copula.

3.1.5.3. Ali-Mikhail-Haq Archimedean copula

The AMH (Ali-Mikhail-Haq) copula (Ali et al., 1978) has as generator function:

$$\Psi(u) = \ln\left(\frac{1-\theta(1-u)}{u}\right)$$

Being θ the copula parameter, with the domain $\theta \in [0,1)$. The inverse is determined as:

$$\Psi^{-1}(u) = \frac{1-\theta}{e^{u}-\theta}$$

Generally, this copula is for bivariate random variables, but the moment generator functions allows to generalize for n variables. The generalization is provided by the following expression:

$$C_{\theta}^{AMH}(u) = \frac{(1-\theta)}{\prod_{j=1}^{n}\left((1-\theta)u_j^{-1}+\theta\right)-\theta}$$

In the following picture, there are shown two AMH copulas with parameters $\theta_1 = 0.6$ and $\theta_2 = 0.75$

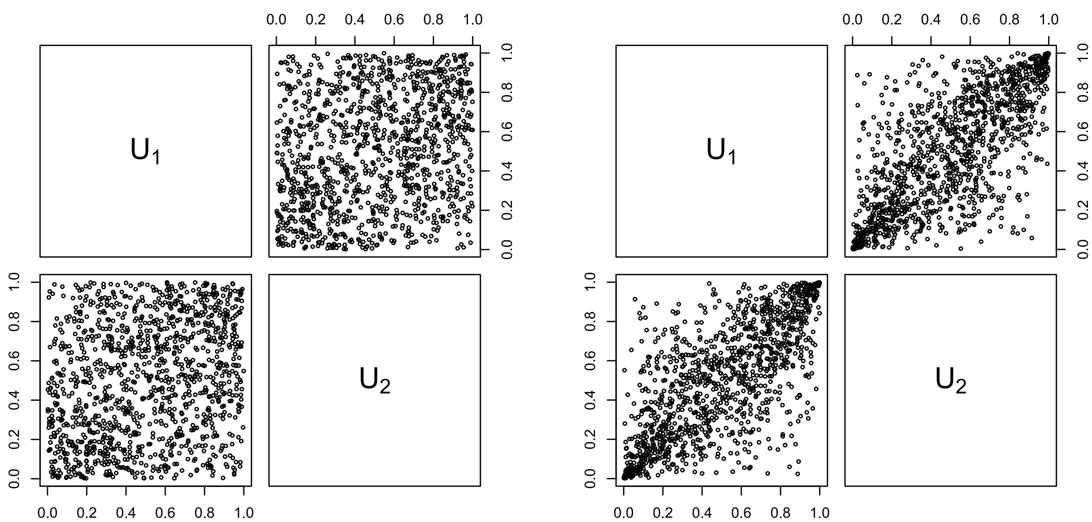

Figure 5. AMH copula examples (left $\boldsymbol{\theta_1 = 0.6}$, right $\boldsymbol{\theta_2 = 0.75}$)

It is important to remark that for the case where the parameter tends to 0, the AMH copula tends to be the independence copula.

3.1.5.4. Clayton Archimedean copula

The Clayton copula (Clayton, 1978) is a bivariate copula with generation function:

$$\Psi(u) = \frac{1}{\theta}\left(u^{-\theta} - 1\right)$$

With a domain $\theta \in [-1, \infty) \setminus \{0\}$. Its inverse is determined by:

$$\Psi^{-1}(u) = (1 + \theta u)^{-\frac{1}{\theta}}$$

That in the bivariate case with all the domain is determined by:

$$C_{\theta}(u_1, u_2) = \max\{u_1^{-\theta} + u_2^{-\theta} - 1; 0\}^{-\theta}$$

As it happens with the Archimedean copulas, even if originally were thought for the bivariate case, they can be generalized for the multivariate. As for the clayton copula the multivariate case with $n \geq 3$ with domain $\theta \in (0, \infty)$:

$$C_{\theta}^{C}(u) = \left(1 - n + \sum_{j=1}^{n} u_j^{-\theta}\right)^{-\frac{1}{\theta}}$$

In the following picture, there are shown two Clayton copulas with parameters $\theta_1 = 1$ and $\theta_2 = 5$:

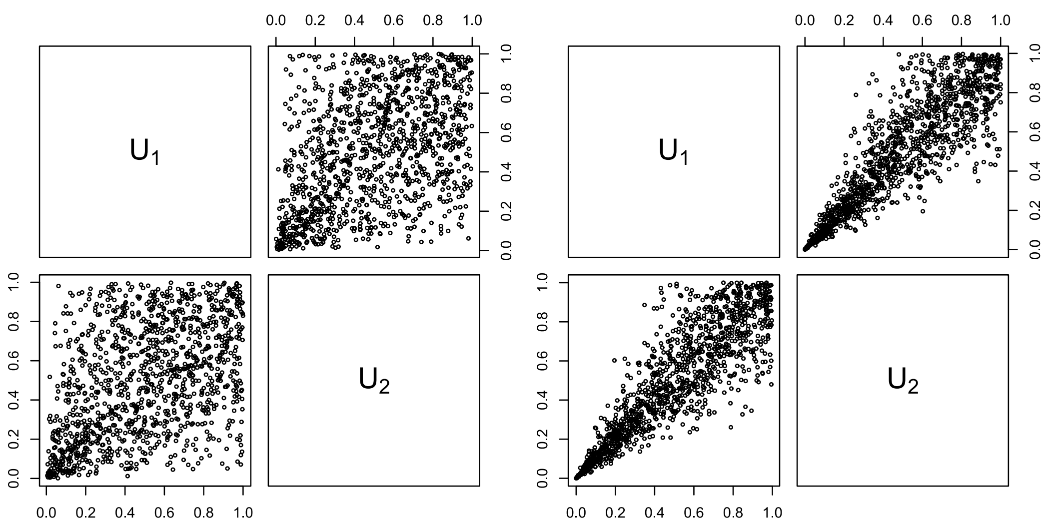

Figure 6. Clayton copula examples (left $\boldsymbol{\theta_1 = 1}$, right $\boldsymbol{\theta_2 = 5}$)

It can be observed this copula is non-symmetrical, an important property when modelling some dependence structures in insurance.

3.1.5.5. Frank Archimedean copula

The Frank copula (FRANK, 1975) is a symmetrical copula with a generator function:

$$\Psi(u) = -\ln\left(\frac{e^{-\theta u} - 1}{e^{-\theta} - 1}\right)$$

With the domain $\theta \in \mathbb{R} \setminus \{0\}$ and the inverse is determined by:

$$\Psi^{-1(u)} = -\frac{1}{\theta}\ln\left(1 + e^{-u}\left(e^{-\theta} - 1\right)\right)$$

As the rest, it can be generalized for the multivariate case with a domain $\theta \in (0, \infty)$:

$$C_\theta^F(u) = -\frac{1}{\theta}\ln\left(\frac{\prod_{j=1}^{n}\left(1 - e^{-\theta\, u_j}\right)}{(1 - e^{-\theta})^{n-1}}\right)$$

In the following picture, there are shown two Frank copulas with parameters $\theta_1 = -3$ and $\theta_2 = 5$:

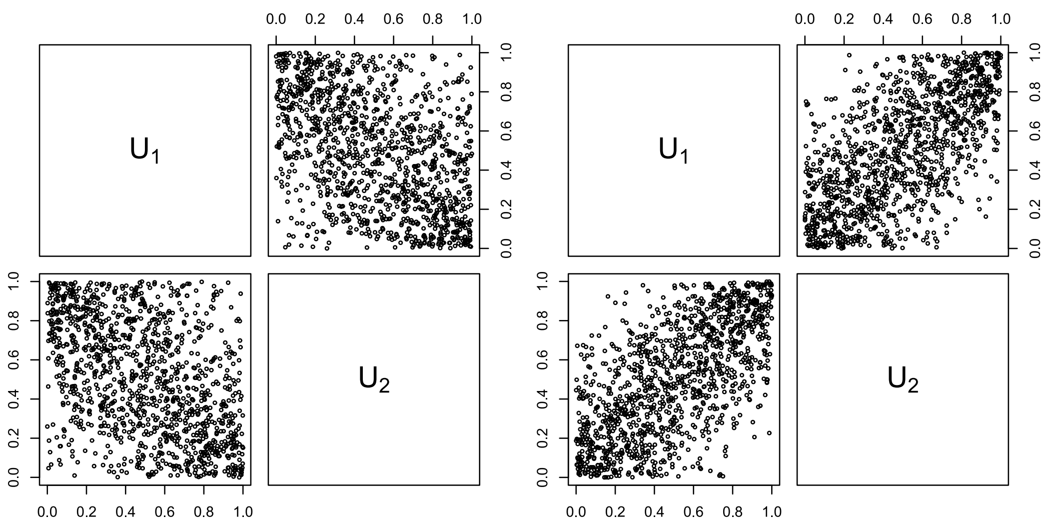

Figure 7. Clayton copula examples (left $\boldsymbol{\theta_1 = -3}$, right $\boldsymbol{\theta_2 = 5}$)

3.1.5.6. Gumbel-Hougaard Archimedean copula

The Gumbel-Hougaard copula, more known as Gumbel copula (Gumbel, 1960) has a generator function of this copula is the following:

$$\Psi(u) = (-\ln(u))^{\theta}$$

With the domain $\theta \in \mathbb{R} \setminus \{0\}$. The inverse is determined by:

$$\Psi^{-1}(u) = e^{-u^{1/\theta}}$$

As for the other Archimedean copulas even if the original copula is a bivariate copula, it can be generalized for multivariate case with the restricted domain of $\theta \in [1, \infty)$:

$$C_{\theta}^{GH}(u) = e^{-\left(\sum_{j=1}^{n}(-\ln(u_j))^{\theta}\right)^{\frac{1}{\theta}}}$$

In the following plot, two cases of the bivariate Gumbel Hougaard copula are shown for reference. It is remarkable how heavily tailed this copula is showing great correlation as the parameter is increased.

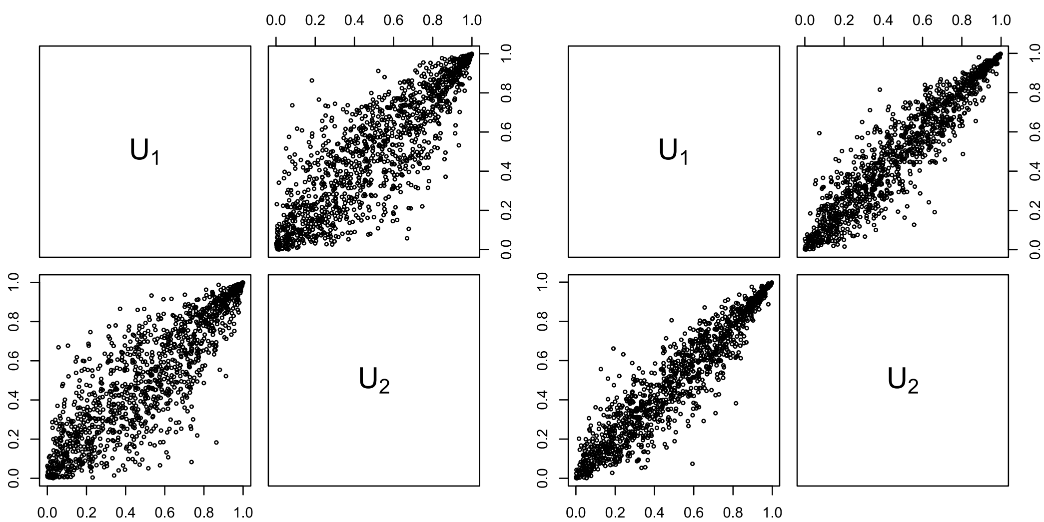

Figure 8. Gumbel-Hougaard copula examples (left $\boldsymbol{\theta_1 = 3}$, right $\boldsymbol{\theta_2 = 5}$)

3.1.5.7. Joe copula

The Joe copula (Joe, 1997) has a generator function of this copula is as follows:

$$\Psi(u) = -\ln\left(1 - (1-u)^{\theta}\right)$$

The domain of the parameter is determined by $\theta \in [1, \infty)$ and with the following inverse:

$$\Psi^{-1}(u) = 1 - (1 - e^{-u})^{\frac{1}{\theta}}$$

The generalized function for the multivariate case of the Archimedean Joe copula is determined for the domain $\theta \in [1, \infty)$:

$$C_{\theta}^{J}(u)1 - \left(1 - \prod_{j=1}^{n}\left(1 - \left(1 - u_j\right)^{\theta}\right)\right)^{\frac{1}{\theta}}$$

A couple of examples of the Joe copula are shown for the parameters $\boldsymbol{\theta_1 = 2}$ and $\boldsymbol{\theta_2 = 5}$.

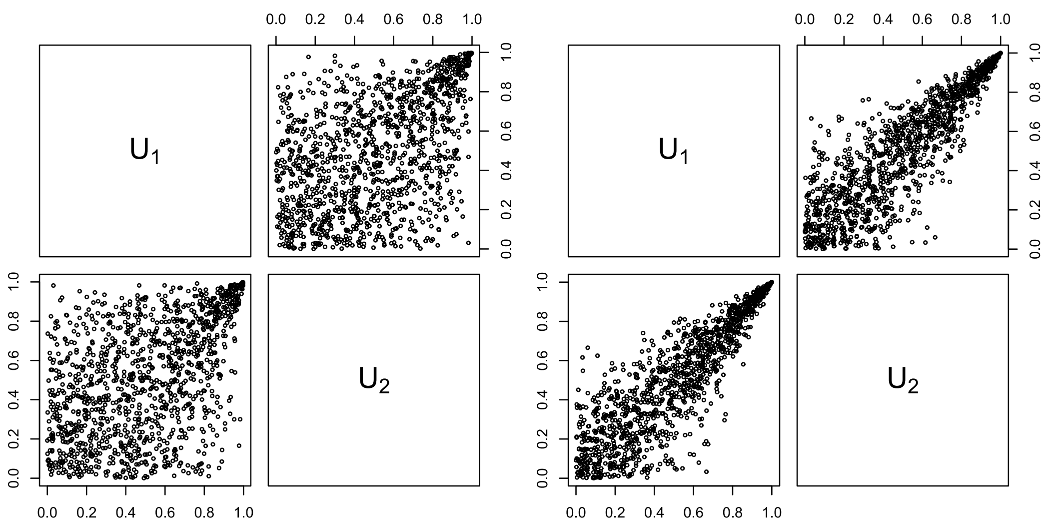

Figure 9. Joe copula examples (left $\boldsymbol{\theta_1 = 2}$, right $\boldsymbol{\theta_2 = 5}$)

It is remarkable that this copula is not symmetrical as happens with the Clayton copula. These copulas are commonly used in the dependence modelling of the insurance claims and costs related to the claims.

3.1.6. Nested copulas

As previously mentioned, the Archimedean copulas are usually a bivariate density functions that are not commonly found for multivariate analysis. Therefore, when determining high dimensional copula dependencies, it is common to use copula nesting systems. The nesting systems link the bivariate or multivariate copula dependencies by relating the copula cumulative density value with another input. By doing this, high dimension multivariate analysis can be performed by making low order relationships and later joining with the estimated copulas.

An example of a nested Archimedean copula scheme would be the following:

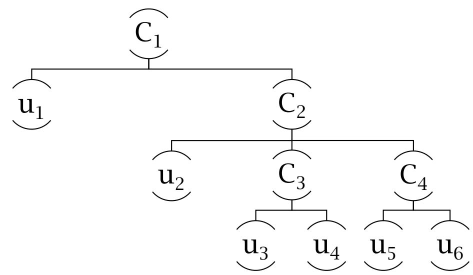

Figure 10. Nested copula scheme

The numerical expression of the scheme plotted above would be as follows:

$$C(u) = C_1\left(u_1, C_2\big(u_2, C_3(u_3, u_4), C_4(u_5, u_6)\big)\right)$$

In the following plot it is shown a Frank nested copula with the dependency $C_1\big(u_1, C_2(u_2, u_3)\big)$ with parameters $\theta_1 = 4$ and $\theta_2 = 7$:

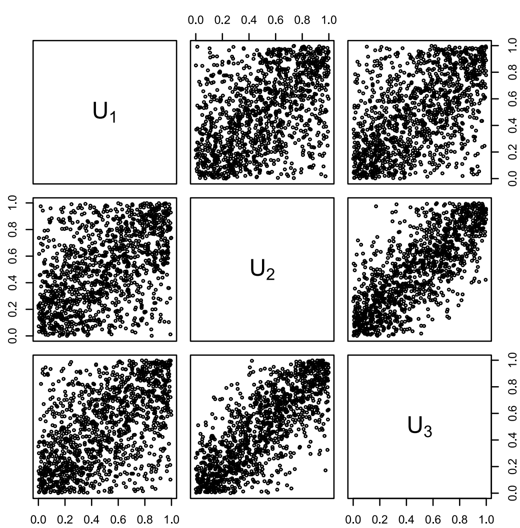

Figure 11. Nested Frank copula of parameters $\boldsymbol{\theta_1 = 4}$ and $\boldsymbol{\theta_2 = 7}$

3.1.7. Copula estimation

In practice, the underlying copula estimation is usually the main objective when working with copulas. For that available data will be available and it is important to decide which is the best estimation method given the case. There are two most used methods, method of moments and maximum likelihood method.

As previously mentioned, the copula can be estimated by the method of moments. In this case, the moments would be the Spearman Rho or the Kendall Tau amongst other correlation measures. For that, each copula has been developed so the moment of the copula is formulated as a function of its parameter.

For example, the Kendall Tau of the Ali-Mikhail-Haq copula as a function of its parameter is determined by the following equation:

$$\tau_\theta = 1 - \frac{2\left((1-\theta)^2 \log(1-\theta) + \theta\right)}{3\theta^2}$$

Therefore, to calibrate the estimated copula, the sample Kendall Tau and the copula θ shall be calculated so both moments match. This way the estimation will be determined by:

$$g_\tau(\theta) = \tau(C_\theta) \qquad \text{and} \qquad g_{\rho_s}(\theta) = \rho_s(C_\theta)$$

So, this way it is satisfied that:

$$\theta = g_\tau^{-1}(\tau) \qquad \text{and} \qquad \theta = g_{\rho_s}^{-1}(\rho_s)$$

In practice, as it happens for most density function both univariate and multivariate, the maximum likelihood method is used. The maximum likelihood method will be approximated by the following expression:

$$\theta_n = \sup_{\theta \in \Theta} \sum_{j=1}^{n} \ln c_\theta \left(F_{1\gamma_1}(X_{j\,1}), F_{2,\gamma_2}(X_{j\,2}), \dots, F_{n,\gamma_n}(X_{jn})\right)$$

Which is usually numerically solved since it is not always feasible or even possible to obtain an expression. Also, in most cases the maximum likelihood method is used since it usually involves a lower standard error of the estimation.

In some cases, for the copula estimation, especially for high dimensionality due to the numerical algorithm convergence problems, a mixed method is used where the starting parameters are obtained by method of moments and later the maximum likelihood method is used.

3.1.8. Goodness of fit test

As for any statistical approach, it is not sufficient to obtain the best fit, but it is also necessary to test if there is sufficient statistical significance that the estimated function adjusts to the data.

For the copulas, there are not many testing procedures, but the most common one is the multivariate generalization of the Cramer Von Mises (Genest et al., 2009). The generalization of this test is as follows:

$$S_n^{\text{gof}} = \int_{[0,1]^d} n\left(C_n(u) - C_{\theta_n}(u)\right)^2 dC_n(u)$$

That can be expressed for a sample as:

$$S_n^{\text{gof}} = \sum_{j=1}^{n}\left(C_n(U_{j,n}) - C_{\theta_n}(U_{j,n})\right)^2$$

The null hypothesis of the test would be the data coming from the specified copula.

This goodness of fit method is implemented in the R package “copula” and it is resolved by simulation. The algorithm calculated by simulation the distribution of the S_n^{gof} statistic so the p value can be calculated.

There are also some other statistics as the R_n (Genest et al., 2013) where the review of the testing but the most generalized one is the previously mentioned S_n^{gof}.

3.2. Factorial analysis with principal components decomposition

The factorial analysis through PCA is quite generalized for any multivariate analysis. It aims to reduce the dimensionality of the data by obtaining the so as called Principal components that capture the main variation of the data.

When dealing with the interest rates, the objective variable is now a curve, which means it is composed of many points. When analyzing all these point values, it is not feasible to perform an analysis of each point since most of them are usually correlated and the high dimensionality problem is usually difficult to handle. Therefore, what is common is to obtain the underlying factors that determine the global behavior of these points. This analysis is performed by Eiopa in the document CEIOPS-DOC-66/10 Calibration of Market Risk Module of January of 2010.

In the shocks calibration paper is shown how they are considering this analysis for the interest rate shocks calibration. The following plot is obtained from Eiopa calibration paper where the results are explained and detailed how they are used to determine the shocks.

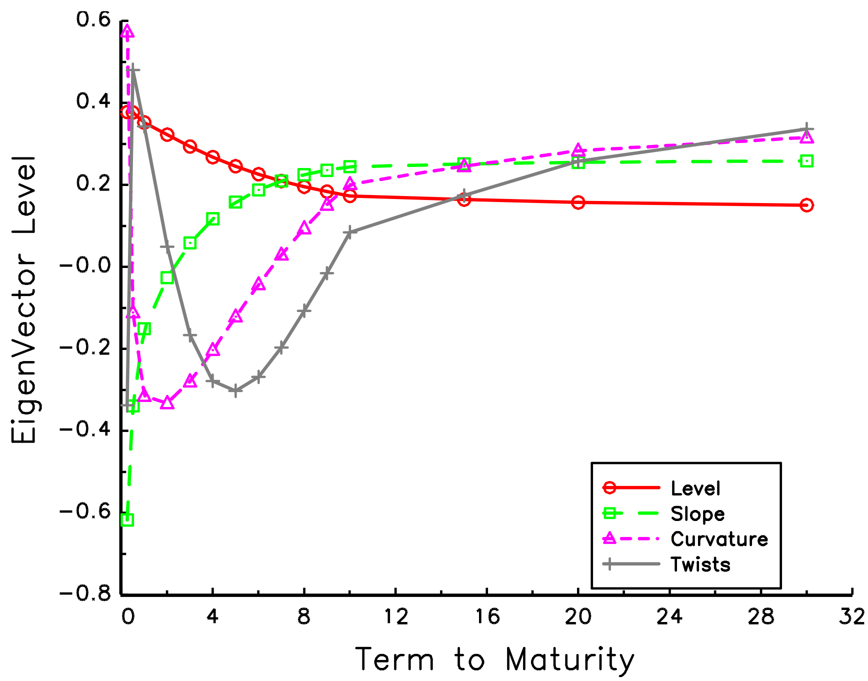

Figure 12. Principal component analysis of interest rate curve (EIOPA, 2010a)

It is mentioned that the 99,98% of the variance is explained with the four PC. The first principal component is indicated as the level, meaning that it reflects the parallel changes the curve can have. The second principal component indicates the slope, indicating how the short-term rates change while the long term rates have the opposite effect. The third principal component indicates the curvature, and fourth one the twist of the curve.

As for the interest rate shock up and shock down, as described in their paper (EIOPA, 2010a) the factor are subsequently used in a regression model so they sensitivity of each factor is obtained and then the annual percentage rates can be obtained.

The problem with this methodology is that the regression model underlies the assumption of a gaussian random variables. While this affirmation is in most cases true and generally accepted, this has not been demonstrated in the Eiopa documentation and is one of the motivations of this work. Additionally, no time dependency is considered in Eiopa analysis, which is well known to be present in any financial time series as it happens with the interest rates.

3.1. ARIMA- GARCH models

It has been clearly demonstrated that most of the financial series can be modelled as ARIMA-GARCH processes. The ARIMA-GARCH models are indeed two models, one model that calculates the mean ARIMA and another one for the variance GARCH.

The ARIMA process re usually composed by the AR component, which is referred as the autoregressive component, and the MA component, that refers as the moving average. Also, it is common to differentiate the tine series since one of the requirements for the time series analysis is the stationarity of the time series. As for the variance the GARCH (generalized autoregressive conditional heteroskedasticity), usually there is a autoregressive component of both the volatility or the squared residuals. Inside the variance estimation of this second model, usually arise some other approaches such as the exponential GARCH or the GJR-GARCH.

3.1.1. ARIMA models

The ARIMA models, commonly known as Box-Jenkins (Box et al., 1970) are a mean models governed by the following expression:

$$y_t = \mu + \phi y_{t-1} + \theta \varepsilon_{t-1} + \varepsilon_t$$

They have a regressive component that refers to the previous time series value denoted as y_{t-1} and a moving average component referring to the previous residual ε_{t-1}. The model can be generalized to k previous periods.

This models usually have the assumption that they time series depend on the previous values.

3.1.2. Standard GARCH models

The standard GARCH (Generalized autoregressive conditional heteroskedasticity) model is a volatility model (Bollerslev, 1986) whose model equation is determined by the following expression:

$$\sigma_t^2 = \omega + \alpha \varepsilon_{t-1}^2 + \beta \sigma_{t-1}^2$$

The GARCH models consider the volatility of the process to be heteroskedastic, so they calculate the conditional variance for each time based in the previous volatility and residuals. They are widely used in financial series modelling. They are particularly relevant since most finance regulations require risk measures based in volatility, and this model allows to measure and predict the volatility of the time series.

As the rest of the models, are usually estimated by maximum likelihood.

3.1.3. Exponential GARCH models

The exponential GARCH volatility model (D. B. Nelson, 1991) incorporates the effect of the asymmetry of the estimation.

The exponential GARCH is determined by the following equation:

$$\sigma_t^2 = e^{\omega + \alpha(|y_{t-1}| - \mathbb{E}[|y_{t-1}|]) + \gamma y_{t-1} + \beta \ln \sigma_{t-1}^2}$$

It is considered by the leverage effect γ so it is considered differently the positive and the negative shocks. This way a negative effect will be captured with a value of $\gamma - \alpha$ for negative innovations while $\gamma + \alpha$ for positive innovations. The assumption is that there is some kind of asymmetry in how it affects the market the positive and negative innovations.

The mean model remains an ARIMA model, this model is only a variance model.

3.1.4. GJR-GARCH models

The GJR-GARCH model extends the definitions of the standard GARCH model considering asymmetric volatility (Glosten et al., 1993) and the model intends to account for the asymmetric response of the volatility when positive and negative shocks occur.

The model is introduced by the following equation:

$$\sigma_t^2 = \omega + (\alpha + \gamma\, I_{t-1})\varepsilon_{t-1}^2 + \beta\sigma_{t-1}^2$$

The asymmetry is introduced by the γ parameter (so called leverage) which by the function I_{t-1} adds the additional weight when the given condition is produced (usually a negative shock). The premise is that negative market shocks usually have more effect on market volatility than positive ones. Usually, bad news have a bigger effects than the good ones, something that usually occurs in financial markets.

The GJR-GARCH model is a volatility model while the mean model is usually an ARIMA model.

4. Available data

For the work, the one year ahead distributions pretend to be estimated for the interest rate curve, the equity, and the currency. For the interest rate the yield curve is modelled by considering the time dependence. Same approach is going to be considered for equity and currency.

As for the interest rate curve, a factorial analysis is performed by the statistical analysis of principal components. Once obtained the principal components, it is determined how many are representative of the total variance of all curve points. The data is extracted from Refinitiv as the Euro Swap for the periods of 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 15, 20, 25 and 30 years.

Many other factors linked to the interest rates and considered in Solvency II regulation such as the spread risk and other factors related to the interest rates are not considered for simplicity. Also, the analysis is limited to the 30 years, it is not estimated and reconstructed the full yield curve further by using the ultimate forward rate for simplicity and due to some limitations of a maximum increase of the UFR for every period.

For the equity, the EUROSTOXX600 index is used. It is considered to be representative of any equity portfolio in the Eurozone since it is composed by the 600 most relevant companies in the Eurozone.

As for the equity, the Euro index should be used as the most representative Euro value indicator for the currency shock under Solvency II. Instead, since the Euro index time series is not available up to the starting date (24-09-1999), the exchange rate between the Euro and USD is used.

At total, the time series have weekly data from the 24-09-1999 to 03-02-2023, being a total of 1219 data. The estimations time series models are estimated for this periodicity and the values one year ahead are obtained by the simulation of 52 consecutive data.

5. Methodology

5.1. PCA for the interest rates

The PCA involves the transformation and scalation of the points into its PC through matrix transformation. After doing it with the interest rate yield curve with the data previously described, the following PC matrix is obtained.

	EUR1Y	EUR2Y	EUR3Y	EUR4Y	EUR5Y	EUR6Y	EUR7Y	EUR8Y	EUR9Y	EUR10Y	EUR12Y	EUR15Y	EUR20Y	EUR25Y	EUR30Y
PC1	-0,250	-0,255	-0,258	-0,259	-0,260	-0,260	-0,260	-0,260	-0,260	-0,260	-0,260	-0,259	-0,258	-0,257	-0,257
PC2	-0,539	-0,417	-0,302	-0,202	-0,119	-0,048	0,010	0,057	0,096	0,128	0,178	0,231	0,282	0,305	0,317
PC3	0,563	0,088	-0,097	-0,188	-0,231	-0,237	-0,226	-0,201	-0,172	-0,143	-0,093	-0,010	0,165	0,339	0,478
PC4	0,381	-0,005	-0,309	-0,328	-0,226	-0,111	-0,006	0,083	0,162	0,238	0,344	0,356	0,090	-0,219	-0,444
PC5	-0,383	0,556	0,313	0,031	-0,163	-0,221	-0,318	-0,250	-0,156	-0,001	0,184	0,313	0,197	0,009	-0,108
PC6	0,169	-0,450	0,051	0,346	0,212	0,232	-0,247	-0,294	-0,284	-0,104	0,002	0,198	0,405	0,080	-0,316
PC7	0,032	-0,091	-0,050	0,124	0,005	0,285	-0,432	-0,247	-0,048	0,415	0,346	-0,014	-0,455	-0,195	0,323
PC8	-0,085	0,412	-0,491	-0,102	-0,099	0,684	0,052	-0,050	-0,188	-0,155	-0,120	0,034	0,058	0,094	-0,043
PC9	0,024	-0,122	0,080	0,486	-0,759	0,132	0,002	0,143	0,232	-0,223	-0,020	0,090	-0,013	-0,112	0,060
PC10	-0,019	0,036	0,097	-0,034	-0,227	0,097	-0,231	0,032	0,167	0,399	0,000	-0,588	0,164	0,445	-0,340
PC11	-0,027	0,184	-0,453	0,404	0,103	-0,276	-0,065	-0,002	0,061	0,316	-0,229	-0,176	0,392	-0,378	0,147
PC12	-0,008	0,106	-0,402	0,436	0,120	-0,314	0,095	-0,107	0,041	-0,141	0,232	0,060	-0,394	0,478	-0,203
PC13	0,005	-0,016	0,016	0,023	0,004	-0,015	-0,259	0,264	-0,015	0,327	-0,653	0,461	-0,258	0,203	-0,087
PC14	0,007	-0,052	0,091	0,074	-0,283	-0,055	0,590	-0,214	-0,556	0,437	-0,036	0,021	-0,052	0,037	-0,009
PC15	-0,002	-0,002	0,042	-0,080	0,005	0,069	0,218	-0,722	0,573	0,067	-0,267	0,115	-0,019	0,013	-0,011

Table 1. PC transformation matrix

When trying to understand the factors defined by the PC, by checking the first row (which indicates the linear transformation from PC to the variable), it can be observed that the value is nearly -0,25 for any swap maturity. This means that any increase in value in the PC1 will derive in a constant increase of the whole swap curve. When analyzing the second, it can be observed that the increase in value is almost linear. By following this approach, the principal components linear transformation coefficients are plotted for the first 4 principal components:

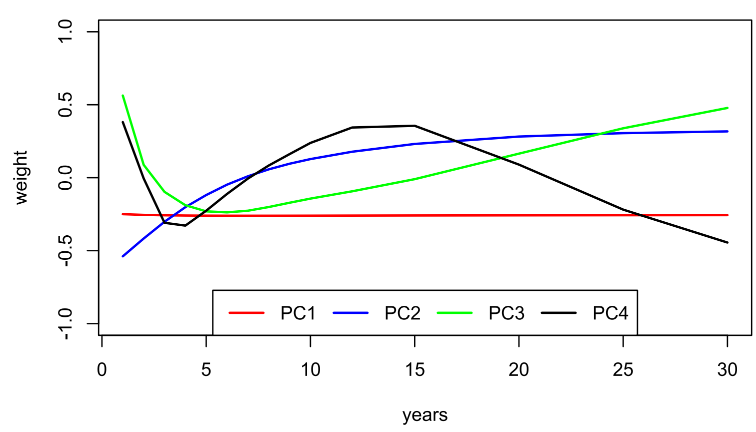

Figure 13. PC (1 to 4) linear transformation coefficients

It can be noticed that the first is a straight line indicating the level, while the second determines the slope, the third the curvature and the fourth would be the twist of the curve.

As for the number of principal components to consider for modelling, there is a big trade-off between the number of principal components and the fit of the estimation to the real curve. When working with copulas, the number of dimensions is usually a key factor and especially sensitive for the sample size. The greater the number of dimensions, the greater it will be the power loss of the statistical testing. When working with principal components, it common to decide the number of PC to use as a function of the explained variance. For the interest rate model presented here, the variance explained by each component is shown in the following table:

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9	PC10	PC11	PC12	PC13	PC14	PC15
Var Explained	98,14%	1,68%	0,15%	0,02%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%
Cummulative	98,14%	99,83%	99,97%	99,99%	100,00%	100,00%	100,00%	100,00%	100,00%	100,00%	100,00%	100,00%	100,00%	100,00%	100,00%

Table 2. Variance explained by each principal component

Since by using 3 principal components, the 99,97% of variance is explained, it is considered that three principal components will be sufficient to have a good performance when estimating the curve.

Additionally, all the PC time series are plotted so it can be observed the variation explained by each principal component. Additionally, the curves are shown in two moments of time and the fit as a function of the number of PC used for the estimation. It can be observed that 3 PC usually offers a very good performance when estimating the curve points.

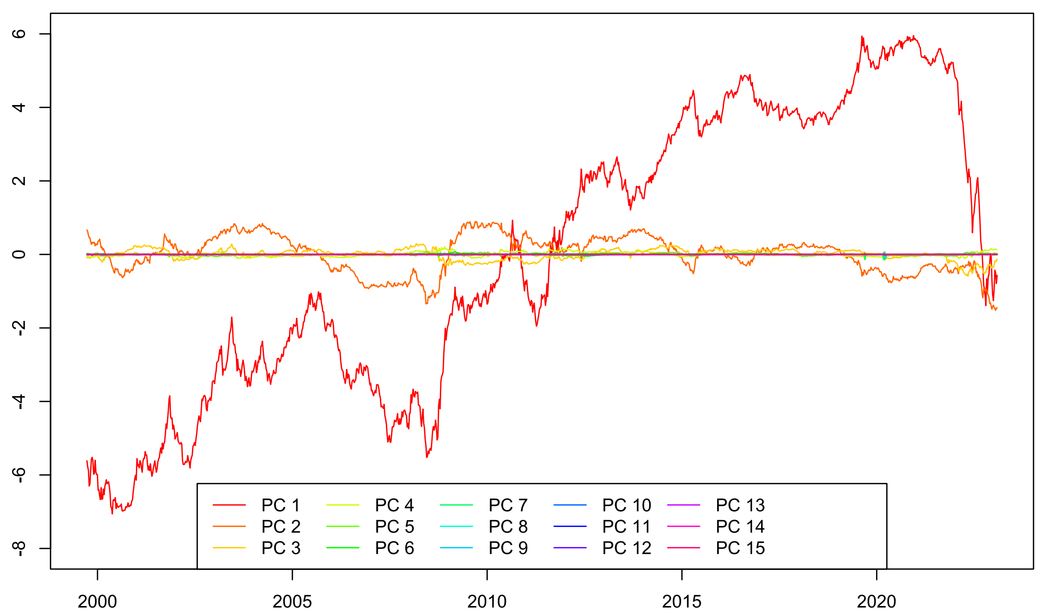

Figure 14. Interest rate PC time series

In the plot it can be observed how the PC1 is the most significant one (explaining most of the variance). The second and third are the following largest explaining the rest up to the 100%. In the pictures below, it can be observed that three principal components adjust very well to the data, even for inverted curves.

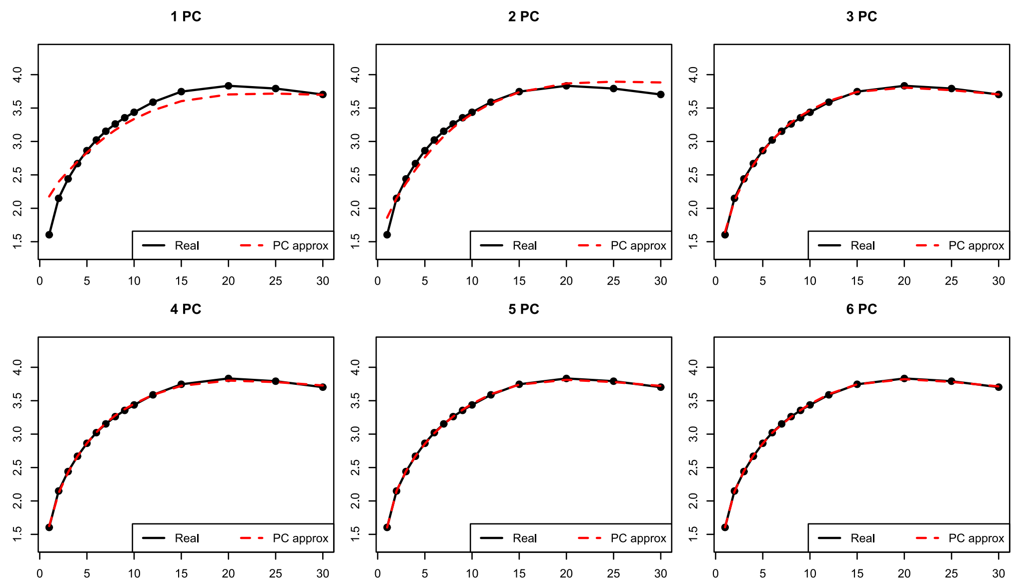

Figure 15. 2011-03-18 Interest rate curve and approximation by PCA

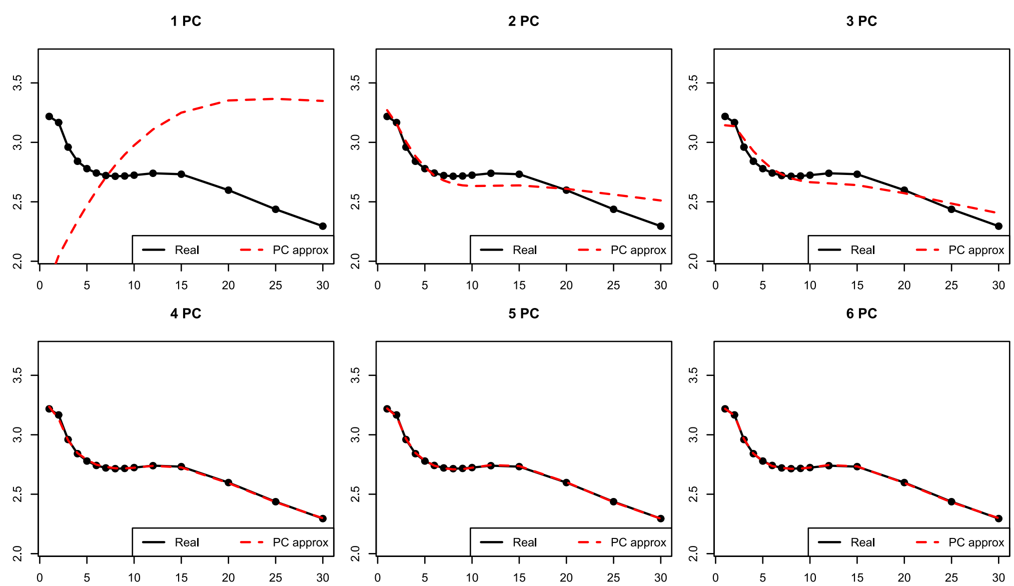

Figure 16. 2023-02-03 Interest rate curve and approximation by PCA

Therefore, it is considered that three principal components are sufficient to reliably represent the yield curve and its possible movements during the time. This way, the three principal components time series are estimated.

In the following plots, it is shown how each principal component affects the yield curve. In the plot the three principal components are changed only for each plot, meaning that they only reflect how the change affects to the principal component. This way it is clearly observable what is the effect each principal component has on the yield curve.

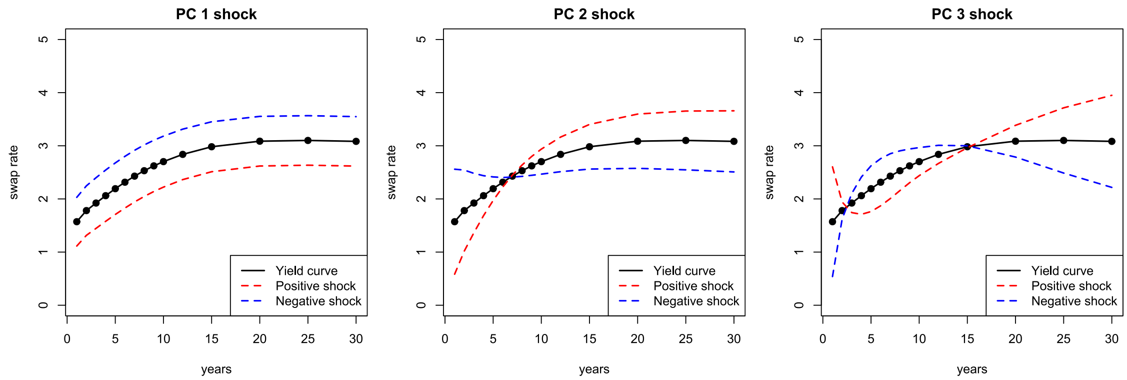

Figure 17. PC effect on yield curve

Then, the new yield curve will be replicated by the combination of these three shocks so it adapts to the new yield curve.

5.2. ARIMA-GARCH series estimation

The following plotted time series are estimated. The time series are checked to be stationary first and if rejected, they are differentiated and checked again. As for the Equity and Currency series, they are calculated as returns while the PC are calculated as absolute values.

The five-time series are estimated by ARIMA-GARCH processes with different distribution error distribution functions. The error is supposed to be a scaled distribution, considered as Normal, t-Student, Normal-Inverse Gaussian (nig) and Generalized error distribution (ged). As for the ARIMA, the AR and MA components are checked up to a third order for both components. The GARCH components are also checked up to a third order.

In R software, the "rugarch" package is used for the time series estimation and later simulation.

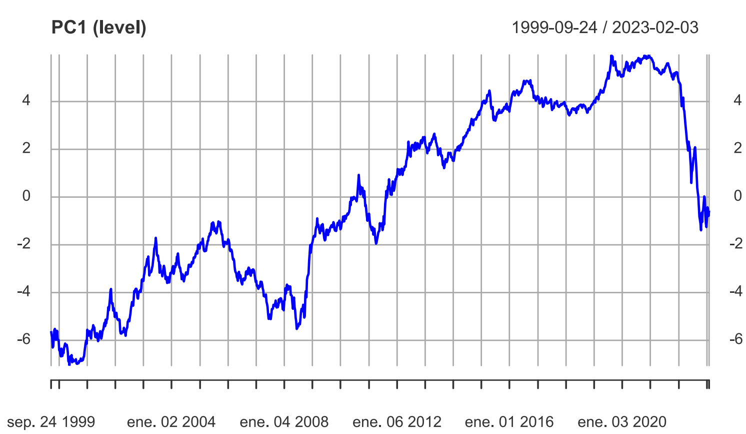

Figure 18. Interest rate PC1 (level)

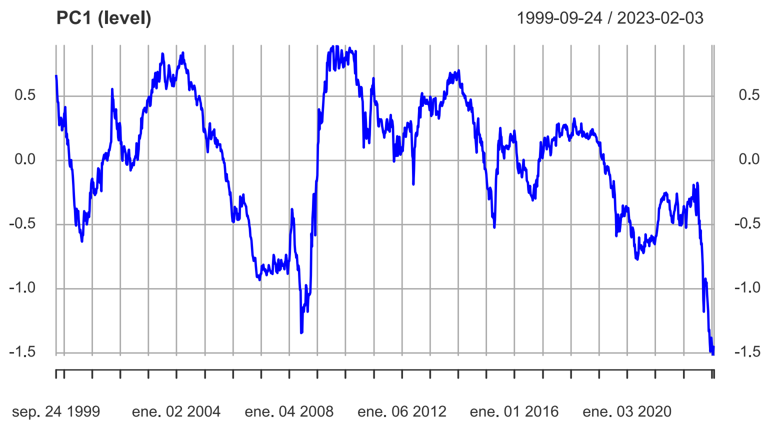

Figure 19. Interest rate PC2 (slope)

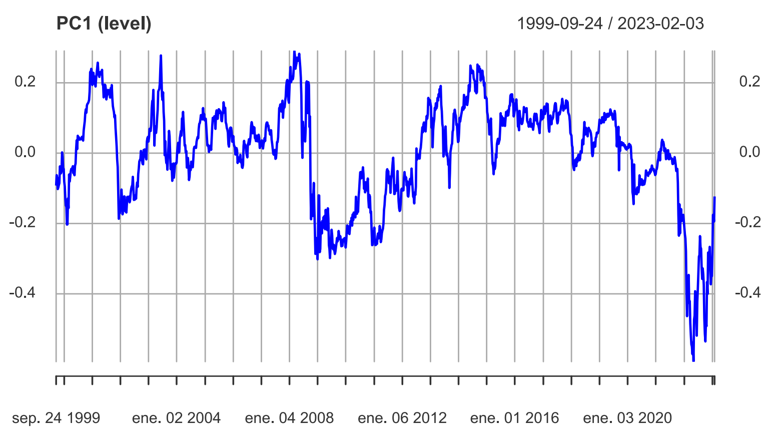

Figure 20. Interest rate PC3 (curvature)

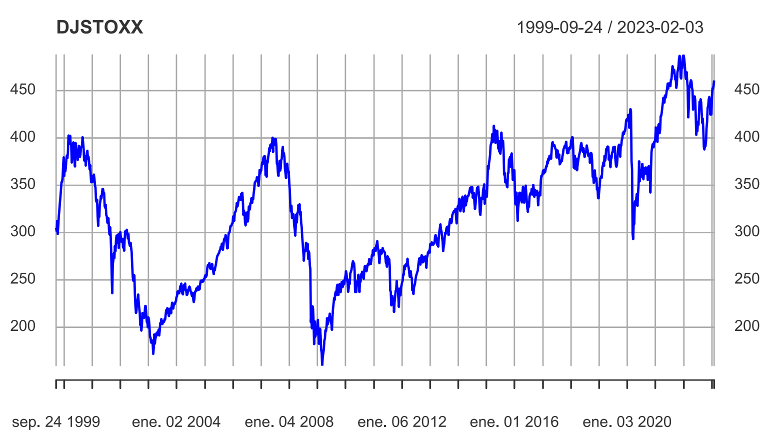

Figure 21. DJSTOXX Index (EUROSTOXX600)

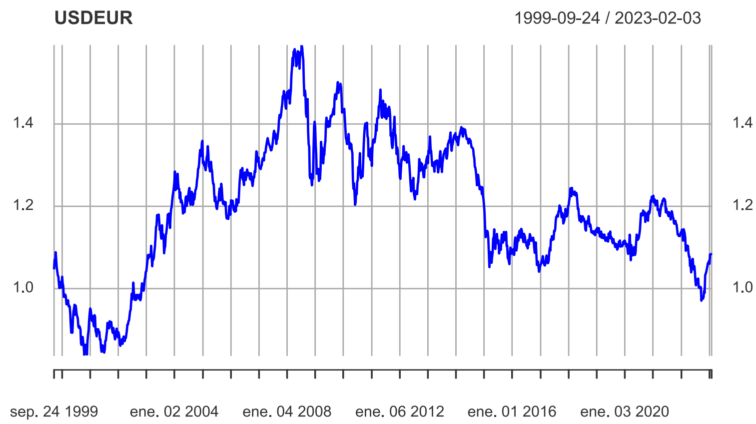

Figure 22. USDEUR

For the series estimation, the GARCH component has been estimated as standard GARCH, exponential GARCH and gjr GARCH, the model proposed by Glosten, Jagannathan and Runkle in 1993. Once estimated, the best model has been selected based in a equally weighted ranked average of the information criterion of Akaike, Bayes, Shibata and Hannan-Quinn.

5.2.1. PC series

5.2.1.1. Stationarity

First it has been checked the series are stationary. Without differentiating, the stationarity null hypothesis is rejected for all the series. After differentiating, the following results are obtained:

	statistic	**pvalue**	**alternative.hypothesis**
Dickey-Fuller	-9,383	0,01	Non-stationary
Phillips-Perron	-1303,842	0,01	Non-stationary
KPSS	0,371	0,09	Stationary

Table 3. PC1 differentiated series stationarity test

	statistic	**pvalue**	**alternative.hypothesis**
Dickey-Fuller	-9,007	0,01	Non-stationary
Phillips-Perron	-1269,441	0,01	Non-stationary
KPSS	0,135	0,10	Stationary

Table 4. PC2 differentiated series stationarity test

	statistic	**pvalue**	**Null hypothesis**
Dickey-Fuller	-10,407	0,01	Non-stationary
Phillips-Perron	-1244,377	0,01	Non-stationary
KPSS	0,035	0,10	Stationary

Table 5. PC3 differentiated series stationarity test

It is checked that all the tests reject the non-stationary hypothesis when it is the null hypothesis and also do not reject it when being non-stationary the null hypothesis. The R software package “tseries” does not provide p values lower than 0,01 nor greater than 0,1. Therefore considering the confidence interval considered is the 0,5% it shall be rejected when 0,01 is obtained, and not rejected when greater than 0,01.

The stationarity tests, Phillips and Perron (C. R. Nelson & Plosser, 1982), KPSS (Kwiatkowski et al., 1992), Dickey-Fuller (Said & Dickey, 1984) are performed.

5.2.1.2. Model

Once the time series has been differentiated, the best fitted model are shown.

The three models are a standard GARCH model with an error distributed as a skewed normal-inverse gaussian function, and t-Student.

The fitted model coefficients are indicated in the following table:

	PC1			PC2			PC3		
	Estimate	Std. Error	Pr(>|t|)	Estimate	Std. Error	Pr(>|t|)	Estimate	Std. Error	Pr(>|t|)
ar1	-	-	-	-	-	-	-	-	-
ar2	-	-	-	-	-	-	-	-	-
ar3	-	-	-	-	-	-	-	-	-
ma1	-	-	-	-	-	-	-	-	-
ma2	-	-	-	-	-	-	-	-	-
ma3	-	-	-	-	-	-	-	-	-
omega	0,0006	0,0003	0,0346	0,0002	0,0001	0,0137	0,0001	0,0000	0,0025
alpha1	-	-	-	0,1027	0,0293	0,0005	0,1346	0,0356	0,0002
alpha2	0,1372	0,0260	0,0000	0,1240	0,0353	0,0004	0,1740	0,0464	0,0002
alpha3	-	-	-	-	-	-	-	-	-
beta1	-	-	-	-	-	-	-	-	-
beta2	0,8538	0,0273	0,0000	0,7111	0,0635	0,0000	0,6230	0,0708	0,0000
beta3	-	-	-	-	-	-	-	-	-
skew	-0,1914	0,0605	0,0015	-	-	-	-	-	-
shape	3,2084	0,9830	0,0011	6,2800	1,0917	0,0000	4,8732	0,7164	0,0000

Table 6. PC time series model estimated coefficients.

It is observed that all the coefficients are significant with a confidence interval of 0,5%.

The innovations follow a scale distribution function. The distribution followed by each component is:

- PC1: skewed normal-inverse gaussian distribution (nig)
- PC2: t-Student distribution (dof = 6,28)
- PC3: t-Student distribution (dof = 4,87)

As for the directionality test of the model, it is not performed any check since the models do not have any autoregressive component and therefore, the estimated mean is always zero.

5.2.1.3. Residuals diagnosis

For the diagnosis, the residuals are checked to be independent and identically distributed (iid) and fit the estimated distributions.

The following residuals histogram plot is obtained for the residuals. As for the residual of the PC1 it can be slightly observed that there is some skewness to the negative values that is captured by the model fit. From the graph it is clearly observed there is not much normality, that is one of the underlying assumptions of EIOPA.

The PC2 and PC3 follow a t-Student distribution. It is observed that they have very long tails in comparison to the tails that can be observed in a normal distribution.

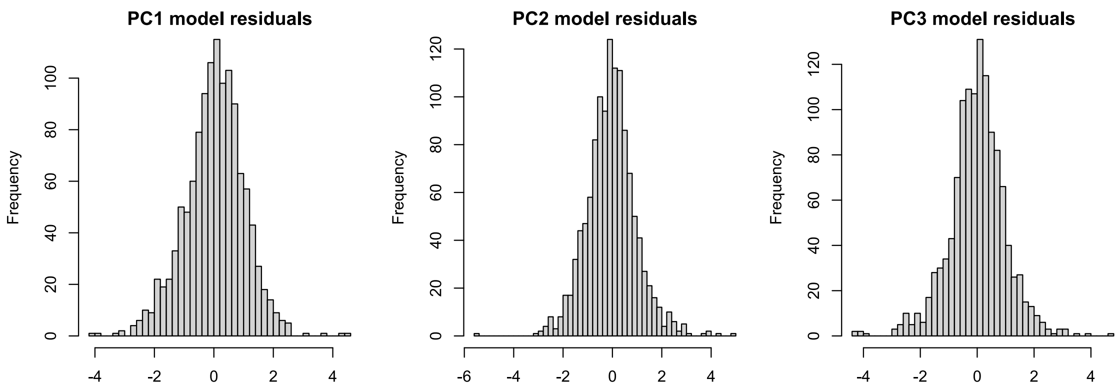

Figure 23. Residuals histogram

When testing the independence of the residuals, the package “rugarch” provides the Ljung-Box independence test, for different lags obtaining the following results.

	PC1		PC2		PC3	
lags	**statistic**	**pvalue**	**statistic**	**pvalue**	**statistic**	**pvalue**
1	0,750	0,386	0,174	0,677	0,716	0,398
2	5,022	0,081	0,174	0,917	2,334	0,311
3	8,973	0,030	0,490	0,921	4,588	0,205
4	11,170	0,025	1,251	0,870	7,628	0,106
5	12,943	0,024	2,596	0,762	7,816	0,167
6	14,515	0,024	3,970	0,681	8,436	0,208
7	14,519	0,043	3,985	0,782	9,790	0,201
8	14,995	0,059	12,026	0,150	9,798	0,279
9	15,064	0,089	12,186	0,203	10,306	0,326
10	15,661	0,110	12,537	0,251	11,521	0,318
15	22,392	0,098	19,840	0,178	13,885	0,534
20	30,671	0,060	23,429	0,268	18,669	0,543

Table 7. Ljung-Box test for residuals independence

It is validated that there is not statistical significance to reject the null hypothesis of residuals independence. Therefore, the residuals are not considered to have any significant autocorrelation so they can be resampled for bootstrap approach.

Additionally, the autocorrelations are plotted below. Note there are not significant autocorrelations for any lag order.

Additionally, the ARCH test is performed for the null hypothesis or homoscedasticity. It needs to be verified the homogeneity in variance. The test performs a regression by residuals for different lags by ensuring that no conditional variance is in the sample, therefore all the regression parameters shall be zero. Otherwise, any conditional dependence would be observed in the data and therefore the null hypothesis would be rejected since the variance could be estimated.

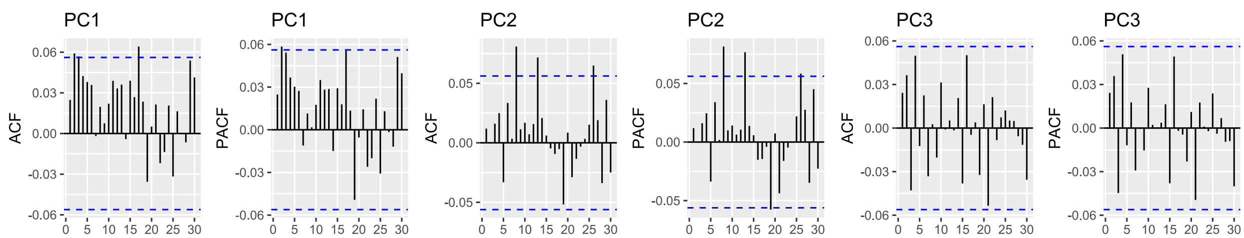

Figure 24. Autocorrelations and partial autocorrelation factors (PC1, PC2 and PC3)

The results obtained by the test for various lags orders obtained from the "rugarch" package are the followings:

	PC1		**PC2**		**PC3**	
lags	**statistic**	**pvalue**	**statistic**	**pvalue**	**statistic**	**pvalue**
7	1,124	0,289	0,119	0,730	0,150	0,205
9	2,568	0,408	0,461	0,913	0,234	0,506
11	3,163	0,566	2,228	0,738	0,792	0,865

Table 8. ARCH LM test for heteroscedasticity

It is verified that there is no statistical significance to reject the null hypothesis of constant variance. The autocorrelation and the partial correlations are plotted to verify there is no dependence.

Also, the goodness of fit is also tested for the distributions. The tests performed for the goodness of fit are the Jarque-Bera test, by obtaining the uniform distribution and then transforming to normal for these distributions not following a normal distribution. Also, the Kolmogorov–Smirnov test is performed by measuring the maximum distance between the cumulative empirical distribution and the estimated analytical one. The last is the Cramer-Von-Mises test, that integrated the squared distance between the empirical distribution function and the estimated analytical one and checking it is significant.

Additionally, the t test is performed to check the residuals have null mean and there is no bias in the estimation. It is validated by the following results:

	PC1		PC2		PC3	
test	**statistic**	**pvalue**	**statistic**	**pvalue**	**statistic**	**pvalue**
t	1,042	0,298	-1,619	0,106	0,316	0,752

Table 9. residuals t test

As for the goodness of fit tests, all the tests indicate there is not sufficient statistical significance to reject the null hypothesis, therefore it is validated the residuals follow the previously indicated distributions.

The results obtained for the test are the following.

	PC1		PC2		PC3	
test	**statistic**	**pvalue**	**statistic**	**pvalue**	**statistic**	**pvalue**
Jarque-Bera	0,441	0,802	4,072	0,131	2,642	0,267
Kolmogorov–Smirnov	0,023	0,562	0,037	0,073	0,024	0,497
Cramer-Von-Mises	0,138	0,430	0,532	0,033	0,153	0,382
Distribution	Nig		t-Student		t-Student	

Table 10. Goodness of fit tests

Additionally, a visual check by a qqplot and by plotting the cumulative functions are performed validating the previously shown results.

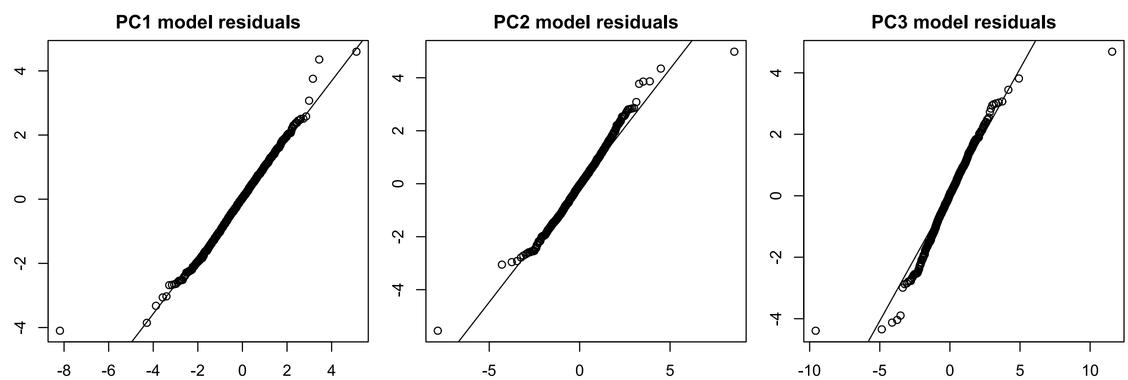

Figure 25. residuals and estimation qqplot

It is observed that the quantiles match almost perfectly and as usual, the biggest deviations are produced in the distribution tails.

As for the cumulative plots, the following results are obtained:

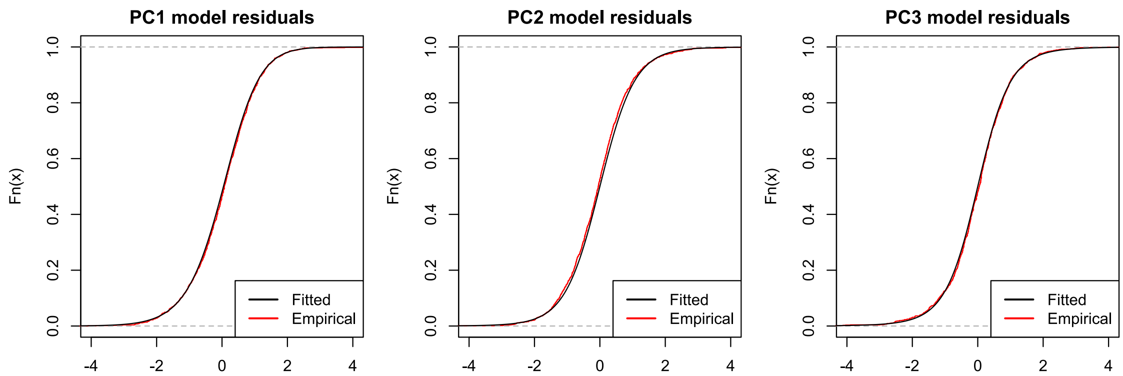

Figure 26. Cumulative plot of empirical and estimated distribution

It is observed that the fitted distribution is very close to the empirical distribution and therefore the residuals have been tested to be iid and distributed as specified.

5.2.1.4. Back testing

For the residuals, the VaR back testing has been performed for alpha 1% and 5% at a 0.5% confidence interval. It has not been tested at alpha 0,5% since there are only 1219 data, therefore the statistical power of the test would be very poor. The expected shortfall is not tested since Solvency II does not consider this risk measure.

For the back testing the sample has been divided in rolling windows with a rolling window of 100 data. The following results are obtained for the PC components:

	PC1			PC2			PC3		
alpha	**unconditional**	**conditional**	**duration**	**unconditional**	**conditional**	**duration**	**unconditional**	**conditional**	**duration**
1%	0,019	0,003	0,047	0,070	0,143	0,951	0,818	0,846	0,602
5%	0,900	0,992	0,757	0,797	0,771	0,708	0,598	0,833	0,687
95%	0,891	0,312	0,117	0,179	0,277	0,160	0,900	0,820	0,167
99%	0,611	0,747	0,575	0,336	0,588	0,664	0,515	0,745	0,223

Table 11. Value at Risk (VaR) back testing p values

As for the null hypothesis of the unconditional (Kupiec, 1999) and conditional (P. F. Christoffersen, 1998)Value at Risk Excedance Test, it cannot be rejected the null hypothesis of correct exceedances for the unconditional and for the conditional the correct exceedances and the independence. Also, the VaR Duration Test (P. Christoffersen & Pelletier, 2004) is performed to test the no predictability of times between VaR exceedances.

As for the test, all have been successful except the conditional of the PC1 which has a p value of 0,003 which considering all the testing performed, it is a reasonable value, and it is not considered to be relevant for the model acceptance.

Therefore, the PC model is considered valid.

5.2.2. Equity

5.2.2.1. Stationarity

For the equity model, the EUROSTOXX600 has been used as a representative factor of the European equity market.

First it has been checked the series are stationary. Without differentiating, the stationarity null hypothesis is rejected for all the series. After differentiating, the following results are obtained:

	statistic	pvalue	alternative.hypothesis
Dickey-Fuller	-10,699	0,01	stationary
Phillips-Perron	-1289,970	0,01	stationary
KPSS	0,094	0,10	non stationary

Table 12. DJSTOXX differentiated series stationarity test

It is checked that all the tests reject the non-stationary hypothesis when it is the null hypothesis and do not reject it when being non-stationary the null hypothesis. The R software package "tseries" does not provide p values lower than 0,01 nor greater than 0,1. Therefore considering the confidence interval considered is the 0,5% it shall be rejected when 0,01 is obtained, and not rejected when greater than 0,01.

5.2.2.2. Model

Once the time series has been differentiated, the best fitted model is shown.

The model is a standard GARCH model with an error distributed as a skewed normal-inverse gaussian function.

The fitted model coefficients are indicated in the following table:

	DJSTOXX		
	Estimate	**Std. Error**	**Pr(>|t|)**
ar1	-	-	-
ar2	-	-	-
ar3	-	-	-
ma1	-0,121267263	0,029895533	4,98419E-05
ma2	-	-	-
ma3	-	-	-
omega	0,0000	0,0000	0,0025
alpha1	0,1353	0,0306	0,0000
alpha2	-	-	-
alpha3	-	-	-
beta1	0,8294	0,0356	0,0000
beta2	-	-	-
beta3	-	-	-
skew	-0,5552	0,0490	0,0000
shape	2,2946	0,5724	0,0001

Table 13. DJSTOXX time series model estimated coefficients.

It is observed that all the coefficients are significant with a confidence interval of 0,5%.

The innovations follow a scale skewed normal-inverse gaussian (nig) distribution function.

5.2.2.3. Residuals diagnosis

For the diagnosis, the residuals are checked to be independent and identically distributed (iid) and fit the estimated distributions.

The following residuals histogram plot is obtained for the residuals.

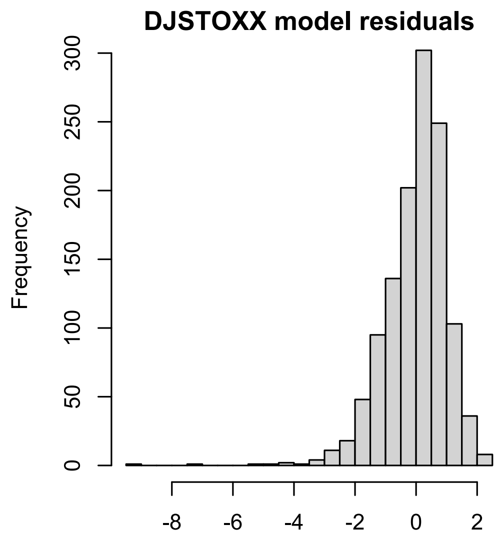

Figure 27. Residuals histogram

When testing the independence of the residuals, the package “rugarch” provides the Ljung-Box independence test, for different lags obtaining the following results.

	DJSTOXX	
lags	**statistic**	**pvalue**
1	11,489	0,001
2	13,249	0,001
3	14,479	0,002
4	14,480	0,006
5	14,798	0,011
6	14,827	0,022
7	15,803	0,027
8	16,373	0,037
9	16,651	0,054
10	16,664	0,082
15	27,671	0,024
20	30,124	0,068

Table 14. Ljung-Box test for residuals independence

It is validated that there is not statistical significance to reject the null hypothesis of residuals independence. Therefore, the residuals are not considered to have any significant autocorrelation so they can be resampled for bootstrap approach.

Additionally, the autocorrelations are plotted below. Note there are not significant autocorrelations for any lag order.

Additionally, the ARCH test is performed for the null hypothesis or homoscedasticity. It needs to be verified the homogeneity in variance. The test performs a regression by residuals for different lags by ensuring that no conditional variance is in the sample, therefore all the regression parameters shall be zero. Otherwise, any conditional dependence would be observed in the data and therefore the null hypothesis would be rejected since the variance could be estimated.

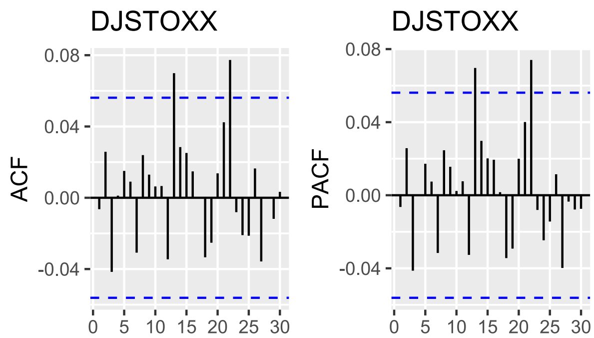

Figure 28. Autocorrelations and partial autocorrelation factors

The results obtained by the test for various lags orders obtained from the “rugarch” package are the followings:

	DJSTOXX	
lags	**statistic**	**pvalue**
7	0,280	0,597
9	0,617	0,873
11	1,660	0,841

Table 15. ARCH LM test for heteroscedasticity

It is verified that there is no statistical significance to reject the null hypothesis of constant variance. The autocorrelation and the partial correlations are plotted to verify there is no dependence.

Additionally, the t test is performed to check the residuals have null mean and there is no bias in the estimation. It is validated by the following results:

	DJSTOXX	
test	**statistic**	**pvalue**
t	-0,090	0,928

Table 16. residuals t test

As for the goodness of fit tests, all the tests indicate there is not sufficient statistical significance to reject the null hypothesis, therefore it is validated the residuals follow the previously indicated distributions.

The results obtained for the test are the following.

	DJSTOXX	
test	**statistic**	**pvalue**
Jarque-Bera	1,069	0,586
Kolmogorov–Smirnov	0,017	0,888
Cramer-Von-Mises	0,068	0,761
Distribution	Nig	

Table 17. Goodness of fit tests

Additionally, a visual check by a qqplot and by plotting the cumulative functions are performed validating the previously shown results.

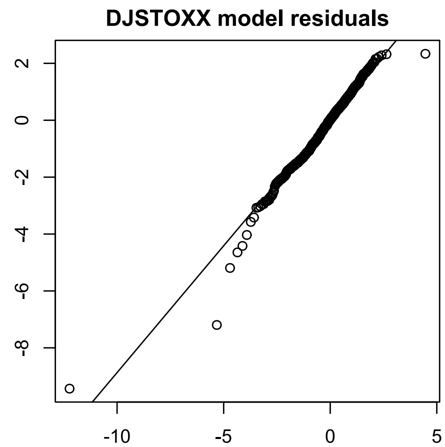

Figure 29. residuals and estimation qqplot

It is observed that the quantiles match almost perfectly and as usual, the biggest deviations are produced in the distribution tails, especially in this one since it is heavily skewed. As for the cumulative plots, the following results are obtained:

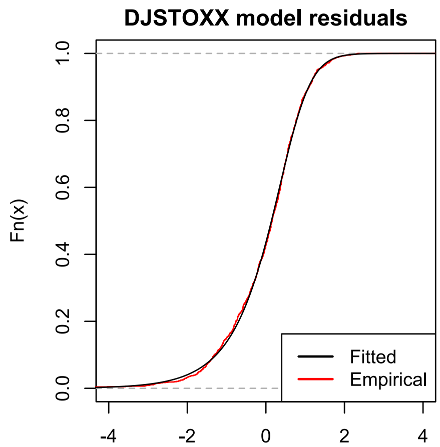

Figure 30. Cumulative plot of empirical and estimated distribution

It is observed that the fitted distribution is very close to the empirical distribution and therefore the residuals have been tested to be iid and distributed as specified.

5.2.2.4. Back testing

For the back testing as it has been performed for the PC the sample in the previous section, it has been divided in rolling windows with a rolling window of 100 data. The following results are obtained:

	DJSTOXX		
alpha	**unconditional**	**conditional**	**duration**
1%	0,198	0,414	0,965
5%	0,078	0,079	0,654
95%	0,600	0,852	0,017
99%	0,956	0,886	0,089

Table 18. Value at Risk (VaR) back testing p values

As for the test, all have been successful therefore, the equity model is considered valid.

5.2.3. Currency

5.2.3.1. Stationarity

For the currency model, the USDEUR has been used as a representative factor of the Euro currency.

First it has been checked the series are stationary. Without differentiating, the stationarity null hypothesis is rejected for all the series. After differentiating, the following results are obtained:

	statistic	**pvalue**	**alternative.hypothesis**
Dickey-Fuller	-9,984	0,01	stationary
Phillips-Perron	-1246,386	0,01	stationary
KPSS	0,127	0,10	non stationary

Table 19. USDEUR differentiated series stationarity test

It is checked that all the tests reject the non-stationary hypothesis when it is the null hypothesis and do not reject it when being non-stationary the null hypothesis. The R software package “tseries” does not provide p values lower than 0,01 nor greater than 0,1. Therefore considering the confidence interval considered is the 0,5% it shall be rejected when 0,01 is obtained, and not rejected when greater than 0,01.

5.2.3.2. Model

Once the time series has been differentiated, the best fitted model is shown.

The model is a standard GARCH model with an error distributed as a skewed normal-inverse gaussian function.

The fitted model coefficients are indicated in the following table:

	USDEUR		
	Estimate	Std. Error	Pr(>|t|)
ar1	-	-	-
ar2	-	-	-
ar3	-	-	-
ma1	-	-	-
ma2	-	-	-
ma3	-	-	-
omega	0,0000	0,0000	0,2137
alpha1	0,0736	0,0176	0,0000
alpha2	-	-	-
alpha3	-	-	-
beta1	0,9112	0,0202	0,0000
beta2	-	-	-
beta3	-	-	-
skew	-	-	-
shape	6,2800	1,0917	0,0000

Table 20. USDEUR time series model estimated coefficients.

It is observed that all the coefficients are significant with a confidence interval of 0,5%.

The innovations follow a scale normal distribution function.

5.2.3.3. Residuals diagnosis

For the diagnosis, the residuals are checked to be independent and identically distributed (iid) and fit the estimated distributions.

The following residuals histogram plot is obtained for the residuals.

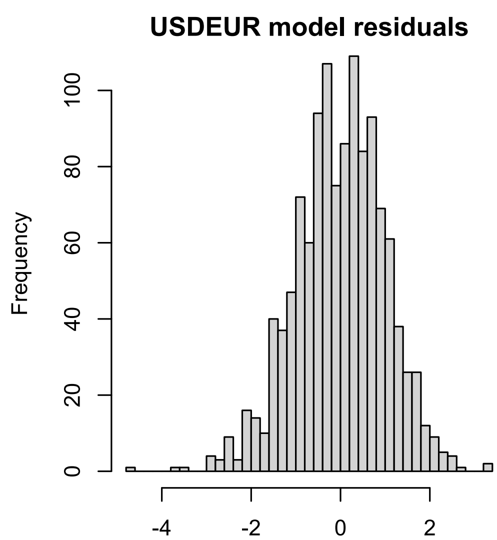

Figure 31. Residuals histogram

When testing the independence of the residuals, the package “rugarch” provides the Ljung-Box independence test, for different lags obtaining the following results.

	USDEUR	
lags	**statistic**	**pvalue**
1	0,030	0,862
2	0,045	0,978
3	0,255	0,968
4	4,349	0,361
5	5,767	0,330
6	6,225	0,398
7	6,310	0,504
8	9,706	0,286
9	10,390	0,320
10	10,462	0,401
15	18,729	0,226
20	26,592	0,147

Table 21. Ljung-Box test for residuals independence

It is validated that there is no statistical significance to reject the null hypothesis of residuals independence. Therefore, the residuals are not considered to have any significant autocorrelation so they can be resampled for bootstrap approach.

Additionally, the autocorrelations are plotted below. Note there are not significant autocorrelations for any lag order.

Additionally, the ARCH test is performed for the null hypothesis or homoscedasticity. It needs to be verified the homogeneity in variance. The test performs a regression by residuals for different lags by ensuring that no conditional variance is in the sample, therefore all the regression parameters shall be zero. Otherwise, any conditional dependence would be observed in the data and therefore the null hypothesis would be rejected since the variance could be estimated.

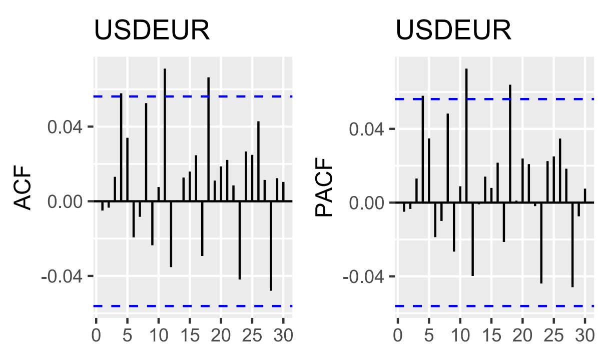

Figure 32. Autocorrelations and partial autocorrelation factors

The results obtained by the test for various lags orders obtained from the “rugarch” package are the followings:

	USDEUR	
lags	**statistic**	**pvalue**
7	0,821	0,365
9	2,320	0,455
11	2,599	0,669

Table 22. ARCH LM test for heteroscedasticity

It is verified that there is no statistical significance to reject the null hypothesis of constant variance. The autocorrelation and the partial correlations are plotted to verify there is no dependence.

Additionally, the t test is performed to check the residuals have null mean and there is no bias in the estimation. It is validated by the following results:

	USDEUR	
test	**statistic**	**pvalue**
t	-0,298	0,766

Table 23. residuals t test

As for the goodness of fit tests, all the tests indicate there is not sufficient statistical significance to reject the null hypothesis, therefore it is validated the residuals follow the previously indicated distributions.

The results obtained for the test are the following.

	USDEUR	
test	**statistic**	**pvalue**
Jarque-Bera	22,405	0,000
Kolmogorov–Smirnov	0,026	0,407
Cramer-Von-Mises	0,115	0,518
Distribution	Normal	

Table 24. Goodness of fit tests

Additionally, a visual check by a qqplot and by plotting the cumulative functions are performed validating the previously shown results.

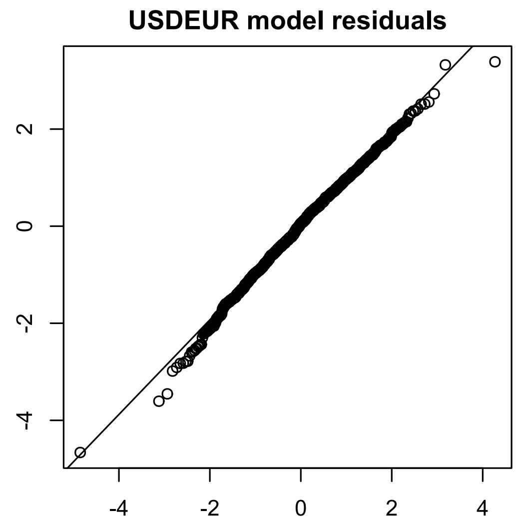

Figure 33. residuals and estimation qqplot

It is observed that the quantiles match almost perfectly and as usual, the biggest deviations are produced in the distribution tails, especially in this one since it is heavily skewed.

As for the cumulative plots, the following results are obtained:

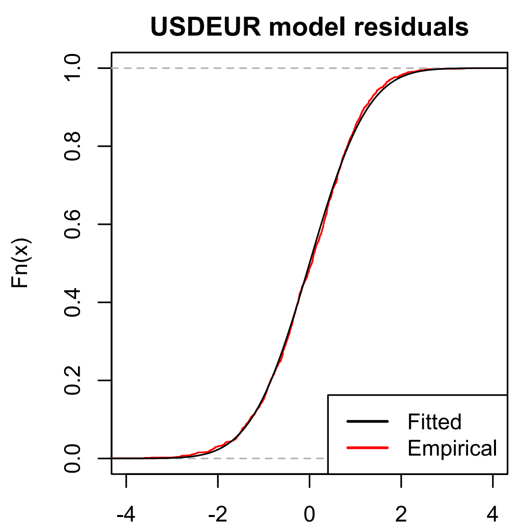

Figure 34. Cumulative plot of empirical and estimated distribution

It is observed that the fitted distribution is very close to the empirical distribution and therefore the residuals have been tested to be iid and distributed as specified.

5.2.3.4. Back testing

For the back testing as it has been performed the same as previous section, it has been divided in rolling windows with a rolling window of 100 data. The following results are obtained:

	USDEUR		
alpha	**unconditional**	**conditional**	**duration**
1%	0,060	0,127	0,560
5%	0,491	0,453	0,362
95%	0,001	0,003	0,858
99%	0,176	0,384	0,181

Table 25. Value at Risk (VaR) back testing p values

As for the test, all have been successful therefore, the currency model is considered valid.

5.3. Copula estimation

5.3.1. Data study

Once all the residuals have been obtained and validated as iid by breaking their time dependence, their dependance from one to the other though copula models will be determined. For that, as a first step, the visual check of the residuals dependence is performed by plotting all their dependencies as a correlogram:

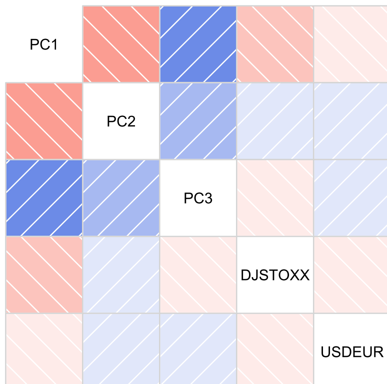

Figure 35. Time series residuals dependence

It is observable that there is a strong dependence between the PC1, PC2, and PC3, while it decreases when relating the Equity (DJSTOXX) and almost nonexistent for Currency (USDEUR). The values plotted by colors are indicated in the following table where it is shown that there is almost no correlation between the variables.

	PC1	PC2	PC3	DJSTOXX	USDEUR
PC1	1,000	-0,291	0,393	-0,204	-0,047
PC2	-0,291	1,000	0,268	0,062	0,056
PC3	0,393	0,268	1,000	-0,018	0,041
DJSTOXX	-0,204	0,062	-0,018	1,000	-0,079
USDEUR	-0,047	0,056	0,041	-0,079	1,000

Table 26. Correlation between residuals

All the plots of the variables are shown in the following plot so the dependencies can easily be observed. For the PC1, PC2 and PC3, the dependencies are very clear and with the sign indicated in the matrix above. The dependencies of the PC1 and Equity are very clear while not that much for the PC2 and PC3 with equity.

In any case, there seems to be symmetrical which reinforces the Eiopa underlying assumption of elliptical dependence, but the tails are very large to be the normal copula.

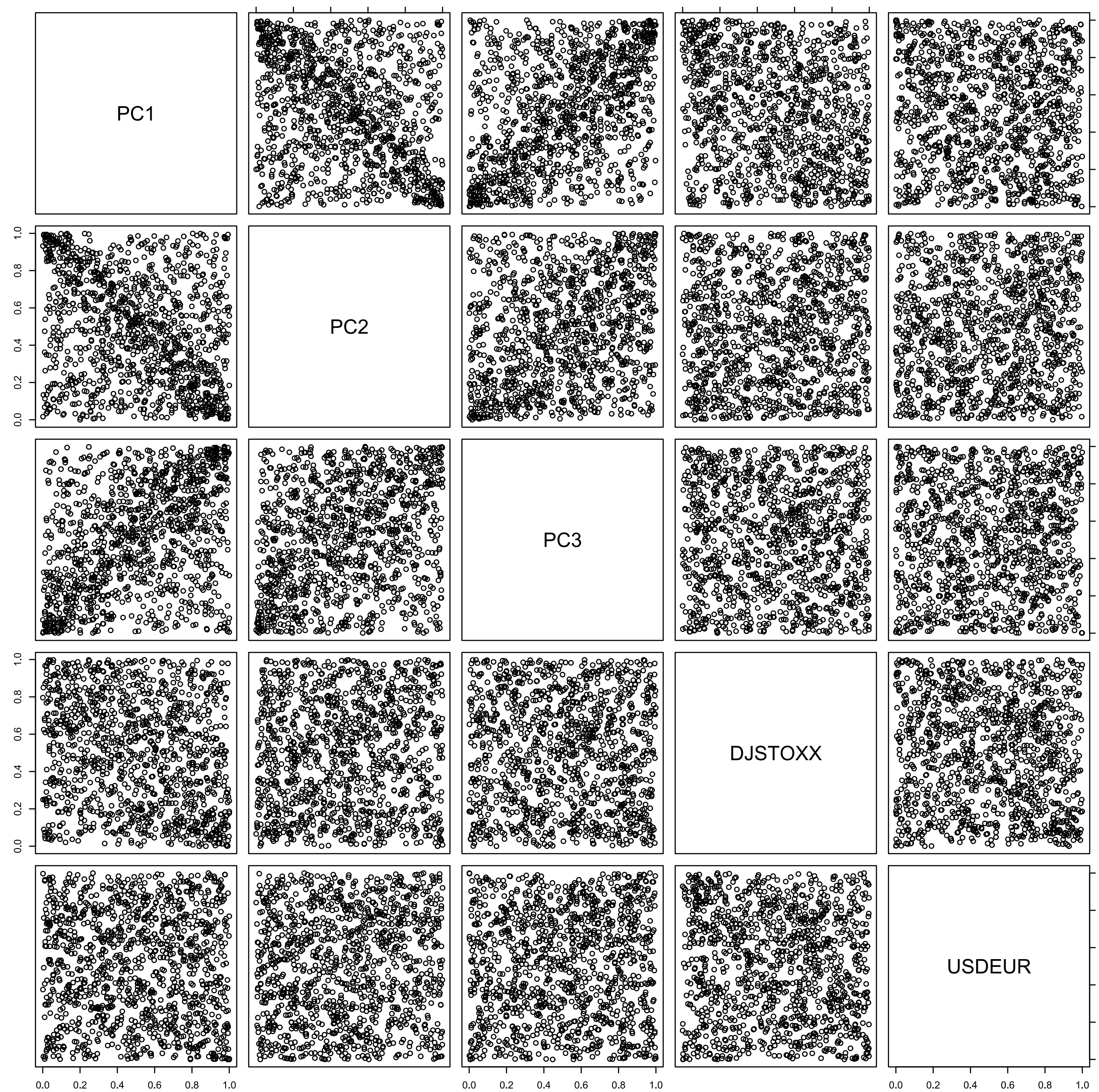

Figure 36. Residuals uniform plot

It is noted that the values of the correlation between the PC and the USDEUR are almost negligible, which suggests that there is no dependency between the interest rates and the USDEUR. To verify this, it is not sufficient to determine the correlation is not significant. It would be enough if both variables would be gaussian and would be related by a gaussian copula, but when studying this type of dependencies is not enough to validate this hypothesis. For that, one option would be to test all the relationships between the variables two by two and see which of them is significant by performing an independency test and, looking for a possible underlying copula. Note there could be some copulas that have a correlation of zero, but still stablish dependence between two variables, that would be in many cases discarded from the analysis and losing the real dependence that could be critical for the risk analysis.

After doing that, the results obtained are the following:

u1	u2	pvalue (indep)	copula	loglike	pvalue
PC1	PC2	0,0005	t-Student (df=3)	99,38	0,0475
PC1	PC3	0,0005	t-Student (df=4)	143,80	0,0025
PC1	DJSTOXX	0,0005	t-Student (df=15)	29,41	0,0954
PC1	USDEUR	0,0974	t-Student (df=7)	15,27	0,3282
PC2	PC1	0,0005	t-Student (df=3)	99,38	0,0435
PC2	PC3	0,0005	t-Student (df=7)	59,56	0,4011
PC2	DJSTOXX	0,0235	t-Student (df=13)	5,24	0,3312
PC2	USDEUR	0,0395	t-Student (df=42)	2,10	0,5180
PC3	PC1	0,0005	t-Student (df=4)	143,80	0,0015
PC3	PC2	0,0005	t-Student (df=7)	59,56	0,4011
PC3	DJSTOXX	0,5050	t-Student (df=18)	1,63	0,7577
PC3	USDEUR	0,0994	t-Student (df=7)	15,41	0,4351
DJSTOXX	PC1	0,0005	t-Student (df=15)	29,41	0,1204
DJSTOXX	PC2	0,0235	t-Student (df=13)	5,24	0,3392
DJSTOXX	PC3	0,5050	t-Student (df=18)	1,63	0,7478
DJSTOXX	USDEUR	0,0025	t-Student (df=7)	16,99	0,0884
USDEUR	PC1	0,0974	t-Student (df=7)	15,27	0,3252
USDEUR	PC2	0,0395	t-Student (df=42)	2,10	0,5170
USDEUR	PC3	0,0994	t-Student (df=7)	15,41	0,4161
USDEUR	DJSTOXX	0,0025	t-Student (df=7)	16,99	0,1174

Table 27. Bivariate independence test and bivariate best fit

It is important to note that the lowest value provided by the software is 0.005 since the p value is obtained by sampling the independence copula and obtaining the distribution of the statistic.

It is noted that the correlations are significant for the PC1, PC2, PC3 and DJSTOXX couples, as it is backed by the tables, where the likelihoods are maximum. As for the USDEUR, it is noted that there is not sufficient evidence to reject the independence. Therefore, it is not going to be considered more in the analysis. It is remarkable that all the dependencies are usually ruled by the student t distribution, and not by a normal copula, which is the underlying assumption under the standard formula. Additionally, the degrees of freedom of the student t copula are very low for the dependent variables, which indicates that they will have very long tails and have a different behavior of the normal copula.

It is important to note that the original correlations considered by Eiopa for this risk is 0,25. Which means that Eiopa takes a conservative approach when setting this value in the correlation matrix with no statistical back up. Also, the correlations between the interest rates and the Equity market are not as high as Eiopa approach of the 0,5. Therefore, it has not been considered the currency for the underlying copula because the independence cannot be rejected and therefore no copula dependence will be significant for that time series.

5.3.2. Copula estimation

As previously mentioned, it will only be considered four variables for the estimation, the PC1, PC2, PC3 and DJSTOXX.

For the copula testing, even if in the bivariate case, the student t copula is the most frequent one, is does not mean it happens the same for multivariate case. Therefore, all the copulas described in the state of art have been estimated. For the estimation, the maximum likelihood method has been used. To ensure convergence, the iteration estimation starting value, has been estimated by the method of moments by estimating the Spearman Rho of the sample and setting the values for the underlying copula in the case of the normal elliptical copula (and Kendall Tau for t-Student copula) where the convergency could be an issue. As for the Archimedean copulas, this is not a common issue.

Also, the p value has been obtained by the methods described in the state of art. The statistic distribution has been obtained by simulation and then, compared with the sample statistic so the p value is obtained, and the significance of the null hypothesis reject can be calculated.

The results obtained from the analysis are the following ones:

copula	loglikelihood	parameter	gof pvalue
Normal	310,58	NA	0,0634
t-Student (df=6)	418,16	6,0000	0,1733
Clayton	16,93	0,0732	0,0005
Frank	1,65	0,1257	0,0005
Joe	0,27	1,0043	0,0005

Table 28. Copula fit testing

Note the t-Student copula has the highest log likelihood being about one third greater than the normal copula.

As for the estimated elliptical copulas, the parameters are as follows for the normal and t-Student copula:

Normal			
1	-0,3014	0,4085	0,2810
-0,3014	1	-0,2150	0,0663
0,4085	-0,2150	1	-0,0210
0,2810	0,0663	-0,0210	1

t-Student			
1	-0,3224	0,4281	0,2871
-0,3224	1	-0,2174	0,0674
0,4281	-0,2174	1	-0,0216
0,2871	0,0674	-0,0216	1

Table 29. Spearman rho of estimated elliptical copulas

It can be observed that the Spearman Rho correlation is slightly greater in the t-Student copula, meaning that the assumption of the underlying copula being normal implies an underestimation of the risk. This reinforces the idea that dependence analysis is important for proper risk estimation. Later in the document, the normal copula and the t-Student copula are compared to quantify the risk underestimation implied in the normality underlying assumption.

Even if for this dimensionality it is difficult to provide graphical support, the bivariate cumulative plot is provided for the elliptical copulas (normal and t Student). It is observed that the student t fits slightly better than the normal, and both adjust much better than the Clayton copula.

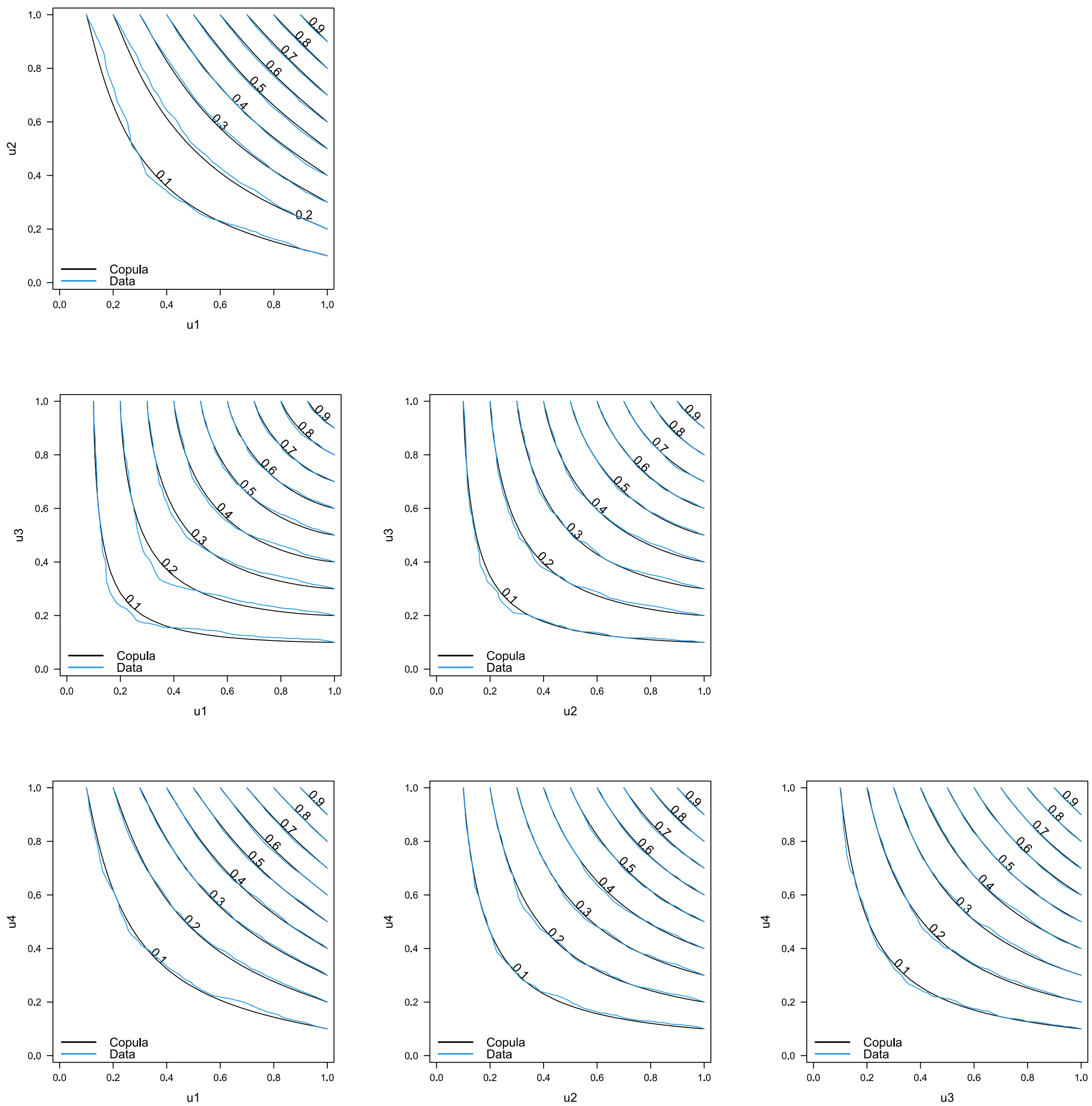

Figure 37. Normal copula fit

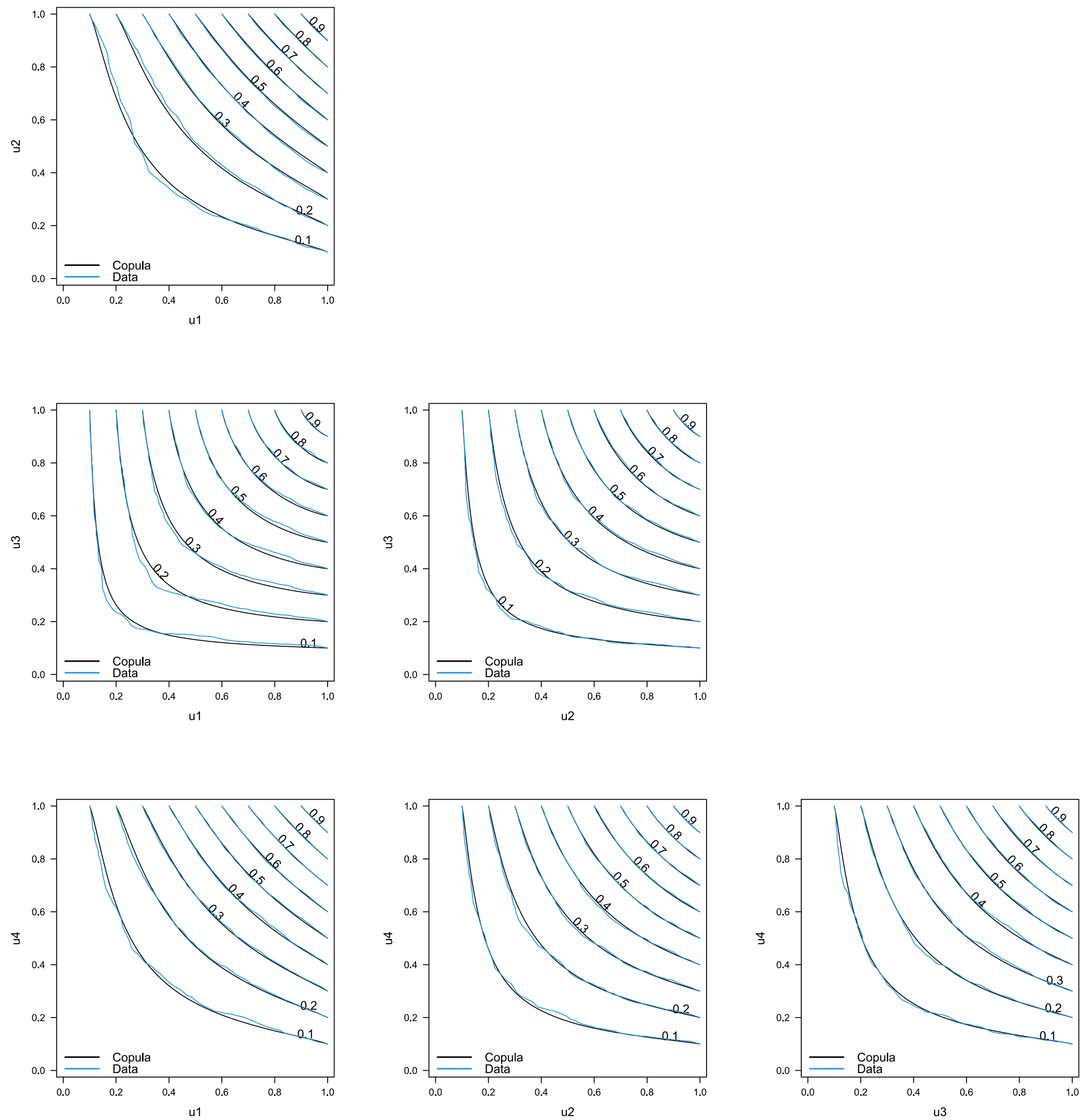

Figure 38. tStudent copula fit

It can be observed that the t Student copula fits slightly better than the normal copula. It is also noticed that for most extreme quantiles, the better fit is more noticeable. For example, it can be observed in the u1, u3 cumulative plot for the 0,1 quantile. It is noticed that it fits better, and the normal copula does not adjust as good as the t Student copula.

Also, for reference, the Clayton copula is plotted so it can be observed that the fit is not very accurate in comparison to the two previous ones.

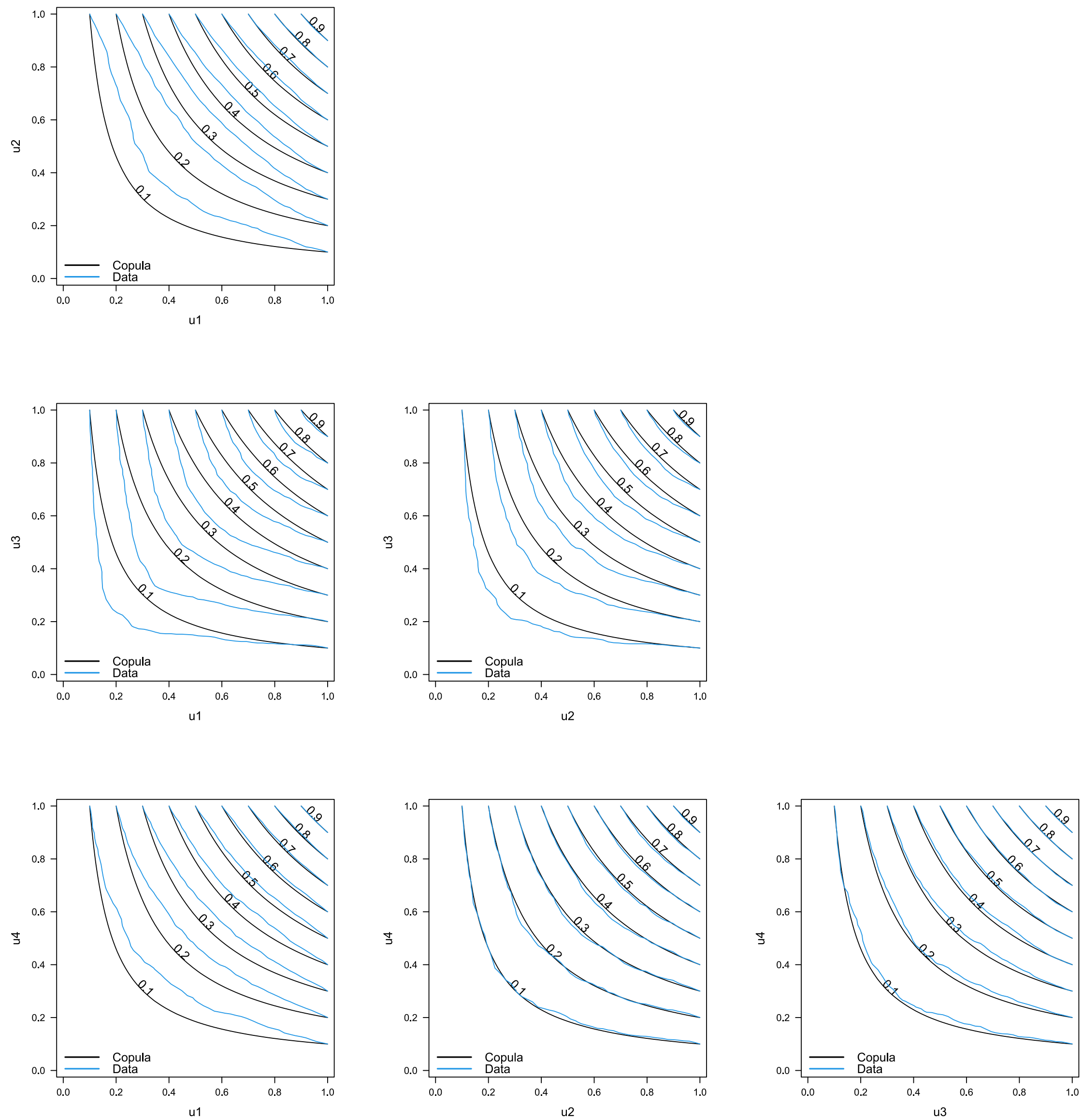

Figure 39. Clayton copula fit

Therefore, it is considered that the student t copula is clearly the best fit and it will be used for the simulation.

5.4. Simulation and marginal models results

For the simulation, once the copula defined the model is used for the simulation and generation of the new weekly residuals so the distribution of the one year ahead values are obtained. Therefore, it is obtained by stochastic scenarios with the real probabilities observed and validated in available data.

This model not only provides the individual shocks of the interest rate and the equity model, it also provides the correlation between both shocks and how to aggregate them.

5.4.1. Interest rate model marginal

One of the strengths of this analysis is the possibility to provide not only one shock to the interest rate, but also the least favorable scenario depending on the long-term portfolio. It is well known the duration is a unique value that summarizes the portfolio change in value when a parallel movement of the yield curve is produced. However, this is not the only case it can happen when analyzing the interest rate yield curve changes, it can be given the case the slope changes, or even the curvature meaning that the simplistic approach of duration, could be not effective in some cases.

From the simulation, the following distribution is obtained.

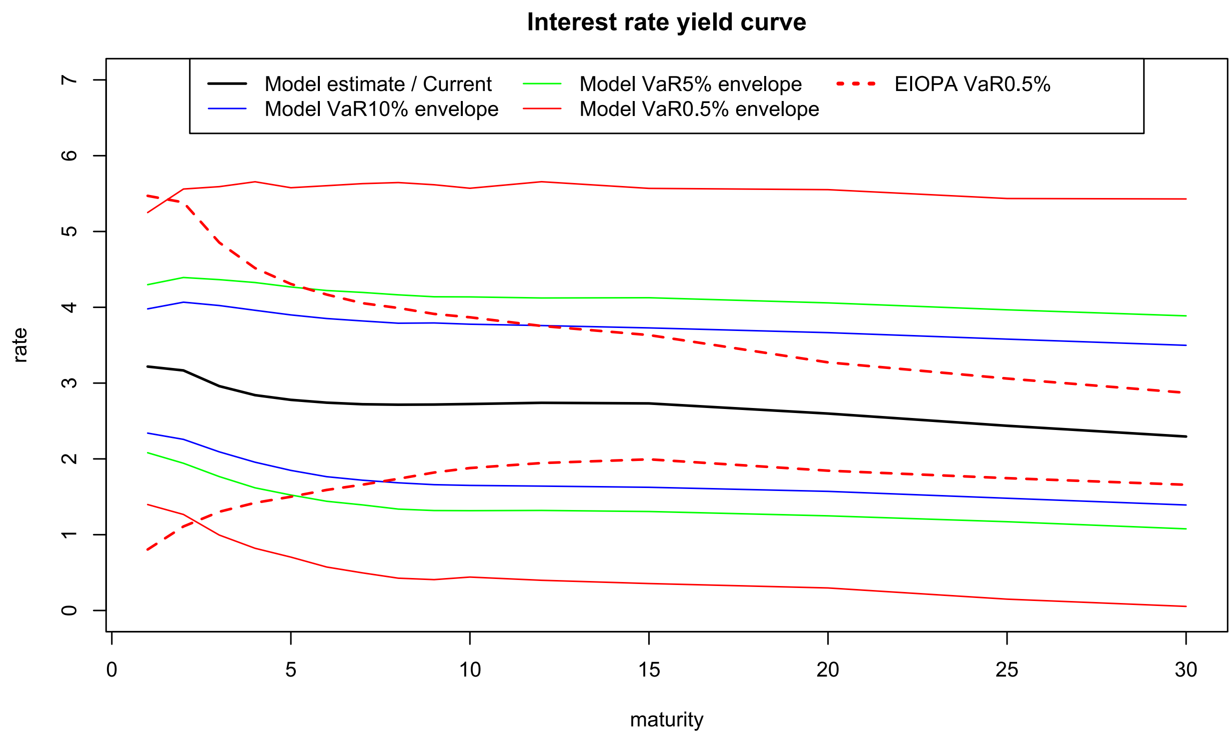

Figure 40. Interest rate model one year ahead distribution

It is remarkable that for the interest rate shocks provided by Eiopa in Solvency II article 166 and 167 does not even reach the two years behind when the interest rates where very low, meaning that the model is not measuring the interest rate risk properly.

As for the proposed model, it captures all the values of the low interest rates and covers the two years behind low interest rate period. Additionally, the model proposed, and the plotted values are the envelopes, meaning that these shocks are conservatively the worst-case scenarios.

It is important to remark that the model is a random walk, meaning that the expected value is the same it is currently. This is especially relevant since it is a starting point to provide a market consistent model, and no market expectations should be considered. Otherwise, the model could influence the investing strategy of the insurance companies, so they are aligned with the market estimations and therefore reduce their capital requirements.

5.4.2. Equity model marginal

The equity model provides the relative shock based on the time series. It calculated the same way it is calculated the interest rate, by simulating the weekly residuals and estimating the time series one year ahead.

The final portfolio value ratio distribution is plotted in the following histogram. This model also has an estimated portfolio value of the same value as the current value, even if in this case to be a market consistent model it should incorporate the risk-free rate interest. In any case for the real-world probabilities, this approach is more appropriate.

Note the interest rate shock incorporating the Symmetric Adjustment established by the Solvency II regulation article 169 are very close to the ones proposed by the model, which means that the volatility captured by Eiopa and the model are similar.

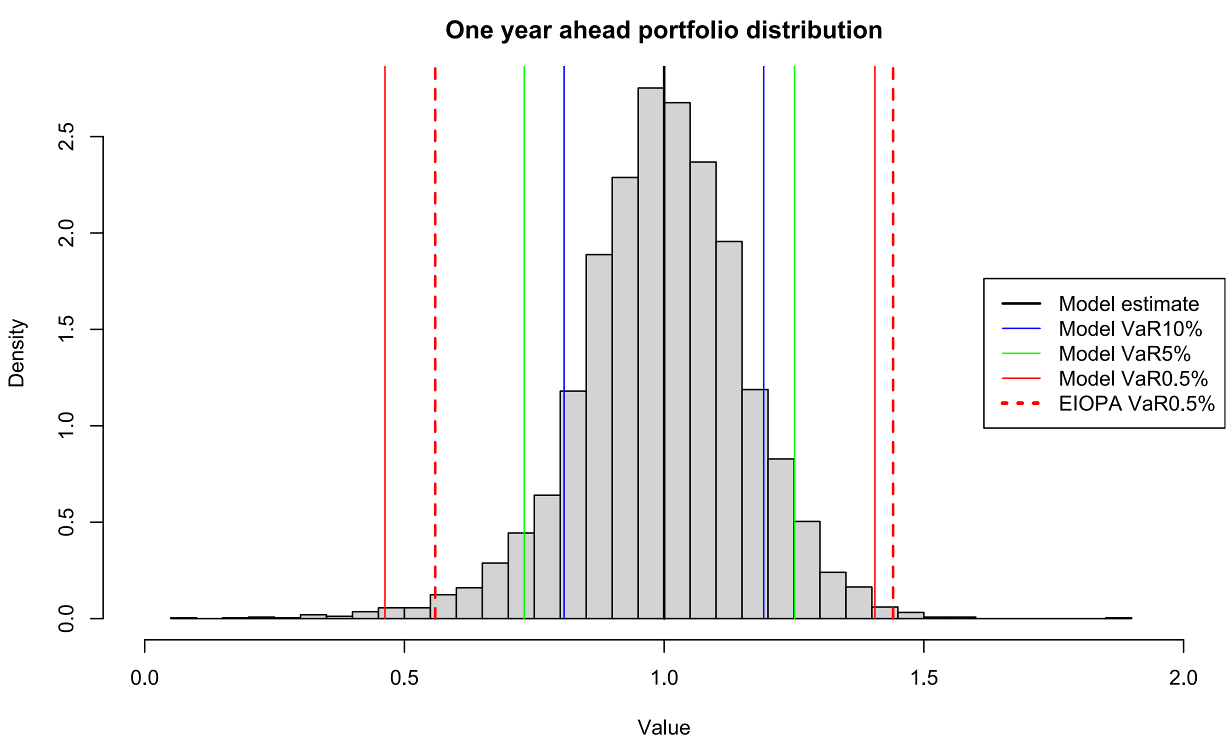

Figure 41. Equity model one year ahead value ratio distribution

5.5. Results and Aggregation

For the aggregation, since the standard formula application is not feasible anymore considering the dependencies described in the work, three reference portfolios have been modelled.

The modelled portfolios are considered as a representative sample of the current European insurance industry. The three portfolios are composed by 90% of value in government bonds for which it is assumed the interest rate does not have any liquidity spread. The other 10% of the portfolio would be equity assets, represented by the equity model.

The difference between the three portfolios is only the bonds maturities, that is different for each portfolio, so a different duration is obtained for each portfolio. The bonds portfolio is composed as described in the following table.

5.5.1. Portfolio 1

	Maturity	Cupon	Nominal
Bond 1	8	0,02	1000
Bond 2	10	0,04	1000
Bond 3	5	0,03	1000
Bond 4	3	0,01	1000
Equity	-	-	500

Table 30. Portfolio 1 composition (Financial duration = 6 years)

Being the financial duration of the portfolio 6 years, which would be a short-term portfolio of an insurance company. For the shown portfolio, the shocks in the total portfolio value for the interest rate shocks and the equity shocks are as follows:

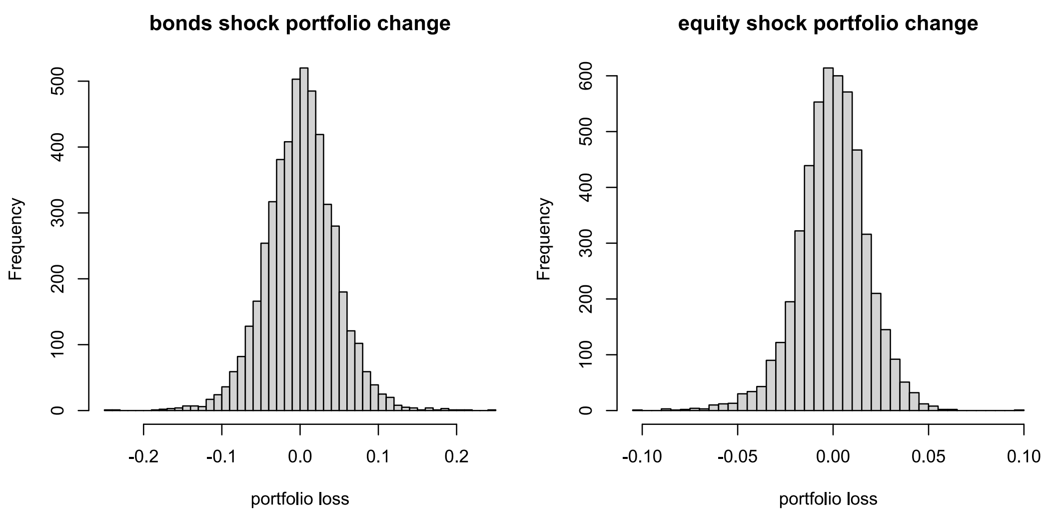

Figure 42. Interest rate and equity shock effect in total portfolio 1 value

As for the correlation, since three copula variables are implicit in the interest rate, and these correlate with one variable, there is no possibility of setting one correlation analytically. Additionally, as previously indicated, the interest cannot be reduced to a single interest rate curve, therefore depending on the portfolio the shocks will have a value or another, and the same shock could have a great impact in one portfolio while it would not in another one with different bonds structure. To analyze this dependence, the pseudo probabilities have been plotted:

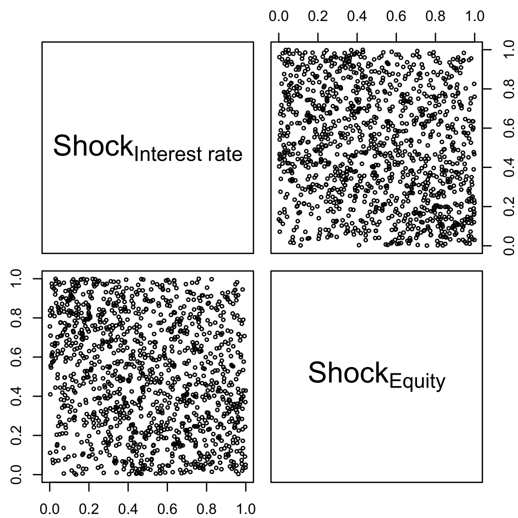

Figure 43. Portfolio 1 shocks pseudo probabilities plot

It is observable there is correlation, indeed the interesting thing is that the Pearson correlation of the shocks is negative (-0,185) and not positive as it is indicated by the correlations provided by the Solvency II regulation. Indeed, the regulator does not only consider it as positive, but also is considered as high as 0,5. Therefore, the Solvency II regulation is proven to be very conservative when determining the shocks. Also notice, the tails correspond to a student t copula and not the normal copula used by the regulator.

The following chart shows the shocks and the aggregated shock value for the 0,5% confidence interval for the rates up and equity down:

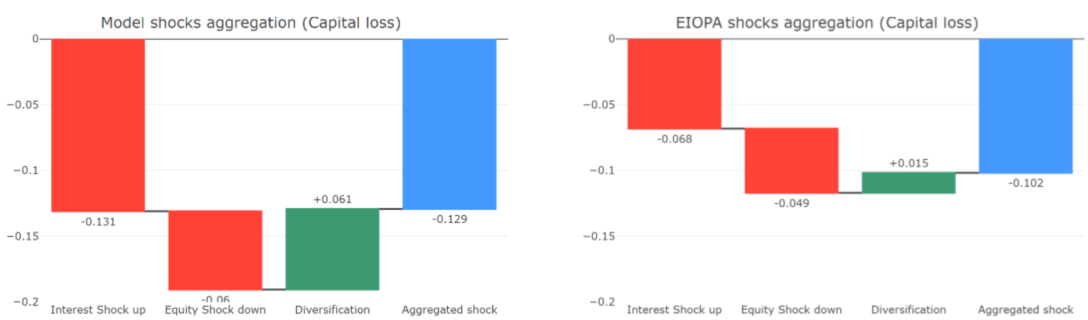

Figure 44. Proposed model and EIOPA shocks and aggregation for VaR99,5% (shock down)

Note that the equity shock has been almost fully diversified by the equity shock in terms of aggregation. These results show that an Equity portfolio can be partially hedged with bonds portfolio in terms of risk measurement. It is also noticeable that the shocks are sensibly greater than the ones proposed by Eiopa in Solvency II regulations. This is probably the most relevant finding of this work, that the interest rate shock is not covering the real risk the insurance companies are exposed to.

The opposite shock has also been plotted when the interest rates go down and when the equity is up. To make a full analysis, the complementary cases shall also be analyzed, when interest rates are down, and equity is down and the opposite one. In any case, it is not considered to be relevant since that would not stress the portfolio and would provide no value for the current analysis.

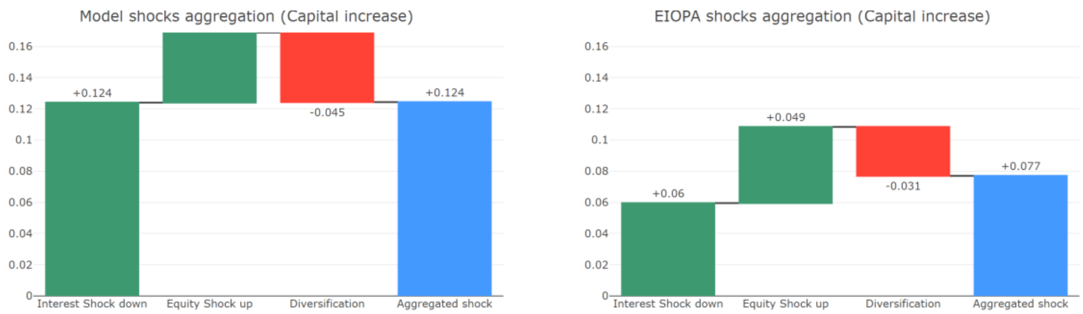

Figure 45. Proposed model and EIOPA shocks and aggregation for VaR99,5% (shock up)

For this case it is noted that the results are similar to the obtained in the other shock analysis when determining the Eiopa interest rate shock is by far not sufficient. As for the risk aggregation, it can be observed that the negative correlation is also noticeable here so the shock aggregation fully compensates capital increase for the interest rate marginal shock.

5.5.2. Portfolio 2

As for the portfolio 2, the composition is changed so the composition is as follows:

	Maturity	**Cupon**	**Nominal**
Bond 1	5	0,01	1000
Bond 2	10	0,02	1000
Bond 3	15	0,04	1000
Bond 4	20	0,03	1000
Equity	-	-	500

Figure 46. Portfolio 1 composition (Financial duration = 10 years)

As for the shocks, the following plot shows the histogram for portfolio 2. It is noticeable that the interest shocks values are greater than that for portfolio 1 since the duration is greater and therefore the sensitivity to the rates change increases also.

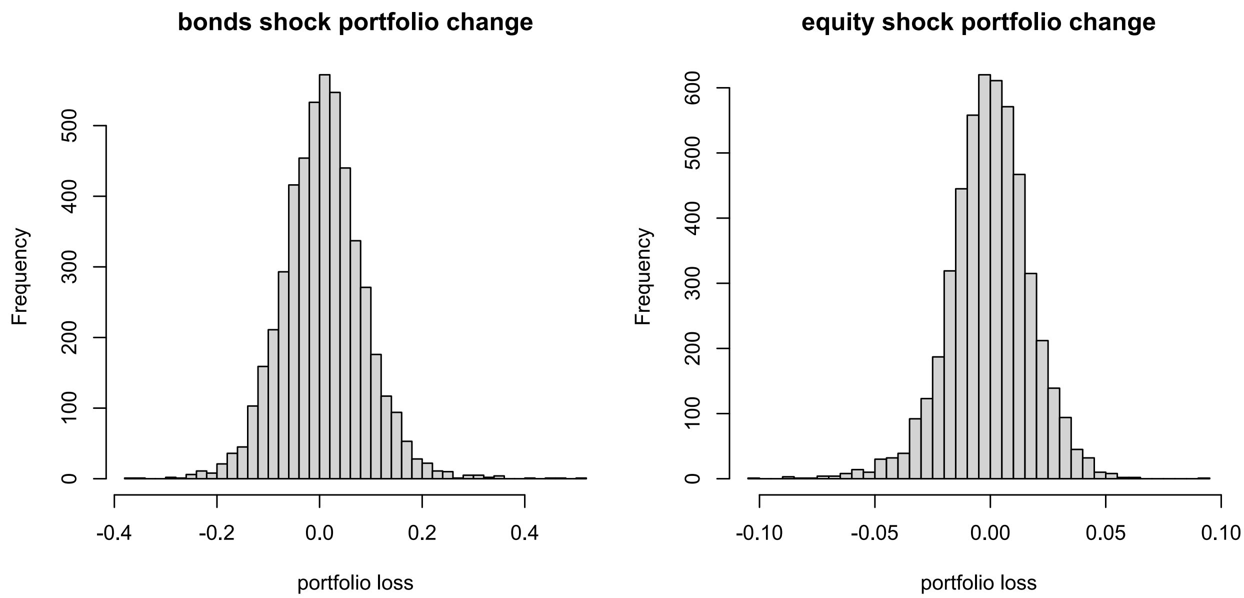

Figure 47. Interest rate and equity shock effect in total portfolio 2 value

As for the dependencies, the following pseudo probabilities are obtained:

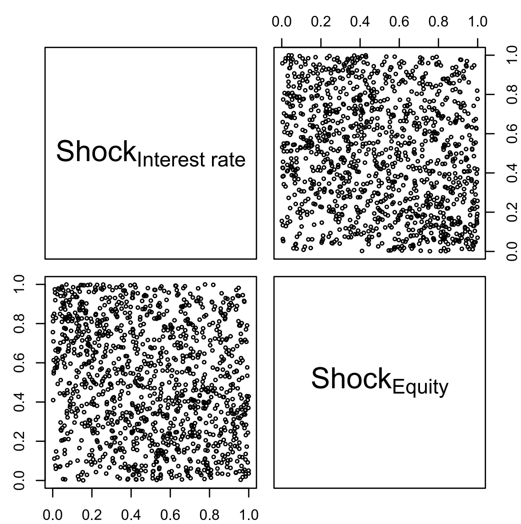

Figure 48. Portfolio 2 shocks pseudo probabilities plot

The plotted correlation is very similar to the one observed in portfolio 1. Indeed, the Pearson correlation of the shocks would be -0,185, also negative. Notice the tails are clearly the ones corresponding to a student t distribution as it was happening also with the portfolio 1.

As for the shocks, the following plots are obtained. It is remarkable that the underestimation of the shocks with the regulator approach is getting bigger in proportion as the duration increases. This makes sense since the greatest difference observed in the interest rates curves was for the longest terms.

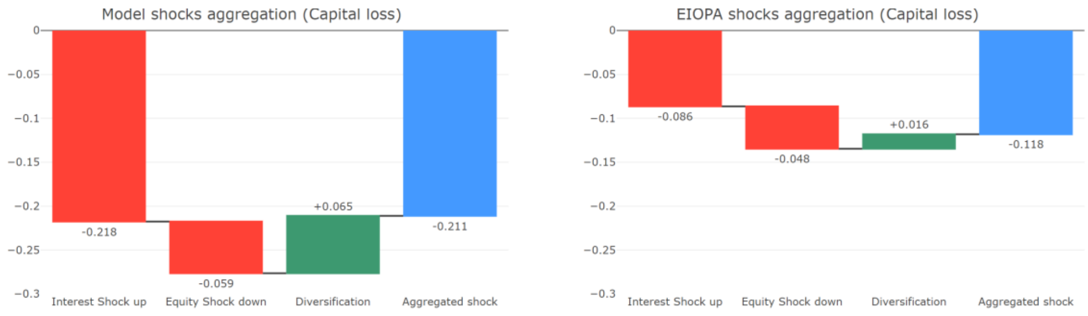

Figure 49. Proposed model and EIOPA shocks and aggregation for VaR99,5% (shock up)

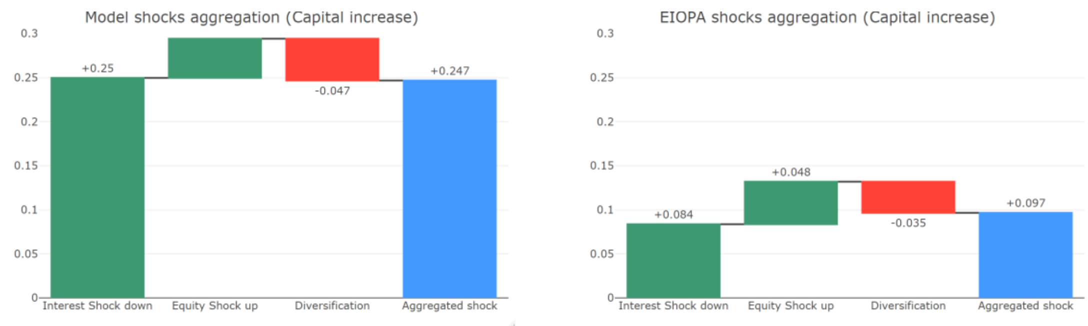

Figure 50. Proposed model and EIOPA shocks and aggregation for VaR99,5% (shock down)

As expected, the shocks are greater due to the greatest financial duration of the portfolio.

5.5.1. Portfolio 3

The third portfolio is shown for the greatest duration. The portfolio is composed as follows.

	Maturity	Cupon	Nominal
Bond 1	10	0,01	1000
Bond 2	15	0,02	1000
Bond 3	20	0,04	1000
Bond 4	25	0,03	1000
Equity	-	-	500

Table 31. Portfolio 1 composition (Financial duration = 14 years)

As for the distribution of the shocks the following histograms are obtained:

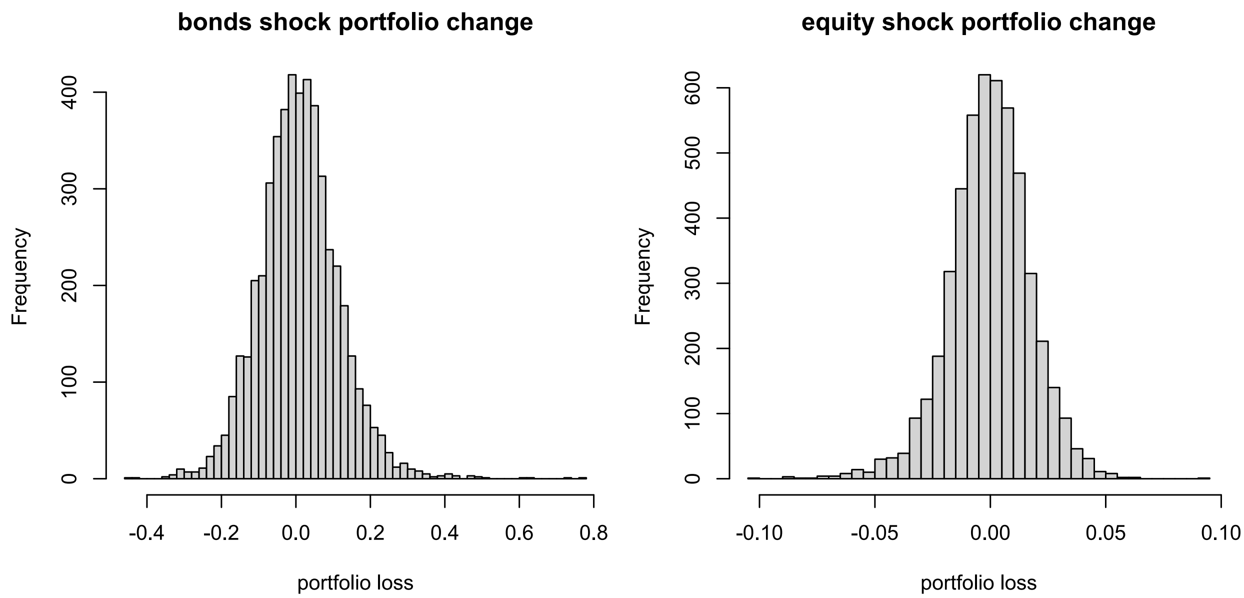

Figure 51. Interest rate and equity shock effect in total portfolio 3 value

Note that the tails of the distribution are much greater for this portfolio meaning that the risk is maximum for this portfolio. This makes sense considering this is the portfolio with the greatest duration.

As for the correlations, the obtained plot is the following:

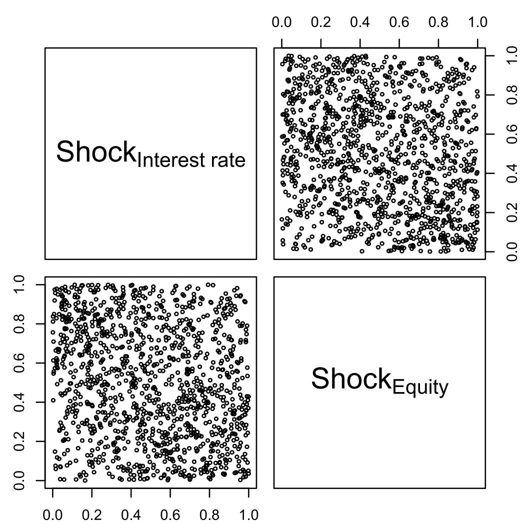

Figure 52. Portfolio 3 shocks pseudo probabilities plot

In this case, negative Pearson correlation is also observed, but in this case slightly smaller (-0,183).

As for the shocks aggregation the following results are obtained:

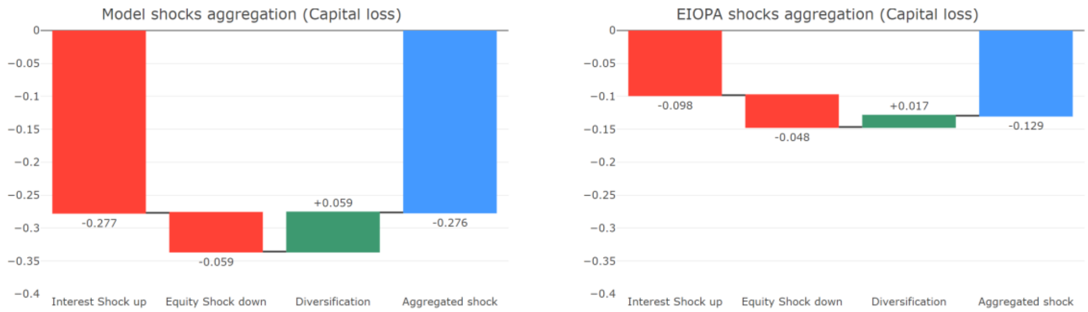

Figure 53. Proposed model and EIOPA shocks and aggregation for VaR99,5% (shock up)

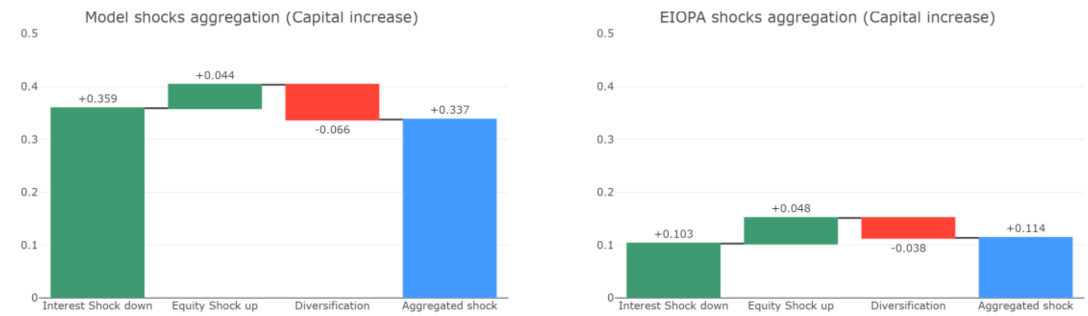

Figure 54. Proposed model and EIOPA shocks and aggregation for VaR99,5% (shock down)

Note the underestimation of the risk, is proportionally greater for long term portfolios, being more than three times higher than the regulators shock.

Therefore, the risk is being underestimated mainly because the interest model is not considering the shocks properly. On the other hand, the aggregation method is conservative so there it diminishes this effect partially. In any case the underestimation of the interest rate model is too much, and it indicates that the interest rates shocks can be underestimated up to 70% of the proposed model value.

5.5.2. Comparison with the normal copula

It has been well observed that the normal copula has a much lower tail dependencies in comparison to the student t. Therefore, it is an important analysis the comparison between the standard formula normality underlying assumptions to the proposed model of a student t underlying dependence.

For the analysis it is plotted the waterfall diagram of the shocks comparison for the interest rate shock up and equity down for the three portfolios above. It is observed that the interest rate shock is clearly underestimated with a difference of about 10% for the interest rate.

Also note the Equity shock shall not have any difference since the marginal distributions do not depend on the underlying copula. Due to the simulation size setting of 10.000 (limited by the computation capacity), when sampling the VaR99,5%, the variance of the estimation is high.

Figure 55. Interest rate and equity shocks for portfolio 1 under Student t copula (left) and normal copula (right)

Figure 56. Interest rate and equity shocks for portfolio 2 under Student t copula (left) and normal copula (right)

Figure 57. Interest rate and equity shocks for portfolio 3 under Student t copula (left) and normal copula (right)

	Interest shock diff	**Equity shock diff**	**Agreggated shock diff**
Portfolio 1	9,15%	1,24%	5,28%
Portfolio 2	12,36%	1,24%	10,59%
Portfolio 3	11,34%	1,24%	12,38%

Table 32. Shocks differences between normal and student t copula

It is observed that the differences are more significant as the duration of the portfolio increases reaching a maximum of 11,34% difference for the long-term portfolio. This is especially relevant since it would imply that the normality assumption is not valid for the interest rate shock analysis and not conservative.

6. Conclusions

The current research aimed to check the validity of the Solvency II regulation market shocks and correlations, especially in the current times of the highest sudden interest rates hikes in decades. From that analysis the following points are obtained.

- The EIOPA approach to elliptical correlations structure is valid and the standard formula approach shows to be backed up by statistical evidence of the real behavior of the market shocks correlation. Nevertheless, the data suggest that the underlying dependence is a student t copula and not a normal copula, so the real tail dependency is greater than that considered. This could mean that an underestimation of about 10% could be present in the shocks.

- The correlations proposed by EIOPA are too conservative, being even of the contrary sign of the ones on the Solvency II regulation. This happens because the correlation of the interest rate with equity proves to be negative and considered positive in the regulatory correlations. As for the currency, it has been observed there is no correlation between the interest rate and equity, so it is also being overestimated. Considering all the shocks have been calibrated for 99,5%, it is not aligned with the statistical approach to provide conservative values in the calculations that are not backed up by the data.

- The interest rate shocks proposed by the regulator have proven to be not enough to cover the real risk of interest rate changes. In some cases, for long term portfolios, the shocks do not even reach the 30% of the observed value. Also, they are very simplistic for the complexity of a time dependent yield curve that can affect in different ways to differently structured portfolios.

- The interest rate shock as a percentage of the interest rate is not a reasonable approach, especially when the interest rates are very low, as it has been happening the latest years. Therefore, when determining the increase or decrease of interest as percentage of the interest rate shock, when being close to zero, or even negative, does not properly capture the risk.

From this work it is observed that the main current problem in the present moment is the interest rate model, which is not capturing the interest rate risk properly as it has proven to be true during last year interest rates hikes. Therefore, it is proposed a copula-based approach considering the time dependence for the interest rate risk estimation and new shocks calibrations since the current EIOPA approach assumes normality when there is not. Also, this model could capture the risk of more heterogeneous portfolios better than the currently defined shock and serve for the internal model of insurance companies.

Additionally, the correlations shall be reviewed to somehow capture the student t copula behavior properly and not the normal copula that has proven to be not capturing the full risk.

For future works, it is proposed to include some other risks dependence analysis included in the market module such as spread or even concentration. Also, some other dependencies such as vine copulas or nested copulas could be an interesting approach and could capture some dependencies and provide more statistically significant results.

Citations

Ali, M. M., Mikhail, N. N., & Haq, M. S. (1978). A class of bivariate distributions including the bivariate logistic. *Journal of Multivariate Analysis*, *8*(3), 405–412.

Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, *31*(3), 307–327.

Box, G. E., Jenkins, G. M., & Reinsel, G. (1970). Time series analysis: Forecasting and control Holden-day San Francisco. *BoxTime Series Analysis: Forecasting and Control Holden Day1970*.

Christoffersen, P. F. (1998). Evaluating interval forecasts. *International Economic Review*, 841–862.

Christoffersen, P., & Pelletier, D. (2004). Backtesting value-at-risk: A duration-based approach. *Journal of Financial Econometrics*, *2*(1), 84–108.

Clayton, D. G. (1978). A model for association in bivariate life tables and its application in epidemiological studies of familial tendency in chronic disease incidence. *Biometrika*, *65*(1), 141–151.

Durbin, J., & Stuart, A. (1951). Inversions and rank correlation coefficients. *Journal of the Royal Statistical Society: Series B (Methodological)*, *13*(2), 303–309.

EIOPA. (2010a). *SCR Standard Formula Article 111b Calibration of Market Risk Module*. CEIOPS. https://register.eiopa.europa.eu/CEIOPS-Archive/Documents/Advices/CEIOPS-L2-Advice-Market-risk-calibration.pdf

EIOPA. (2010b). *Solvency II Calibration Paper*. CEIOPS. https://register.eiopa.europa.eu/CEIOPS-Archive/Documents/Advices/CEIOPS-Calibration-paper-Solvency-II.pdf

FRANK, M. (1975). SIMULTANEOUS ASSOCIATIVITY OF C (X, Y) AND X+ YC (X, Y)-PRELIMINARY REPORT. *NOTICES OF THE AMERICAN MATHEMATICAL SOCIETY*, *22*(5), A562–A562.

Fréchet, M. (1935). Sur la définition axiomatique d’une classe d’espaces vectoriels distanciés applicables vectoriellement sur l’espace de Hilbert. *Annals of Mathematics*, 705–718.

Genest, C., Huang, W., & Dufour, J.-M. (2013). A regularized goodness-of-fit test for copulas. *Journal de La Société Française de Statistique*, *154*(1), 64–77.

Genest, C., Rémillard, B., & Beaudoin, D. (2009). Goodness-of-fit tests for copulas: A review and a power study. *Insurance: Mathematics and Economics*, *44*(2), 199–213.

Glosten, L. R., Jagannathan, R., & Runkle, D. E. (1993). On the relation between the expected value and the volatility of the nominal excess return on stocks. *The Journal of Finance*, *48*(5), 1779–1801.

Gumbel, E. J. (1960). Bivariate exponential distributions. *Journal of the American Statistical Association*, *55*(292), 698–707.

Joe, H. (1997). *Multivariate models and multivariate dependence concepts*. CRC press.

Kupiec, P. H. (1999). Risk capital and VaR. *The Journal of Derivatives*, *7*(2), 41–52.

Kwiatkowski, D., Phillips, P. C., Schmidt, P., & Shin, Y. (1992). Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root? *Journal of Econometrics*, *54*(1–3), 159–178.

Nelson, C. R., & Plosser, C. (1982). Trends and random walks in macro-economic time series, journal of Monetary Economics. *September*, *10*(2), 139–162.

Nelson, D. B. (1991). Conditional heteroskedasticity in asset returns: A new approach. *Econometrica: Journal of the Econometric Society*, 347–370.

Rüschendorf, L. (1981). Sharpness of Fréchet-bounds. *Zeitschrift Für Wahrscheinlichkeitstheorie Und Verwandte Gebiete*, *57*(2), 293–302.

Said, S. E., & Dickey, D. A. (1984). Testing for unit roots in autoregressive-moving average models of unknown order. *Biometrika*, *71*(3), 599–607.

Sklar, M. (1959). Fonctions de répartition à n dimensions et leurs marges. *Annales de l'ISUP*, *8*(3), 229–231.

Bibliography

Albrecher, H., Beirlant, J., & Teugels, J. L. (2017). Reinsurance: actuarial and statistical aspects.

Charpentier, A. (Ed.). (2014). Computational actuarial science with R.

Cossette, H., Marceau, E., Mtalai, I., & Veilleux, D. (2018). Dependent risk models with Archimedean copulas: A computational strategy based on common mixtures and applications. *Insurance: Mathematics and Economics*, *78*, 53-71.

Czado, C. (2019). Analyzing dependent data with vine copulas. *Lecture Notes in Statistics, Springer*, *222*.

Durante, F., & Sempi, C. (2015). Principles of copula theory.

Fredricks, G. A., & Nelsen, R. B. (2007). On the relationship between Spearman's rho and Kendall's tau for pairs of continuous random variables. *Journal of statistical planning and inference*, *137*(7), 2143-2150.

Ghalanos, A., Ghalanos, M. A., & Rcpp, L. (2018). Package ‘rugarch’. *R Team Cooperation*, 43.

Joe, H. (2014). Dependence modeling with copulas.

Joe, H., & Kurowicka, D. (Eds.). (2011). Dependence modeling: vine copula handbook.

Lyubchich, V. (2020). Elements of Copula Modeling With R: by Marius Hofert, Ivan Kojadinovic, Martin Mächler, and Jun Yan. Cham, Switzerland: Springer, 2018, 267 pp., $84.99, ISBN: 978-3-319-89634-2, ISBN: 978-3-319-89635-9 (eBook), https://doi. org/10.1007/978-3-319-89635-9. *Technometrics*, *62*(3), 416-416.

Nelsen, R. B. (2007). An introduction to copulas.

Schmid, F., & Schmidt, R. (2007). Multivariate conditional versions of Spearman's rho and related measures of tail dependence. *Journal of Multivariate Analysis*, *98*(6), 1123-1140.

Trivedi, P. K., & Zimmer, D. M. (2007). Copula modeling: an introduction for practitioners. *Foundations and Trends® in Econometrics*, *1*(1), 1-111.

Workshop on Copula Theory and Its Applications. (2010). Copula theory and its applications: proceedings of the workshop held in Warsaw, 25-26 September 2009.

Annexes

Main Code

```
# clear work space
rm(list = ls())

# 0. Packages and functions ----

library("ggplot2")
library("tidyverse")
library("dplyr")
library("fpp2")
library("cowplot")
library("tseries")
library("copula")
library("zoo")
library("rugarch")
library("xts")
library("goftest")
#install.packages("R.utils")
library("R.utils")
#install.packages("xlsx")
library("xlsx")
# load all user functions
source("functions.R")
#install.packages("latticeExtra")
library("latticeExtra")
#install.packages("corrgram")
library("corrgram")
library("plotly")

old.par <- par(mar = c(0, 0, 0, 0))

# 1. Import and clean data ----

## Import from Github ----

# Define all the bonds and the urls
url <-
"https://raw.githubusercontent.com/asiergs/financial_data/main/EUROIS.csv"

## 1.1 Import rates ----
swaps_df <- read.csv(url)
swaps_df2 <- na.omit(swaps_df)

## 1.2 Clean rates data----

# Order the swaps by time
names <- names(swaps_df)
times <- c(0, as.numeric(gsub("[A-Z]", "",names[-1])))

for (i in 2:length(names)){
  if (substring(names[i],nchar(names[i])) != "Y"){
```

```
    names[i] <- paste0(names[i],"Y")
  }
}

names(swaps_df2) <- names

names(times) <- names
times <- sort(times)

swaps_df2 <- swaps_df2[,names(times)]

times <- times[-1]

# Expand the data to work with the data
swaps_df3 <- swaps_df2 %>% pivot_longer(cols = colnames(swaps_df2)[-1],
                                        values_to = "rate")

# Get the time for each swipe and convert date char to date format
swaps_df3 <- swaps_df3 %>% mutate(date = as.Date(Code, format = "%d/%m/%Y"),
                                  time = as.numeric(gsub("[A-Z]", "",name)))

# Get only the columns of interest
swaps_df3 <- swaps_df3 %>% select(date,name,time,rate)

# Check there is no missing data (shall not be with the na.omit())
swaps_df3 %>% group_by(date) %>% count() %>% summarise(max = max(n),
                                                        min = min(n))

## Convert to weekly

start <- min(swaps_df3$date)
start_weekday <- wday(start, week_start = 1)

# Extract the last day from week only
swaps_df3 <- swaps_df3 %>%
  mutate(weekday = wday(date, week_start = 1),
         days_from_start = date - start + start_weekday,
         week = as.numeric(trunc(days_from_start/7))) %>%
  group_by(week) %>% mutate(max_wday = max(weekday)) %>%
  filter(weekday == max_wday) %>% ungroup()

# Convert to matrix

swaps_df4 <- swaps_df3 %>% pivot_wider(id_cols = date, values_from = rate)
dates <- swaps_df4$date

# match all columns in same time

swaps_matrix <- as.matrix(swaps_df4[,-1])
rownames(swaps_matrix) <- as.character(swaps_df4$date)

## 1.3 Import EURSTOXX600 and USDEUR data and join it ----

url <-
"https://raw.githubusercontent.com/asiergs/financial_data/main/DJSTOXX.csv"
```

```
DJSTOXX <- read.csv(url) %>% mutate(date = as.Date(Date, format =
"%d/%m/%Y")) %>%
  select(date, DJSTOXX)

url <-
"https://raw.githubusercontent.com/asiergs/financial_data/main/USDEUR.csv"

USDEUR <- read.csv(url) %>% mutate(date = as.Date(Date, format = "%d/%m/%Y"))
%>%
  select(date, USDEUR)

market_df <- swaps_df4 %>% left_join(DJSTOXX, join_by(date)) %>%
  left_join(USDEUR, join_by(date))

# 2. PCA & Factorial analysis----

scaled_swaps <- scale(swaps_matrix)

center <- attr(scaled_swaps,"scaled:center")
scale <- attr(scaled_swaps,"scaled:scale")

## 2.1 Get the values of the Principal Components ----

correl_matrix <- cov(scaled_swaps)
eigen_swaps <- eigen(correl_matrix)

PC <- scaled_swaps %*% eigen_swaps$vectors
var_cum_explained <- cumsum(eigen_swaps$values)/sum(eigen_swaps$values)
write.xlsx(as.data.frame(var_cum_explained), file =
"tables/var_explained.xlsx")

# undo principal components
PC %*% t(eigen_swaps$vectors)

PC_mat <- t(eigen_swaps$vectors)
colnames(PC_mat) <- names(times)
rownames(PC_mat) <- paste0("PC",1:15)

write.xlsx(as.data.frame(PC_mat), file = "tables/PC_matrix.xlsx")

## 2.2 Fit of the Principal Components to curve ----
dim <- length(times)
curve <- matrix(0,ncol = dim, nrow = dim)

swaps_red_list <- list()
for (i in 1:dim){
  # The not selected PC are set to 0
  PC_red <- PC
  PC_red[,-(1:i)] <- 0
  # The PC are undone and unscaled
  swaps_red_list[[i]]  <- unscale(PC_red %*%
t(eigen_swaps$vectors),scale,center)
}

date <- as.character(swaps_df4$date[1220])
ylim <- c(min(swaps_matrix[date,])-0.25,max(swaps_matrix[date,])+0.5)
```

```
# Graphs of how PC affect the curve

sample_curve <- unscale(PC_show %*% t(eigen_swaps$vectors),scale,center)

for(i in 1:3){
  svg(file = paste0("plots/PCA_",i,"_shock.svg"), width = 5, height = 5)
  par(mfrow=c(1,1), oma = c(0,0,0,0), mar = c(4,4,2,1))
  vector <- c(rep(0,i-1),1,rep(0,15-i))
  plot(times,sample_curve, type="p", lty = 1, pch = 19,
       lwd = 2, ylim = c(0,5), ylab = "swap rate", main = paste0("PC ",i,"
shock"),
       xlab = "years")
  lines(times,sample_curve, type="l", lty = 1,
        lwd = 2)
  sample_curve_shock <- unscale(vector %*%
t(eigen_swaps$vectors),scale,center)
  lines(times,sample_curve_shock, type = "l", lty = 2, lwd = 2, col = "red")
  sample_curve_shock <- unscale(-vector %*%
t(eigen_swaps$vectors),scale,center)
  lines(times,sample_curve_shock, type = "l", lty = 2, lwd = 2, col = "blue")
  legend("bottomright",legend = c("Yield curve", "Positive shock", "Negative
shock"),
         lty = c(1,2,2), lwd = c(2,2,2), ncol=1, col=
c("black","red","blue"))
  dev.off()
}


#### INPUT to change graph
n <- 6
#n <- dim
#### INPUT to change graph

svg(file = 'plots/PCA2.svg', width = 10, height = 6)
par(mfrow=c(ceiling(n/3),3), oma = c(2,2,2,2), mar = c(1.5,1.5,4,1.5))
for (i in 1:n) {
  plot(times,swaps_matrix[date,1:dim], type="p", lty = 1, pch = 19,
       lwd = 2, ylim = ylim, ylab = "swap", main = paste0(i," PC"), xlab =
"years")
  lines(times,swaps_matrix[date,1:dim], type="l", lty = 1,
        lwd = 2)
  lines(times,swaps_red_list[[i]][date,1:dim], type = "l", lty = 2, lwd = 2,
col = "red")
  legend("bottomright",legend = c("Real", "PC approx"), lty = 1:2, lwd =
c(2,2), ncol=2,
         col= c("black","red"))
}
dev.off()

## 2.3 PC components interpretation ----
svg(file = 'plots/PCA_inerpret.svg', width = 7, height = 4)
par(mfrow=c(1,1), mar = c(5,5,1,1),oma = c(0,0,0,0))
plot(times, eigen_swaps$vectors[,1], type = "l", col = "red",
     ylim = c(-1,1), lwd = 2, ylab =  "weight", xlab = "years")
lines(times, eigen_swaps$vectors[,2], type = "l", col = "blue", lwd = 2)
```

```
lines(times, eigen_swaps$vectors[,3], type = "l", col = "green", lwd = 2)
lines(times, eigen_swaps$vectors[,4], type = "l", col = "black", lwd = 2)
legend("bottom", legend = paste0("PC",1:4), lwd = rep(2,4),
       col = c("red","blue","green","black"), ncol = 4)
dev.off()

colors <- rainbow(dim)

svg(file = 'plots/PCA_time.svg', width = 10, height = 6)
par(mfrow=c(1,1), oma = c(1,1,1,1), mar = c(1,1,1,1))
plot(dates,PC[,1],type="l",col=colors[1], ylim = c(-8,6), xlab = "time", ylab
= "")
for (i in 2:dim) {
  lines(dates,PC[,i],col=colors[i])
}

legend("bottom",legend = paste("PC",1:dim), col = colors,
       lwd = rep(1,12),ncol=6)
dev.off()

# 3. Time series estimation ----

colnames(PC) <- paste0("PC",seq(1,15))
tsPCX <- xts(PC, order.by = swaps_df4$date)
tsPC <- tsPCX

tsPC1 <- tsPCX[,1]
tsPC2 <- tsPCX[,2]
tsPC3 <- tsPCX[,3]


par(old.par)
plot(dates, tsPC1, type = "l", col = "red")
lines(dates, tsPC2, type = "l",
      col = "blue")
lines(dates, tsPC3, type = "l",
      col = "green")

## 3.1 PC1 ----

### 3.1.1 Stationarity check ----

tsPC <- tsPCX[,1]

svg(file = 'plots/PC1_hist.svg', width = 7, height = 4)
par(mfrow=c(1,1), mar = c(5,5,1,1),oma = c(0,0,0,0))
plot(tsPC, type = "l", main = "PC1 (level)", col = "blue")
dev.off()

ggdraw() +
  draw_plot(ggAcf(tsPC), x = 0, y = .5, width = .5, height = .5) +
  draw_plot(ggPacf(tsPC), x = .5, y = .5, width = .5, height = .5) +
  draw_plot(autoplot(tsPC), x = 0, y = 0, width = 1, height = 0.5)

adf.test(tsPC)
pp.test(tsPC)
```

```
kpss.test(tsPC)

dtsPC <- diff_narm(tsPC)

ggdraw() +
  draw_plot(ggAcf(dtsPC), x = 0, y = .5, width = .5, height = .5) +
  draw_plot(ggPacf(dtsPC), x = .5, y = .5, width = .5, height = .5) +
  draw_plot(autoplot(dtsPC), x = 0, y = 0, width = 1, height = 0.5)

station <- list()
station[[1]] <- adf.test(dtsPC)
station[[2]] <- pp.test(dtsPC)
station[[3]] <- kpss.test(dtsPC)

statistic <- c()
pvalue <- c()
alternative <- c()

for (i in 1:3) statistic <- c(statistic, station[[i]]$statistic)
for (i in 1:3) pvalue <- c(pvalue, station[[i]]$p.value)
for (i in 1:3) alternative <- c(alternative, station[[i]]$alternative)
alternative[3] <- "non stationary"

station <- data.frame("statistic" = statistic, "pvalue" = pvalue,
"alternative hypothesis"=
                        alternative, row.names = c("Dickey-Fuller",
"Phillips-Perron", "KPSS"))

write.xlsx(station, file = "tables/stationarityPC1.xlsx")

### 3.1.2 Estimating the model ARMA GARCH ----

#### 3.1.2.1 ARIMA sGARCH model fitting ----

type.model <- "sGARCH"

filename_results_dtsPC <- paste0("results_",type.model,"_dts",
                                 colnames(tsPC),".csv")
filename_param_dtsPC <- paste0("param_",type.model,"_dts",
                               colnames(tsPC),".csv")

# tests multiple ARMA order GARCH order models
results <- tryCatch(read.csv(filename_results_dtsPC), error = function(e) 1,
                    warning = function(w) 1)
# check if there is a file with the results and otherwise, calculate them
if (typeof(results) == "double"){
  comb_full <- crossing(ar1 = 0:1, ar2 = 0:1, ar3 = 0:1,
                        ma1 = 0:1, ma2 = 0:1, ma3 = 0:1,
                        alpha1 = 0:1, alpha2 = 0:1, alpha3 = 0:1,
                        beta1 = 0:1, beta2 = 0:1, beta3 = 0:1,
                        mean = c(TRUE, FALSE),
                        distr = c("std","norm","ged","nig"))
  write.csv(comb_full, filename_param_dtsPC, row.names = FALSE)
  results <- ARIMA_GARCH_test(param = comb_full, ts = dtsPC,
                              filename = filename_results_dtsPC,
                              type.model = type.model)
```

```
  param <- comb_full
} else param <- read.csv(filename_param_dtsPC)

# filter only best models
alpha <- 1 - 0.995

valid_models <- results %>% mutate(rankAIC = rank(-akaike),
                                   rankBIC = rank(-bic),
                                   rankSHIB = rank(-shibata),
                                   rankHANQ = rank(-hannan_quinn),
                                   score = rankAIC + rankBIC + rankSHIB +
                                     rankHANQ) %>%
  filter(max_p_value < alpha, max_estim < 1) %>% arrange(desc(score)) %>%
  select(-rankAIC, -rankBIC, -rankSHIB, -rankHANQ)

# see models selected
param_valid <- param[valid_models$N,]
valid_models

valid_models$distr <- param_valid$distr
param_valid

# see one particular models of those selected
(model_sGarch_PC1 <- fit_GARCH(dtsPC, param[valid_models$N[95],], type.model
= type.model))

#### 3.1.2.2 ARIMA eGARCH model fitting ----

type.model <- "eGARCH"

filename_results_dtsPC <- paste0("results_",type.model,"_dts",
                                 colnames(tsPC),".csv")
filename_param_dtsPC <- paste0("param_",type.model,"_dts",
                               colnames(tsPC),".csv")

# tests multiple ARMA order GARCH order models
results <- tryCatch(read.csv(filename_results_dtsPC), error = function(e) 1,
                    warning = function(w) 1)
# check if there is a file with the results and otherwise, calculate them
if (typeof(results) == "double"){
  comb_full <- crossing(ar1 = 0:1, ar2 = 0:1, ar3 = 0:1,
                        ma1 = 0:1, ma2 = 0:1, ma3 = 0:1,
                        alpha1 = 0:1, alpha2 = 0:1, alpha3 = 0:1,
                        beta1 = 0:1, beta2 = 0:1, beta3 = 0:1,
                        mean = c(TRUE, FALSE),
                        distr = c("std","norm","ged","nig"))
  write.csv(comb_full, filename_param_dtsPC, row.names = FALSE)
  results <- ARIMA_GARCH_test(param = comb_full, ts = dtsPC,
                              filename = filename_results_dtsPC,
                              type.model = type.model)
  param <- comb_full
} else param <- read.csv(filename_param_dtsPC)

# filter only best models
alpha <- 1 - 0.995
```

```
valid_models <- results %>% mutate(rankAIC = rank(-akaike),
                                   rankBIC = rank(-bic),
                                   rankSHIB = rank(-shibata),
                                   rankHANQ = rank(-hannan_quinn),
                                   score = rankAIC + rankBIC + rankSHIB +
                                     rankHANQ) %>%
  filter(max_p_value < alpha, max_estim < 1) %>% arrange(desc(score)) %>%
  select(-rankAIC, -rankBIC, -rankSHIB, -rankHANQ)

# see models selected
param_valid <- param[valid_models$N,]
valid_models

valid_models$distr <- param_valid$distr

# see one particular models of those selected
model_eGarch_PC1 <- fit_GARCH(dtsPC, param[valid_models$N[12],], type.model =
type.model)

#### 3.1.2.3 ARIMA gjrGARCH model fitting ----

type.model <- "gjrGARCH"

filename_results_dtsPC <- paste0("results_",type.model,"_dts",
                                 colnames(tsPC),".csv")
filename_param_dtsPC <- paste0("param_",type.model,"_dts",
                               colnames(tsPC),".csv")

# tests multiple ARMA order GARCH order models
results <- tryCatch(read.csv(filename_results_dtsPC), error = function(e) 1,
                    warning = function(w) 1)
# check if there is a file with the results and otherwise, calculate them
if (typeof(results) == "double"){
  comb_full <- crossing(ar1 = 0:1, ar2 = 0:1, ar3 = 0:1,
                        ma1 = 0:1, ma2 = 0:1, ma3 = 0:1,
                        alpha1 = 0:1, alpha2 = 0:1, alpha3 = 0:1,
                        beta1 = 0:1, beta2 = 0:1, beta3 = 0:1,
                        mean = c(TRUE, FALSE),
                        distr = c("std","norm","ged","nig"))
  write.csv(comb_full, filename_param_dtsPC, row.names = FALSE)
  results <- ARIMA_GARCH_test(param = comb_full, ts = dtsPC,
                              filename = filename_results_dtsPC,
                              type.model = type.model)
  param <- comb_full
} else param <- read.csv(filename_param_dtsPC)

# filter only best models
alpha <- 1 - 0.995

valid_models <- results %>% mutate(rankAIC = rank(-akaike),
                                   rankBIC = rank(-bic),
                                   rankSHIB = rank(-shibata),
                                   rankHANQ = rank(-hannan_quinn),
                                   score = rankAIC + rankBIC + rankSHIB +
                                     rankHANQ) %>%
  filter(max_p_value < alpha, max_estim < 0.9) %>% arrange(desc(score)) %>%
```

```r
  select(-rankAIC, -rankBIC, -rankSHIB, -rankHANQ)

# see models selected
param_valid <- param[valid_models$N,]
valid_models

valid_models$distr <- param_valid$distr

# see one particular models of those selected
model_gjrGarch_PC1 <- fit_GARCH(dtsPC, param[valid_models$N[3],], type.model
= type.model)

### 3.1.3 VaR backtesting ####

model_dtsPC1 <- model_sGarch_PC1
model <- model_sGarch_PC1

spec <- fit_GARCH_spec(param[valid_models$N[9],], type.model = type.model)

modelroll <- ugarchroll(
  spec = spec, data=dtsPC, n.ahead = 1, forecast.length = 1119, refit.every =
100,
  refit.window = c("recursive"), solver = "hybrid", calculate.VaR = TRUE,
  VaR.alpha = c(0.01,0.05,0.95,0.99))

VaR001 <- modelroll@forecast$VaR[,"alpha(1%)"]
VaR005 <- modelroll@forecast$VaR[,"alpha(5%)"]
VaR095 <- modelroll@forecast$VaR[,"alpha(95%)"]
VaR099 <- modelroll@forecast$VaR[,"alpha(99%)"]
real <- modelroll@forecast$VaR[,"realized"]

mean(real<VaR001)
mean(real<VaR005)
mean(real<VaR095)
mean(real<VaR099)

VaR <- list()
VaR[[1]] <- VaRTest(alpha = 0.01, real, VaR001, conf.level = 0.995)
VaR[[2]] <- VaRTest(alpha = 0.05, real, VaR005, conf.level = 0.995)
VaR[[3]] <- VaRTest(alpha = 0.05, -real, -VaR095, conf.level = 0.995)
VaR[[4]] <- VaRTest(alpha = 0.01, -real, -VaR099, conf.level = 0.995)

VaRDur <- list()
VaRDur[[1]] <- VaRDurTest(alpha = 0.01, real, VaR001, conf.level = 0.995)
VaRDur[[2]] <- VaRDurTest(alpha = 0.05, real, VaR005, conf.level = 0.995)
VaRDur[[3]] <- VaRDurTest(alpha = 0.01, -real, -VaR095, conf.level = 0.995)
VaRDur[[4]] <- VaRDurTest(alpha = 0.05, -real, -VaR099, conf.level = 0.995)

DAC <- list()
DAC[[1]] <- DACTest(modelroll@forecast$density$Mu, real, test = "PT",
conf.level = 0.95)
DAC[[2]] <- DACTest(modelroll@forecast$density$Mu, real, test = "AG",
conf.level = 0.95)

results <- matrix(NA,ncol = 3, nrow = 4)
colnames(results) <- c("unconditional", "conditional", "duration")
```

```
rownames(results) <- c(0.01,0.05,0.95,0.99)
for (i in 1:4){
  results[i,] <- c(VaR[[i]]$uc.LRp, VaR[[i]]$cc.LRp, VaRDur[[i]]$LRp)
}

write.xlsx(as.data.frame(results), file = "tables/backtestPC1.xlsx")

## 3.2 PC2 ----

### 3.2.1 Stationarity check ----

tsPC <- tsPCX[,2]

svg(file = 'plots/PC2_hist.svg', width = 7, height = 4)
par(mfrow=c(1,1), mar = c(5,5,1,1),oma = c(0,0,0,0))
plot(tsPC, type = "l", main = "PC1 (level)", col = "blue")
dev.off()

ggdraw() +
  draw_plot(ggAcf(tsPC), x = 0, y = .5, width = .5, height = .5) +
  draw_plot(ggPacf(tsPC), x = .5, y = .5, width = .5, height = .5) +
  draw_plot(autoplot(tsPC), x = 0, y = 0, width = 1, height = 0.5)

adf.test(tsPC)
pp.test(tsPC)
kpss.test(tsPC)

dtsPC <- diff_narm(tsPC)

ggdraw() +
  draw_plot(ggAcf(dtsPC), x = 0, y = .5, width = .5, height = .5) +
  draw_plot(ggPacf(dtsPC), x = .5, y = .5, width = .5, height = .5) +
  draw_plot(autoplot(dtsPC), x = 0, y = 0, width = 1, height = 0.5)

station <- list()
station[[1]] <- adf.test(dtsPC)
station[[2]] <- pp.test(dtsPC)
station[[3]] <- kpss.test(dtsPC)

statistic <- c()
pvalue <- c()
alternative <- c()

for (i in 1:3) statistic <- c(statistic, station[[i]]$statistic)
for (i in 1:3) pvalue <- c(pvalue, station[[i]]$p.value)
for (i in 1:3) alternative <- c(alternative, station[[i]]$alternative)
alternative[3] <- "non stationary"

station <- data.frame("statistic" = statistic, "pvalue" = pvalue,
"alternative hypothesis"=
                        alternative, row.names = c("Dickey-Fuller",
"Phillips-Perron", "KPSS"))

write.xlsx(station, file = "tables/stationarityPC2.xlsx")

### 3.2.2 Estimating the model ARMA GARCH ----
```

```
#### 3.2.2.1 ARIMA sGARCH model fitting ----

type.model <- "sGARCH"

filename_results_dtsPC <- paste0("results_",type.model,"_dts",
                                 colnames(tsPC),".csv")
filename_param_dtsPC <- paste0("param_",type.model,"_dts",
                               colnames(tsPC),".csv")

# tests multiple ARMA order GARCH order models
results <- tryCatch(read.csv(filename_results_dtsPC), error = function(e) 1,
                    warning = function(w) 1)
# check if there is a file with the results and otherwise, calculate them
if (typeof(results) == "double"){
  comb_full <- crossing(ar1 = 0:1, ar2 = 0:1, ar3 = 0:1,
                        ma1 = 0:1, ma2 = 0:1, ma3 = 0:1,
                        alpha1 = 0:1, alpha2 = 0:1, alpha3 = 0:1,
                        beta1 = 0:1, beta2 = 0:1, beta3 = 0:1,
                        mean = c(TRUE, FALSE),
                        distr = c("std","norm","ged","nig"))
  write.csv(comb_full, filename_param_dtsPC, row.names = FALSE)
  results <- ARIMA_GARCH_test(param = comb_full, ts = dtsPC,
                              filename = filename_results_dtsPC,
                              type.model = type.model)
  param <- comb_full
} else param <- read.csv(filename_param_dtsPC)

# filter only best models
alpha <- 1 - 0.995

valid_models <- results %>% mutate(rankAIC = rank(-akaike),
                                   rankBIC = rank(-bic),
                                   rankSHIB = rank(-shibata),
                                   rankHANQ = rank(-hannan_quinn),
                                   score = rankAIC + rankBIC + rankSHIB +
                                     rankHANQ) %>%
  filter(max_p_value < alpha, max_estim < 1) %>% arrange(desc(score)) %>%
  select(-rankAIC, -rankBIC, -rankSHIB, -rankHANQ)

# see models selected
param_valid <- param[valid_models$N,]
valid_models

valid_models$distr <- param_valid$distr

# see one particular models of those selected
(model_sGarch_PC2 <- fit_GARCH(dtsPC, param[valid_models$N[12],], type.model
= type.model))


#### 3.2.2.2 ARIMA eGARCH model fitting ----

type.model <- "eGARCH"

filename_results_dtsPC <- paste0("results_",type.model,"_dts",
```

```
                                    colnames(tsPC),".csv")
filename_param_dtsPC <- paste0("param_",type.model,"_dts",
                                  colnames(tsPC),".csv")

# tests multiple ARMA order GARCH order models
results <- tryCatch(read.csv(filename_results_dtsPC), error = function(e) 1,
                    warning = function(w) 1)
# check if there is a file with the results and otherwise, calculate them
if (typeof(results) == "double"){
  comb_full <- crossing(ar1 = 0:1, ar2 = 0:1, ar3 = 0:1,
                        ma1 = 0:1, ma2 = 0:1, ma3 = 0:1,
                        alpha1 = 0:1, alpha2 = 0:1, alpha3 = 0:1,
                        beta1 = 0:1, beta2 = 0:1, beta3 = 0:1,
                        mean = c(TRUE, FALSE),
                        distr = c("std","norm","ged","nig"))
  write.csv(comb_full, filename_param_dtsPC, row.names = FALSE)
  results <- ARIMA_GARCH_test(param = comb_full, ts = dtsPC,
                              filename = filename_results_dtsPC,
                              type.model = type.model)
  param <- comb_full
} else param <- read.csv(filename_param_dtsPC)

# filter only best models
alpha <- 1 - 0.995

valid_models <- results %>% mutate(rankAIC = rank(-akaike),
                                   rankBIC = rank(-bic),
                                   rankSHIB = rank(-shibata),
                                   rankHANQ = rank(-hannan_quinn),
                                   score = rankAIC + rankBIC + rankSHIB +
                                     rankHANQ) %>%
  filter(max_p_value < alpha, max_estim < 1) %>% arrange(desc(score)) %>%
  select(-rankAIC, -rankBIC, -rankSHIB, -rankHANQ)

# see models selected
param_valid <- param[valid_models$N,]
valid_models

valid_models$distr <- param_valid$distr

# see one particular models of those selected
model_eGarch_PC2 <- fit_GARCH(dtsPC, param[valid_models$N[3],], type.model =
type.model)

#### 3.2.2.3 ARIMA gjrGARCH model fitting ----

type.model <- "gjrGARCH"

filename_results_dtsPC <- paste0("results_",type.model,"_dts",
                                    colnames(tsPC),".csv")
filename_param_dtsPC <- paste0("param_",type.model,"_dts",
                                  colnames(tsPC),".csv")

# tests multiple ARMA order GARCH order models
results <- tryCatch(read.csv(filename_results_dtsPC), error = function(e) 1,
                    warning = function(w) 1)
```

```
# check if there is a file with the results and otherwise, calculate them
if (typeof(results) == "double"){
  comb_full <- crossing(ar1 = 0:1, ar2 = 0:1, ar3 = 0:1,
                        ma1 = 0:1, ma2 = 0:1, ma3 = 0:1,
                        alpha1 = 0:1, alpha2 = 0:1, alpha3 = 0:1,
                        beta1 = 0:1, beta2 = 0:1, beta3 = 0:1,
                        mean = c(TRUE, FALSE),
                        distr = c("std","norm","ged","nig"))
  write.csv(comb_full, filename_param_dtsPC, row.names = FALSE)
  results <- ARIMA_GARCH_test(param = comb_full, ts = dtsPC,
                              filename = filename_results_dtsPC,
                              type.model = type.model)
  param <- comb_full
} else param <- read.csv(filename_param_dtsPC)

# filter only best models
alpha <- 1 - 0.995

valid_models <- results %>% mutate(rankAIC = rank(-akaike),
                                   rankBIC = rank(-bic),
                                   rankSHIB = rank(-shibata),
                                   rankHANQ = rank(-hannan_quinn),
                                   score = rankAIC + rankBIC + rankSHIB +
                                     rankHANQ) %>%
  filter(max_p_value < alpha, max_estim < 1) %>% arrange(desc(score)) %>%
  select(-rankAIC, -rankBIC, -rankSHIB, -rankHANQ)

# see models selected
param_valid <- param[valid_models$N,]
valid_models

valid_models$distr <- param_valid$distr

# see one particular models of those selected
model_gjrGarch_PC2 <- fit_GARCH(dtsPC, param[valid_models$N[3],], type.model
= type.model)

### 3.2.3 VaR backtesting ####

model_dtsPC2 <- model_sGarch_PC2
model <- model_sGarch_PC2

spec <- fit_GARCH_spec(param[valid_models$N[9],], type.model = type.model)

modelroll <- ugarchroll(
  spec=spec, data=dtsPC, n.ahead = 1, forecast.length = 1119, refit.every =
100,
  refit.window = c("recursive"), solver = "hybrid", calculate.VaR = TRUE,
  VaR.alpha = c(0.01,0.05,0.95,0.99))

VaR001 <- modelroll@forecast$VaR[,"alpha(1%)"]
VaR005 <- modelroll@forecast$VaR[,"alpha(5%)"]
VaR095 <- modelroll@forecast$VaR[,"alpha(95%)"]
VaR099 <- modelroll@forecast$VaR[,"alpha(99%)"]
real <- modelroll@forecast$VaR[,"realized"]
```

```
mean(real<VaR001)
mean(real<VaR005)
mean(real<VaR095)
mean(real<VaR099)

VaR <- list()
VaR[[1]] <- VaRTest(alpha = 0.01, real, VaR001, conf.level = 0.995)
VaR[[2]] <- VaRTest(alpha = 0.05, real, VaR005, conf.level = 0.995)
VaR[[3]] <- VaRTest(alpha = 0.05, -real, -VaR095, conf.level = 0.995)
VaR[[4]] <- VaRTest(alpha = 0.01, -real, -VaR099, conf.level = 0.995)

VaRDur <- list()
VaRDur[[1]] <- VaRDurTest(alpha = 0.01, real, VaR001, conf.level = 0.995)
VaRDur[[2]] <- VaRDurTest(alpha = 0.05, real, VaR005, conf.level = 0.995)
VaRDur[[3]] <- VaRDurTest(alpha = 0.01, -real, -VaR095, conf.level = 0.995)
VaRDur[[4]] <- VaRDurTest(alpha = 0.05, -real, -VaR099, conf.level = 0.995)

DAC <- list()
DAC[[1]] <- DACTest(modelroll@forecast$density$Mu, real, test = "PT",
conf.level = 0.95)
DAC[[2]] <- DACTest(modelroll@forecast$density$Mu, real, test = "AG",
conf.level = 0.95)

results <- matrix(NA,ncol = 3, nrow = 4)
colnames(results) <- c("unconditional", "conditional", "duration")
rownames(results) <- c(0.01,0.05,0.95,0.99)
for (i in 1:4){
  results[i,] <- c(VaR[[i]]$uc.LRp, VaR[[i]]$cc.LRp, VaRDur[[i]]$LRp)
}

write.xlsx(as.data.frame(results), file = "tables/backtestPC2.xlsx")

## 3.3 PC3 ----

### 3.3.1 Stationarity check ----

tsPC <- tsPCX[,3]

svg(file = 'plots/PC3_hist.svg', width = 7, height = 4)
par(mfrow=c(1,1), mar = c(5,5,1,1),oma = c(0,0,0,0))
plot(tsPC, type = "l", main = "PC1 (level)", col = "blue")
dev.off()


ggdraw() +
  draw_plot(ggAcf(tsPC), x = 0, y = .5, width = .5, height = .5) +
  draw_plot(ggPacf(tsPC), x = .5, y = .5, width = .5, height = .5) +
  draw_plot(autoplot(tsPC), x = 0, y = 0, width = 1, height = 0.5)

adf.test(tsPC)
pp.test(tsPC)
kpss.test(tsPC)

dtsPC <- diff_narm(tsPC)

ggdraw() +
```

```
  draw_plot(ggAcf(dtsPC), x = 0, y = .5, width = .5, height = .5) +
  draw_plot(ggPacf(dtsPC), x = .5, y = .5, width = .5, height = .5) +
  draw_plot(autoplot(dtsPC), x = 0, y = 0, width = 1, height = 0.5)

station <- list()
station[[1]] <- adf.test(dtsPC)
station[[2]] <- pp.test(dtsPC)
station[[3]] <- kpss.test(dtsPC)

statistic <- c()
pvalue <- c()
alternative <- c()

for (i in 1:3) statistic <- c(statistic, station[[i]]$statistic)
for (i in 1:3) pvalue <- c(pvalue, station[[i]]$p.value)
for (i in 1:3) alternative <- c(alternative, station[[i]]$alternative)
alternative[3] <- "non stationary"

station <- data.frame("statistic" = statistic, "pvalue" = pvalue,
"alternative hypothesis"=
                        alternative, row.names = c("Dickey-Fuller",
"Phillips-Perron", "KPSS"))

write.xlsx(station, file = "tables/stationarityPC3.xlsx")

### 3.3.2 Estimating the model ARMA GARCH ----

#### 3.3.2.1 ARIMA sGARCH model fitting ----

type.model <- "sGARCH"

filename_results_dtsPC <- paste0("results_",type.model,"_dts",
                                 colnames(tsPC),".csv")
filename_param_dtsPC <- paste0("param_",type.model,"_dts",
                               colnames(tsPC),".csv")

# tests multiple ARMA order GARCH order models
results <- tryCatch(read.csv(filename_results_dtsPC), error = function(e) 1,
                    warning = function(w) 1)
# check if there is a file with the results and otherwise, calculate them
if (typeof(results) == "double"){
  comb_full <- crossing(ar1 = 0:1, ar2 = 0:1, ar3 = 0:1,
                        ma1 = 0:1, ma2 = 0:1, ma3 = 0:1,
                        alpha1 = 0:1, alpha2 = 0:1, alpha3 = 0:1,
                        beta1 = 0:1, beta2 = 0:1, beta3 = 0:1,
                        mean = c(TRUE, FALSE),
                        distr = c("std","norm","ged","nig"))
  write.csv(comb_full, filename_param_dtsPC, row.names = FALSE)
  results <- ARIMA_GARCH_test(param = comb_full, ts = dtsPC,
                              filename = filename_results_dtsPC,
                              type.model = type.model)
  param <- comb_full
} else param <- read.csv(filename_param_dtsPC)

# filter only best models
alpha <- 1 - 0.995
```

```
valid_models <- results %>% mutate(rankAIC = rank(-akaike),
                                   rankBIC = rank(-bic),
                                   rankSHIB = rank(-shibata),
                                   rankHANQ = rank(-hannan_quinn),
                                   score = rankAIC + rankBIC + rankSHIB +
                                     rankHANQ) %>%
  filter(max_p_value < alpha, max_estim < 1) %>% arrange(desc(score)) %>%
  select(-rankAIC, -rankBIC, -rankSHIB, -rankHANQ)

# see models selected
param_valid <- param[valid_models$N,]
valid_models

valid_models$distr <- param_valid$distr

# see one particular models of those selected
(model_sGarch_PC3 <- fit_GARCH(dtsPC, param[valid_models$N[26],], type.model
= type.model))


#### 3.3.2.2 ARIMA eGARCH model fitting ----

type.model <- "eGARCH"

filename_results_dtsPC <- paste0("results_",type.model,"_dts",
                                 colnames(tsPC),".csv")
filename_param_dtsPC <- paste0("param_",type.model,"_dts",
                               colnames(tsPC),".csv")

# tests multiple ARMA order GARCH order models
results <- tryCatch(read.csv(filename_results_dtsPC), error = function(e) 1,
                    warning = function(w) 1)
# check if there is a file with the results and otherwise, calculate them
if (typeof(results) == "double"){
  comb_full <- crossing(ar1 = 0:1, ar2 = 0:1, ar3 = 0:1,
                        ma1 = 0:1, ma2 = 0:1, ma3 = 0:1,
                        alpha1 = 0:1, alpha2 = 0:1, alpha3 = 0:1,
                        beta1 = 0:1, beta2 = 0:1, beta3 = 0:1,
                        mean = c(TRUE, FALSE),
                        distr = c("std","norm","ged","nig"))
  write.csv(comb_full, filename_param_dtsPC, row.names = FALSE)
  results <- ARIMA_GARCH_test(param = comb_full, ts = dtsPC,
                              filename = filename_results_dtsPC,
                              type.model = type.model)
  param <- comb_full
} else param <- read.csv(filename_param_dtsPC)

# filter only best models
alpha <- 1 - 0.995

valid_models <- results %>% mutate(rankAIC = rank(-akaike),
                                   rankBIC = rank(-bic),
                                   rankSHIB = rank(-shibata),
                                   rankHANQ = rank(-hannan_quinn),
                                   score = rankAIC + rankBIC + rankSHIB +
```

```
                                    rankHANQ) %>%
  filter(max_p_value < alpha, max_estim < 1) %>% arrange(desc(score)) %>%
  select(-rankAIC, -rankBIC, -rankSHIB, -rankHANQ)

# see models selected
param_valid <- param[valid_models$N,]
valid_models

valid_models$distr <- param_valid$distr

# see one particular models of those selected
model_eGarch_PC3 <- fit_GARCH(dtsPC, param[valid_models$N[3],], type.model =
type.model)

#### 3.3.2.3 ARIMA gjrGARCH model fitting ----

type.model <- "gjrGARCH"

filename_results_dtsPC <- paste0("results_",type.model,"_dts",
                                 colnames(tsPC),".csv")
filename_param_dtsPC <- paste0("param_",type.model,"_dts",
                               colnames(tsPC),".csv")

# tests multiple ARMA order GARCH order models
results <- tryCatch(read.csv(filename_results_dtsPC), error = function(e) 1,
                    warning = function(w) 1)
# check if there is a file with the results and otherwise, calculate them
if (typeof(results) == "double"){
  comb_full <- crossing(ar1 = 0:1, ar2 = 0:1, ar3 = 0:1,
                        ma1 = 0:1, ma2 = 0:1, ma3 = 0:1,
                        alpha1 = 0:1, alpha2 = 0:1, alpha3 = 0:1,
                        beta1 = 0:1, beta2 = 0:1, beta3 = 0:1,
                        mean = c(TRUE, FALSE),
                        distr = c("std","norm","ged","nig"))
  write.csv(comb_full, filename_param_dtsPC, row.names = FALSE)
  results <- ARIMA_GARCH_test(param = comb_full, ts = dtsPC,
                              filename = filename_results_dtsPC,
                              type.model = type.model)
  param <- comb_full
} else param <- read.csv(filename_param_dtsPC)

# filter only best models
alpha <- 1 - 0.995

valid_models <- results %>% mutate(rankAIC = rank(-akaike),
                                   rankBIC = rank(-bic),
                                   rankSHIB = rank(-shibata),
                                   rankHANQ = rank(-hannan_quinn),
                                   score = rankAIC + rankBIC + rankSHIB +
                                     rankHANQ) %>%
  filter(max_p_value < alpha, max_estim < 1) %>% arrange(desc(score)) %>%
  select(-rankAIC, -rankBIC, -rankSHIB, -rankHANQ)

# see models selected
param_valid <- param[valid_models$N,]
valid_models
```

```r
valid_models$distr <- param_valid$distr

# see one particular models of those selected
model_gjrGarch_PC3 <- fit_GARCH(dtsPC, param[valid_models$N[3],], type.model
= type.model)

### 3.3.3 VaR backtesting ####

model_dtsPC3 <- model_sGarch_PC3
model <- model_sGarch_PC3

spec <- fit_GARCH_spec(param[valid_models$N[9],], type.model = type.model)

modelroll=ugarchroll(
  spec=spec, data=dtsPC, n.ahead = 1, forecast.length = 1119, refit.every =
100,
  refit.window = c("recursive"), solver = "hybrid", calculate.VaR = TRUE,
  VaR.alpha = c(0.01,0.05,0.95,0.99))

VaR001 <- modelroll@forecast$VaR[,"alpha(1%)"]
VaR005 <- modelroll@forecast$VaR[,"alpha(5%)"]
VaR095 <- modelroll@forecast$VaR[,"alpha(95%)"]
VaR099 <- modelroll@forecast$VaR[,"alpha(99%)"]
real <- modelroll@forecast$VaR[,"realized"]

mean(real<VaR001)
mean(real<VaR005)
mean(real<VaR095)
mean(real<VaR099)

VaR <- list()
VaR[[1]] <- VaRTest(alpha = 0.01, real, VaR001, conf.level = 0.995)
VaR[[2]] <- VaRTest(alpha = 0.05, real, VaR005, conf.level = 0.995)
VaR[[3]] <- VaRTest(alpha = 0.05, -real, -VaR095, conf.level = 0.995)
VaR[[4]] <- VaRTest(alpha = 0.01, -real, -VaR099, conf.level = 0.995)

VaRDur <- list()
VaRDur[[1]] <- VaRDurTest(alpha = 0.01, real, VaR001, conf.level = 0.995)
VaRDur[[2]] <- VaRDurTest(alpha = 0.05, real, VaR005, conf.level = 0.995)
VaRDur[[3]] <- VaRDurTest(alpha = 0.01, -real, -VaR095, conf.level = 0.995)
VaRDur[[4]] <- VaRDurTest(alpha = 0.05, -real, -VaR099, conf.level = 0.995)

DAC <- list()
DAC[[1]] <- DACTest(modelroll@forecast$density$Mu, real, test = "PT",
conf.level = 0.95)
DAC[[2]] <- DACTest(modelroll@forecast$density$Mu, real, test = "AG",
conf.level = 0.95)

results <- matrix(NA,ncol = 3, nrow = 4)
colnames(results) <- c("unconditional", "conditional", "duration")
rownames(results) <- c(0.01,0.05,0.95,0.99)
for (i in 1:4){
  results[i,] <- c(VaR[[i]]$uc.LRp, VaR[[i]]$cc.LRp, VaRDur[[i]]$LRp)
}
```

```r
write.xlsx(as.data.frame(results), file = "tables/backtestPC3.xlsx")

## 3.X STOXX600 ----

DJSTOXX <- xts(market_df$DJSTOXX, order.by = market_df$date)

### 3.X.1 Stationarity check ----

tsPC <- DJSTOXX
colnames(tsPC) <- "DJSTOXX"

svg(file = 'plots/DJSTOXX.svg', width = 7, height = 4)
par(mfrow=c(1,1), mar = c(5,5,1,1),oma = c(0,0,0,0))
plot(tsPC, type = "l", main = "DJSTOXX", col = "blue")
dev.off()

ggdraw() +
  draw_plot(ggAcf(tsPC), x = 0, y = .5, width = .5, height = .5) +
  draw_plot(ggPacf(tsPC), x = .5, y = .5, width = .5, height = .5) +
  draw_plot(autoplot(tsPC), x = 0, y = 0, width = 1, height = 0.5)

adf.test(tsPC)
pp.test(tsPC)
kpss.test(tsPC)

dtsPC <- diff_narm(tsPC)/tsPC

ggdraw() +
  draw_plot(ggAcf(dtsPC), x = 0, y = .5, width = .5, height = .5) +
  draw_plot(ggPacf(dtsPC), x = .5, y = .5, width = .5, height = .5) +
  draw_plot(autoplot(dtsPC), x = 0, y = 0, width = 1, height = 0.5)

station <- list()
station[[1]] <- adf.test(dtsPC)
station[[2]] <- pp.test(dtsPC)
station[[3]] <- kpss.test(dtsPC)

statistic <- c()
pvalue <- c()
alternative <- c()

for (i in 1:3) statistic <- c(statistic, station[[i]]$statistic)
for (i in 1:3) pvalue <- c(pvalue, station[[i]]$p.value)
for (i in 1:3) alternative <- c(alternative, station[[i]]$alternative)
alternative[3] <- "non stationary"

station <- data.frame("statistic" = statistic, "pvalue" = pvalue,
"alternative hypothesis"=
                        alternative, row.names = c("Dickey-Fuller",
"Phillips-Perron", "KPSS"))

write.xlsx(station, file = "tables/stationarityDJSTOXX.xlsx")

#### 3.X.2.1 ARIMA sGARCH model fitting ----

type.model <- "sGARCH"
```

```
filename_results_dtsPC <- paste0("results_",type.model,"_dts",
                                 colnames(tsPC),".csv")
filename_param_dtsPC <- paste0("param_",type.model,"_dts",
                               colnames(tsPC),".csv")

# tests multiple ARMA order GARCH order models
results <- tryCatch(read.csv(filename_results_dtsPC), error = function(e) 1,
                    warning = function(w) 1)
# check if there is a file with the results and otherwise, calculate them
if (typeof(results) == "double"){
  comb_full <- crossing(ar1 = 0:1, ar2 = 0:1, ar3 = 0:1,
                        ma1 = 0:1, ma2 = 0:1, ma3 = 0:1,
                        alpha1 = 0:1, alpha2 = 0:1, alpha3 = 0:1,
                        beta1 = 0:1, beta2 = 0:1, beta3 = 0:1,
                        mean = c(TRUE, FALSE),
                        distr = c("std","norm","ged","nig"))
  write.csv(comb_full, filename_param_dtsPC, row.names = FALSE)
  results <- ARIMA_GARCH_test(param = comb_full, ts = dtsPC,
                              filename = filename_results_dtsPC,
                              type.model = type.model)
  param <- comb_full
} else param <- read.csv(filename_param_dtsPC)

# filter only best models
alpha <- 1 - 0.995

valid_models <- results %>% mutate(rankAIC = rank(-akaike),
                                   rankBIC = rank(-bic),
                                   rankSHIB = rank(-shibata),
                                   rankHANQ = rank(-hannan_quinn),
                                   score = rankAIC + rankBIC + rankSHIB +
                                     rankHANQ) %>%
  filter(max_p_value < alpha, max_estim < 1) %>% arrange(desc(score)) %>%
  select(-rankAIC, -rankBIC, -rankSHIB, -rankHANQ)

# see models selected
param_valid <- param[valid_models$N,]
valid_models

valid_models$distr <- param_valid$distr

# see one particular models of those selected
(model_sGarch_DJSTOXX <- fit_GARCH(dtsPC, param[valid_models$N[8],],
type.model = type.model))

### 3.X.3 VaR backtesting ####

model_dtsDJSTOXX <- model_sGarch_DJSTOXX
model <- model_sGarch_DJSTOXX

spec <- fit_GARCH_spec(param[valid_models$N[9],], type.model = type.model)

modelroll=ugarchroll(
  spec=spec, data=dtsPC, n.ahead = 1, forecast.length = 1119, refit.every =
100,
```

```
  refit.window = c("recursive"), solver = "hybrid", calculate.VaR = TRUE,
  VaR.alpha = c(0.01,0.05,0.95,0.99))

VaR001 <- modelroll@forecast$VaR[,"alpha(1%)"]
VaR005 <- modelroll@forecast$VaR[,"alpha(5%)"]
VaR095 <- modelroll@forecast$VaR[,"alpha(95%)"]
VaR099 <- modelroll@forecast$VaR[,"alpha(99%)"]
real <- modelroll@forecast$VaR[,"realized"]

mean(real<VaR001)
mean(real<VaR005)
mean(real<VaR095)
mean(real<VaR099)

VaR <- list()
VaR[[1]] <- VaRTest(alpha = 0.01, real, VaR001, conf.level = 0.995)
VaR[[2]] <- VaRTest(alpha = 0.05, real, VaR005, conf.level = 0.995)
VaR[[3]] <- VaRTest(alpha = 0.05, -real, -VaR095, conf.level = 0.995)
VaR[[4]] <- VaRTest(alpha = 0.01, -real, -VaR099, conf.level = 0.995)

VaRDur <- list()
VaRDur[[1]] <- VaRDurTest(alpha = 0.01, real, VaR001, conf.level = 0.995)
VaRDur[[2]] <- VaRDurTest(alpha = 0.05, real, VaR005, conf.level = 0.995)
VaRDur[[3]] <- VaRDurTest(alpha = 0.01, -real, -VaR095, conf.level = 0.995)
VaRDur[[4]] <- VaRDurTest(alpha = 0.05, -real, -VaR099, conf.level = 0.995)

DAC <- list()
DAC[[1]] <- DACTest(modelroll@forecast$density$Mu, real, test = "PT",
conf.level = 0.95)
DAC[[2]] <- DACTest(modelroll@forecast$density$Mu, real, test = "AG",
conf.level = 0.95)

results <- matrix(NA,ncol = 3, nrow = 4)
colnames(results) <- c("unconditional", "conditional", "duration")
rownames(results) <- c(0.01,0.05,0.95,0.99)
for (i in 1:4){
  results[i,] <- c(VaR[[i]]$uc.LRp, VaR[[i]]$cc.LRp, VaRDur[[i]]$LRp)
}

write.xlsx(as.data.frame(results), file = "tables/backtestDJSTOXX.xlsx")

## 3.Y USDEUR ----

USDEUR <- xts(market_df$USDEUR, order.by = market_df$date)

### 3.Y.1 Stationarity check ----

tsPC <- USDEUR
colnames(tsPC) <- "USDEUR"

svg(file = 'plots/USDEUR.svg', width = 7, height = 4)
par(mfrow=c(1,1), mar = c(5,5,1,1),oma = c(0,0,0,0))
plot(tsPC, type = "l", main = "USDEUR", col = "blue")
dev.off()
```

```
ggdraw() +
  draw_plot(ggAcf(tsPC), x = 0, y = .5, width = .5, height = .5) +
  draw_plot(ggPacf(tsPC), x = .5, y = .5, width = .5, height = .5) +
  draw_plot(autoplot(tsPC), x = 0, y = 0, width = 1, height = 0.5)

adf.test(tsPC)
pp.test(tsPC)
kpss.test(tsPC)

dtsPC <- diff_narm(tsPC)/tsPC

ggdraw() +
  draw_plot(ggAcf(dtsPC), x = 0, y = .5, width = .5, height = .5) +
  draw_plot(ggPacf(dtsPC), x = .5, y = .5, width = .5, height = .5) +
  draw_plot(autoplot(dtsPC), x = 0, y = 0, width = 1, height = 0.5)

station <- list()
station[[1]] <- adf.test(dtsPC)
station[[2]] <- pp.test(dtsPC)
station[[3]] <- kpss.test(dtsPC)

statistic <- c()
pvalue <- c()
alternative <- c()

for (i in 1:3) statistic <- c(statistic, station[[i]]$statistic)
for (i in 1:3) pvalue <- c(pvalue, station[[i]]$p.value)
for (i in 1:3) alternative <- c(alternative, station[[i]]$alternative)
alternative[3] <- "non stationary"

station <- data.frame("statistic" = statistic, "pvalue" = pvalue,
"alternative hypothesis"=
                        alternative, row.names = c("Dickey-Fuller",
"Phillips-Perron", "KPSS"))

write.xlsx(station, file = "tables/stationarityUSDEUR.xlsx")

#### 3.1.2.1 ARIMA sGARCH model fitting ----

type.model <- "sGARCH"

filename_results_dtsPC <- paste0("results_",type.model,"_dts",
                                 colnames(tsPC),".csv")
filename_param_dtsPC <- paste0("param_",type.model,"_dts",
                               colnames(tsPC),".csv")

# tests multiple ARMA order GARCH order models
results <- tryCatch(read.csv(filename_results_dtsPC), error = function(e) 1,
                    warning = function(w) 1)
# check if there is a file with the results and otherwise, calculate them
if (typeof(results) == "double"){
  comb_full <- crossing(ar1 = 0:1, ar2 = 0:1, ar3 = 0:1,
                        ma1 = 0:1, ma2 = 0:1, ma3 = 0:1,
                        alpha1 = 0:1, alpha2 = 0:1, alpha3 = 0:1,
                        beta1 = 0:1, beta2 = 0:1, beta3 = 0:1,
                        mean = c(TRUE, FALSE),
```

```
                         distr = c("std","norm","ged","nig"))
  write.csv(comb_full, filename_param_dtsPC, row.names = FALSE)
  results <- ARIMA_GARCH_test(param = comb_full, ts = dtsPC,
                              filename = filename_results_dtsPC,
                              type.model = type.model)
  param <- comb_full
} else param <- read.csv(filename_param_dtsPC)

# filter only best models
alpha <- 1 - 0.995

valid_models <- results %>% mutate(rankAIC = rank(-akaike),
                                   rankBIC = rank(-bic),
                                   rankSHIB = rank(-shibata),
                                   rankHANQ = rank(-hannan_quinn),
                                   score = rankAIC + rankBIC + rankSHIB +
                                     rankHANQ) %>%
  filter(max_p_value < alpha, max_estim < 1) %>% arrange(desc(score)) %>%
  select(-rankAIC, -rankBIC, -rankSHIB, -rankHANQ)

# see models selected
param_valid <- param[valid_models$N,]
valid_models

valid_models$distr <- param_valid$distr

# see one particular models of those selected
(model_sGarch_USDEUR <- fit_GARCH(dtsPC, param[valid_models$N[9],],
type.model = type.model))

### 3.Y.3 VaR backtesting ####

model_dtsUSDEUR <- model_sGarch_USDEUR
model  <- model_sGarch_USDEUR

spec <- fit_GARCH_spec(param[valid_models$N[9],], type.model = type.model)

modelroll=ugarchroll(
  spec=spec, data=dtsPC, n.ahead = 1, forecast.length = 1119, refit.every =
100,
  refit.window = c("recursive"), solver = "hybrid", calculate.VaR = TRUE,
  VaR.alpha = c(0.01,0.05,0.95,0.99))

VaR001 <- modelroll@forecast$VaR[,"alpha(1%)"]
VaR005 <- modelroll@forecast$VaR[,"alpha(5%)"]
VaR095 <- modelroll@forecast$VaR[,"alpha(95%)"]
VaR099 <- modelroll@forecast$VaR[,"alpha(99%)"]
real <- modelroll@forecast$VaR[,"realized"]

mean(real<VaR001)
mean(real<VaR005)
mean(real<VaR095)
mean(real<VaR099)

VaR <- list()
VaR[[1]] <- VaRTest(alpha = 0.01, real, VaR001, conf.level = 0.995)
```

```
VaR[[2]] <- VaRTest(alpha = 0.05, real, VaR005, conf.level = 0.995)
VaR[[3]] <- VaRTest(alpha = 0.05, -real, -VaR095, conf.level = 0.995)
VaR[[4]] <- VaRTest(alpha = 0.01, -real, -VaR099, conf.level = 0.995)

VaRDur <- list()
VaRDur[[1]] <- VaRDurTest(alpha = 0.01, real, VaR001, conf.level = 0.995)
VaRDur[[2]] <- VaRDurTest(alpha = 0.05, real, VaR005, conf.level = 0.995)
VaRDur[[3]] <- VaRDurTest(alpha = 0.01, -real, -VaR095, conf.level = 0.995)
VaRDur[[4]] <- VaRDurTest(alpha = 0.05, -real, -VaR099, conf.level = 0.995)

DAC <- list()
DAC[[1]] <- DACTest(modelroll@forecast$density$Mu, real, test = "PT",
conf.level = 0.95)
DAC[[2]] <- DACTest(modelroll@forecast$density$Mu, real, test = "AG",
conf.level = 0.95)

results <- matrix(NA,ncol = 3, nrow = 4)
colnames(results) <- c("unconditional", "conditional", "duration")
rownames(results) <- c(0.01,0.05,0.95,0.99)
for (i in 1:4){
  results[i,] <- c(VaR[[i]]$uc.LRp, VaR[[i]]$cc.LRp, VaRDur[[i]]$LRp)
}

write.xlsx(as.data.frame(results), file = "tables/backtestUSDEUR.xlsx")

## 3.0 Summary ----

write.xlsx(as.data.frame(model_dtsPC1@fit$matcoef), file =
"tables/modelPC1.xlsx")
write.xlsx(as.data.frame(model_dtsPC2@fit$matcoef), file =
"tables/modelPC2.xlsx")
write.xlsx(as.data.frame(model_dtsPC3@fit$matcoef), file =
"tables/modelPC3.xlsx")
write.xlsx(as.data.frame(model_dtsDJSTOXX@fit$matcoef), file =
"tables/modelDJSTOXX.xlsx")
write.xlsx(as.data.frame(model_dtsUSDEUR@fit$matcoef), file =
"tables/modelUSDEUR.xlsx")

write.xlsx(as.data.frame(infocriteria(model_dtsPC1)), file =
"tables/infocriteriaPC1.xlsx")
write.xlsx(as.data.frame(infocriteria(model_dtsPC2)), file =
"tables/infocriteriaPC2.xlsx")
write.xlsx(as.data.frame(infocriteria(model_dtsPC3)), file =
"tables/infocriteriaPC3.xlsx")
write.xlsx(as.data.frame(infocriteria(model_dtsDJSTOXX)), file =
"tables/modelDJSTOXX.xlsx")
write.xlsx(as.data.frame(infocriteria(model_dtsUSDEUR)), file =
"tables/modelUSDEUR.xlsx")


# 4. Copula estimation and diagnosis ----

## 4.1 Get residuals and analysis ----

model_dtsPC1
model_dtsPC2
```

```
model_dtsPC3
model_dtsDJSTOXX
model_dtsUSDEUR

models <- list(model_dtsPC1, model_dtsPC2, model_dtsPC3, model_dtsDJSTOXX,
model_dtsUSDEUR)
models_name <- c("dtsPC1", "dtsPC2", "dtsPC3", "dtsDJSTOXX", "dtsUSDEUR")

for (i in 1:length(models)){
  model <- models[[i]]

  # get standarized residuals
  s_res <- residuals(model)/sigma(model)
  svg(file = paste0("plots/hist_",models_name[i],".svg"), width = 4, height =
4)
  par(mfrow=c(1,1), mar = c(2,4,1,1),oma = c(0,0,0,0))
  hist(s_res, breaks = 40, main = paste0(gsub("dts","",models_name[i]),
                                         " model residuals"), xlab = "")
  dev.off()
  sd(s_res)

  svg(file = paste0("plots/autocorr_",models_name[i],".svg"), width = 4,
height = 2.5)
  ggdraw() +
    draw_plot(ggAcf(s_res, main = paste0(gsub("dts","",models_name[i])), xlab
= ""),
              x = 0, y = 0, width = .5, height = 1) +
    draw_plot(ggPacf(s_res, main = paste0(gsub("dts","",models_name[i])),
xlab = ""),
              x = 0.5, y = 0, width = .5, height = 1)
  dev.off()

  coef <- model@fit$coef

  lbtest <- matrix(NA, nrow = 20, ncol = 2)
  for (j in 1:20) {
    lb <- Box.test(s_res,lag = j, "Ljung-Box")
    lbtest[j,1] <- lb$statistic
    lbtest[j,2] <- lb$p.value
  }
  write.xlsx(as.data.frame(lbtest), file = paste0("tables/lbtest_",
                                                  models_name[i],".xlsx"))


  if (is.na(coef["skew"])) skew <- 1 else skew <- coef["skew"]
  if (is.na(coef["shape"])) shape <- 5 else shape <- coef["shape"]

  dist_model <- model@model$modeldesc$distribution

  # get cummulative distribution values
  u_res <- pdist(distribution = dist_model, s_res, skew = skew, shape =
shape)
  u_res <- abs(u_res)
  hist(u_res, breaks = 40)

  # plot it for reference
```

```
  svg(file = paste0("plots/qqplot_",models_name[i],".svg"), width = 4, height
= 4)
  par(mfrow=c(1,1), mar = c(2,2,2,1),oma = c(0,0,0,0))
  qqplot(rdist(distribution = dist_model,100000,
               skew = skew, shape = shape),coredata(s_res),
         main = paste0(gsub("dts","",models_name[i]), " model residuals"),
         xlab = "")
  qqline(coredata(s_res))
  dev.off()

  svg(file = paste0("plots/cumplot_",models_name[i],".svg"), width = 4,
height = 4)
  par(mfrow=c(1,1), mar = c(2,4,2,1),oma = c(0,0,0,0))
  plot(ecdf(coredata(s_res)), col = "red", xlim = c(-4,4),
       main = paste0(gsub("dts","",models_name[i]), " model residuals"), xlab
= "")
  lines(seq(-8,8,0.025),pdist(seq(-8,8,0.025),distribution = dist_model,
                               skew = skew, shape = shape), col = "black")
  legend("bottomright", legend = c("Fitted", "Empirical"), lwd = c(2,2),
         col = c("black", "red"), ncol = 1)
  dev.off()


  # goodness of fit tests
  jb <- jarque.bera.test(qnorm(u_res))
  ks <- ks.test(rdist(distribution = dist_model,100000,
                      skew = skew, shape = shape),coredata(s_res))
  cvm <- cvm.test(qnorm(u_res),"norm")
  t <- t.test(s_res)

  goftests <- matrix(NA, ncol = 2, nrow = 4)
  goftests[1,] <- c(jb$statistic, jb$p.value)
  goftests[2,] <- c(ks$statistic, ks$p.value)
  goftests[3,] <- c(cvm$statistic, cvm$p.value)
  goftests[4,] <- c(t$statistic[[1]], t$p.value[[1]])
  rownames(goftests) <- c("jarque-bera", "ks", "cvm", "t")

  write.xlsx(as.data.frame(goftests), file = paste0("tables/goftest_",
                                                      models_name[i],".xlsx"))

  if(i==1){
    s_res_all <- s_res
    # set u_res_all so it is a xts class object
    u_res_all <- s_res
    u_res_all[,1] <- u_res
  } else {
    s_res_all <- merge(s_res_all, s_res)
    u_res_all <- merge(u_res_all, u_res)
  }
}

colnames(s_res_all) <- models_name
colnames(u_res_all) <- models_name

## 4.2 Two by two copula estimation ----
```

```
pairs(coredata(s_res_all))

series_names <- paste0(gsub("dts","",models_name))

svg(file = paste0("plots/copula_all.svg"), width = 12, height = 12)
pairs(coredata(u_res_all),oma = c(2,2,0,0))
dev.off()

colnames(u_res_all) <- series_names
R <- cor(u_res_all)
svg(file = paste0("plots/corrgram.svg"), width = 4, height = 4)
corrgram(R,oma = c(1,1,1,1))
dev.off()

# independence tests and see what copula
d <- indepTestSim(nrow(u_res_all), p = 2)
combos <- crossing(u1 = 1:5, u2 = 1:5) %>% filter(u1 != u2)
indep_pvalue <- c()
loglike <- c()
pvalue <- c()
copula <- c()
for (i in 1:nrow(combos)){
  u1 <- combos[i,1]
  u2 <- combos[i,2]
  indep_pvalue <- c(indep_pvalue, indepTest(coredata(u_res_all[,c(u1,u2)]),
d)$pvalue)
  results <- testCopula(coredata(u_res_all[,c(u1,u2)]))
  results <- na.omit(results)
  loglike <- c(loglike, max(results$log.likelihood, na.rm = TRUE))
  pvalue <- c(pvalue, max(results$gof.pvalue))
  copula <- c(copula, results$copula[max(results$log.likelihood, na.rm =
TRUE)==results$log.likelihood])
}

results <- cbind(combos, indep_pvalue, loglike,  pvalue, copula)
results$u1 <- series_names[results$u1]
results$u2 <- series_names[results$u2]

write.xlsx(results, file = "tables/2by2copulas.xlsx")

## 4.3 Unique copula estimation ----

u <- coredata(u_res_all[,1:4])
N <- 1000
opt.meth = "L-BFGS-B"

nc <- fitCopula(copula = normalCopula(dim = 4, dispstr = "un"), data = u,
                optim.method = opt.meth, method = "irho")

nc@loglik <- sum(log(dCopula(u, nc@copula)))

(gofnc <- gofCopula(normalCopula(dim = 4, dispstr = "un"), method = "Sn",
                    estim.method = "mpl", N = N,x = u, simulation = "mult"))

tc <- fitCopula(copula = tCopula(dim = 4, dispstr = "un"), data = u,
                optim.method = opt.meth, method = "itau.mpl")
```

```
(goftc <- gofCopula(tCopula(dim = 4, dispstr = "un", df.fixed = TRUE,
                            df = round(tc@estimate[7])),
                    estim.method = "mpl", N = N,x = u, simulation = "mult"))

# if error change optim.method as opt.meth = "Nelder-Mead"
tc <- fitCopula(copula = tCopula(dim = 4, dispstr = "un",
                                 df = round(tail(tc@estimate,1)), df.fixed =
TRUE),
                data = u,optim.method = opt.meth, method = "ml")


u_t2 <- rCopula(n = 1219, copula = tc@copula)
u_t <- pCopula(u = u, copula = tc@copula)
pairs(u_t2)
pairs(u)

u_t2 <- pCopula(u = u_t2, copula = tc@copula)
hist(u_t)
hist(u_t2)

fc <- fitCopula(copula = frankCopula(dim = 4), data = u,
                optim.method = opt.meth, method = "ml")

(goffc <- gofCopula(copula = frankCopula(dim = 4), data = u,
                    estim.method = "mpl", N = N,x = u, simulation = "mult"))

jc <- fitCopula(copula = joeCopula(dim = 4), data = u,
                optim.method = opt.meth, method = "ml")

(gofjc <- gofCopula(copula = joeCopula(dim = 4), data = u,
                    estim.method = "mpl", N = N,x = u, simulation = "mult"))

cc <- fitCopula(copula = claytonCopula(dim = 4), data = u,
                optim.method = opt.meth, method = "ml")

(gofcc <- gofCopula(copula = claytonCopula(dim = 4), data = u,
                    estim.method = "mpl", N = N,x = u, simulation = "mult"))

results <- data.frame("copula" = c("Normal","t-Student",
                                   "Clayton", "Frank","Joe"),
                      "log-likelihood" = c(nc@loglik,tc@loglik,cc@loglik,
                                           fc@loglik,jc@loglik),
                      "gof pvalue" = c(gofnc$p.value,goftc$p.value,

gofcc$p.value,goffc$p.value,gofjc$p.value),
                      "parameter" = c(NA,
round(tail(tc@copula@parameters,1),0),
                                      cc@estimate, fc@estimate, jc@estimate))

write.xlsx(results, file = "tables/fit_copulas.xlsx")

write.xlsx(as.data.frame(nc@estimate), file = "tables/normal_param.xlsx")
write.xlsx(as.data.frame(tc@estimate), file = "tables/tStud_param.xlsx")
```

```
combos <- crossing(u1 = 1:4, u2 = 1:4) %>% filter(u1 != u2) %>% as.matrix()

## 4.4. Copula fit plots ----
# it wont create the graphs properly in the loop so it is created manually

i <- 1
u_graph <- u_res_all[,combos[i,]]
u_rand <- rCopula(n = 1000000, copula = nc@copula)
u <- seq(0, 1, length.out = 50)
grid <- as.matrix(expand.grid(u1 = u, u2 = u))
val <- cbind(grid, z = C.n(grid, X = u_rand[,combos[i,]]))
cpG <- contourplot2(val, region = FALSE,
                    key = list(corner = c(0.01, 0.01),
                               lines = list(col = c(1,4), lwd = 2),
                               text = list(c("Copula",
                                             "Data")), oma =c(0,0,0,0),mar =
c(2,2,2,1)),
                    xlab= paste0("u",combos[i,1]),
                    ylab = paste0("u",combos[i,2]))
u <- seq(0, 1, length.out = 50)
grid <- as.matrix(expand.grid(u1 = u, u2 = u))
val <- cbind(grid, z = C.n(grid, X = u_graph))
cpCn <- contourplot2(val, region = FALSE, labels = FALSE, col = 4, oma =
c(0,0,0,0),mar = c(2,2,2,1))
file <- paste0("plots/fit_normal_",combos[i,1],"_",combos[i,2],".svg")
svg(file = file, width = 5, height = 5)
cpG + cpCn
dev.off()
i <- i + 1

i <- 1
u_graph <- u_res_all[,combos[i,]]
u_rand <- rCopula(n = 1000000, copula = tc@copula)
u <- seq(0, 1, length.out = 50)
grid <- as.matrix(expand.grid(u1 = u, u2 = u))
val <- cbind(grid, z = C.n(grid, X = u_rand[,combos[i,]]))
cpG <- contourplot2(val, region = FALSE,
                    key = list(corner = c(0.01, 0.01),
                               lines = list(col = c(1,4), lwd = 2),
                               text = list(c("Copula",
                                             "Data")), oma =c(0,0,0,0),mar =
c(2,2,2,1)),
                    xlab= paste0("u",combos[i,1]),
                    ylab = paste0("u",combos[i,2]))
u <- seq(0, 1, length.out = 50)
grid <- as.matrix(expand.grid(u1 = u, u2 = u))
val <- cbind(grid, z = C.n(grid, X = u_graph))
cpCn <- contourplot2(val, region = FALSE, labels = FALSE, col = 4, oma =
c(0,0,0,0),mar = c(2,2,2,1))
file <- paste0("plots/fit_t_",combos[i,1],"_",combos[i,2],".svg")
svg(file = file, width = 5, height = 5)
cpG + cpCn
dev.off()
i <- i + 1
```

```
i <- 1
u_graph <- u_res_all[,combos[i,]]
u_rand <- rCopula(n = 1000000, copula = cc@copula)
u <- seq(0, 1, length.out = 50)
grid <- as.matrix(expand.grid(u1 = u, u2 = u))
val <- cbind(grid, z = C.n(grid, X = u_rand[,combos[i,]]))
cpG <- contourplot2(val, region = FALSE,
                    key = list(corner = c(0.01, 0.01),
                               lines = list(col = c(1,4), lwd = 2),
                               text = list(c("Copula",
                                             "Data")), oma =c(0,0,0,0),mar =
c(2,2,2,1)),
                    xlab= paste0("u",combos[i,1]),
                    ylab = paste0("u",combos[i,2]))
u <- seq(0, 1, length.out = 50)
grid <- as.matrix(expand.grid(u1 = u, u2 = u))
val <- cbind(grid, z = C.n(grid, X = u_graph))
cpCn <- contourplot2(val, region = FALSE, labels = FALSE, col = 4, oma =
c(0,0,0,0),mar = c(2,2,2,1))
file <- paste0("plots/fit_clayton_",combos[i,1],"_",combos[i,2],".svg")
svg(file = file, width = 5, height = 5)
cpG + cpCn
dev.off()
i <- i + 1

# 5. Simulation ####

n_sim <- 5000

u_boot <- rCopula(n = n_sim*52, copula = bc@copula)
colnames(u_boot) <- colnames(u_res_all)[1:4]

## 5.1 Interest rate model ----

### 5.1.1 PC1 ----
nPC <- 1
model <- model_dtsPC1
dtsPC <- diff_narm(tsPCX[,nPC])
pb <- txtProgressBar(1,n_sim)
for (i in 1:n_sim){
  pos <- (i-1)*52+(1:52)
  simulated <- sim_ugarch(model, u_boot[pos,nPC])
  complete <- c(simulated, dtsPC)
  series <- cumsum(complete) + coredata(tsPCX[1,nPC])[[1]]
  if (i == 1) series_sim <- series else series_sim <- cbind(series_sim,
series)
  setTxtProgressBar(pb, i)
}

colnames(series_sim) <- paste0("sim_",1:n_sim)
series_sim_PC1 <- series_sim

### 5.1.2 PC2 ----
nPC <- 2
model <- model_dtsPC2
dtsPC <- diff_narm(tsPCX[,nPC])
```

```
for (i in 1:n_sim){
  pos <- (i-1)*52+(1:52)
  simulated <- sim_ugarch(model, u_boot[pos,nPC])
  complete <- c(simulated, dtsPC)
  series <- cumsum(complete) + coredata(tsPCX[1,nPC])[[1]]
  if (i == 1) series_sim <- series else series_sim <- cbind(series_sim,
series)
  setTxtProgressBar(pb, i)
}

colnames(series_sim) <- paste0("sim_",1:n_sim)
series_sim_PC2 <- series_sim

### 5.1.3 PC3 ----
nPC <- 3
model <- model_dtsPC3
dtsPC <- diff_narm(tsPCX[,nPC])

for (i in 1:n_sim){
  pos <- (i-1)*52+(1:52)
  simulated <- sim_ugarch(model, u_boot[pos,nPC])
  complete <- c(simulated, dtsPC)
  series <- cumsum(complete) + coredata(tsPCX[1,nPC])[[1]]
  if (i == 1) series_sim <- series else series_sim <- cbind(series_sim,
series)
  setTxtProgressBar(pb, i)
}

colnames(series_sim) <- paste0("sim_",1:n_sim)
series_sim_PC3 <- series_sim

## 5.2 EuroStoxx600 model ----

nPC <- 4
model <- model_dtsDJSTOXX
dtsPC <- diff_narm(DJSTOXX)/lag(DJSTOXX,1)
init <- coredata(DJSTOXX[-1][1])[[1]]
init <- init/(1+coredata(dtsPC[1])[[1]])

for (i in 1:n_sim){
  pos <- (i-1)*52+(1:52)
  simulated <- sim_ugarch(model, u_boot[pos,nPC])
  complete <- c(simulated, dtsPC)
  # in this case to undo is not simply cumulative sum since it is a
percentage
  series <- cumprod(1+complete)*init
  if (i == 1) series_sim <- series else series_sim <- cbind(series_sim,
series)
  setTxtProgressBar(pb, i)
}

colnames(series_sim) <- paste0("sim_",1:n_sim)
series_sim_DJSTOXX <- series_sim

## 5.3 Undo PC and organize data ----
```

```
market_df_sim <- mutate(market_df, type = "hist")
PC_sim <- matrix(data = 0, nrow = 52, ncol = 15)
market_sub_df <- market_df_sim[1:52,]
dates_sim <- index(tail(series_sim,52))

for (i in 1:n_sim) {
  # undo PC components to swaps curve
  PC1_sim <- series_sim_PC1[,i]
  PC2_sim <- series_sim_PC2[,i]
  PC3_sim <- series_sim_PC3[,i]
  PC_sim[,1:3] <- unlist(coredata(tail(cbind(PC1_sim,PC2_sim,PC3_sim),52)))
  # see PC decomposition code for reference
  swaps_sub <- PC_sim %*% t(eigen_swaps$vectors)
  swaps_sub <- unscale(swaps_sub, scale, center)
  # add also the EUROSTOXX series
  DJSTOXX_sim <- as.vector(unlist(coredata(tail(series_sim_DJSTOXX[,i],52))))
  # bind to already existing data
  market_sub_df2 <- as.data.frame(swaps_sub) %>% mutate(type =
paste0("sim_",i))
  market_sub_df2$date <- dates_sim
  market_sub_df2$DJSTOXX <- DJSTOXX_sim
  market_sub_df[,colnames(market_sub_df2)] <-
market_sub_df2[,colnames(market_sub_df2)]
  market_df_sim <- rbind(market_df_sim,market_sub_df)
}

## 5.4 Plot results ----

### 5.4.1 Yield curve ----

yield <- market_df_sim %>% filter(date == (head(dates_sim,1)-7)) %>%
  subset(select = -c(USDEUR,type,date)) %>% as.matrix()
estim <- market_df_sim %>% filter(date == tail(dates_sim,1)) %>%
  subset(select = -c(USDEUR,type,date)) %>% as.matrix()

quantiles <- c(0.005, 0.05,0.1)

env_top <- apply(estim, 2, quantile, probs = quantiles)
env_bot <- apply(estim, 2, quantile, probs = 1-quantiles)

# art 166 and 167
up <- c(.7,.7,.64,.59,.55,.52,.49,.47,.44,.42,0.37,.33,.26,.26-5/70*.06,
        .26-10/70*.06)
down <- c(.75,.65,.56,.5,.46,.42,.39,.36,.33,.31,.29,.27,.29,.29-5/70*0.09,
          .29-10/70*0.09)

eiopa_up <- (1+up)*yield[1,1:length(times)]
eiopa_down <- (1-down)*yield[1,1:length(times)]

cols <- rainbow(length(quantiles))

svg(file = "plots/shocks_interest.svg", width = 10, height = 6)
par(mfrow=c(1,1), mar = c(4,4,3,1),oma = c(0,0,0,0))
plot(times, yield[1,1:15], type = "l", ylim = c(0,7), col = "black", lwd =
"2",
```

```
    xlab = "maturity", ylab = "rate", main = "Interest rate yield curve")
for (i in 1:nrow(env_top)){
  lines(times, env_top[i,1:15], type = "l", col = cols[i])
  lines(times, env_bot[i,1:15], type = "l", col = cols[i])
}

lines(times, eiopa_down, type = "l", lty = 2, col = "red", lwd = 2)
lines(times, eiopa_up, type = "l", lty = 2, col = "red", lwd = 2)
legend("top", legend = c("Model estimate / Current", "Model VaR10% envelope",
                         "Model VaR5% envelope", "Model VaR0.5% envelope",
                         "EIOPA VaR0.5%"),
       lwd = c(2,1,1,1,3), lty = c(1,1,1,1,3),
       col = c("black", "blue", "green", "red", "red"), ncol = 3)
dev.off()

### 5.4.2 Equity ----

stock <- yield[,"DJSTOXX"]
stock_estim <- estim[,"DJSTOXX"]

stock_top <- quantile(stock_estim, quantiles)/stock
stock_bot <- quantile(stock_estim, 1-quantiles)/stock

# art 169
SA_interval <- DJSTOXX[paste(tail(dates,1)-3*365,tail(dates,1),sep ="::")]
SA <- 1/2*(stock-mean(SA_interval))/mean(SA_interval)
SA <- max(min(0.1,SA),-0.1)
eiopa_stock_up <- (1 + 0.39 + SA)
eiopa_stock_down <- (1 - 0.39 - SA)

svg(file = "plots/shocks_equity.svg", width = 10, height = 6)
par(mfrow=c(1,1), mar = c(4,4,3,1),oma = c(0,0,0,0))
hist(stock_estim/stock, breaks = 50, freq = FALSE,
    xlab = "Value",
    main = "One year ahead portfolio distribution",
    xlim = c(0,2))
lines(c(1,1), c(0,100), type = "l", col = "black", lwd = 2)
for (i in 1:nrow(env_top)){
  lines(c(stock_top[i],stock_top[i]), c(0,100), type = "l", col = cols[i])
  lines(c(stock_bot[i],stock_bot[i]), c(0,100), type = "l", col = cols[i])
}

lines(c(eiopa_stock_up,eiopa_stock_up), c(0,100), type = "l", col = "red",
      lwd = 2, lty = 2)
lines(c(eiopa_stock_down,eiopa_stock_down), c(0,100), type = "l", col =
"red",
      lwd = 2, lty = 2)
legend("right", legend = c("Model estimate", "Model VaR10%", "Model VaR5%",
                           "Model VaR0.5%", "EIOPA VaR0.5%"),
       lwd = c(2,1,1,1,3), lty = c(1,1,1,1,3),
       col = c("black", "blue", "green", "red", "red"), ncol = 1)
dev.off()

### 5.5.1 Portfolio 1 calculation ----

portfolio <- list(bond = list(mat = 8, cupon = 0.02, qty = 1000),
```

```
                  bond = list(mat = 10, cupon = 0.04, qty = 1000),
                  bond = list(mat = 5, cupon = 0.03, qty = 1000),
                  bond = list(mat = 3, cupon = 0.01, qty = 1000),
                  stock = list(price = 500))

portfolio_value <- portfolio_value_calc(portfolio, yield[1:length(times)],
                                        times, 1, duration = TRUE)
portfolio_value <- c(portfolio_value, "total" = sum(portfolio_value[1:2]))
portfolio_eiopa_results <- portfolio_value_calc(portfolio, eiopa_up, times,
                                                eiopa_stock_down)

portfolio_eiopa_results2 <- portfolio_value_calc(portfolio, eiopa_down,
times,
                                                 eiopa_stock_up)

#### 5.5.1.1 Standard formula ----

eiopa_cor <- matrix(c(1,0.5,0.5,1), nrow = 2, ncol = 2)
eiopa_cor2 <- matrix(c(1,0,0,1), nrow = 2, ncol = 2)

shocks_eiopa <- c(portfolio_eiopa_results["bonds"]-portfolio_value["bonds"],
                  portfolio_eiopa_results["stocks"]-
portfolio_value["stocks"])

shocks_eiopa["total"] <- -sqrt((-shocks_eiopa) %*% eiopa_cor %*% (-
shocks_eiopa))
names(shocks_eiopa) <- paste0("eiopa_shock_",c("bonds","stocks","total"))

shocks_eiopa2 <- c(portfolio_eiopa_results2["bonds"]-
portfolio_value["bonds"],
                   portfolio_eiopa_results2["stocks"]-
portfolio_value["stocks"])

shocks_eiopa2["total"] <- sqrt((-shocks_eiopa2) %*% eiopa_cor2 %*% (-
shocks_eiopa2))
names(shocks_eiopa2) <- paste0("eiopa_shock_",c("bonds","stocks","total"))

#### 5.5.1.2 Stochastic model ----

for (i in 1:n_sim){
  portfolio_sim <- portfolio_value_calc(portfolio, estim[i,1:length(times)],
times,
                                        stock_estim[i]/as.vector(stock))
  if (i == 1) portfolio_sim_results <- portfolio_sim else
    portfolio_sim_results <- rbind(portfolio_sim_results,portfolio_sim)
}

results_sim_portfolio <- market_df_sim %>% filter(date == tail(dates_sim,1))
%>%
  mutate(current_bond_value = portfolio_value["bonds"],
         current_stock_value = portfolio_value["stocks"],
         bonds_duration = portfolio_value["duration"])

results_sim_portfolio$stochastic_bonds_value <-
portfolio_sim_results[,"bonds"]
```

```
results_sim_portfolio$stochastic_stocks_value <-
portfolio_sim_results[,"stocks"]

results_sim_portfolio <- results_sim_portfolio %>%
  mutate(bonds_change = stochastic_bonds_value - current_bond_value,
         stocks_change = stochastic_stocks_value - current_stock_value,
         portfolio_change = bonds_change + stocks_change)

vaR <- quantile(results_sim_portfolio$portfolio_change,c(0.005,0.995))

svg(file = "plots/histogram_portfolio1.svg", width = 10, height = 5)
par(mfrow=c(1,2), mar = c(4.5,4,3,1),oma = c(0,0,0,0))
hist(results_sim_portfolio$bonds_change/portfolio_value["total"],60,
     main = "bonds shock portfolio change", xlab = "portfolio loss")
hist(results_sim_portfolio$stocks_change/portfolio_value["total"],60,
     main = "equity shock portfolio change", xlab = "portfolio loss")
dev.off()
par(mfrow=c(1,1), mar = c(4.5,4,3,1),oma = c(0,0,0,0))

varnames <- c(expression("Shock"["Interest rate"]),
expression("Shock"["Equity"]))
u_stock_bonds <- cbind(pobs(results_sim_portfolio$bonds_change),
                       pobs(results_sim_portfolio$stocks_change))

cor(results_sim_portfolio$bonds_change,results_sim_portfolio$stocks_change)

svg(file = "plots/corel_portfolio1.svg", width = 5, height = 5)
pairs(head(u_stock_bonds,1219),
      labels = varnames, cex = 0.5, oma = c(2,2,2,2))
dev.off()

VaR_rates <- quantile(results_sim_portfolio$bonds_change,c(0.005,0.995))
VaR_stock <- quantile(results_sim_portfolio$stocks_change,c(0.005,0.995))

#### 5.5.1.3 Results ----

portfolio1 <- c(portfolio_value,shocks_eiopa,stoch_shock_bonds =
VaR_rates[1],
                stoch_shock_stocks = VaR_stock[1], stoch_shock_total =
vaR[1])

##### Waterfall diagram Shock Model down ----
range <-
min(VaR_rates[1]+VaR_stock[1],shocks_eiopa[1]+shocks_eiopa[2])/portfolio_valu
e["total"]
range <- sign(range)*ceiling(abs(range*10))/10
shocks <- c(VaR_rates[1],VaR_stock[1], vaR[1])/portfolio_value["total"]
x <- list("Interest Shock up", "Equity Shock down", "Diversification",
"Aggregated shock")
measure <- c("relative", "relative", "relative", "total")
y <- c(shocks[1], shocks[2], shocks[3]-sum(shocks[1:2]),0)
text <- as.character(abs(round(c(y[1:3],shocks[3]),3)))
signs <- c("-","","+")[sign(c(y[1:3],shocks[3]))+2]
text <- paste(signs, text, sep = "")
data <- data.frame(x=factor(x,levels=x),measure,text,y)
plot_ly(data, name = "", type = "waterfall", measure = ~measure,
```

```
        x = ~x, textposition = "outside", y= ~y, text =~text,
        connector = list(line = list(color= "rgb(63, 63, 63)"))) %>%
  layout(title = "Model shocks aggregation (Capital loss)",
         xaxis = list(title = ""),
         yaxis = list(title = "",range = list(range, 0)),
         autosize = TRUE,
         showlegend = FALSE)

##### Waterfall diagram Shock EIOPA down ----
shocks <- c(shocks_eiopa[1],shocks_eiopa[2],
shocks_eiopa[3])/portfolio_value["total"]
x <- list("Interest Shock up", "Equity Shock down", "Diversification",
"Aggregated shock")
measure <- c("relative", "relative", "relative", "total")
y <- c(shocks[1], shocks[2], shocks[3]-sum(shocks[1:2]),0)
text <- as.character(abs(round(c(y[1:3],shocks[3]),3)))
signs <- c("-","","+")[sign(c(y[1:3],shocks[3]))+2]
text <- paste(signs, text, sep = "")
data <- data.frame(x=factor(x,levels=x),measure,text,y)
plot_ly(data, name = "", type = "waterfall", measure = ~measure,
        x = ~x, textposition = "outside", y= ~y, text =~text,
        connector = list(line = list(color= "rgb(63, 63, 63)"))) %>%
  layout(title = "EIOPA shocks aggregation (Capital loss)",
         xaxis = list(title = ""),
         yaxis = list(title = "",range = list(range, 0)),
         autosize = TRUE,
         showlegend = FALSE)

##### Waterfall diagram Shock Model up ----
range <-
max(VaR_rates[2]+VaR_stock[2],shocks_eiopa2[1]+shocks_eiopa2[2])/portfolio_va
lue["total"]
shocks <- c(VaR_rates[2],VaR_stock[2], vaR[2])/portfolio_value["total"]
x <- list("Interest Shock down", "Equity Shock up", "Diversification",
"Aggregated shock")
measure <- c("relative", "relative", "relative", "total")
y <- c(shocks[1], shocks[2], shocks[3]-sum(shocks[1:2]),0)
text <- as.character(abs(round(c(y[1:3],shocks[3]),3)))
signs <- c("-","","+")[sign(c(y[1:3],shocks[3]))+2]
text <- paste(signs, text, sep = "")
data <- data.frame(x=factor(x,levels=x),measure,text,y)
plot_ly(data, name = "", type = "waterfall", measure = ~measure,
        x = ~x, textposition = "outside", y= ~y, text =~text,
        connector = list(line = list(color= "rgb(63, 63, 63)"))) %>%
  layout(title = "Model shocks aggregation (Capital increase)",
         xaxis = list(title = ""),
         yaxis = list(title = "",range = list(0, range)),
         autosize = TRUE,
         showlegend = FALSE)

##### Waterfall diagram Shock EIOPA up ----
shocks <- c(shocks_eiopa2[1],shocks_eiopa2[2],
shocks_eiopa2[3])/portfolio_value["total"]
x <- list("Interest Shock down", "Equity Shock up", "Diversification",
"Aggregated shock")
measure <- c("relative", "relative", "relative", "total")
```

```
y <- c(shocks[1], shocks[2], shocks[3]-sum(shocks[1:2]),0)
text <- as.character(abs(round(c(y[1:3],shocks[3]),3)))
signs <- c("-","","+")[sign(c(y[1:3],shocks[3]))+2]
text <- paste(signs, text, sep = "")
data <- data.frame(x=factor(x,levels=x),measure,text,y)
plot_ly(data, name = "", type = "waterfall", measure = ~measure,
        x = ~x, textposition = "outside", y= ~y, text =~text,
        connector = list(line = list(color= "rgb(63, 63, 63)"))) %>%
  layout(title = "EIOPA shocks aggregation (Capital increase)",
         xaxis = list(title = ""),
         yaxis = list(title = "",range = list(0, range)),
         autosize = TRUE,
         showlegend = FALSE)

### 5.5.2 Portfolio 2 calculation ----

portfolio <- list(bond = list(mat = 10, cupon = 0.02, qty = 1000),
                  bond = list(mat = 15, cupon = 0.04, qty = 1000),
                  bond = list(mat = 20, cupon = 0.03, qty = 1000),
                  bond = list(mat = 5, cupon = 0.01, qty = 1000),
                  stock = list(price = 500))

portfolio_value <- portfolio_value_calc(portfolio, yield[1:length(times)],
                                        times, 1, duration = TRUE)
portfolio_value <- c(portfolio_value, "total" = sum(portfolio_value[1:2]))
portfolio_eiopa_results <- portfolio_value_calc(portfolio, eiopa_up, times,
                                               eiopa_stock_down)

portfolio_eiopa_results2 <- portfolio_value_calc(portfolio, eiopa_down,
times,
                                                eiopa_stock_up)

#### 5.5.1.1 Standard formula ----

eiopa_cor <- matrix(c(1,0.5,0.5,1), nrow = 2, ncol = 2)
eiopa_cor2 <- matrix(c(1,0,0,1), nrow = 2, ncol = 2)

shocks_eiopa <- c(portfolio_eiopa_results["bonds"]-portfolio_value["bonds"],
                  portfolio_eiopa_results["stocks"]-
portfolio_value["stocks"])

shocks_eiopa["total"] <- -sqrt((-shocks_eiopa) %*% eiopa_cor %*% (-
shocks_eiopa))
names(shocks_eiopa) <- paste0("eiopa_shock_",c("bonds","stocks","total"))

shocks_eiopa2 <- c(portfolio_eiopa_results2["bonds"]-
portfolio_value["bonds"],
                   portfolio_eiopa_results2["stocks"]-
portfolio_value["stocks"])

shocks_eiopa2["total"] <- sqrt((-shocks_eiopa2) %*% eiopa_cor2 %*% (-
shocks_eiopa2))
names(shocks_eiopa2) <- paste0("eiopa_shock_",c("bonds","stocks","total"))

#### 5.5.1.2 Stochastic model ----
```

```
for (i in 1:n_sim){
  portfolio_sim <- portfolio_value_calc(portfolio, estim[i,1:length(times)],
times,
                                        stock_estim[i]/as.vector(stock))
  if (i == 1) portfolio_sim_results <- portfolio_sim else
    portfolio_sim_results <- rbind(portfolio_sim_results,portfolio_sim)
}

results_sim_portfolio <- market_df_sim %>% filter(date == tail(dates_sim,1))
%>%
  mutate(current_bond_value = portfolio_value["bonds"],
         current_stock_value = portfolio_value["stocks"],
         bonds_duration = portfolio_value["duration"])

results_sim_portfolio$stochastic_bonds_value <-
portfolio_sim_results[,"bonds"]
results_sim_portfolio$stochastic_stocks_value <-
portfolio_sim_results[,"stocks"]

results_sim_portfolio <- results_sim_portfolio %>%
  mutate(bonds_change = stochastic_bonds_value - current_bond_value,
         stocks_change = stochastic_stocks_value - current_stock_value,
         portfolio_change = bonds_change + stocks_change)

vaR <- quantile(results_sim_portfolio$portfolio_change,c(0.005,0.995))

svg(file = "plots/histogram_portfolio2.svg", width = 10, height = 5)
par(mfrow=c(1,2), mar = c(4.5,4,3,1),oma = c(0,0,0,0))
hist(results_sim_portfolio$bonds_change/portfolio_value["total"],60,
     main = "bonds shock portfolio change", xlab = "portfolio loss")
hist(results_sim_portfolio$stocks_change/portfolio_value["total"],60,
     main = "equity shock portfolio change", xlab = "portfolio loss")
dev.off()
par(mfrow=c(1,1), mar = c(4.5,4,3,1),oma = c(0,0,0,0))

varnames <- c(expression("Shock"["Interest rate"]),
expression("Shock"["Equity"]))
u_stock_bonds <- cbind(pobs(results_sim_portfolio$bonds_change),
                       pobs(results_sim_portfolio$stocks_change))

cor(results_sim_portfolio$bonds_change,results_sim_portfolio$stocks_change)

svg(file = "plots/corel_portfolio2.svg", width = 5, height = 5)
pairs(head(u_stock_bonds,1219),
      labels = varnames, cex = 0.5, oma = c(2,2,2,2))
dev.off()

VaR_rates <- quantile(results_sim_portfolio$bonds_change,c(0.005,0.995))
VaR_stock <- quantile(results_sim_portfolio$stocks_change,c(0.005,0.995))

#### 5.5.2.3 Results ----

portfolio2 <- c(portfolio_value,shocks_eiopa,stoch_shock_bonds =
VaR_rates[1],
                stoch_shock_stocks = VaR_stock[1], stoch_shock_total =
vaR[1])
```

```
##### Waterfall diagram Shock Model down ----
range <-
min(VaR_rates[1]+VaR_stock[1],shocks_eiopa[1]+shocks_eiopa[2])/portfolio_valu
e["total"]
range <- sign(range)*ceiling(abs(range*10))/10
shocks <- c(VaR_rates[1],VaR_stock[1], vaR[1])/portfolio_value["total"]
x <- list("Interest Shock up", "Equity Shock down", "Diversification",
"Aggregated shock")
measure <- c("relative", "relative", "relative", "total")
y <- c(shocks[1], shocks[2], shocks[3]-sum(shocks[1:2]),0)
text <- as.character(abs(round(c(y[1:3],shocks[3]),3)))
signs <- c("-","","+")[sign(c(y[1:3],shocks[3]))+2]
text <- paste(signs, text, sep = "")
data <- data.frame(x=factor(x,levels=x),measure,text,y)
plot_ly(data, name = "", type = "waterfall", measure = ~measure,
        x = ~x, textposition = "outside", y= ~y, text =~text,
        connector = list(line = list(color= "rgb(63, 63, 63)"))) %>%
  layout(title = "Model shocks aggregation (Capital loss)",
         xaxis = list(title = ""),
         yaxis = list(title = "",range = list(range, 0)),
         autosize = TRUE,
         showlegend = FALSE)

##### Waterfall diagram Shock EIOPA down ----
shocks <- c(shocks_eiopa[1],shocks_eiopa[2],
shocks_eiopa[3])/portfolio_value["total"]
x <- list("Interest Shock up", "Equity Shock down", "Diversification",
"Aggregated shock")
measure <- c("relative", "relative", "relative", "total")
y <- c(shocks[1], shocks[2], shocks[3]-sum(shocks[1:2]),0)
text <- as.character(abs(round(c(y[1:3],shocks[3]),3)))
signs <- c("-","","+")[sign(c(y[1:3],shocks[3]))+2]
text <- paste(signs, text, sep = "")
data <- data.frame(x=factor(x,levels=x),measure,text,y)
plot_ly(data, name = "", type = "waterfall", measure = ~measure,
        x = ~x, textposition = "outside", y= ~y, text =~text,
        connector = list(line = list(color= "rgb(63, 63, 63)"))) %>%
  layout(title = "EIOPA shocks aggregation (Capital loss)",
         xaxis = list(title = ""),
         yaxis = list(title = "",range = list(range, 0)),
         autosize = TRUE,
         showlegend = FALSE)

##### Waterfall diagram Shock Model up ----
range <-
max(VaR_rates[2]+VaR_stock[2],shocks_eiopa2[1]+shocks_eiopa2[2])/portfolio_va
lue["total"]
range <- sign(range)*ceiling(abs(range*10))/10
shocks <- c(VaR_rates[2],VaR_stock[2], vaR[2])/portfolio_value["total"]
x <- list("Interest Shock down", "Equity Shock up", "Diversification",
"Aggregated shock")
measure <- c("relative", "relative", "relative", "total")
y <- c(shocks[1], shocks[2], shocks[3]-sum(shocks[1:2]),0)
text <- as.character(abs(round(c(y[1:3],shocks[3]),3)))
signs <- c("-","","+")[sign(c(y[1:3],shocks[3]))+2]
```

```
text <- paste(signs, text, sep = "")
data <- data.frame(x=factor(x,levels=x),measure,text,y)
plot_ly(data, name = "", type = "waterfall", measure = ~measure,
        x = ~x, textposition = "outside", y= ~y, text =~text,
        connector = list(line = list(color= "rgb(63, 63, 63)"))) %>%
  layout(title = "Model shocks aggregation (Capital increase)",
         xaxis = list(title = ""),
         yaxis = list(title = "",range = list(0, range)),
         autosize = TRUE,
         showlegend = FALSE)

##### Waterfall diagram Shock EIOPA up ----
shocks <- c(shocks_eiopa2[1],shocks_eiopa2[2],
shocks_eiopa2[3])/portfolio_value["total"]
x <- list("Interest Shock down", "Equity Shock up", "Diversification",
"Aggregated shock")
measure <- c("relative", "relative", "relative", "total")
y <- c(shocks[1], shocks[2], shocks[3]-sum(shocks[1:2]),0)
text <- as.character(abs(round(c(y[1:3],shocks[3]),3)))
signs <- c("-","","+")[sign(c(y[1:3],shocks[3]))+2]
text <- paste(signs, text, sep = "")
data <- data.frame(x=factor(x,levels=x),measure,text,y)
plot_ly(data, name = "", type = "waterfall", measure = ~measure,
        x = ~x, textposition = "outside", y= ~y, text =~text,
        connector = list(line = list(color= "rgb(63, 63, 63)"))) %>%
  layout(title = "EIOPA shocks aggregation (Capital increase)",
         xaxis = list(title = ""),
         yaxis = list(title = "",range = list(0, range)),
         autosize = TRUE,
         showlegend = FALSE)

### 5.5.3 Portfolio 3 calculation ----

portfolio <- list(bond = list(mat = 15, cupon = 0.02, qty = 1000),
                  bond = list(mat = 20, cupon = 0.04, qty = 1000),
                  bond = list(mat = 25, cupon = 0.03, qty = 1000),
                  bond = list(mat = 10, cupon = 0.01, qty = 1000),
                  stock = list(price = 500))

portfolio_value <- portfolio_value_calc(portfolio, yield[1:length(times)],
                                        times, 1, duration = TRUE)
portfolio_value <- c(portfolio_value, "total" = sum(portfolio_value[1:2]))
portfolio_eiopa_results <- portfolio_value_calc(portfolio, eiopa_up, times,
                                                eiopa_stock_down)

portfolio_eiopa_results2 <- portfolio_value_calc(portfolio, eiopa_down,
times,
                                                 eiopa_stock_up)

#### 5.5.1.1 Standard formula ----

eiopa_cor <- matrix(c(1,0.5,0.5,1), nrow = 2, ncol = 2)
eiopa_cor2 <- matrix(c(1,0,0,1), nrow = 2, ncol = 2)

shocks_eiopa <- c(portfolio_eiopa_results["bonds"]-portfolio_value["bonds"],
```

```
                  portfolio_eiopa_results["stocks"]-
portfolio_value["stocks"])

shocks_eiopa["total"] <- -sqrt((-shocks_eiopa) %*% eiopa_cor %*% (-
shocks_eiopa))
names(shocks_eiopa) <- paste0("eiopa_shock_",c("bonds","stocks","total"))

shocks_eiopa2 <- c(portfolio_eiopa_results2["bonds"]-
portfolio_value["bonds"],
                   portfolio_eiopa_results2["stocks"]-
portfolio_value["stocks"])

shocks_eiopa2["total"] <- sqrt((-shocks_eiopa2) %*% eiopa_cor2 %*% (-
shocks_eiopa2))
names(shocks_eiopa2) <- paste0("eiopa_shock_",c("bonds","stocks","total"))

#### 5.5.1.2 Stochastic model ----

for (i in 1:n_sim){
  portfolio_sim <- portfolio_value_calc(portfolio, estim[i,1:length(times)],
times,
                                        stock_estim[i]/as.vector(stock))
  if (i == 1) portfolio_sim_results <- portfolio_sim else
    portfolio_sim_results <- rbind(portfolio_sim_results,portfolio_sim)
}

results_sim_portfolio <- market_df_sim %>% filter(date == tail(dates_sim,1))
%>%
  mutate(current_bond_value = portfolio_value["bonds"],
         current_stock_value = portfolio_value["stocks"],
         bonds_duration = portfolio_value["duration"])

results_sim_portfolio$stochastic_bonds_value <-
portfolio_sim_results[,"bonds"]
results_sim_portfolio$stochastic_stocks_value <-
portfolio_sim_results[,"stocks"]

results_sim_portfolio <- results_sim_portfolio %>%
  mutate(bonds_change = stochastic_bonds_value - current_bond_value,
         stocks_change = stochastic_stocks_value - current_stock_value,
         portfolio_change = bonds_change + stocks_change)

vaR <- quantile(results_sim_portfolio$portfolio_change,c(0.005,0.995))

svg(file = "plots/histogram_portfolio3.svg", width = 10, height = 5)
par(mfrow=c(1,2), mar = c(4.5,4,3,1),oma = c(0,0,0,0))
hist(results_sim_portfolio$bonds_change/portfolio_value["total"],60,
     main = "bonds shock portfolio change", xlab = "portfolio loss")
hist(results_sim_portfolio$stocks_change/portfolio_value["total"],60,
     main = "equity shock portfolio change", xlab = "portfolio loss")
dev.off()
par(mfrow=c(1,1), mar = c(4.5,4,3,1),oma = c(0,0,0,0))

varnames <- c(expression("Shock"["Interest rate"]),
expression("Shock"["Equity"]))
u_stock_bonds <- cbind(pobs(results_sim_portfolio$bonds_change),
```

```
                    pobs(results_sim_portfolio$stocks_change))

cor(results_sim_portfolio$bonds_change,results_sim_portfolio$stocks_change)

svg(file = "plots/corel_portfolio3.svg", width = 5, height = 5)
pairs(head(u_stock_bonds,1219),
      labels = varnames, cex = 0.5, oma = c(2,2,2,2))
dev.off()

VaR_rates <- quantile(results_sim_portfolio$bonds_change,c(0.005,0.995))
VaR_stock <- quantile(results_sim_portfolio$stocks_change,c(0.005,0.995))

#### 5.5.3.3 Results ----

portfolio3 <- c(portfolio_value,shocks_eiopa,stoch_shock_bonds =
VaR_rates[1],
                stoch_shock_stocks = VaR_stock[1], stoch_shock_total =
vaR[1])

##### Waterfall diagram Shock Model down ----
range <-
min(VaR_rates[1]+VaR_stock[1],shocks_eiopa[1]+shocks_eiopa[2])/portfolio_valu
e["total"]
range <- sign(range)*ceiling(abs(range*10))/10
shocks <- c(VaR_rates[1],VaR_stock[1], vaR[1])/portfolio_value["total"]
x <- list("Interest Shock up", "Equity Shock down", "Diversification",
"Aggregated shock")
measure <- c("relative", "relative", "relative", "total")
y <- c(shocks[1], shocks[2], shocks[3]-sum(shocks[1:2]),0)
text <- as.character(abs(round(c(y[1:3],shocks[3]),3)))
signs <- c("-","","+")[sign(c(y[1:3],shocks[3]))+2]
text <- paste(signs, text, sep = "")
data <- data.frame(x=factor(x,levels=x),measure,text,y)
plot_ly(data, name = "", type = "waterfall", measure = ~measure,
        x = ~x, textposition = "outside", y= ~y, text =~text,
        connector = list(line = list(color= "rgb(63, 63, 63)"))) %>%
  layout(title = "Model shocks aggregation (Capital loss)",
         xaxis = list(title = ""),
         yaxis = list(title = "",range = list(range, 0)),
         autosize = TRUE,
         showlegend = FALSE)

##### Waterfall diagram Shock EIOPA down ----
shocks <- c(shocks_eiopa[1],shocks_eiopa[2],
shocks_eiopa[3])/portfolio_value["total"]
x <- list("Interest Shock up", "Equity Shock down", "Diversification",
"Aggregated shock")
measure <- c("relative", "relative", "relative", "total")
y <- c(shocks[1], shocks[2], shocks[3]-sum(shocks[1:2]),0)
text <- as.character(abs(round(c(y[1:3],shocks[3]),3)))
signs <- c("-","","+")[sign(c(y[1:3],shocks[3]))+2]
text <- paste(signs, text, sep = "")
data <- data.frame(x=factor(x,levels=x),measure,text,y)
plot_ly(data, name = "", type = "waterfall", measure = ~measure,
        x = ~x, textposition = "outside", y= ~y, text =~text,
        connector = list(line = list(color= "rgb(63, 63, 63)"))) %>%
```

```r
  layout(title = "EIOPA shocks aggregation (Capital loss)",
         xaxis = list(title = ""),
         yaxis = list(title = "",range = list(range, 0)),
         autosize = TRUE,
         showlegend = FALSE)

##### Waterfall diagram Shock Model up ----
range <-
max(VaR_rates[2]+VaR_stock[2],shocks_eiopa2[1]+shocks_eiopa2[2])/portfolio_va
lue["total"]
range <- sign(range)*ceiling(abs(range*10))/10
shocks <- c(VaR_rates[2],VaR_stock[2], vaR[2])/portfolio_value["total"]
x <- list("Interest Shock down", "Equity Shock up", "Diversification",
"Aggregated shock")
measure <- c("relative", "relative", "relative", "total")
y <- c(shocks[1], shocks[2], shocks[3]-sum(shocks[1:2]),0)
text <- as.character(abs(round(c(y[1:3],shocks[3]),3)))
signs <- c("-","","+")[sign(c(y[1:3],shocks[3]))+2]
text <- paste(signs, text, sep = "")
data <- data.frame(x=factor(x,levels=x),measure,text,y)
plot_ly(data, name = "", type = "waterfall", measure = ~measure,
        x = ~x, textposition = "outside", y= ~y, text =~text,
        connector = list(line = list(color= "rgb(63, 63, 63)"))) %>%
  layout(title = "Model shocks aggregation (Capital increase)",
         xaxis = list(title = ""),
         yaxis = list(title = "",range = list(0, range)),
         autosize = TRUE,
         showlegend = FALSE)

##### Waterfall diagram Shock EIOPA up ----
shocks <- c(shocks_eiopa2[1],shocks_eiopa2[2],
shocks_eiopa2[3])/portfolio_value["total"]
x <- list("Interest Shock down", "Equity Shock up", "Diversification",
"Aggregated shock")
measure <- c("relative", "relative", "relative", "total")
y <- c(shocks[1], shocks[2], shocks[3]-sum(shocks[1:2]),0)
text <- as.character(abs(round(c(y[1:3],shocks[3]),3)))
signs <- c("-","","+")[sign(c(y[1:3],shocks[3]))+2]
text <- paste(signs, text, sep = "")
data <- data.frame(x=factor(x,levels=x),measure,text,y)
plot_ly(data, name = "", type = "waterfall", measure = ~measure,
        x = ~x, textposition = "outside", y= ~y, text =~text,
        connector = list(line = list(color= "rgb(63, 63, 63)"))) %>%
  layout(title = "EIOPA shocks aggregation (Capital increase)",
         xaxis = list(title = ""),
         yaxis = list(title = "",range = list(0, range)),
         autosize = TRUE,
         showlegend = FALSE)

## 5.5.4 Summary ----

portfolios <- rbind(portfolio1, portfolio2, portfolio3)

write.xlsx(as.data.frame(rbind(portfolios, portfolios_tstudent)),"data.xlsx")
portfolios_tstudent
```

```
cols <- rainbow(5)
for (i in 1:3){
  barplot(abs(portfolios[i,c(1:2,4:10)]), col = cols[c(2,2,1,3,3,1,4,4,1)],
          main = paste0("bond duration = ",
round(portfolios[i,"duration"],2)))
}

# Some plots ----
swaps_df3 %>% mutate(coding = fct_reorder(name, sort(time))) %>%
  ggplot() + geom_line(mapping = aes(x = date, y = rate, color = coding))

swaps_df4 %>% ggplot() + geom_line(mapping = aes(x = date, y = TREUR1Y))

bond_all_df %>% filter(date == init_date) %>% ggplot() +
  geom_line(mapping = aes(x = maturity, y = rate))


bond_all_df$date


dates <- rev((bond_all_df %>% filter(date > init_date, maturity == 1))$date)

for (i in dates){
  sum(bond_all_df$date == i)
}

for(i in dates){
  bond_all_df %>%
    ggplot(aes(times, rate)) +
    geom_line() +
    scale_x_continuous() +
    theme_bw() +
    labs(title = paste0("Year: ",2020), x = 'rate', y = 'maturity')
}
```

Functions

```
mutate <- dplyr::mutate
count <- dplyr::count
filter <- dplyr::filter
group_by <- dplyr::group_by
ungroup <- dplyr::ungroup
wday <- lubridate::wday

diff_narm <- function(ts, lag = 1){na.omit(diff(ts, lag = lag))}

unscale <- function(matrix, scale, center){
  t(apply(matrix,1,function(matrix) matrix * scale + center ))
}

ARIMA_GARCH_test <- function(param, ts, filename, type.model, max_order = 3,
                             start = 1, save_every = 1000){

  n <- nrow(param)

  if (start != 1){
    results <- as.matrix(read.csv(filename))
  } else{
    results <- matrix(NA, nrow = n, ncol = 7)
    colnames(results) <- c("N","max_p_value", "akaike",
                           "bic","shibata", "hannan_quinn","max_estim")
  }

  # do not show warnings
  defaultW <- getOption("warn")
  options(warn = -1)

  for (i in start:n){
    par <- param[i,]
    par_fixed <- par[,!names(par) %in% c("distr", "mean")]
    par_fixed <- par_fixed[, as.vector(par_fixed[1,] != 1)]
    fixed.pars = as.list(par_fixed)
    mean <- par[,"mean"][[1]]
    distr <- par[,"distr"][[1]]

    Garch <- ugarchspec(variance.model = list(model = type.model,
                                              garchOrder = c(max_order,
                                                             max_order,
                                                             max_order)),
                        mean.model = list(armaOrder = c(max_order,
max_order),
                                          include.mean = mean),
                        distribution.model = distr, fixed.pars = fixed.pars)

    model <- tryCatch(withTimeout(ugarchfit(spec = Garch, data = ts),
                                  timeout = 30, onTimeout = c("error")),
                      error = function(e) 1)

    if (typeof(model)!="double") {
      # omega is not checked since it must be there for the model to converge
```

```
      model_info <- model@fit$matcoef[,c("Pr(>|t|)"," Estimate")]
      model_info <- model_info[rownames(model_info)!="omega",]
      p.value.max <- max(model_info[,c("Pr(>|t|)")],0, na.rm = TRUE)
      model_info <- model_info[rownames(model_info)!="skew",]
      model_info <- model_info[rownames(model_info)!="shape",]
      max_estim <- max(abs(c(model_info[," Estimate"],0)))

      infocriteria <- tryCatch(infocriteria(model), error = function(e) 1)

      if (length(infocriteria) == 1){
        akaike <- Inf
        bic <- Inf
        shibata <- Inf
        hannan_quinn <- Inf
      } else {
        akaike <- infocriteria[1]
        bic <- infocriteria[2]
        shibata <- infocriteria[3]
        hannan_quinn <- infocriteria[4]
      }
      results[i,] <- c(i, p.value.max, akaike, bic, shibata, hannan_quinn,
                       max_estim)
    }
    if (i%%save_every == 0) {
      write.csv(as.data.frame(results),filename, row.names = FALSE)
      print(paste0(i, " out of ", n, " - progress ", round(i/n*100, 2), "%"))
      print(Sys.time())
    }
  }
  # show warnings again
  options(warn = defaultW)

  write.csv(as.data.frame(results),filename, row.names = FALSE)

  as.data.frame(results)
}

testCopula <- function(u, method = "mult", opt.meth = "L-BFGS-B"){
  n <- length(u[,1])
  # Normal copula
  normc <- fitCopula(copula = normalCopula(), data = u, method = "irho")
  resultnorm <- gofCopula(copula = normalCopula(param = normc@estimate), x =
u,
                          simulation = method)
  BICnorm <- 1*log(n) - 2*normc@loglik
  AICcnorm <- 2*1 - 2*normc@loglik + (2*1^2+2*1)/(n-1-1)
  # tStudent copula
  tc <- fitCopula(copula = tCopula(), data = u, optim.method = opt.meth)
  # if error change optim.method as opt.meth = "Nelder-Mead"
  tc <- fitCopula(copula = tCopula(df = round(tc@estimate[2]), df.fixed =
TRUE),
                  data = u)
  # library only implemented for df = integer
  df = tc@copula@parameters[[2]]
  resultt <- gofCopula(copula = tCopula(param = tc@estimate, df = df,
                                        df.fixed = TRUE),
```

```
                          x = u, simulation = method)
  BICt<- 2*log(n) - 2*tc@loglik
  AICct <- 2*2 - 2*tc@loglik + (2*2^2+2*2)/(n-2-1)
  # AMH copula
  amhc <- fitCopula(copula = amhCopula(), data = u)
  resultamh <- gofCopula(copula = amhCopula(param = amhc@estimate), x = u,
                          simulation = method)
  BICamh <- 1*log(n) - 2*amhc@loglik
  AICcamh <- 2*1 - 2*amhc@loglik + (2*1^2+2*1)/(n-1-1)
  # Clayton copula
  cc <- fitCopula(copula = claytonCopula(), data = u)
  resultc <- gofCopula(copula = claytonCopula(param = cc@estimate), x = u,
                          simulation = method)
  BICc <- 1*log(n) - 2*cc@loglik
  AICcc <- 2*1 - 2*cc@loglik + (2*1^2+2*1)/(n-1-1)
  # Frank copula
  fc <- fitCopula(copula = frankCopula(), data = u)
  resultf <- gofCopula(copula = frankCopula(param = fc@estimate), x = u,
                          simulation = method)
  BICf <- 1*log(n) - 2*fc@loglik
  AICcf <- 2*1 - 2*fc@loglik + (2*1^2+2*1)/(n-1-1)
  # Gumbel-Hougaard copula
  ghc <- fitCopula(copula = gumbelCopula(), data = u)
  resultgh <- gofCopula(copula = gumbelCopula(param = ghc@estimate), x = u,
                          simulation = method)
  BICgh <- 1*log(n) - 2*ghc@loglik
  AICcgh <- 2*1 - 2*ghc@loglik + (2*1^2+2*1)/(n-1-1)
  # Joe copula
  jc <- fitCopula(copula = joeCopula(), data = u)
  resultj <- gofCopula(copula = joeCopula(param = jc@estimate), x = u,
                          simulation = method)
  BICj <- 1*log(n) - 2*jc@loglik
  AICcj <- 2*1 - 2*jc@loglik + (2*1^2+2*1)/(n-1-1)
  data.frame("copula" = c("Normal",paste0("t-Student"," (df=",df,")"),"AMH",
                        "Clayton", "Frank","Gumbel-Hougaard","Joe"),
             "log-likelihood" =
c(normc@loglik,tc@loglik,amhc@loglik,cc@loglik,
                                  fc@loglik,ghc@loglik,jc@loglik),
             "parameter" =
c(normc@estimate,tc@estimate,amhc@estimate,cc@estimate,
                             fc@estimate,ghc@estimate,jc@estimate),
             "s.e." =
sqrt(c(normc@var.est,tc@var.est,amhc@var.est,cc@var.est,
                             fc@var.est,ghc@var.est,jc@var.est)),
             "gof pvalue" = c(resultnorm$p.value,resultt$p.value,

resultamh$p.value,resultc$p.value,resultf$p.value,
                              resultgh$p.value,resultj$p.value),
             "BIC" = c(BICnorm,BICt,BICamh,BICc,BICf,BICgh,BICj),
             "AICc" = c(AICcnorm,AICct,AICcamh,AICcc,AICcf,AICcgh,AICcj))
}

# a function to fit a GARCH model based in the notation of parameters
proposed

fit_GARCH <- function(ts, par, type.model, max_order = 3){
```

```
  par_fixed <- par[,!names(par) %in% c("distr", "mean")]
  par_fixed <- par_fixed[, as.vector(par_fixed[1,] != 1)]
  fixed.pars = as.list(par_fixed)
  mean <- par[,"mean"][[1]]
  distr <- par[,"distr"][[1]]

  sGarch <- ugarchspec(variance.model = list(model = type.model,
                                             garchOrder =
c(max_order,max_order,max_order)),
                       mean.model = list(armaOrder = c(max_order, max_order),
                                         include.mean = mean),
                       distribution.model = distr, fixed.pars = fixed.pars)
  ugarchfit(spec = sGarch, data = dtsPC)
}


fit_GARCH_spec <- function(par, type.model, max_order = 3){
  par_fixed <- par[,!names(par) %in% c("distr", "mean")]
  par_fixed <- par_fixed[, as.vector(par_fixed[1,] != 1)]
  fixed.pars = as.list(par_fixed)
  mean <- par[,"mean"][[1]]
  distr <- par[,"distr"][[1]]
  ugarchspec(variance.model = list(model = type.model,
                                   garchOrder =
c(max_order,max_order,max_order)),
             mean.model = list(armaOrder = c(max_order, max_order),
                               include.mean = mean),
             distribution.model = distr, fixed.pars = fixed.pars)
}

# convert the fitted model to a spec

fit_to_spec <- function(model){
  model.type = model@model$modeldesc$vmodel
  fixed.pars <-
as.list(model@model$pars[!is.na(model@model$pars[,"LB"]),"Level"])
  if (!any(names(fixed.pars) == c("gamma1"))){
    fixed.pars$gamma1 <- 0
    fixed.pars$gamma2 <- 0
    fixed.pars$gamma3 <- 0
  }
  maxOrder <- model@model$maxOrder
  distr <- model@model$modeldesc$distribution
  Garch <- ugarchspec(variance.model = list(model = type.model,
                                            garchOrder = c(maxOrder,
                                                           maxOrder,
                                                           maxOrder)),
                      mean.model = list(armaOrder = c(maxOrder, maxOrder)),
                      distribution.model = distr, fixed.pars = fixed.pars)
  Garch
}

# simulate paths providing the residuals as uniform values (p)

sim_ugarch <- function(fit, u){
  # u = random uniform (0.0 to 1.0) vector of residuals in order
```

```
  n <- length(u)
  newdates <-tail(fit@model$modeldata$index,1) + 7*seq(1,n)
  coef <- fit@fit$coef
  distr <- fit@model$modeldesc$distribution
  if (is.na(coef["skew"])) skew <- 1 else skew <- coef["skew"]
  if (is.na(coef["shape"])) shape <- 5 else shape <- coef["shape"]
  sres <- matrix(data = qdist(distribution = distr, p = u, shape = shape,
                              skew = skew),
                 nrow = n, ncol = 1)
  ts <- ugarchsim(fit, n.sim = n, m.sim = 1, custom.dist =
list(name="custom",
                                                                   distfit =
sres))
  xts(ts@simulation$seriesSim, order.by = newdates)
}

portfolio_value_calc <- function(portfolio, shock_swaps, times, shock_stock,
                                 duration = FALSE){
  # portfolio: list with bond and stock info
  # rates: swap rates in percentage
  # times: the times corresponding with the rates
  # shock_stock: the final value as fraction
  # duration: if duration is wanted as output
  rates <- approx(times, shock_swaps, 1:30)
  value_bond <- 0
  value_stock <- 0
  dur <- 0
  for (i in 1:length(portfolio)) {
    if (names(portfolio[i])=="bond"){
      bond <- portfolio[[i]]
      cashflow <- c(rep(bond$cupon, bond$mat - 1), 1 + bond$cupon)
      PV_rates <- rates$y[1:bond$mat]
      PV_cashflows <- cashflow/(1+PV_rates/100)^(1:bond$mat)
      value_bond <- value_bond + sum(PV_cashflows)*bond$qty
      dur <- dur + sum(PV_cashflows*1:bond$mat)*bond$qty
    } else value_stock <- value_stock + portfolio[[i]]$price*shock_stock
  }
  dur <- dur/value_bond
  if (duration == TRUE) {
    c("bonds" = value_bond, "stocks" = value_stock, "duration" = dur)
  } else c("bonds" = value_bond, "stocks" = value_stock)
}
```