Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Scenario analysis and stress-testing with expert, predictive and risk-correcting views.

Abdelaziz Belqadhi

Under the supervision of
Prof. Didier Sornette and Dr. Peter Cauwels

MASTER THESIS - 2014
Spec. MSc. Quantitative Finance

Abstract

The goal of this master thesis is to implement scenario analysis tools addressing various distributional properties of financial risk factors with a powerful and flexible framework that works with a limited set of assumptions. A few applications, which we cover throughout this thesis, come to mind: integrate multiple conflicting outlooks in a sound way, stress-test statistical measures, correct risk assessments with quantitative models or embed predictive trends with data mining. Indeed, we improved value-at-risk estimations at several levels with views from different models. In another case study, we implemented and assessed a data mining algorithm for crash forecasts in order to modify the risk model's features with that information. We explore Attilio Meucci's perspective that formulates the general problem of embedding statistical views as a linearly constrained convex program. We detail the precise modus operandi on how we act on specific views, which allows us to write the form that the constraints take. We especially document and test, for risk drivers in equity markets, the numerical procedure that modifies a model's statistics or risk measures. For these quantities, confidence intervals are computed in order to take into account the estimation risk and increase robustness, crucial in asset allocation. We describe, in a logical order, a coherent process to perform scenario analysis. Indeed, we work our way from a univariate specification and benefit from its foundations to deal with the multivariate case. As an application in the multivariate case, we performed a robust portfolio optimization with views on expectation vector, correlation and we showed that returns are less volatile and more consistent. We then discuss the interactions between both settings and the construction variety of multivariate representations with the use of copula separation and combination algorithms. In fact, we developed an algorithm that allows us to process univariate views when we start with a multivariate specification, and we also found that Student copulas have the best goodness of fit for equity returns, even when we put them up against canonical vine copulas. Finally, we implemented Meucci's liquidity-integrated extension that includes trading effects. It enables us to consider scenarios with additional features such as various execution horizons, liquidation policies, and we numerically computed the liquidity score as a measure of liquidity risk.

Contents

Acknowledgements

I am very grateful to professor Farkas for enabling the thesis under this form, and Prof. Sornette to allow me working under his guidance for my master thesis as well as giving me a great deal of flexibility in determining my research plan. I would also like to thank Dr. Peter Cauwels, who always kept an attentive ear and gave me, along with professor Sornette, valuable feedback during our meetings. Both were very available and quickly helped me solve my issues. In the chair of Entrepreneurial Risks, I had very interesting discussions with Zalán Forró and Diego Ardila Alvarez, and I thank them for their time. I was really amazed by the entrepreneurial spirit in this group, and I wish you all the best in your future activities.

I dedicate this thesis to my wonderful parents Jamal Eddine and Nadia, to my beautiful sister Boutaina and my whole Moroccan family, who always believe in me and support me through thick and thin. Allah ikhellikoum lili bessaha ou elaafia inchallah. This endeavour would also have been less amazing without the hilarious moments I shared with my favorite quantitative finance mates in Zurich: Pavel Riabouchkine, Kevin Soobratty and Antoine Lyson. We always discussed about the real things and to them I say: salutations à mes flingueurs de Gwada ! You guys are starting to make your place in this business, so best of luck. Special big-ups to our successors, who in time became very good friends: Wail El Allali, Tom Noppe, Laurent Oberholzer, Ryan Kurniawan and Gerard Foster. Un abrazo fuerte to my Lausanne mates and hermanos: Cristian Sacon, Mauricio Alvarez and Matthieu Stigler. May our friendship continue to bloom even further and dance our hearts out until we can't stand anymore: on compte dessus ? My old friends Ludovic Peter and Lauro Sexto made my WG-life sweeter, and I am happy we still keep in touch. Besos a mis salseras preferidas Marion Mazacz and Christelle Brulhart, my adorable partners who made my Zurich nightlife much more awesome. Last but not least, I strongly hug Adrien Lücker and Arnaud Monnard, who permanently resided with me at ETH and kept me alive and kicking. Our A-A-A group had a AAA rating of comradeship and I can't imagine living in this city without them. Puissent les réunions au sommet continuer !

Introduction

Scenario analysis is a technique that quantifies the effects of diverse economic events or risk factor states on a financial portfolio. Its counterpart, stress-testing, exposes the portfolio to adverse market movements and has a stronger focus on the downside and what can go wrong. These activities trace their origins back to the RiskMetrics methodology and value at risk (VaR) models used since 1993 with historical price movements to simulate loss amounts on a portfolio level. However, qualitative event analyses to mitigate risks and prepare contingencies were already in place since decades, this approach was pioneered by the oil company Shell for corporate scenario planning and then spilled over to financial institutions. Scenario analysis is used in portfolio management to embed subjective views arising from an analyst's research or a portfolio manager's outlook on specific asset developments. Stress-testing is used in risk management to identify portfolio weaknesses and determine if the capital is sufficient to withstand losses. Traditionally, scenario analysis has been used with the very popular Black-Litterman model, which considers a normal market in equilibrium described by returns, and where views are on the expectations. However, the market is not necessarily normal or in equilibrium, and we would like to incorporate views on risk factors more generally and not only returns. Moreover, views on expectations are very restrictive, since risk factors are often represented by a simulated distribution, and we therefore want to incorporate views on every feature of the distribution. Regarding valuation, if an asset is priced by a non-linear function of the risk factors, we want to easily explore the impact on an asset or a portfolio when we distort the risk factors and subject them to specific scenarios. The framework we adopt here is a method introduced by Meucci and inspired by well-known concepts from information theory. We detail how we will process views on common statistics and implement them for equities on Matlab. We also explore a spectrum of applications in risk estimation, stress-testing and asset allocation. The main idea of this scenario analysis method is that, starting from a representation with realizations of the risk factors and associated prior probabilities, our views can be incorporated as optimization constraints that keep the realizations untouched but modify the probability masses.

We start with the univariate case where the simulated realizations are given by a vector and we process views on a risk factor's: expectation, variance, quantiles for which the Value-At-Risk is a special case, and expected shortfall. We validate the model by adjusting our simulation procedure with a step including the generation of a robust, deterministic grid to guarantee a match between the analytical solution and the numerical one. Once this is done, any subjective view on these features can be given by the analyst or the asset manager. We do not make any assumption on the prior risk model and do not need it to be in equilibrium, so we can have views on prices or complex derivatives for example. After taking care of expert views, we move on to risk-correcting views

with a case study on Value-At-Risk assessment and improvement. Views can come from human beings in a manual way, but they can also come from various models in order to combine their strengths in different areas. We backtest the value-at-risk and select the view given by the model who performed the best regarding the VaR estimation for each confidence level. The last type of view we cover in the univariate case is the predictive kind, where a model's properties can be enhanced by other models or data mining outputs. The detection of a crash zone would lead to a view that alters the expected returns of an index for example and provide an early-warning signal. We develop a data mining method for crash detection that uses the Johansen-Ledoit-Sornette fitted parameters as inputs, along with other features, and discuss the results.

We then move on to the multivariate case, where the joint realizations are now represented by a matrix, and the dependence structure comes into play. Views can apply to the linear dependence, as evidenced by the covariance or correlation, but also a nonlinear dependence structure, represented by the copula. However, we use another algorithm to incorporate views on a copula, and it will allow us to combine any copula to individual risk factor dynamics but also separate the copula from the marginals if we already have a prior multivariate specification. We looked at the tails of the distribution in the univariate case, now we will also explore the tail-dependence, i.e joint extreme realizations. We visualize, evaluate and stress-test bivariate tail-dependences for various dependence structures. Model selection criterias will help us classify the dependence structure and choose the best ones. We also revisit predictive views with an application to asset allocation, where our prior views are sample-based. Sample estimates are often known to be unstable and yield a high volatility, we can get very large positive but also negative returns. We implement a robust procedure to update the sample estimates of expected returns and covariance, which are then used as views on the model's prior expectation and covariance. We compute the ex-ante and realized returns of the portfolio in both cases. Finally, we integrate the market model for equities with a liquidity adjustment, enabling a better scenario description since we take into account the trading impact. Since the liquidity integration retains the scenario-probability representation, we can hence leverage all the previous tools and moreover study the effect of different execution horizons, liquidity diversifications or liquidity schedule on the portfolio. We conclude the master thesis with an overview of our results and what the next steps are.

Chapter 1

Univariate scenario analysis

We present in this chapter the tools needed to perform scenario analysis in a one-dimensional setting, if we consider a stock for example. Considering each risk factor of a system or a portfolio individually is the starting point of a structured framework for scenario analysis or stress-testing. We use a simulation-based setting and combine several innovations and perspectives in a more complete, coherent and powerful manner to perform risk analysis. This process can be done in a subjective manner, with a human intervention determining what statistical specifications the scenarios should have. This is useful for example if an analyst researched a company and wants to change the risk model for its stock price according to his financial analysis. In stress-testing, a risk manager can determine himself the stress values and stress types. We will also discuss how to incorporate views that arise from different models, and this will be useful in correcting risk model weaknesses and enhance their accuracy. We can combine models in different areas of the distribution, as we will see with tail measures for example. A model can also be predictive and in this case we want to incorporate the output as a model view that doesn't correct the risk model but that gives an investment value. Data mining is a tool that allows to do this, and we will therefore develop and assess a machine learning technique for crash and rebound detection.

1.1 Fully flexible views

Fully flexible probabilities, named by Meucci in [Meu08], are a way to represent a distribution with scenarios and probabilities attached to them, and fully flexible views are the alteration of the statistical properties of this representation. Statistical properties can be the standard deviation or the quantiles for example, and we will write down how we can constrain them to take a certain value, which requires solving an optimization problem. We need to correct and validate this framework with the gaussian distribution in which we know the analytical solution of the posterior. With such a representation, we can generate confidence intervals for each rich statistic, and specify the confidence we have in each view.

1.1.1 Entropy pooling

Before presenting the entropy pooling, we have to discuss how we are going to obtain the market density at the investment horizon. The steps to model and manage the P&L distribution are taken from [Meu11b]. The first step is the quest of invariance, we are looking for patterns that are independent and identically distributed over time, we extract these invariants from the risk drivers that we identify. The set of risk drivers totally determines the price of the securities at any time t. For stocks, the suitable risk driver is $Y_t \equiv \ln(S_t)$ where S_t is the stock price. We take X as the distribution of the profit and loss of the stock S_t. Observe that Y_t is not independent and identically distributed, however if we take $\varepsilon_{t\to t+1} = Y_{t+1} - Y_t$, we can detect with two simple graphical tests if they are good candidates. We look at the time series of $x_{t+1} \equiv \varepsilon_{t\to t+1}$, and split it in two time series, denoting by $\tilde{t}$ the starting time and $\big[.\big]$ the integer part:

$$\begin{aligned} x_t \quad , \quad & t = \tilde{t},\, \tilde{t}+\tau,\ldots\ \tilde{t} + \left[\frac{t-\tilde{t}}{2\tilde{t}}\right] \\ x_t \quad , \quad & t = \left(\left[\frac{t-\tilde{t}}{2\tilde{t}}\right] + 1\right)\tau,\ldots\ t. \end{aligned} \tag{1.1.1}$$

We examine the respective histograms and they should look very similar if we have the same distribution. Then we examine the scatter-plot of the time series until the current time against its lagged values, and independance tells us we should have symmetry with respect to both axes while identical distribution means the scatter plot must bear the form of a circular cloud.

The second step is the estimation of the invariant distribution. Sometimes we have more than one invariant for a risk driver or many risk drivers with associated invariants, and these invariants are correlated among each other, so we need to estimate the multivariate joint distribution, and we will use the copula-marginal algorithm, presented afterwards for the risk driver distributions at the investment horizon, after a univariate estimation for the sake of calculability. Estimation risk has to be addressed because we do not know the true distribution, or as Donald Rumsfeld puts it, "the unknown unknowns", so we compute confidence intervals for each risk statistic.

The third step is the projection of the risk drivers distribution to the investment horizon, for that we need to go back to the connection and dynamics between the invariants and the risk drivers. We thus obtain the distribution of the risk driver $Y_{t+\tau}$, and then use the pricing function, along with the information i_t at time, to finally obtain the price at the investment horizon:

$$P_{t+\tau} = p\big(Y_{t+\tau},\, i_t\big). \tag{1.1.2}$$

The distribution of the profit and loss is $X \equiv P_{t+\tau} - P_t$. We usually perform entropy pooling for the risk invariants, such as the stock log-returns, or the P&L distribution. For the stock, since $Y_t \equiv \ln(S_t)$, the pricing function is $p(y) = e^y$.

The P&L approximation of order 1 of the stock reads:

$$\Pi_{t\to t+\tau} \approx S_t\big(\ln(S_{t+\tau}) - \ln(S_t)\big) \tag{1.1.3}$$

We present here the entropy pooling method, first introduced in [Meu08], but slightly differently by taking a univariate context instead of a multivariate one. The reason, as we'll see later, is because the processing of extreme views is applied to univariate distributions and we have fast and accurate algorithms to generate the scenarios in this case; moreover, it allows more combination possibilities to glue flexible copulas to the univariate risk factors in order to obtain a joint distribution.

Suppose that we have an arbitrary market model, also called base-case distribution or "prior" in our setting: it can be stationary or not, parametric or nonparametric, fat-tailed or Gaussian. This market model can therefore be returns of any asset type, security prices, profit and losses, but in general it represents risk factors. We would like to stress-test the market model or incorporate subjective views to measure the consequences of these views on the distribution and hence on the pricing of assets or portfolios dependent on these risk factors. The output is the updated or stressed distribution which we also call "posterior", it incorporates the views by imposing the least distortion on the original distribution.

Let us suppose we have a random variable for one risk factor X, $\mathcal{I}_t$ represents the information at time t and τ the time length to the investment horizon. X is actually the market distribution at the investment horizon, so at time $t+\tau$, and we represent it by its probability density function $X \sim f_X$.

The entropy pooling method generalizes the Black-Litterman model which assumes that the market applies to normal returns in equilibrium, that views are portfolios applying on expectations only and that the optimization framework is mean-variance. This method removes all these barriers.

We will consider nonlinear functions of the market, which we call $V \equiv g(X) \sim f_V$, and g is a K-dimensional random variable. This means that as a special case we can express statements on prices. If X represents the stock log-returns of IBM for example, then we can express views on a nonlinear function of X that represents the pricing function. Views that users will define will contradict in some way g and thus we will have $V \sim \tilde{f}_V \neq f_V$. We can express views on expectations, medians, volatilities, quantiles by solving the same mathematical problem, but for views on conditional value-at-risk we solve a slightly different, recursive problem.

To compute the posterior distribution, let us define the relative entropy between two distributions f and g, also called Kullback-Leibler divergence:

$$\mathcal{E}(f,g) = \int f(x)\big[\ln f(x) - \ln g(x)\big]dx. \tag{1.1.4}$$

The relative entropy is zero if $f = g$, and increases the more different they are. We will thus minimize the relative entropy to get the updated distribution:

$$\tilde{f}_X = \underset{f\in\mathbb{V}}{\operatorname{argmin}}\,\mathcal{E}(f, f_X) \tag{1.1.5}$$

where V is the set of all view-consistent distributions.

Finally, we present multiple opinions. Suppose we have M collaborators who specify their views with each collaborator having a confidence level $c_m \in \big[0,\,1\big]$. We take the

average weighted by the different confidence levels for each collaborator to get the pooled posterior distribution:

$$\tilde{f}_X^{\mathbf{c}} = \sum_{m=1}^{M} c_m f_X^{(m)} \tag{1.1.6}$$

where $f_X^{(m)}$ are the posterior distributions for the m-th collaborator.

The computational cost of the algorithm is low because the convex program to be solved has a number of variables equal to the number of views, and we don't need to reprice since the scenarios are the same but only the probabilities have changed.

We have 1 risk factor, and we will generate J scenarios, where J is very large. The market model X is thus represented by an $J \times 1$ vector $\mathcal{X}$ of simulations. The different columns will display the scenario probabilities for each risk factor, so the n-th column is a $J \times 1$ vector. The scenario probabilities are also of dimension $J \times 1$. The K views are thus a $J \times K$ matrix, with the k-th view of the j-th scenario being:

$$\nu_{j,k} = g_k\big(X_j\big). \tag{1.1.7}$$

Now we don't simulate new scenarios but we use the same J scenarios with different probability masses $\tilde{\mathbf{p}}$. General views can be written as linear constraints on the updated probabilities:

$$\underline{\mathbf{a}} \leq \mathbf{A}\tilde{\mathbf{p}} \leq \overline{\mathbf{a}} \tag{1.1.8}$$

where $\mathbf{A}$, $\underline{\mathbf{a}}$ and $\overline{\mathbf{a}}$ are expressions of $\mathcal{X}$. The relative entropy is discretized according to the number of scenarios:

$$\mathcal{E}(\tilde{\mathbf{p}}, \mathbf{p}) = \sum_{j=1}^{J} \tilde{\mathbf{p}}_j \big[\ln(\tilde{\mathbf{p}}_j) - \ln(\mathbf{p}_j)\big] \tag{1.1.9}$$

The posterior distribution can be rewritten as:

$$\tilde{\mathbf{p}} = \underset{\underline{\mathbf{a}} \leq \mathbf{A}f \leq \overline{\mathbf{a}}}{\operatorname{argmin}} \mathcal{E}(f, \mathbf{p}) \tag{1.1.10}$$

We solve the entropy minimization problem by taking the Lagrangian, setting it equal to 0 to get the dual Lagrangian which we prefer to maximize because the optimization acts on the number of views, and then we obtain $\tilde{\mathbf{p}}$ by a suitable transformation. If we define the pairs ($\mathbf{L}$, $\mathbf{l}$) and ($\mathbf{H}$, $\mathbf{h}$) to respectively contain the inequality and equality constraints, the problem reads:

$$\tilde{\mathbf{p}} = \underset{\substack{\text{s.t.}\, \mathbf{L}f \leq \mathbf{l} \\ \mathbf{H}f = \mathbf{h}}}{\operatorname{argmin}} \left\{ \sum_{j=1}^{J} \tilde{f}_j \big[\ln(f_j) - \ln(\mathbf{p}_j)\big] \right\} \tag{1.1.11}$$

The Lagrangian is:

$$\mathcal{L}(\mathbf{f},\, \boldsymbol{\lambda},\, \boldsymbol{\nu}) = \boldsymbol{f}'\Big(\ln(\mathbf{f}) - \ln(\mathbf{p})\Big) + \boldsymbol{\lambda}'\Big(\mathbf{L}\mathbf{f} - \mathbf{l}\Big) + \boldsymbol{\nu}'\Big(\mathbf{H}\mathbf{f} - \mathbf{h}\Big). \tag{1.1.12}$$

The solution for the first order condition is:

$$\mathbf{f}(\boldsymbol{\lambda},\, \boldsymbol{\nu}) = e^{\ln(\mathbf{p})-\mathbf{1}-\boldsymbol{L}'\boldsymbol{\lambda}-\boldsymbol{H}'\boldsymbol{\nu}}. \tag{1.1.13}$$

Taking the Lagrange dual function $\mathcal{G}(\boldsymbol{\lambda},\, \boldsymbol{\nu}) = \mathcal{L}\Big(\mathbf{f}(\boldsymbol{\lambda},\, \boldsymbol{\nu}),\, \boldsymbol{\lambda},\, \boldsymbol{\nu}\Big)$ and maximizing it yields

$$(\boldsymbol{\lambda}^*,\, \boldsymbol{\nu}^*) = \underset{\boldsymbol{\lambda}\geq\mathbf{0},\, \boldsymbol{\nu}}{\operatorname{argmax}}\, \mathcal{G}(\boldsymbol{\lambda},\, \boldsymbol{\nu}). \tag{1.1.14}$$

The posterior probability vector is finally:

$$\tilde{\mathbf{p}} = \mathbf{f}(\boldsymbol{\lambda}^*,\, \boldsymbol{\nu}^*). \tag{1.1.15}$$

We have seen earlier that the type of views will determine the form of the matrix $\mathbf{A}$, and we will specify the form of this matrix every time we encounter a new view. The entropy pooling function was written in Matlab with the help of the fmincon function, since we have a constrained convex optimization problem. Obviously the parameters are the matrix $\mathbf{A}$, the value of the view represented in a vector $\mathbf{b}$, and finally the prior probability vector $\mathbf{p}$. We have validated the results and improved our program with Meucci's code, especially since he sped up the code by providing the gradient and the hessian as options. All of Meucci's codes can be found in the Matlab Central file exchange under "Attilio Meucci", so all further references to his codes are located there.

1.1.2 Expectation and variance stress-testing : model validation

We now write the method for processing views on the expectation and the variance of any distribution represented by the scenarios $\mathcal{X}$ and a prior distribution $\mathbf{p}$. We have gone further than Meucci and programmed a function which determines the matrix form $\mathbf{A}$ in this case and asks the user for his views on expectation and variance. This is actually the case for all types of views we will encounter, Meucci provided the general code but the specific constraints were absent, and this was an extension we developed in Matlab.

Expectation and variance views

- Recognize that the expectation in the scenario-probability representation is written as:

$$\sum_{j=1}^{J} p_j \mathcal{X}_j = \hat{m}$$

- The variance in the scenario-probability representation is written as:

$$\sum_{j=1}^{J} p_j \left(\mathcal{X}_j - \hat{m} \right)^2 = \hat{\sigma}^2$$

- A view on expectation $\tilde{m}$ and variance $\tilde{\sigma}$ amounts to choosing a new set of probability masses $\tilde{p}_j$ such as:

$$\begin{aligned} \sum_{j=1}^{J} \tilde{p}_j \mathcal{X}_j &= \tilde{m}, \\ \sum_{j=1}^{J} \tilde{p}_j \left(\mathcal{X}_j - \hat{m} \right)^2 &= \tilde{\sigma}^2. \end{aligned}$$

- We can write the views as linear equality system on the posterior probability masses $\tilde{p}_j$

$$A\tilde{\mathbf{p}} = b$$

where

$$A = \begin{pmatrix} \mathcal{X}_1 & \mathcal{X}_2 & \cdots & \mathcal{X}_J \\ \left(\mathcal{X}_1 - \hat{m} \right)^2 & \left(\mathcal{X}_2 - \hat{m} \right)^2 & \cdots & \left(\mathcal{X}_J - \hat{m} \right)^2 \end{pmatrix}$$

and

$$b = \begin{pmatrix} \tilde{m} \\ \tilde{\sigma}^2 \end{pmatrix}$$

- We solve the relative entropy minimization problem:

$$\tilde{\mathbf{p}} = \underset{A\tilde{\mathbf{p}}=b}{\operatorname{argmin}} \sum_{j=1}^{J} \tilde{p}_j \big[\ln(\tilde{p}_j) - \ln(p_j) \big]$$

- We keep the same scenarios $\mathcal{X}$ and replace $\mathbf{p}$ with $\tilde{\mathbf{p}}$, the expectation and variance views are now satisfied.

We start here with the simplest example by fitting a gaussian distributions to the log-returns of the S&P 500 index. Although the normal distribution has its flaws, it is important, for validation purposes, to test our model against analytical solutions that can be derived in this case. We also have to consider that many funds are still using the normal distribution in their investment activities, so it will be interesting to see the difference it produces. We start with the S&P 500 index and determine the parameters

μ and σ of the fitted normal distribution for the log-returns. We then generate $J = 10^5$ Monte-Carlo scenarios and apply two kinds of stress-tests to the expectation and standard deviation (or equivalently the variance) of the log-returns distribution, a mild one and an extreme one. By mild we mean that the views we force are relatively near to the prior whereas for the extreme case we take views that will modify the distribution consequently. We summarize the stress-types and stress values in Table 1:

		Scenarios		
		Base-case scenario	Mild stress-test	Extreme stress-test
Stress type	Expectation	0.043	-0.2	-0.8
	Variance	0.6563	1	2

Table 1: Stress-test

For a prior normal distribution

$$X \sim N(\mu, \sigma^2) \tag{1.1.16}$$

if we consider views on the expectation and the variance only, it can be proven analytically that the Kullback-Leibler divergence minimization produces a posterior distribution belonging to the class of normal distributions. If we write the views as:

$$\mathbb{V} \quad : \quad \begin{cases} \tilde{\mathbb{E}}[X] = \tilde{\mu} \\ \tilde{Var}(X) = \tilde{\sigma}^2 \end{cases} \tag{1.1.17}$$

then the posterior reads

$$X \sim N(\tilde{\mu}, \tilde{\sigma}^2). \tag{1.1.18}$$

Since we have 2 views, we therefore have a $J \times 2$ matrix composed of $\nu_{j,1}$ and $\nu_{j,2}$, where $\nu_{j,k}$ has been defined earlier and is a function of the $j-$th scenario.

The numerical constraint for the expectation view can be written as:

$$\sum_{i=1}^{J} \tilde{p}_i \mathcal{X}_i \quad = \quad \tilde{\mu}, \tag{1.1.19}$$

where the view is on $\nu_{j,1} = \mathcal{X}_j$, and $\mathcal{X}_i$ are the scenarios simulated from the normal distribution. The expectation value $\tilde{\mu}$ is defined exogenously and $\tilde{p}_i$ is the updated probability vector.

The numerical constraint for the variance view can be written as:

$$\sum_{i=1}^{J} \tilde{p}_i \Big(\mathcal{X}_i - \mu \Big)^2 \quad = \quad \tilde{\sigma}^2, \tag{1.1.20}$$

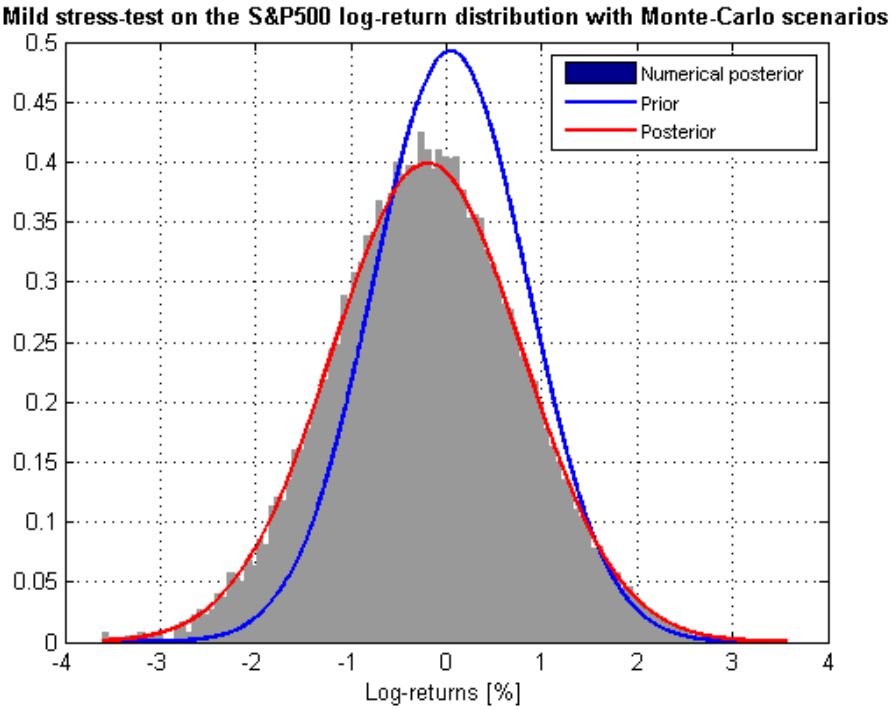

Figure 2: Mild stress-test

where the view is on $\nu_{j,2} = \left(\mathcal{X}_j - \mu\right)^2$, and its value $\tilde{\sigma}^2$ is defined exogenously.

We therefore have a benchmark for our numerical results. We observe in Fig. 2 that for a mild stress-test, the numerical and the analytical posterior match. However, for an extreme stress-test, the tails are not covered correctly and some probabilities even blow up as shown in Fig. 3, demonstrating a significant departure between the numerical and analytical solution. We would need a very large number of Monte-Carlo scenarios to get the right numerical solution. We address the problem of computing a correct posterior distribution, even when we have an extreme stress-test, by using a stable grid proposed in [MAK12].

We will not start with Monte-Carlo methods to generate the grid probabilities pair x_j, p_j because we either don't have enough scenarios to cover the tails of the distribution, or we need a very large number of simulations. We will take an approach which retains the flexibility and the scope of the views specification, specifying a deterministic grid and setting $p_j \equiv \int_{I_j} f(x)dx$ where

$$I_j \equiv \left[x_j - \frac{x_j - x_{j-1}}{2}, x_j + \frac{x_{j+1} - x_j}{2}\right] \tag{1.1.21}$$

and f is the density function. To dispose of a grid that covers the tails, we find the lower and upper extremes $\underline{x}$ and $\overline{x}$ such that the probability that the risk factor is smaller than the lower extreme (or larger than the upper extreme) is taken to be $\epsilon \approx 10^{-9}$. The grid points are the roots of the Hermite polynomial of order J $H_J(x) = (-1)^J e^{\frac{-x^2}{2}} \frac{d^J e^{\frac{-x^2}{2}}}{dx^J}$,

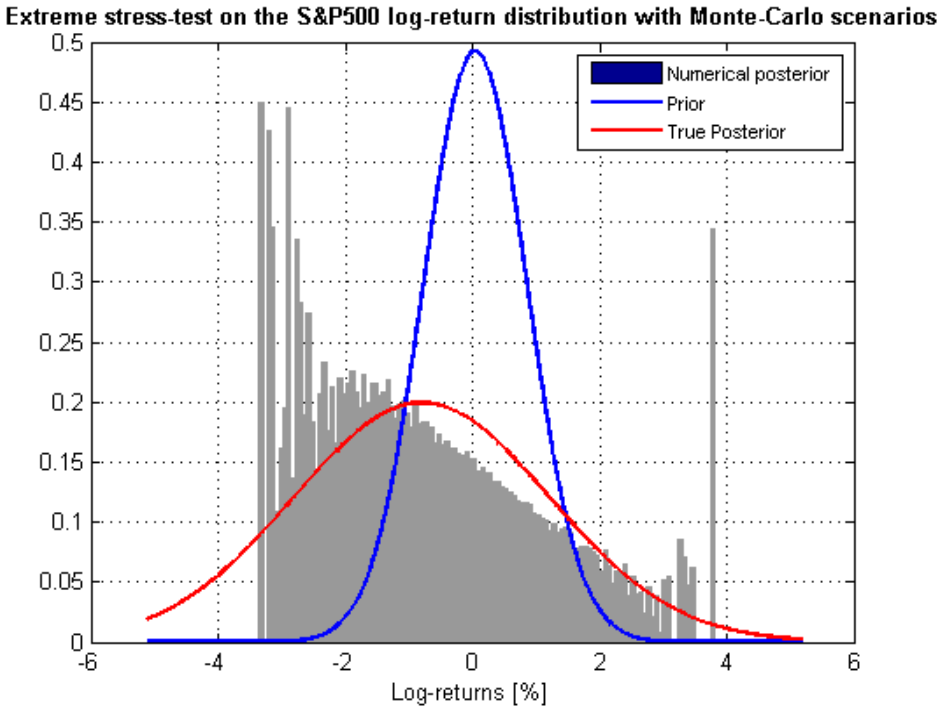

Figure 3: Extreme stress-test

that we translate and dilate to cover the lower and upper extremes. Now that the grid is defined, we apply the algorithms in the numerical optimization section and the adjustment needed for an extreme view in the previous section, to obtain the updated probabilities $\tilde{p}$. We can generate afterwards good quality Monte Carlo scenarios by first computing the posterior cumulative distribution function at the grid points:

$$\tilde{F}(x_j) \equiv \sum_{s=1}^{j} \tilde{p}_s. \tag{1.1.22}$$

We then simulate uniform distributions and use an interpolation operator on the uniforms that approximates the inverse distribution function, providing us the posterior, adjusted Monte Carlo scenarios.

We come back to our extreme stress-test example, where the standard fully flexible views framework fails because the Monte-Carlo method breaks down numerically. Instead, we follow the fully flexible extreme views framework introduced in [MAK12]. Meucci provided the Gauss-Hermite grid and the function computing the robust prior probabilities on the intervals I_j, the views on expectation and variance in this framework have however been programmed by us. We take a Gauss-Hermite grid with only $J = 10^3$ scenarios compared to $J = 10^5$ Monte-Carlo scenarios in the basic method. The result in Fig. 4 is compelling and demonstrates the power of using this grid.

As we have stated earlier, we can use this Gauss-Hermite grid to generate high-quality Monte-Carlo scenarios. We choose the number of scenarios $J = 10^5$. We generate J uniform random numbers, and then use an interpolation function which takes as inputs:

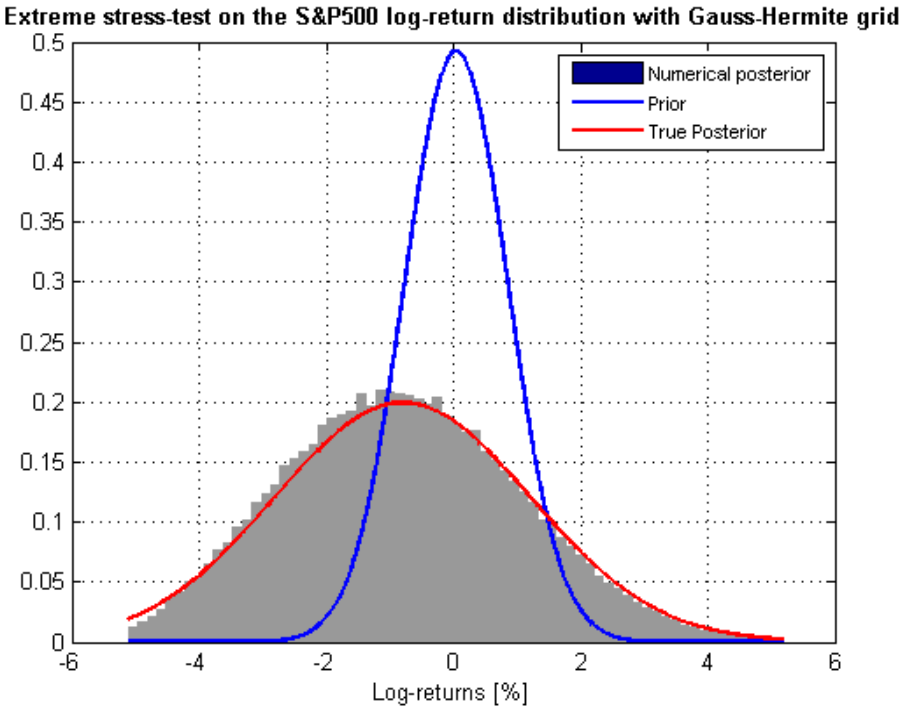

Figure 4: Extreme stress-test revisited

the J uniforms, the posterior cumulative distribution and the 10^3 deterministic grid scenarios. The outputs are the J posterior Monte-Carlo scenarios, which are equally weighted. We plot the numerical posterior with the Gauss-Hermite grid-adjusted Monte-Carlo scenarios in Fig. 5. The analytical and the numerical solution match, so we will always use a Gauss-Hermite grid in the future to process the tails before generating Monte-Carlo scenarios.

In the first Fig. 6, we suppose that we have one user who expresses his views but is not fully sure about them, meaning that he assigns a confidence level to his statement, which will temper the posterior probability and bring it closer to the prior distribution. Hence, the lower the confidence level, the closer the posterior will be to the prior. Let's plot the impact of the confidence level on the posterior probability for a stress-test. We take 5 different confidence levels, from 0.2 to 1 with increments of 0.2

In the second Fig. 8, we suppose we have 3 users who are assigned different confidence levels, which can correspond to their track-record, their level of seniority, it amounts to the level of faith we put in their views basically. The Table 7 displays the stakeholders with their views. We naturally expect that the user with the highest confidence level will bring the opinion-pooled posterior distribution closest to his posterior.

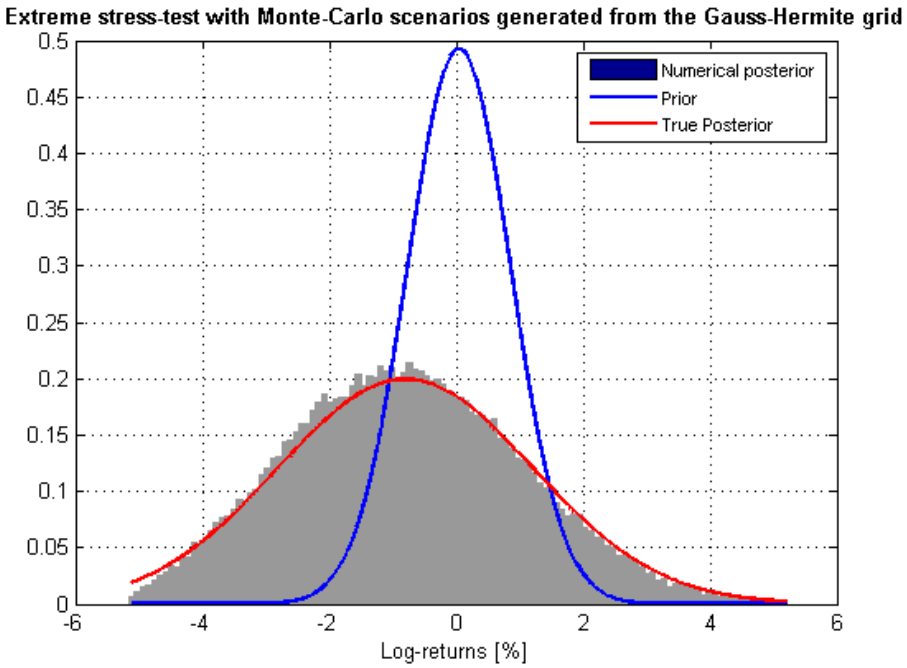

Figure 5: Gauss-Hermite adjusted Monte-Carlo scenarios

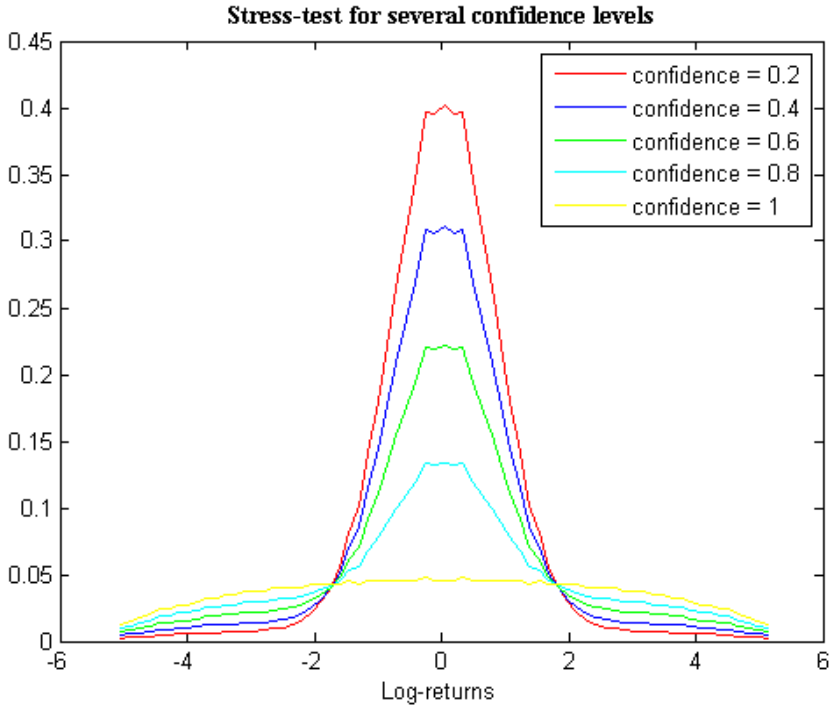

Figure 6: Confidence powers in a view

		User confidence levels		
		Junior analyst = 0.1	Asset manager = 0.3	Director = 0.6
Stress type	Expectation	0.3	-0.3	- 1
	Variance	0.35	0.7	0.2

Table 7: Opinion pooling: expert views

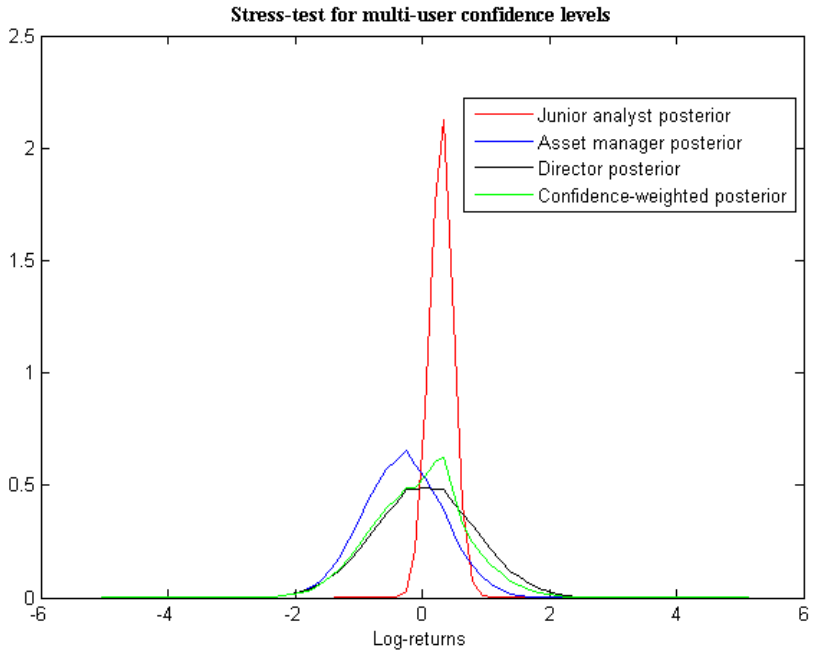

Figure 8: Opinion pooling visualized

To conclude this section, we computed stable Monte-Carlo scenarios and enabled mild and extreme stress-tests on the expectation and the variance of any risk factor specified by the fully flexible probabilities. We allowed collaboration with multiple views and users and computed the ensuing distribution that takes into account their outlooks. The general entropy pooling framework can naturally digest much more than only dispersion and location views, and we will now turn to risk measures used by regulators and banks to assess their potential losses and present how the fully flexible views can apply to them.

1.2 Extreme views and tail risk measures

We present two popular risk measures used by financial institutions to quantify the risks on their books, and explain how we can stress-test these quantities. Risk quantifies the uncertainty of the future outcome of a current decision or situation, and the possible outcomes are often described by a probability distribution. Banks however often express risk with one number. Volatility is a common measure of risk but it is only useful when returns are normally distributed. When this is not the case, we introduce other types of risk metrics. To measure extreme risks, the value-at-risk and the expected shortfall

are appropriate. These two measures are also often used for capital allocation, sometimes across several business lines, where the goal is not to provide finance but to absorb the risks undertaken, and are a precondition to optimize shareholder value. Introducing views on these risk metrics is more complicated than other statistical features. We give an important application of model views on value at risk at different confidence levels, and show how we can combine different models to produce better estimates of value-at-risk.

1.2.1 Value-at-risk and expected shortfall views

Value-at-risk and expected shortfall are two risk measures that attempt to quantify the market risk of an asset or a portfolio in a bank or pension fund. The former is the maximum loss over a time window with a high confidence level, whereas the latter measures the severity of the losses when the former amount is exceeded. These risk measures are often computed by models or historical simulation, which means they are only as good as the assumptions or the data behind it. Stress-testing these quantities and therefore covering unusual scenarios is therefore crucial in order for the bank to evaluate its capacity to absorb large losses as well as identify steps to reduce risks and preserve its capital. Before presenting these two measures in more detail, we give the definition of a coherent risk measure is, as theoreticized by Artzner et al. in [ADEH99]. A coherent risk measure displays the following properties:

1. Scalable, twice the risk should give a twice bigger measure,
2. Ranks risk correctly, bigger risks get bigger measures,
3. Allows for diversification, aggregated risks should have a lower measure,
4. Translation invariance with respect to riskless cash flows,
5. Relevance, non-zero risks get non-zero risk measures.

The value-at-risk is the maximum loss from adverse market movements for a given confidence level α (95 %, 99 %) over a given time horizon T (1, 10 days), so it's the level of loss that can occur over the time horizon that is not exceeded in $\alpha\%$ of cases. Mathematically, if we define L as the random variable expressing the loss of a portfolio, then $VaR_\alpha(L)$ is the α-quantile

$$VaR_\alpha(L) = \inf\left\{l \in \mathbb{R} \ : \ \mathbb{P}(L > l) \leq 1 - \alpha\right\} = \inf\left\{l \in \mathbb{R} \ : \ F_L(l) \geq \alpha\right\}$$

The value-at-risk is not a coherent measure of risk, it is relevant, scalable, ranks risks correctly and translation invariant but does not allow for diversification. We now show how we can express a view on the value-at-risk. We remind that $\nu_{j,k}$ are the elements of a $J \times K$ matrix representing a function, possibly non-linear, of the J scenarios $\mathcal{X}$, and K is the number of views. Let k be the column where we express the view on the value-at-risk (let's take it at a 99 % level and with a 1-day horizon). Without loss of generality, let's take $K = 1$. Since we have a scenarios-probabilities representation, we first sort the

scenarios from $\mathcal{X}$, in ascending order, i.e $\mathcal{X}_i \leq \mathcal{X}_j$ if the scenario index satisfies $i < j$. If we have a log-returns distribution, then the value-at-risk is the maximal log-return such that the probability that log-return outcomes are smaller than this value is lower than 1 %. For the next value, the probability exceeds 1 %. This corresponds to the left-tail of the distribution. For the log-return distribution, we thus find the index I such that

$$\sum_{i=1}^{I} p_i \leq 1\% \tag{1.2.1}$$

$$\sum_{i=1}^{I+1} p_i > 1\% \tag{1.2.2}$$

and $\mathcal{X}_I$ is the value-at-risk. A view on VaR amounts to choosing another $\mathcal{X}_{I_{view}}$ as the new value-at-risk and hence an index I_{view} implying that the linear constraint, whose coefficients in front of the probabilities will go into the matrix A, will now be written as:

$$\sum_{i=1}^{I_{view}} \tilde{p}_i \leq 1\% \tag{1.2.3}$$

$$\sum_{i=1}^{I_{view}+1} \tilde{p}_i > 1\% \Rightarrow \sum_{i=1}^{I_{view}+1} -\tilde{p}_i < -1\% \tag{1.2.4}$$

We resume the view on value-at-risk below, which we have programmed entirely.

Value-at-risk views at the α-% level

- Sort the scenarios $\mathcal{X}$ in increasing order.
- Recognize that the value-at-risk at the α-% level in the scenario-probability representation is written as:

$$\begin{aligned} \sum_{i=1}^{I} p_i &\leq \alpha\%, \\ \sum_{i=1}^{I+1} -p_i &< -\alpha\%, \\ VaR_\alpha(X) &= \mathcal{X}_I. \end{aligned}$$

- A view on value-at-risk $\widetilde{VaR}_\alpha(X)$ amounts to choosing a new set of probability masses $\tilde{p}_j$ such as:

$$\begin{aligned} \sum_{i=1}^{\tilde{I}} \tilde{p}_i &\leq \alpha\%, \\ \sum_{i=1}^{\tilde{I}+1} -\tilde{p}_i &< -\alpha\%, \\ \widetilde{VaR}_\alpha(X) &= \mathcal{X}_{\tilde{I}}. \end{aligned}$$

- We can write the view as a linear inequality system on the posterior probability masses $\tilde{p}_j$

$$A\tilde{\mathbf{p}} < b$$

where

$$A = \begin{array}{c} \begin{matrix} 1 & 2 & \cdots & I & I+1 & I+2 & \cdots & J \end{matrix} \\ \begin{pmatrix} 1 & 1 & \cdots & 1 & 0 & 0 & \cdots & 0 \\ -1 & -1 & \cdots & -1 & -1 & 0 & \cdots & 0 \end{pmatrix} \end{array}$$

has dimension $2 \times J$, the row above the first row of A lists the column indices, and

$$b = \begin{pmatrix} \alpha\% \\ -\alpha\% \end{pmatrix}$$

- We solve the relative entropy minimization problem:

$$\tilde{\mathbf{p}} = \underset{A\tilde{\mathbf{p}}<b}{\operatorname{argmin}} \sum_{j=1}^{J} \tilde{p}_j \big[\ln(\tilde{p}_j) - \ln(p_j) \big]$$

- We keep the same scenarios $\mathcal{X}$ and replace $\mathbf{p}$ with $\tilde{\mathbf{p}}$, the view on the value-at-risk is now satisfied.

We have in our example with the log-returns distribution of the S&P 500 a value-at-risk of -1.8339 and we set a view on the VaR of -2.1059, which should make the tail fatter. This gives us the following prior and posterior distributions in Fig. 9. We notice that the prior and posterior differ only in the region of the left tail, where obviously a lower value-at-risk view will imply that the probability masses of the posterior should be higher in that region to maintain the 1 % level.

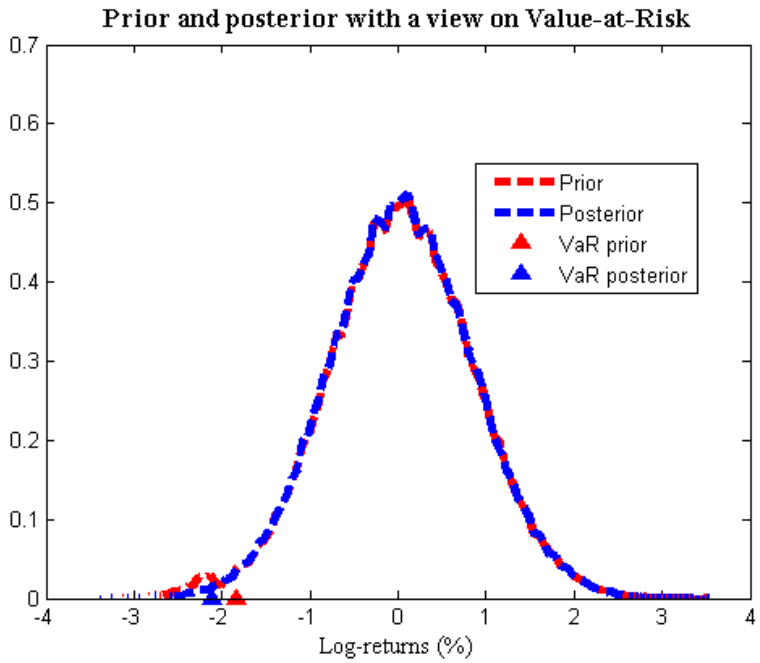

Figure 9: View on VaR: prior and posterior

The second risk measure we introduce is the expected shortfall or CVaR, which enjoys the benefits of diversification thanks to its sub-additivity property contrary to the value-at-risk and is therefore a coherent measure of risk. For a loss variable L, the CVaR is the average loss beyond the value-at-risk and therefore computes the expected loss in a worst-case scenario. The expected shortfall stress-tests have been addressed in the fully flexible extreme views framework in [MAK12] and we follow its implementation. Suppose we have a view on the CVaR:

$$\tilde{\mathbb{E}}\big[X \big| X \leq q\tilde{u}_\gamma\big] \equiv c\tilde{v}_\gamma \tag{1.2.5}$$

$c\tilde{v}_\gamma$ is the target $1-\gamma$ conditional value-at-risk and $q\tilde{u}_\gamma$ the posterior γ-quantile. If we want to add a view on conditional value-at-risk, the problem is that we do not know value-at-risk beforehand and thus cannot transform the view into a linear constraint on the posterior. So we have to find the VaR. Sorting the scenarios in increasing order as we did for the value-at-risk. For $s \in \{1,\dots\ J\}$, we define:

$$p^{(s)} = \underset{q \in \mathcal{C}_s}{\operatorname{argmin}}\, \mathcal{E}(q,p) \tag{1.2.6}$$

$$\mathcal{C}_s : \begin{cases} x_1 q_1 + \cdots + x_s q_s \equiv c\tilde{v}_\gamma \\ q_1 + \cdots + q_s \equiv \gamma. \end{cases} \tag{1.2.7}$$

The constraints are linear, and we can solve the problem by switching to the dual. The solution satisfies the CVaR constraint while being as close as possible to the reference distribution, but the posterior satisfying the CVaR view is the probability vector with

the least distortion among all the $p^{(s)}$ for $s \in \{1,... J\}$, hence: $\tilde{p} = p^{(\tilde{s})}$ where $\tilde{s} = \text{argmin}_{s\in\{1,... J\}} \mathcal{E}(p^{(s)}, p)$. We use the Newton-Raphson method to solve the problem since the computation is really costly, by observing that the relative entropy is a concave function of s and defining $D\mathcal{E}(p^{(s)}, p) = \mathcal{E}(p^{(s+1)}, p) - \mathcal{E}(p^{(s)}, p)$. We initialize with a value $\underline{s} \in \{1,... J\}$ and apply the numerical scheme:

$$\overline{s} = \text{int}\left(\underline{s} - \frac{D\mathcal{E}(p^{(\underline{s})}, p)}{D^2\mathcal{E}(p^{(\underline{s})}, p)}\right) \tag{1.2.8}$$

$$\underline{s} = \overline{s} \tag{1.2.9}$$

where int is the closest integer and D^2 is the second empirical derivative with respect to s.

We write below our algorithm for conditional value-at-risk views. For the programming part, we have taken inspiration from Meucci's code example but rewrote it into a function to avoid duplication and make it reusable.

Expected shortfall views at the α-% level

- Sort the scenarios $\mathcal{X}$ in increasing order.
- Recognize that the expected shortfall at the α-% level in the scenario-probability representation is written as:

$$\begin{aligned}
\sum_{i=1}^{s} p_i \mathcal{X}_i &= es_\alpha, \\
\sum_{i=1}^{s} p_i &< \alpha\%, \\
\sum_{i=1}^{s+1} -p_i &< -\alpha\%, \\
CVaR_\alpha(X) &= es_\alpha.
\end{aligned}$$

- A view on expected shortfall $\widetilde{CVaR}_\alpha(X)$ amounts to choosing a new set of probability masses $\tilde{p}_j$ such as:

$$\begin{aligned}
\sum_{i=1}^{\tilde{s}} \tilde{p}_i \mathcal{X}_i &= \tilde{es}_\alpha, \\
\sum_{i=1}^{\tilde{s}} \tilde{p}_i &< \alpha\%, \\
\sum_{i=1}^{\tilde{s}+1} -\tilde{p}_i &< -\alpha\%, \\
\widetilde{CVaR}_\alpha(X) &= \tilde{es}_\alpha.
\end{aligned}$$

- Solve a first optimization problem for all $s \in \{1,\dots\ J\}$

$$\mathbf{p}^{(s)} = \underset{A\hat{\mathbf{p}}^{(s)}=b}{\operatorname{argmin}} \sum_{j=1}^{J} \hat{p}_j^{(s)} \big[\ln(\hat{p}_j^{(s)}) - \ln(p_j) \big]$$

  where

$$A = \begin{array}{c} \begin{array}{ccccccc} 1 & 2 & \cdots & s & s+1 & \cdots & J \end{array} \\ \begin{pmatrix} \mathcal{X}_1 & \mathcal{X}_2 & \cdots & \mathcal{X}_s & 0 & \cdots & 0 \\ 1 & 1 & \cdots & 1 & 0 & \cdots & 0 \end{pmatrix} \end{array}$$

  has dimension $2 \times J$, the row above the first row of A lists the column indices, and

$$b = \begin{pmatrix} \tilde{es}_\alpha \\ \alpha\% \end{pmatrix}$$

- Report the result

$$\sum_{j=1}^{J} \tilde{p}_j^{(s)} \big[\ln(\tilde{p}_j^{(s)}) - \ln(p_j) \big]$$

  for all $s \in \{1,\dots\ J\}$ and choose

$$\tilde{s} = \underset{s\in\{1,\dots\ J\}}{\operatorname{argmin}} \sum_{j=1}^{J} p_j^{(s)} \big[\ln(p_j^{(s)}) - \ln(p_j) \big].$$

- The posterior probability mass $\tilde{\mathbf{p}}$ is obtained by setting:

$$\tilde{\mathbf{p}} = \mathbf{p}^{(\tilde{s})}.$$

- We keep the same scenarios $\mathcal{X}$ and replace $\mathbf{p}$ with $\tilde{\mathbf{p}}$, the view on the expected shortfall is now satisfied.

Fig. 10 and Fig. 11 show us the value of the Kullback-Leibler divergence, which is minimized, at each iteration and the posterior distributions for each iteration. We notice that the first two iterations produce very similar posteriors. We set to 0.01 and stop our algorithm when the change in the Kullback-Leibler divergence is smaller than this tolerance. At the third iteration, this condition is met and we thus obtain the red posterior which is clearly very different from the two first ones. In Fig. 12 we see the difference between prior and posterior but also the location of both expected shortfalls labeled by the blue and red triangles. We can observe that it wasn't just a shift to the left but a complete rearrangement of the probability masses.

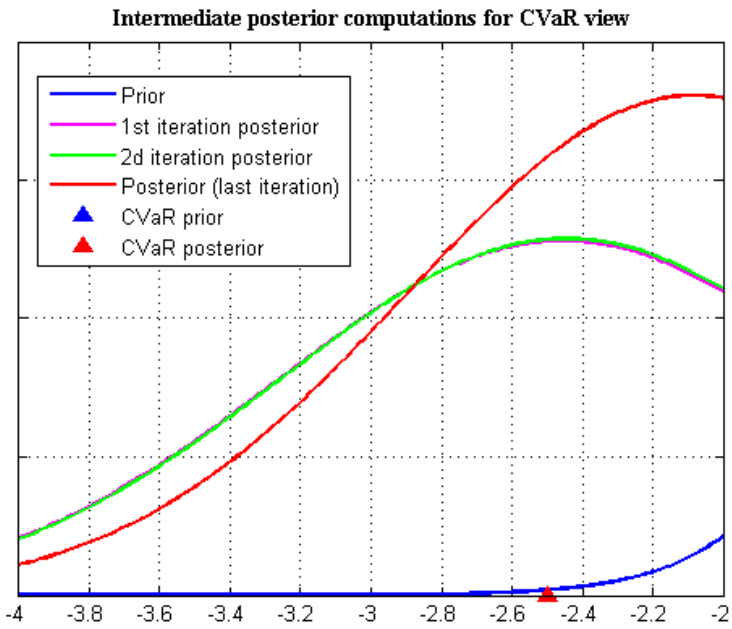

Figure 10: Intermediate computations of CVaR view

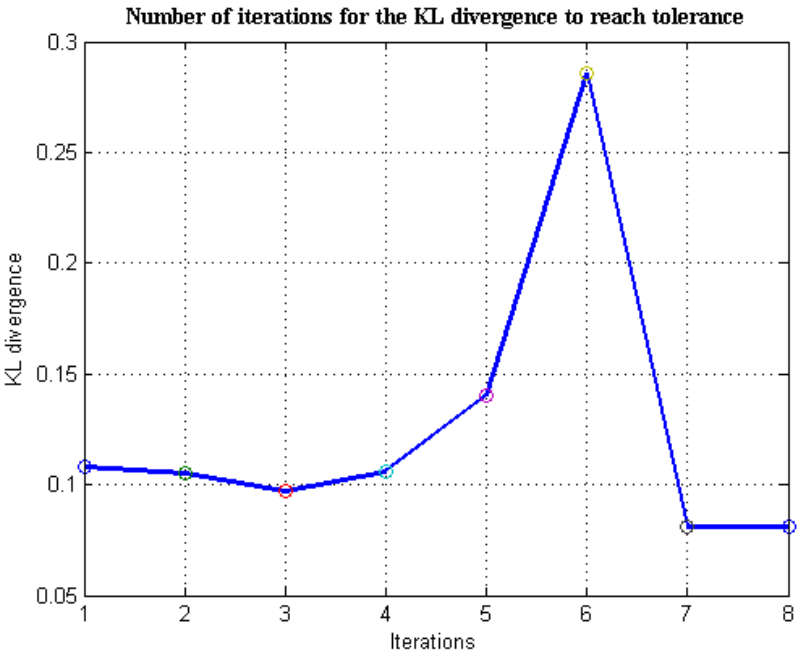

Figure 11: Value of the Kullback-Leibler divergence

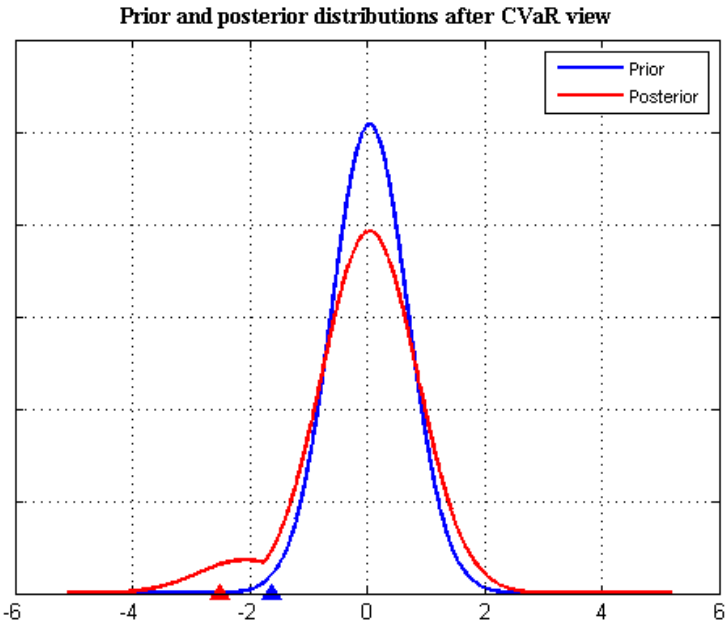

Figure 12: View on CVaR: prior and posterior

1.2.2 Confidence intervals for risk statistics

We know how to compute as well as process views on the expectation, variance, value-at-risk and expected shortfall when we represent the risk factors with a scenario-probability pair. We obtain one number for each statistic, but we have to be aware that there is an estimation risk associated to the computation of that risk number. To increase the robustness of the estimation exercise, we build confidence intervals in this fully flexible probabilities framework. The general applicability of this method to any risk model, however complicated it may be, makes it powerful because we do not need to know the specifications or the parameters characterizing the model but only its scenario-probability specification. Moreover, the methodology is the same for any risk measure, which makes it even simpler. This method is called the effective number of scenarios and defined by Meucci in [Meu12a]. We see how we can construct these confidence bands for the risk numbers in our framework, for a prior distribution as well as for a posterior distribution. The effective number of scenarios $\hat{J}$ is defined as the exponential of the entropy of the probability weights p_j:

$$\hat{J} \equiv e^{-\sum_{j=1}^{J} p_j \ln p_j}. \tag{1.2.10}$$

The entropy measures the concentration of probability mass in the p_j's. The effective number of scenarios counts the scenarios where the probabilities are strictly positive, this number ranges from 1 to J. The maximum is equal to J when all probabilities are equal ($p_j = \frac{1}{J}$), and the minimum is equal to 1 when we emphasize one scenario only, i.e there exists an index j such that $p_j = 1$. It is a measure of the statistical power of a stress-test, the higher the effective number of scenarios, the more powerful the test. If we denote by σ

the true risk number that could be the standard deviation, the skewness, kurtosis, value-at-risk, expected shortfall, of the unknown distribution and $\hat{\sigma}$ the risk number computed in our framework. Statistical consistency ensures that the approximated risk number converges to the true risk number as the number of scenarios grows:

$$\lim_{J\to\infty} \hat{J} = \infty \Rightarrow \lim_{J\to\infty} \hat{\sigma} = \sigma. \tag{1.2.11}$$

For a confidence band

$$\left[\underline{\sigma}, \overline{\sigma}\right] \tag{1.2.12}$$

around σ, here is the procedure to follow that we have programmed from scratch.

Confidence intervals for any risk statistic σ

- we compute the empirical scenario-probability distribution,
- we compute the effective number of scenarios $\hat{J}$ and round it to the nearest integer,
- we repeat the following step $S \approx 1000$ times for $s = 1, \ldots, S$: drawing $\hat{J}$ independant and equally-weighted scenarios from the empirical distribution and computing $\hat{\sigma}^{(s)}$,
- compute the lower and upper quantiles $\underline{\sigma}$ and $\overline{\sigma}$ from the distribution of the risk number $\{\hat{\sigma}^{(s)}\}_{s=1,\ldots,S}$.

	Distribution	
	Prior	Posterior
Effective number scenarios	10000	9177
Expectation	0.043 [0.034 0.052]	-0.19999 [-0.2 -0.19997]
Standard deviation	0.80967 [0.80294 0.81622]	0.98078 [0.98036 0.98122]
VaR	-1.8452 [-1.8797 -1.8122]	-2.5178 [-2.6475 -2.3913]
CVaR	-2.117 [-2.1601 -2.0753]	-2.7099[-2.8678 -2.5602]

Table 13: 75% confidence level for prior and posterior risk statistics

	Distribution	
	Prior	Posterior
Effective number scenarios	10000	9177
Expectation	0.043 [0.0266 0.058]	-0.19999 [-0.2 -0.19995]
Standard deviation	0.80967 [0.79832 0.82222]	0.98078 [0.98017 0.98155]
VaR	-1.8452 [-1.9033 -1.787]	-2.5178 [-2.7938 -2.325]
CVaR	-2.117 [-2.1935 -2.0465]	-2.7099[-3.0193 -2.4747]

Table 14: 95% confidence level for prior and posterior risk statistics

This procedure applies to all types of distributions that are given by the scenario-probability specification and all kinds of risk statistics. We take the mild stress-test example from Table 1 and compute the following statistics as well as their confidence intervals for confidence levels of 75% and 95%: the effective number of scenarios, the expectation, the standard deviation, the value-at-risk and the conditional value-at-risk. We also compute for each risk number its expected value by giving an equal weight to all S resamplings. The results are shown in Table 13 and Table 14. For the prior distribution, we generated $J = 10^5$ equally-weighted Monte-Carlo scenarios, which explains why the effective number of scenarios is equal to 10000. However, the mild stress-test changes the probabilities of the prior distribution and therefore lowers the effective number of scenarios, now equal to 9177. As expected, the confidence bands in brackets are narrower in Table 13 than Table 14. These confidence intervals can serve multiple purposes: they can be used for robust asset allocation and incorporate the estimation uncertainty of the risk measures (such as the mean and CVaR) directly in the optimization process, or for stress-testing by taking the lower or upper end of the brackets instead of the expected value.

1.2.3 Risk-correcting views with backtests

We know how to process views on value-at-risk, albeit in an exogenous way, and this would count as an expert view. However, it is also possible to set a view that doesn't come from a human being, but from a model. You may say that we are already using a model, and if it is not doing a good job at estimating the VaR, then we should change it by another one. This is however simplistic, and our model may be good for example for predicting expected returns and correlations, but bad for downside risks. Our prior model could also be good for value-at-risk estimation at some confidence levels and bad at other thresholds. We want to keep using our prior model for the things it is good for, and refine it by mixing other models that will bring a more accurate view of the VaR with risk-correcting views. This case study was programmed by using fitting procedures available in the statistics toolbox of Matlab, but the backtesting methodology and the confidence interval generation were programmed by hand.

We use a case study with IBM daily log-returns, and we fit a simple Gaussian distribution. We then produce a high-number of simulations with the estimated parameters and compute the Monte-Carlo Value-At-Risk at the 1%, 2.5%, 5% and 10% confidence levels for a 1-day horizon. The results are in Table 15.

		VaR for fitted Gaussian distribution
Confidence levels	1%	-3.9873 %
	2.5%	-3.3331 %
	5%	-2.8379 %
	10%	-2.2153 %

Table 15: Value-at-risk at several confidence levels

The question we can ask ourselves is how flawed the risk estimation is. We use the

Anderson-Darling hypothesis test in order to accept or reject the hypothesis that the sample is coming from a normal distribution. The $p-$value is computed and compared to the significance level for the decision to be made. The principle of the test is to measure a weighted distance between the hypothesized normal distribution and the empirical cumulative distribution, where the weight function is putting the emphasis on the tails and is therefore better at detecting departures from normality. The $p-$value returned at the 5% level is extremely small, which favors a strong rejection of the normal distribution hypothesis. Now we know that our previous risk estimates are not correct. We could choose totally different models and compute the value-at-risk, or we could keep our normal distribution but use more accurate models for VaR specifications. We will explain our choice afterwards.

We start fitting 4 distributions to the IBM log-returns:

- A Student distribution,
- a non-parametric distribution with normal kernel,
- a non-parametric distribution with Epanechnikov kernel,
- a semi-parametric distribution with an empirical cumulative distribution in the center and Pareto in the tails.

We also perform the Anderson-Darling test on these four distributions, this test is not only meant for Gaussian distributions. We report the p-values at the 5% level and the decision in Table 16.

		Hypothesis testing	
		$p-$value	Test decision
Distributions	Student distribution	0.83542	Accept
	Nonparametric normal kernel	0.84465	Accept
	Nonparametric Epanechnikov kernel	0.84465	Accept
	Semi-parametric	0.854632	Accept

Table 16: Anderson-Darling tests for various distributions

They all pass the Anderson-Darling test. Does it mean that we can use any of these models to process views on the value-at-risk instead of the gaussian value-at-risk ? No it does not, and as a matter of fact we are going to introduce more stringent ways to perform the model validation through out-of-sample forecasts. The backtesting methodology is taken from [Ale09] and we implemented it for our case study. If the backtest fails, it means the estimation error is too large or the model has been misspecified. We choose an estimation period of 1000 days which defines the number of days used to estimate the VaR model parameters. We then then roll over the estimation sample 2425 times while keeping the estimation period constant. The testing sample starts after the last day of the estimation sample. Also, since we estimate a daily VaR, the estimation from day k to day $k+1000$ produces a VaR from the $k+1000$-th to the $k+1001$-th day. The realized return from the $k+1000$-th to the $k+1001$-th day is also recorded. We roll the window

forward 1 day and repeat the estimation of the VaR until the end of the entire sample. We therefore have two time series which are going to form the backbone of the backtest: the daily VaR at a specified confidence level, and the realized returns. By comparing these two time series, we can count the number of times the realized returns is lower than the estimated VaR, and we call it a VaR violation or exceedance. For example, if we estimate a 1% VaR and have 2000 observations in the time series, we would expect the VaR to be exceeded 20 times. We can make a statistical test with the number of exceedances. Our backtest assumption is an i.i.d Bernoulli process generation of the daily returns, where success means a VaR violation and is labelled 1. More formally, we define X_{t+1} as the realized return from t to $t+1$, $VaR_{\alpha,t}$ the $\alpha\%$ value-at-risk estimated at time t for the following day, and $I_{\alpha,t}$ as

$$I_{\alpha,t} = \begin{cases} 1, \text{ if } X_{t+1} < VaR_{\alpha,t} \\ 0, \text{ otherwise.} \end{cases} \tag{1.2.13}$$

If the model produces a good VaR assessment, then the probability of success is

$$\mathbb{P}\left(I_{\alpha,t} = 1 \text{ at any time } t \right) = \alpha. \tag{1.2.14}$$

The expected number of successes in a sample with n observations is therefore n times the previous probability if the indicator is independent and identically distributed. Calling $Y_{n,\alpha}$ the number of successes, it is a binomial distribution with parameters n and α, thus:

$$\mathbb{E}\left[Y_{n,\alpha} \right] = n\alpha, \tag{1.2.15}$$

$$V\left(Y_{n,\alpha} \right) = n\alpha(1-\alpha). \tag{1.2.16}$$

When the number of observations n is big enough, the binomial distribution converges to a normal distribution, so we can write a confidence interval C for $Y_{n,\alpha}$ under the null hypothesis that the VaR model is correct:

$$C = \left(n\alpha - 1.96n\sqrt{\alpha(1-\alpha)},\ n\alpha + 1.96n\sqrt{\alpha(1-\alpha)} \right). \tag{1.2.17}$$

If the observed number of exceedances lies outside this confidence interval, this leads to a rejection of the null hypothesis and therefore hints at a misspecification of the VaR model.

We will do a backtest with our four distributions and for the 1%, 2.5%, 5% and 10% confidence levels. We report the confidence intervals, the observed number of exceedances for each distribution and confidence level on one hand, and the rejection results.

The Student distribution is rejected at the 1%, 2.5% and 10% levels but accepted at the 5% level. All other distributions are accepted at the 1%, 2.5% and 5% but all rejected

		Confidence interval $C = (14, 33)$	
		1% level	Test decision
Number exceedances	Student distribution	35	Reject
	Normal kernel	27	Accept
	Epanechnikov kernel	27	Accept
	Semi-parametric	30	Accept

Table 17: Backtesting VaR at the 1% level

		Confidence interval $C = (45, 75)$	
		2.5% level	Test decision
Number exceedances	Student distribution	76	Reject
	Normal kernel	65	Accept
	Epanechnikov kernel	68	Accept
	Semi-parametric	68	Accept

Table 18: Backtesting VaR at the 2.5% level

at the 10%. If we take the value-at-risk view with the gaussian distribution at the 10% level, it lies inside the confidence interval and is thus accepted. When more than one lies inside the confidence interval, we choose the one which is closest to the expected value of the binomial distribution. Based on this case study alone, we can process the views on value-at-risk as following:

- Gaussian VaR at the 10% level,
- Student VaR at the 5% level,
- non-parametric distribution with normal kernel VaR at the 2.5% level,
- non-parametric distribution with Epanechnikov kernel VaR at the 1% level.

We have demonstrated that by combining different risk models, we can improve the risk estimation of the value-at-risk at different confidence levels. The prior risk model is altered with views on the quantiles coming from different models who perform best at the required confidence levels using a backtesting methodology as a performance indicator. This makes the posterior risk model more accurate by construction.

After discussing risk estimation enhancement, we will now move on to predictive scenario analysis. Our prior risk model may be blind to certain future developments, and this is why we want to embed analyst outlooks or other quantitative models, bringing other information to light, in the same way we did with risk-correcting views. The posterior risk model then reflects that new information and should provide a better forward-looking perspective. In the next section, we study a predictive model based on the findings of Johansen, Ledoit and Sornette in [JLS71], who gives us warnings about crashes or more generally change of regimes, so we can update our prior model in order to reflect this valuable information.

		Confidence interval $C = (100, 142)$	
		5% level	Test decision
Number exceedances	Student distribution	119	Accept
	Normal kernel	111	Accept
	Epanechnikov kernel	111	Accept
	Semi-parametric	113	Accept

Table 19: Backtesting VaR at the 5% level

		Confidence interval $C = (213, 271)$	
		10% level	Test decision
Number exceedances	Student distribution	199	Reject
	Normal kernel	190	Reject
	Epanechnikov kernel	198	Reject
	Semi-parametric	203	Reject

Table 20: Backtesting VaR at the 10% level

1.3 Predictive scenario analysis: crashes, rebounds

We saw earlier that views could come from experts or analysts who want to incorporate their assessments, and not leave everything to the risk model which is by definition calibrated with historical data, i.e backward-looking. Views can alternatively originate from risk-correcting models, and are usually used to enhance the prior model's statistical properties, for example by improving value-at-risk or expected shortfall estimates. A last type of view we are going to present here are predictive views, i.e views that foresee where statistical quantities or risk factors are moving, and we will use machine learning to generate these predictions in conjunction with the Johansen-Ledoit-Sornette bubble model in [JLS71]. We then assess its performance. If the predictive power of the data mining method is proven, we can then embed the prediction with the prior risk model, acting on one or more of its features by using Meucci's fully flexible views.

1.3.1 The Johansen-Ledoit-Sornette model

Intuition and theory

We will first explain the intuition of the model before delving in the equations, and we refer to [Sor09] for more detailed discussions. The traditional Johansen-Ledoit-Sornette model assumes that the price satisfies a jump-diffusion stochastic differential equation. The jump component quantifies the loss amplitude if a crash has occurred, if not there is none. Jumps are driven by the crash hazard rate. The crash hazard rate on a time interval is the conditional probability that the crash occurs on this time interval knowing that the crash hasn't happened yet. The structure of the financial system is represented as a hierarchical system, the magnitude of the different agents vary and form structures, from the small retail investor to sovereign wealth funds, and this system determines the functional form of the crash hazard rate. In a simple structure, the crash hazard rate will increase in a power law fashion towards a point in time that we call the critical time.

The structure chosen by Johansen-Ledoit-Sornette in [Sor09], called diamond lattice, displays accelerating log-periodic oscillations on top of the power law, with a divergence at the critical point. The herding behavior among the agents should illustrate the fight between order and disorder. This fight can also be exemplified by fundamental versus noise traders and quantified by the log-periodic power law. The log-periodicity shows the changing balance of power between both trader types and the power law reveals the rising strength of the noise traders who herd. The crash hazard rate, once determined, is connected by a simple relation to the log-price or the log-returns, allowing us to write the equation for these observable quantities and perform statistical analysis.

We now present the traditional JLS model formally, following the exposition in [JLS71]. Introducing an exogenous probability of crash modeled by a jump process j which is 0 if the crash has not occurred, and 1 right after the crash. We denote by $Q(t)$ and $q(t)$ the cumulative distribution function of the crash time and its density, with the crash hazard rate defined as:

$$h(t) = \frac{q(t)}{1-Q(t)} \approx \mathbb{P}\left(\text{"Crash time" in } \left[t, t+dt\right] \middle| \text{"Crash time"} > t\right) \tag{1.3.1}$$

When a crash occurs, the price drops by κ. The stochastic differential equation governing the asset price $p(t)$ is:

$$dp(t) = \mu(t)p(t)dt + \sigma(t)p(t)dW_t - \kappa p(t)dj \tag{1.3.2}$$

where the drift is constrained to satisfy the rational expectation condition

$$\begin{aligned}
\mathbb{E}\left[dp(t)|\mathcal{F}_t\right] &= \mu(t)p(t)dt + \sigma p(t)\,\mathbb{E}\left[dW_t|\mathcal{F}_t\right] - \kappa p(t)\,\mathbb{E}\left[dj|\mathcal{F}_t\right] \\
&= \mu(t)p(t)dt + 0 - \kappa p(t)\Big(1 \times h(t)dt + 0 \times (1-h(t)dt)\Big) \\
&= \mu(t)p(t)dt - \kappa p(t)h(t)dt = 0,
\end{aligned}$$

which gives the condition $\mu(t) = \kappa h(t)$, and by integration, yields:

$$\ln \mathbb{E}\left[p(t)\right] = \ln \mathbb{E}\left[p(t_0)\right] + \int_{t_0}^{t} \kappa h(t')dt', \tag{1.3.3}$$

The power law and the log-periodic accelerating oscillations backed-up from the susceptibility in the hierarchical diamond lattice structure, determine the crash hazard rate's form

$$h(t) = B'(t_c - t)^m + C'(t_c - t)^m \cos(\omega \ln(t_c - t) - \psi) \tag{1.3.4}$$

where t_c is the critical time, and integrating it with the previous condition gives us:

$$\ln \mathbb{E}\left[p(t)\right] = A + B(t_c - t)^m + C(t_c - t)^m \cos(\omega \ln(t_c - t) - \phi) \tag{1.3.5}$$

where

$$\begin{aligned} B &= \frac{-\kappa B'}{m}, \\ C &= \frac{-\kappa C'}{\sqrt{m^2+\omega^2}}. \end{aligned}$$

These dynamics are only valid up to the critical time, and new dynamics take place afterwards.

An efficient numerical implementation

The bubble model presented previously is described by 3 linear parameters (A, B, C) and 4 nonlinear parameters (m, t_c, ω, ϕ). We will show here the results obtained by Filimonov and Sornette in [FS99] to transform the problem into a fitting procedure with 4 linear and 3 nonlinear parameters to decrease the computational cost significantly and improve the stability with a cost function that displays one minimum only instead of multiple candidates. Moreover, the growth rate exponent m and the log-frequency ω are slaved to the critical time t_c, which decreases the complexity of the numerical implementation even stronger.

First we rewrite $\ln \mathbb{E}\left[p(t)\right]$ as:

$$\begin{aligned} \ln \mathbb{E}\left[p(t)\right] &= A + B(t_c-t)^m + C(t_c-t)^m \cos(\omega \ln(t_c-t)) \cos\phi \\ &+ C(t_c-t)^m \sin(\omega \ln(t_c-t)) \sin\phi \\ &= A + B(t_c-t)^m + C_1(t_c-t)^m \cos(\omega \ln(t_c-t)) \\ &+ C_2(t_c-t)^m \sin(\omega \ln(t_c-t)) \end{aligned} \tag{1.3.6}$$

where $C_1 = C\cos\phi$ and $C_2 = C\sin\phi$. A least-squares method with the following cost function is applied

$$\begin{aligned} F(t_c, m, \omega, A, B, C_1, C_2) &= \sum_{i=1}^{N} \Big[\ln p(\tau_i) - A - B(t_c-\tau_i)^m \\ &- C_1(t_c-\tau_i)^m \cos(\omega \ln(t_c-\tau_i)) \\ &- C_2(t_c-\tau_i)^m \sin(\omega \ln(t_c-\tau_i)) \Big]^2. \end{aligned} \tag{1.3.7}$$

We then slave the 4 linear parameters (A, B, C_1, C_2) to the 3 nonlinear parameters (t_c, m, ω) to get the nonlinear optimization

$$\left(\hat{t_c}, \hat{m}, \hat{\omega}\right) = \underset{t_c, m, \omega}{\operatorname{argmin}} F_1(t_c, m, \omega), \tag{1.3.8}$$

$$F_1(t_c, m, \omega) = \underset{A, B, C_1, C_2}{\operatorname{argmin}} F(t_c, m, \omega, A, B, C_1, C_2). \tag{1.3.9}$$

The solution of

$$\left(\hat{A}, \hat{B}, \hat{C_1}, \hat{C_2}\right) = \underset{A, B, C_1, C_2}{\operatorname{argmin}} F(t_c, m, \omega, A, B, C_1, C_2) \tag{1.3.10}$$

is obtained by solving

$$\begin{pmatrix} N & \sum f_i & \sum g_i & \sum h_i \\ \sum f_i & \sum f_i^2 & \sum f_i g_i & \sum f_i h_i \\ \sum g_i & \sum f_i g_i & \sum g_i^2 & \sum g_i h_i \\ \sum h_i & \sum f_i h_i & \sum g_i h_i & \sum h_i^2 \end{pmatrix} \begin{pmatrix} \hat{A} \\ \hat{B} \\ \hat{C}_1 \\ \hat{C}_2 \end{pmatrix} = \begin{pmatrix} \sum y_i \\ \sum y_i f_i \\ \sum y_i g_i \\ \sum y_i h_i \end{pmatrix}$$

where

$$\begin{aligned} y_i &= \ln p(\tau_i), \\ f_i &= (t_c - \tau_i)^m, \\ g_i &= (t_c - \tau_i)^m \cos(\omega \ln(t_c - \tau_i)), \\ h_i &= (t_c - \tau_i)^m \sin(\omega \ln(t_c - \tau_i)), \end{aligned}$$

The second step is to slave the critical exponent and the log-frequency to the critical time to get an optimization problem where we generally have one minimum but no more than three, so we replace

$$\left(\hat{t_c}, \hat{m}, \hat{\omega}\right) = \underset{t_c,m,\omega}{\operatorname{argmin}} F_1(t_c, m, \omega), \tag{1.3.11}$$

$$\tag{1.3.12}$$

by

$$\hat{t_c} = \underset{t_c}{\operatorname{argmin}} \tilde{F}_2(t_c), \tag{1.3.13}$$

$$F_2(t_c) = \min_{\omega,m} F_1(t_c, m, \omega), \tag{1.3.14}$$

$$\left(\tilde{m}(t_c), \tilde{\omega}(t_c)\right) = \underset{\omega,m}{\operatorname{argmin}} F_1(t_c, m, \omega). \tag{1.3.15}$$

Filimonov and Sornette recommend to launch at most 20 searches with the Leverberg-Marquardt nonlinear least squares algorithm or the Nelder-Mead simplex within

$$\begin{aligned} 0.1 &\leq m_0 \leq 0.9 \\ 6 &\leq \omega_0 \leq 13. \end{aligned}$$

For $F_2(t_c)$, we start from different initial points t_{c_0} and use the local search algorithms stated previously. The advantage gained here is not so much the complexity reduction than the dependance of the critical exponent and the log-frequency on the critical time t_c.

The S&P 500: the stock bubble (2004 - 2007)

Professor Sornette and Dr. Peter Cauwels were very kind to provide the Filimonov-Sornette code in Matlab in order to fit the log-periodic-power law to the log-prices. We fit the S&P 500 index with the JLS model and visualize the impact on the fit and the

critical time forecast when we vary the time window but keeping the ending point of the window fixed. The goal here is to show the lack of robustness if we use just one time window, or if we don't use any goodness of fit or filtering criteria. This motivates the use of a machine learning technique that automatically filters away the wrong fits.

Indeed, if we do it naively and without any goodness of fit, we will get many fits and critical times that can be months away from each other. We have two crashes in this period, one that occurs on July 20th, 2007 and correctly identified by two fits which give us July 19th and 20th for the critical time. The other crash happens on October 10th, 2007 and we have 3 fits that are close to this date, with an error of a week. In Fig. 21 we can observe the sensitivity of the fits to the chosen time window. The colored triangles represent the critical times and the solid blue line is chosen as the actual date or ending point of the time window we use to fit the model.

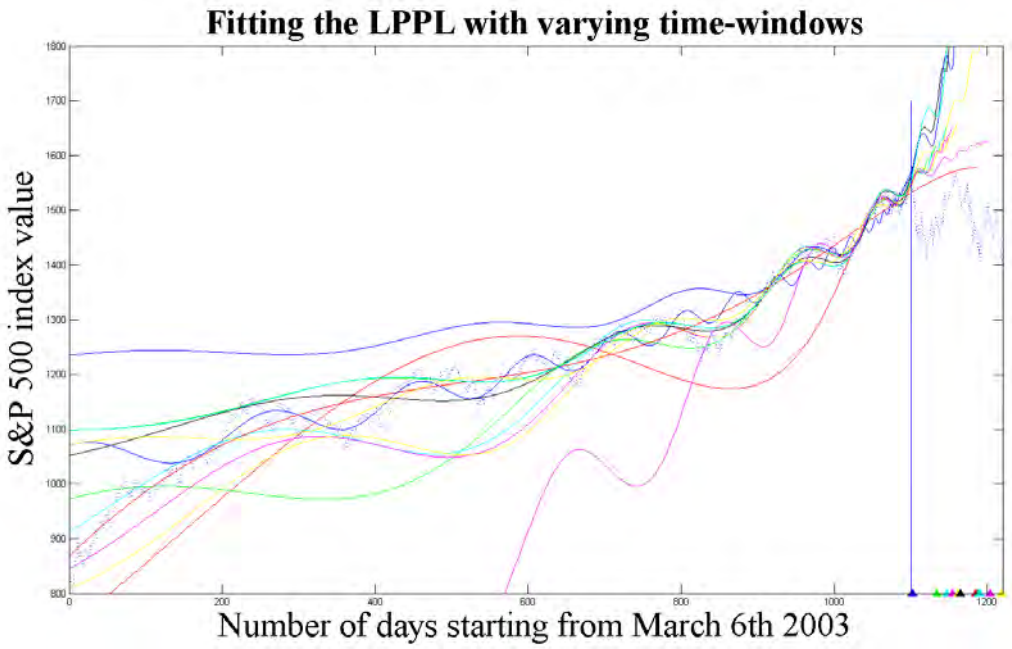

Figure 21: Log-periodic power law fits with no filtering

There are clearly some bad fits, and a filter that is described in [SWYZ71] to remedy this situation and that we can use for bubbles is:

$$b \quad := \quad -Bm - |C|\sqrt{m^2 + \omega^2} \geq 0. \tag{1.3.16}$$

All the fits which do not satisfy this condition are eliminated. This filter however applies to the original implementation of the Johansen-Ledoit-Sornette model, we do not have the parameter C anymore in the Filimonov-Sornette approach but C_1 and C_2. There is a simple relation between them by exploiting $\cos^2 u + \sin^2 u = 1$ for all u:

$$C_1^2 + C_2^2 = C^2.$$

Hence we can rewrite the filter as:

$$b \;:=\; -Bm - \sqrt{(C_1^2 + C_2^2)(m^2 + \omega^2)} \geq 0. \tag{1.3.17}$$

The question however remains how we can eliminate in a more systematic fashion bad fits and keep only the critical times close to a crash, because we are not sure that these two filtering conditions are enough. The next section attempts to answer this question by using the previous filtering condition 1.3.17, but also other indicators that should help us in the filtering task.

1.3.2 Data mining for critical time classification

Problem and goal

Our goal here is to answer the following question: can we tell if a modeled critical time is indeed close to a crash or not ? If it is, then we call this critical time an admissible critical time. We hope that this will help us separate the admissible critical times from the others by performing a classification task and thus sharpening our confidence interval bounds. We will define mathematically the terminology and the setting of our classification method afterwards: a crash, what close means, the number of time windows, the learning and testing set, etc. We just introduce the general idea here.

Obviously we need many candidates to get good confidence intervals for crash forecasts so the first task is to have many time windows. The second one is going to use a data mining algorithm on a known dataset, where crash times are known and thus the modeled critical times can be labeled as either close or not close to a crash. Although we have taken some inspiration from [YRWS12], in our problem the final goal is not to build a crash alarm index for each day but to classify all the critical times on the testing set as close or not close to a crash. We choose a date that separates the learning and the training sets, and we use a machine learning method for the classification task. We can pinpoint the misclassified critical times, either those who were indeed close to a crash and were labeled not close or those in reality not close to a crash and labeled close. This will allow us to assess the predictive accuracy of the algorithm. Once the algorithm has produced the right learner, we evaluate it on a testing set, for which we do not know the critical time labels. Of course we can compute ex-post the classification error, since the testing set is a historical dataset.

Ensemble learning: Random Forests

Ensemble learning is described in [Zho12]. We need a matrix data X, where each column represents the features or predictive variables. Then we prepare the response data Y or prediction outcome, in our case it's categorical data ("close", "not close" for crash detection). The learner is a model generated by an algorithm on the data. To operate our binary classification of critical times, we are going to use an ensemble method, which consists in training multiple learners to solve a same task and combine them. The reason we use an ensemble method is because their generalization ability is much higher than our base learners. Weak learners are thus combined to form a strong learner with a higher

predictive accuracy. Ensemble learning techniques almost always win machine learning competitions, as evidenced by the KDD-cup or the Netflix competition. "Random Forests" is an ensemble learning method introduced by Breiman in [Bre01]. Basically, "Random Forests" uses trees for decision-making and randomizes the tree splits as well as the training sets used. Bootstrap samples are new training sets created by randomly sampling many number of times the original training set. Trees are added until the error no longer decreases, and we call N the number of trees. For each of the N iterations, we select a bootstrap sample and grow a tree on this bootstrap. We then save the tree and move to the next iteration. The output is the average response for regression and the majority vote for classification from all individually trained trees. The percentage of misclassified observations is called the classification error.

We can also use the JLS model to identify anti-bubbles and thus detect rebounds. We then want to classify the critical times as being close to crashes or rebounds or neither. The method works in the same way as for determining whether the critical time is close or not to a crash, and we will include it as an extension while focusing our efforts on the crash detection.

Random Forest application to crash and rebound detection

We will now detail the data mining algorithm for binary classification of critical time proximity to crashes and rebounds hereafter, combining the Random Forests in [Bre01] and the JLS model in [JLS71] and [FS99]. We have programmed this algorithm in Matlab, using the TreeBagger class who creates the ensemble of decision trees, its associated methods, and the code provided by Professor Sornette for fitting the log-periodic power law.

Crash detection

1. **Creating the time windows for the log-prices**
   (a) Choose $w = 90$ window widths, from 110 to 1000 days with an increment $dw = 10$ days between each successive window.
   (b) The 1000 first days of the dataset will be used to build w time windows for the first training date t_1, which is constrained to be the last date or endpoint of all these windows and we will say in this case that it's a generator for these time windows.
   (c) Define an increment parameter $dt = 10$ and the number of training dates N to obtain the subsequent training dates $t_{i+1} = t_i + dt$ for $i = 0, ..., N-1$, each of them generating a new set of w time windows.
   (d) Repeat the procedure until we arrive at the actual analysis date $t_{N+1} = t_{actual}$, which means that we do not know prices after this date and this also implies that training dates close to t_{actual} will produce unlabeled critical times because they extend in a territory where we can't say whether a crash has occurred or not.
2. **Fitting the log-prices with the Johansen-Ledoit-Sornette model**

(a) Define the range $RANGE$ of investigation for the critical times.

(b) We apply the Filimonov-Sornette procedure for all dates t_i (training and actual) and all generated windows, with critical time candidates ranging from $t_i + 1$ to $t_i + RANGE$.

(c) We obtain the parameters A, B, C_1, C_2, m, ω, the residual of the fit q and the bubble filter b.

(d) We choose other parameters who will characterize, along with the 8 factors arising from the JLS fits, each window: the endpoint date as the distance in days from the first date in the dataset d_1, the distance of the critical time from the endpoint d_2, the size of the window s, the number of crashes in the time window nc, the realized return over the time window r, the average daily return dr and the daily volatility of returns $dvol$.

3. **Crashes and critical time labels**

(a) Define $ds = 50$ and the vector $CRASH = \{d \mid P_d = \max P_x, \forall x \in \left[d - ds, d + ds\right] \}$ of all crashes before t_{actual} and $CLOSE = 20$.

(b) For all critical times t_c before t_{actual}, we label them C or NC, with C meaning close to a crash if there exists an index i such that $|t_c - CRASH(i)| \leq CLOSE$.

(c) For all critical times after t_{actual}, we don't know their label so they are the ones we would like to predict.

4. **Critical time classification with Random Forests**

(a) Build the data matrix X where each row contains one observation and there are 15 columns corresponding to our 15 predictor variables.

(b) Build the vector of responses Y where the size is the number of observations, and each element corresponds to the category C or NC.

(c) Turn on the out-of-bag OOBPred feature and use the TreeBagger class to create an ensemble of M trees.

(d) Monitor the out-of-bag classification error with oobError and the mean classification margin with oobMeanMargin.

(e) Choose the number of trees M such that the classification error does not decrease anymore.

(f) Find the fraction of out-of-bag observations per number of grown trees.

(g) Make an out-of-bag prediction with OOBPred and use perfcurv to plot a performance curve showing the true positive rate versus the false positive rate.

(h) Use OOBPermutedVarDeltaErrorChoose and choose a cutoff value to find the indices of the important features.

(i) Grow trees on a reduced but most important set of features and verify that the performance is similar to the one with the full set of features.

(j) Perform the category prediction and compute the classification error on the testing set by using the data matrix $X_{testing}$, for which all observations correspond to the unlabeled critical times after t_{actual}, and the TreeBagger object previously trained.

5. **Confidence interval for critical times**

   (a) Keep only the critical times predicted as C and build confidence interval.

   (b) Build a confidence interval with all critical times irrespective of their labels.

   (c) Compare both confidence intervals and assess improvement in the bounds.

Rebound detection method

1. **Classification method to account for rebounds**

   (a) Define the vector $REBOUND = \{d \mid P_d = \min P_x, \forall x \in \left[d - dt, d + dt\right] \}$ of all rebounds before t_{actual} and $CLOSE = 30$.

   (b) For all critical times t_c before t_{actual}, we label them C or R or O, with C meaning close to a crash if there exists an index i such that $|t_c - CRASH(i)| \leq CLOSE$, R meaning close to a rebound if there exists an index i such that $|t_c - REBOUND(i)| \leq CLOSE$, and O meaning other (a change of regime, bubble deflation for example).

   (c) We can use Random Forests for the multiple classification in the same way we did for the binary classification.

Setting and discussion

We work with the daily time series of the S&P500 index spanning 20 years of data and starting in the 80's. We consider here the simple crash method, described in the previous subsection, and follow step by step the algorithm. We have 400 endpoint dates separated by 10 days each and 90 window widths separated by 10 days each. For each time window, our scanning range for critical times when we fit is one third of the window size in the future. We parallelized the code on the ETH Brutus cluster with parallel for loops in Matlab using the parfor instruction, since the fits can be computed independently, i.e we don't need the result of a previous iteration for the next one. We also the machine learning toolbox in Matlab, where our datasets can be given as arguments to the TreeBagger class (an implementation of "Random Forests"). An adequate number of cores should be used, considering factors such as the speedup, the loss in efficiency when increasing the number of cores and the walltime. In our case, using 128 cores was appropriate. Random Forests can also benefit from parallelization, which makes the classification faster.

We first test the robustness of the classification scheme with respect to the crash definition and what we consider as close critical times to a crash. The two parameters we vary are thus ds and $CLOSE$, which we respectively set to 50, 100 and 10, 30. ds represents the size of the time window we scan to pick our maximum stock price, whereas $CLOSE$ represents all points in time within a radius of $CLOSE$ days from the selected crash time. Fig. 22 shows the results for $ds = 50$ and $CLOSE = 10$. The more trees

we grow, the lower the classification error becomes, which is what one would expect. There is a convergence of the classification error after 100 trees. The classification error indeed is 34% at the start of the algorithm and decreases to 23%. The point where the classification error doesn't decrease anymore will always be the number of trees we choose to make predictions. We inspect whether the results stay robust with respect to changes in ds and $CLOSE$.

Fig. 23 shows the results for $ds = 100$ and $CLOSE = 30$. The classification error is 27% and decreases to 19%. The results do not stay robust with respect to changes in ds and $CLOSE$. Playing a little more with ds and $CLOSE$, we notice that when $CLOSE$ is higher, the classification error worsens; when ds is higher, it improves. We will justify this lack of robustness later on.

We now want to assess the feature relevance by launching the data mining method with only the Johansen-Ledoit-Sornette parameters and two measures that derive from them. Fig. 24 and Fig. 25 with the reduced features show an increase in the classification error from start to convergence compared to the previous Fig. 22 and Fig. 23. Indeed, we start with a 38% classification error and converge to a 25% error in Fig. 24 while it starts at 34% and converges to 23% in Fig. 22. This illustrates the added value of the price-related features.

We have assumed in these techniques that the classification cost is the same for both classes C and NC. We can actually enhance the performance of the method by assigning an unequal misclassification cost to the cost matrix. This situation typically arises when we care about a class more than another, when we want to penalize a wrong classification of a label type. The classification error curve with unequal classification cost in Fig. 26 and Fig. 27 is lower than the one with equal classification cost. For $ds = 50$ and $CLOSE = 10$, the classification error is 30% at the first tree and decreases to 14% compared to 34% decreasing to to 23%. The robustness issue with respect to the crash and crash proximity raises some doubts however.

We examine the critical times with more detail. To that purpose, we display a histogram of all critical times, i.e before the classification, piling up in our testing set as well as crashes and rebounds in respectively red and green. We get a very noisy signal in Fig. 28 which our data mining method should filter. The histogram of critical times labeled as close in Fig. 29 is still noisy. It does not look like a 14% classification error at all because this classification error is the overall classification error, so to investigate this further we must look at individual classification errors. We perform an operation on the confusion matrix to determine the individual classification errors. We get an 86% accuracy for NC label predictions and an 11% accuracy for C predictions. It makes very good predictions for the NC class and very bad ones for the C class.

The discrepancy in classification error between C and NC is explained by looking at the number of occurences of each class in the sample. The frequency of each class is 90% for NC and 10% for C. By increasing ds, we actually have less crashes displayed, so the C proportion becomes lower, and we thus achieve a better classification error. By increasing $CLOSE$, we collect more C labels so our classification error increases. The classification errors now also make sense, because if we have the NC occuring 9 times out of 10, then it also makes sense that we are going to correctly predict the NC 86% of the time if we guess randomly. On the contrary, if we the C class occurs 1 time out of 10, then by guessing randomly we could get an error of 14%. The somewhat low overall classification

error actually hides the bad results for the individual classification of critical times close to a crash, because of the class imbalance. We therefore have an absence of predictive power. We formalize this by looking at the receiver operating characteristic curve (ROC) and plotting the true positive rate against the false positive rate. The further the curve is from the 45 degree line, the more predictive power the data mining method has. Random guesses are therefore on the 45 degree line, which is what our method accomplishes here, as seen in Fig. 30.

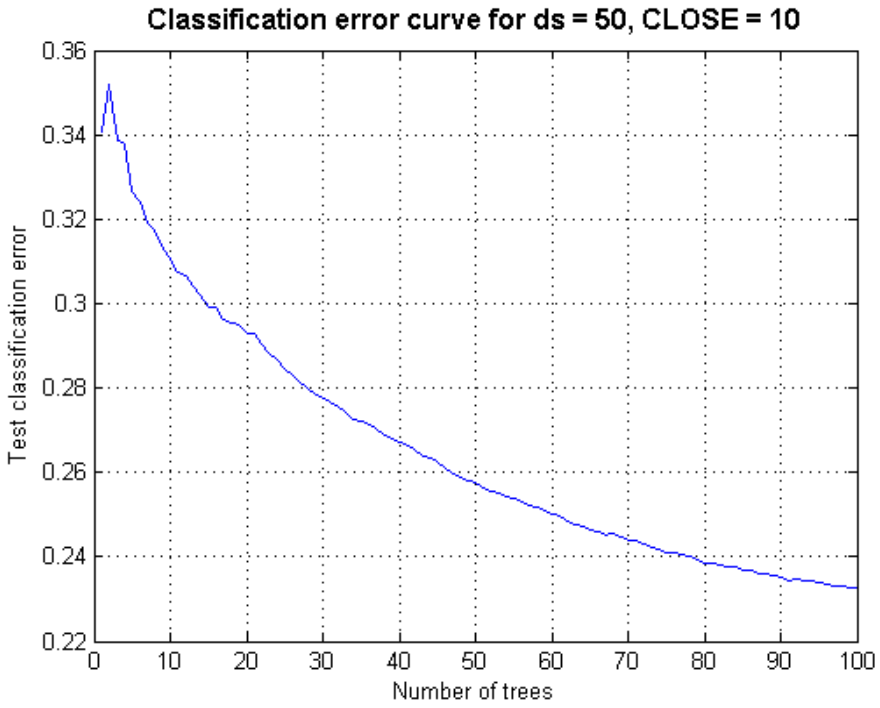

Figure 22: Evolution of the classification error with number of trees grown

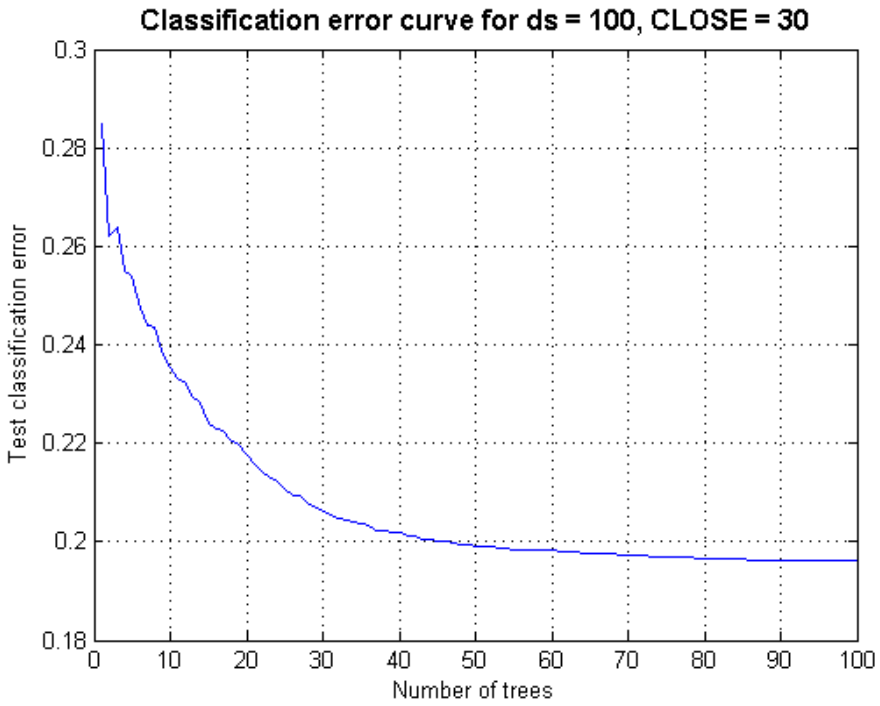

Figure 23: Evolution of the classification error with number of trees grown

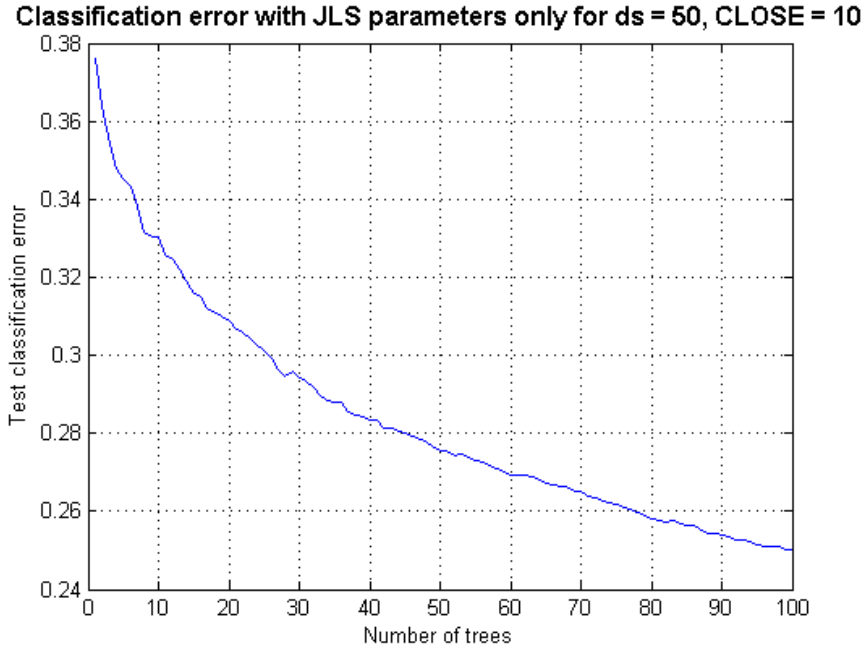

Figure 24: Evolution of the classification error with number of trees grown

Figure 25: Evolution of the classification error with number of trees grown

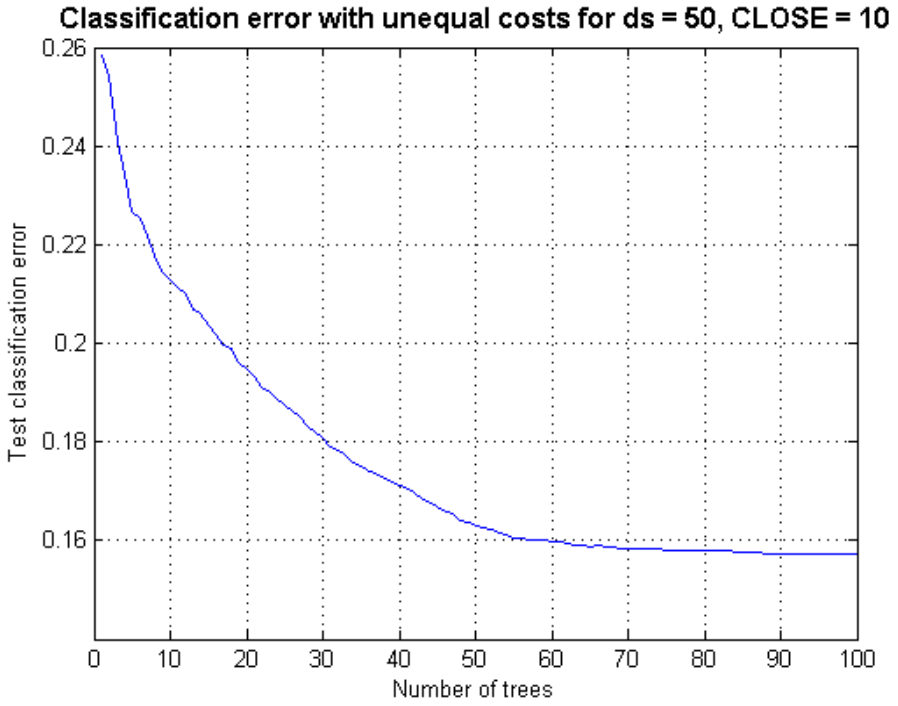

Figure 26: Evolution of the classification error with number of trees grown

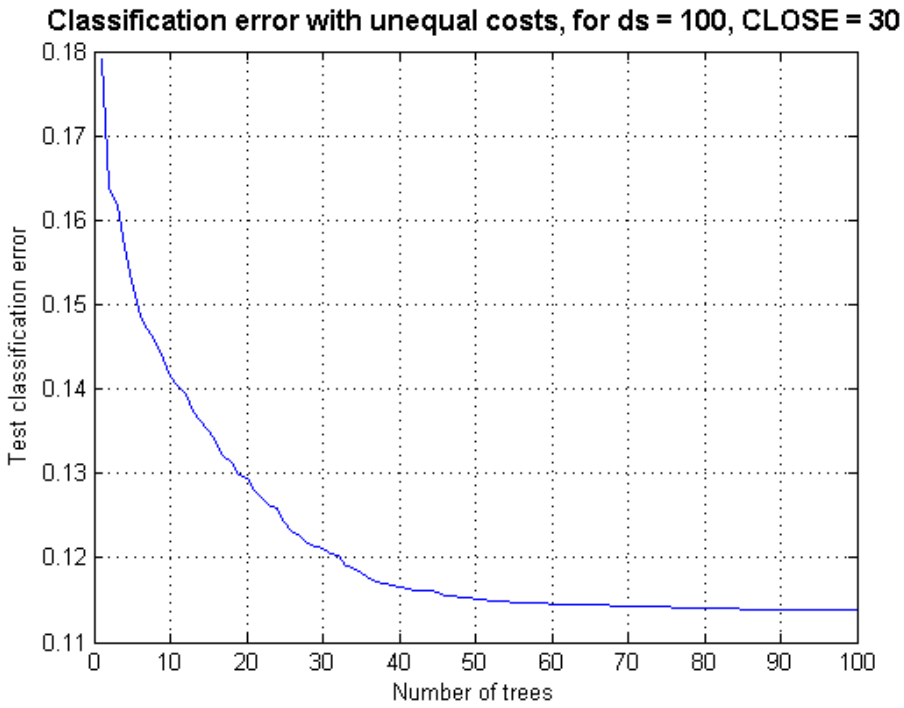

Figure 27: Evolution of the classification error with number of trees grown

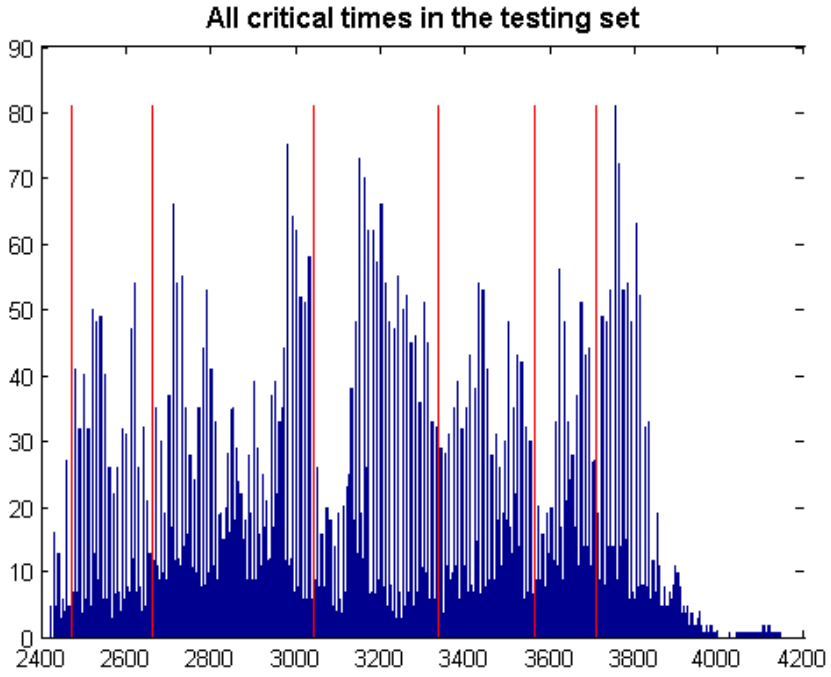

Figure 28: All critical times in the testing set

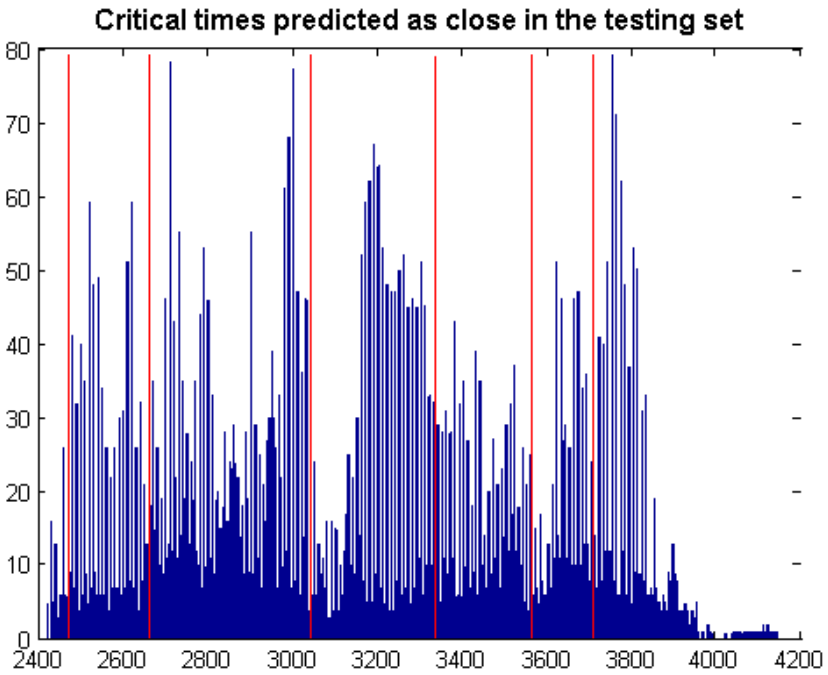

Figure 29: Critical times predicted as close to a crash

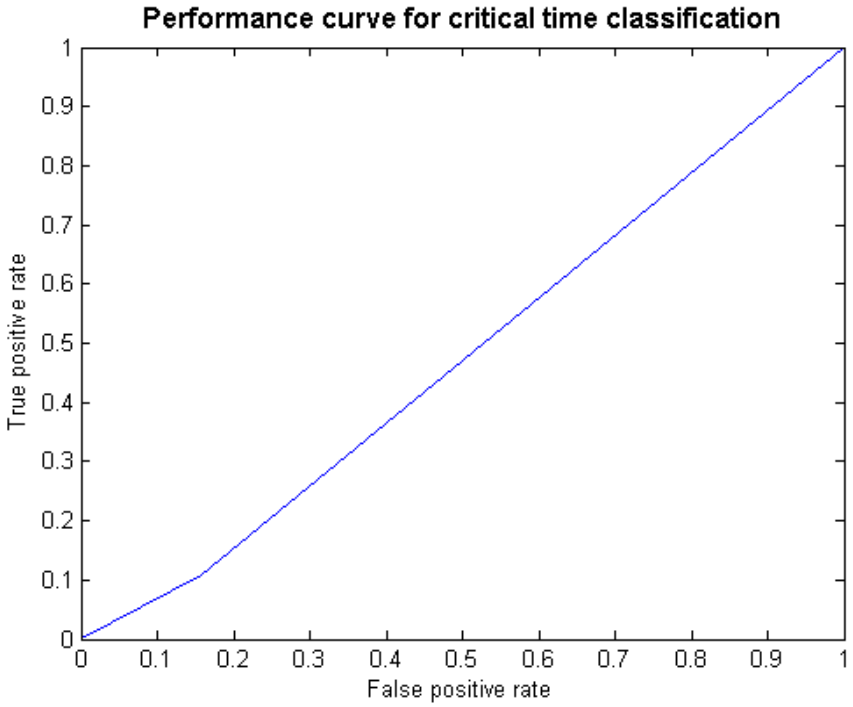

Figure 30: Receiver operating characteristic

We end this section by discussing whether the issue lies with the problem we are solving or the method we are using. First of all, we cannot distinguish between a critical time close to a crash and one that is not. Let's take an extreme example: if we look at the frontier between the C and NC class, there are no distinguishing features between critical times inside the crash proximity perimeter and those who are excluded. Even far from the crash proximity perimeter, many fits produced by the JLS model are actually good, and only the definition of the proximity excludes them from being close to the crash. Our scanning range for critical times is limited, we only go one third of the window size in the future, and if we are far from a crash timewise, then we will not hit the crash proximity perimeter. Moreover, the JLS model is valuable to diagnose endogenous bubbles, so an exogenous crash can in no way be forecast by this theory. However, the way we define our crashes include exogenous crashes as well as endogenous crashes, because we have defined crashes in an objective and quantitative way by taking the maximum over a time period. Taking exogenous crashes in the learning set spoils the prediction, they should therefore be removed. It is easy to get the big and loud endogenous crashes, but we do not know how to detect endogenous crashes on much smaller time scales. The *ds* parameter also influences how many crashes we actually have, so it will have an effect on how critical times are labeled. The absence of classification predictability is an interesting result per se and helped us identify many of the issues we should be dealing with. The next step is to exploit the whole machinery to perform a more meaningful and predictable task that the data mining method will be able to provide an answer for, at least better than random. We can use for example unsupervised learning methods to find patterns if we don't know how to label the data. Once reliable crash or rebound forecasts are produced, they can be used as predictive views for our existing risk model or to stress-test the risk model by taking the crash forecast as the investment horizon.

Chapter 2

Multivariate scenario analysis

In this second chapter, we extend the scenario analysis tools to a multidimensional setting, which is necessary when we work with portfolios of assets or multiple risk factors. Instead of single realizations represented by a scenario vector, we now work with joint realizations described by a scenario matrix. We can still stress risk factor specifications separately with the joint realization perspective, and we show two ways to achieve that: a tedious one constraining all other risk factor statistics, and an easy one using copulas. The novelty here is that the dependence structure can now be stressed: linear dependence such as correlation, or the nonlinear one such as copulas. Dependences can be modified while keeping the individual behaviour of the risk factors unchanged. Views on extreme co-movements are described, and we point out various multidimensional copulas that exhibit different lower and upper tail behaviours. In asset management, more precisely mean-variance optimization, the inputs are estimates of expected returns for each security and the covariance matrix, and views on these quantities are possible in multidimensional scenario analysis. We can therefore replace the risk model's expected return vector and covariance matrix by enhanced estimates and compare performances of the resulting trading strategies. Finally, liquidity risk can also be integrated to market risk in a portfolio, and we visualize the impact of liquidity factors on the P&L of several portfolios. We compute a risk measure associated to liquidity risk, the liquidity score, and discuss the changes of that risk measure in our experiments.

2.1 Fully flexible views on joint scenarios

2.1.1 Multivariate entropy pooling

Theory and general implementation

We represent, as in the univariate case, the joint distribution with scenarios and probabilities, the difference being that the scenarios are represented by a matrix and not a vector anymore, each row will be a joint realization of the underlying risk factors, each column will represent all realizations for a specific risk factor. Again, we invoke the entropy pooling method described in [Meu08], but now in a multivariate setting, to update the probabilities associated with the joint scenarios, and this time we can also process views on prices, portfolio returns or P&Ls, since we have the joint distribution of the risk

drivers.

Our views for the multivariate joint distribution $\boldsymbol{X}$ are expressed as

$$\boldsymbol{V} \equiv \boldsymbol{g}(\boldsymbol{X}) \tag{2.1.1}$$

where $\boldsymbol{g}$ is a $K-$dimensional random variable, meaning we have K views and every component of $\boldsymbol{g}$ is generally a non-linear function of the market.

We now deal with N risk factor, and we will generate J scenarios, where J is very large. The market model X is thus represented by an $J \times N$ vector $\mathcal{X}$ of simulations. The different columns will display the scenario probabilities for each risk factor, so the n-th column is a $J \times 1$ vector. The rows represent the joint scenarios. The scenario probabilities are also of dimension $J \times 1$. The K views are thus a $J \times K$ matrix, with the k-th view of the j-th scenario being:

$$\nu_{j,k} = g_k\big(X_{j,1},\, X_{j,2},\ldots\, X_{j,N}\big). \tag{2.1.2}$$

As for the univariate case, we don't simulate new scenarios but we use the same J scenarios with different probability masses $\tilde{\mathbf{p}}$. General views can be written as linear constraints on the updated probabilities:

$$\underline{\mathbf{a}} \leq \mathbf{A}\tilde{\mathbf{p}} \leq \overline{\mathbf{a}} \tag{2.1.3}$$

where $\mathbf{A}$, $\underline{\mathbf{a}}$ and $\overline{\mathbf{a}}$ are expressions of $\mathcal{X}$.

Generalized empirical distribution

We consider the $(N+1)-$tuplet $\{\bar{x}_{1,j} \ldots \bar{x}_{N,j}; \bar{p}_j\} = \{\bar{\boldsymbol{x}}_j; \bar{p}_j\}$ where the probabilities are given by $\bar{f}_{\boldsymbol{X}}$ and the scenarios do not necessarily occur with the same probabilities. We will use a convenient representation of the risk drivers distribution, which is called the generalized empirical distribution, as presented in [Meu10]:

$$\bar{f}_{\boldsymbol{X}}(\boldsymbol{x}) = \sum_{j=1}^{J} \bar{p}_j \delta^{\bar{\boldsymbol{x}}_j}(\boldsymbol{x}) \tag{2.1.4}$$

For $\boldsymbol{z} \in \mathbb{R}^N$, $\delta^{\boldsymbol{z}}$ is the Dirac function centered in the point $\boldsymbol{z}$ and verifying:

$$\int \delta^{\boldsymbol{z}}(\boldsymbol{x}) g(x) dx = g(z). \tag{2.1.5}$$

If we define an arbitrary function of the risk drivers $\boldsymbol{Y} \equiv y(\boldsymbol{X})$, there is a remarkable property of the generalized empirical distribution function, which is to compute trivially the distribution of Y:

$$\bar{f}_{\boldsymbol{Y}}(\boldsymbol{y}) = \sum_{j=1}^{J} \bar{p}_j \delta^{\bar{\boldsymbol{y}}_j}(\boldsymbol{y}) \tag{2.1.6}$$

where $\bar{\boldsymbol{y}}_j \equiv y(\bar{\boldsymbol{x}}_j)$. We can actually compute any statistics of any transformation of the risk drivers distribution such as the profit and loss for example.

The expected value of $\boldsymbol{Y}$ is:

$$\begin{aligned} \mu_{\boldsymbol{Y}} \equiv \mathbb{E}\left[y(X)\right] &\equiv \int \boldsymbol{y} \bar{f}_{\boldsymbol{Y}}(\boldsymbol{y}) d\boldsymbol{y} \\ &= \sum_{j=1}^{J} \bar{p}_j \int \boldsymbol{y} \delta^{\bar{\boldsymbol{y}}_j}(\boldsymbol{y}) d\boldsymbol{y} = \sum_{j=1}^{J} \bar{p}_j \bar{\boldsymbol{y}}_j \end{aligned} \tag{2.1.7}$$

The covariance between Y and Z is:

$$\Sigma_{Y,Z} \equiv Cov\Big(y(X), z(X)\Big) = \sum_{j=1}^{J} \bar{p}_j \bar{\boldsymbol{y}}_j \bar{\boldsymbol{z}}_j - \mu_{\boldsymbol{Y}} \mu_{\boldsymbol{Z}} \tag{2.1.8}$$

The Value-At-Risk with confidence level c of Y, i.e $\mathbb{P}(\boldsymbol{Y} \leq VaR_{\boldsymbol{Y}}) = 1 - c$, is easily computed as:

$$VaR_{\boldsymbol{Y}} = \max\{y_t\} \text{ such that } \sum_{y_t \leq VaR_{\boldsymbol{Y}}} \bar{p}_j \leq 1 - c. \tag{2.1.9}$$

Finally, the conditional Value-At-Risk CVaR with confidence level c, applied to the generalized empirical distribution, gives:

$$CVaR_{\boldsymbol{Y}} = \frac{\sum_{j=1}^{J} \bar{p}_j \bar{\boldsymbol{y}}_j}{\sum_{j=1}^{J} \bar{p}_j} \text{ such that } \bar{\boldsymbol{y}}_j \leq VaR_{\boldsymbol{Y}}. \tag{2.1.10}$$

Marginal views

In this multivariate setting, we can still process views on a marginal, univariate distribution, by isolating the column of interest and applying the algorithms implemented in the previous chapter. The other risk factors also appear in the constraints because the probability representation applies to a joint realization, we therefore need to ensure that the statistical properties of the other risk factors remain unaltered. We detail the procedure below, this is the first way to do it:

Views on the marginal distribution of the d-th risk factor

- Sort the scenarios $\mathcal{X}_d$ in

$$\mathcal{X} = \begin{pmatrix} \mathcal{X}_{1,1} & \cdots & \mathcal{X}_{k,1} & \cdots & \mathcal{X}_{J,1} \\ \mathcal{X}_{1,2} & \cdots & \mathcal{X}_{k,2} & \cdots & \mathcal{X}_{J,2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathcal{X}_{1,N} & \cdots & \mathcal{X}_{k,N} & \cdots & \mathcal{X}_{J,N} \end{pmatrix}$$

  in increasing order.

- The expectations of the individual risk factors $\mathcal{X}_i$ in the scenario-probability representation are written as:

$$\begin{aligned} \sum_{j=1}^{J} p_j \mathcal{X}_{j,1} &= \hat{m_1}, \\ \sum_{j=1}^{J} p_j \mathcal{X}_{j,2} &= \hat{m_2}, \\ &\vdots \\ \sum_{j=1}^{J} p_j \mathcal{X}_{j,d} &= \hat{m_d}, \\ &\vdots \\ \sum_{j=1}^{J} p_j \mathcal{X}_{j,N} &= \hat{m_N}. \end{aligned}$$

- The variances of the individual risk factors $\mathcal{X}_i$ in the scenario-probability representation are written as:

$$\begin{aligned} \sum_{j=1}^{J} p_j \left(\mathcal{X}_{j,1} - \hat{m_1} \right)^2 &= \hat{\sigma_1}^2, \\ \sum_{j=1}^{J} p_j \left(\mathcal{X}_{j,2} - \hat{m_2} \right)^2 &= \hat{\sigma_2}^2, \\ &\vdots \\ \sum_{j=1}^{J} p_j \left(\mathcal{X}_{j,d} - \hat{m_d} \right)^2 &= \hat{\sigma_d}^2, \\ &\vdots \\ \sum_{j=1}^{J} p_j \left(\mathcal{X}_{j,N} - \hat{m_N} \right)^2 &= \hat{\sigma_N}^2 \end{aligned}$$

- The value-at-risk of the individual risk factors $\mathcal{X}_i$ at the $\alpha\%$ level in the scenario-probability representation are written as:

$$\begin{aligned}
\sum_{i=1}^{I} p_i &\leq \alpha\%, \\
\sum_{i=1}^{I+1} -p_i &< -\alpha\%, \\
VaR_\alpha(X_1) &= \mathcal{X}_{1,I}, \\
VaR_\alpha(X_2) &= \mathcal{X}_{2,I}, \\
&\vdots \\
VaR_\alpha(X_d) &= \mathcal{X}_{d,I}, \\
&\vdots \\
VaR_\alpha(X_N) &= \mathcal{X}_{N,I}.
\end{aligned}$$

- The expected shortfalls of the individual risk factors $\mathcal{X}_i$ at the $\alpha\%$ level in the scenario-probability representation are written as:

$$\begin{aligned}
\sum_{i=1}^{s} p_i &< \alpha\%, \\
\sum_{i=1}^{s+1} -p_i &< -\alpha\%, \\
\sum_{i=1}^{s} p_i \mathcal{X}_{1,i} &= es_{\alpha,1}, \\
\sum_{i=1}^{s} p_i \mathcal{X}_{2,i} &= es_{\alpha,2}, \\
&\vdots \\
\sum_{i=1}^{s} p_i \mathcal{X}_{d,i} &= es_{\alpha,d}, \\
&\vdots \\
\sum_{i=1}^{s} p_i \mathcal{X}_{N,i} &= es_{\alpha,N}, \\
CVaR_\alpha(X_1) &= es_{\alpha,1}, \\
CVaR_\alpha(X_2) &= es_{\alpha,2}, \\
&\vdots \\
CVaR_\alpha(X_d) &= es_{\alpha,d}, \\
&\vdots \\
CVaR_\alpha(X_N) &= es_{\alpha,N}.
\end{aligned}$$

- Processing views on expectation $\tilde{m_d}$ and variance $\tilde{\sigma_d}^2$ amounts to choosing a new set of probability masses $\tilde{p}_j$ such as:

$$\begin{aligned}
\sum_{j=1}^{J} \tilde{p}_j \mathcal{X}_{j,1} &= \hat{m_1}, \\
\sum_{j=1}^{J} \tilde{p}_j \left(\mathcal{X}_{j,1} - \hat{m_1} \right)^2 &= \hat{\sigma_1}^2, \\
\sum_{j=1}^{J} \tilde{p}_j \mathcal{X}_{j,2} &= \hat{m_2}, \\
\sum_{j=1}^{J} \tilde{p}_j \left(\mathcal{X}_{j,2} - \hat{m_2} \right)^2 &= \hat{\sigma_2}^2, \\
&\vdots \\
\sum_{j=1}^{J} \tilde{p}_j \mathcal{X}_{j,d} &= \tilde{m_d}, \\
\sum_{j=1}^{J} \tilde{p}_j \left(\mathcal{X}_{j,d} - \hat{m_d} \right)^2 &= \tilde{\sigma_d}^2, \\
&\vdots \\
\sum_{j=1}^{J} \tilde{p}_j \mathcal{X}_{j,N} &= \hat{m_N} \\
\sum_{j=1}^{J} \tilde{p}_j \left(\mathcal{X}_{j,N} - \hat{m_N} \right)^2 &= \hat{\sigma_N}^2
\end{aligned}$$

- Processing a view on value-at-risk $\widetilde{VaR}_\alpha(X_d)$ amounts to choosing a new set of probability masses $\tilde{p}_j$ such as:

$$\begin{aligned}
\sum_{i=1}^{\tilde{I}} \tilde{p}_i &\leq \alpha\%, \\
\sum_{i=1}^{\tilde{I}+1} -\tilde{p}_i &< -\alpha\%, \\
\widetilde{VaR}_\alpha(X_1) &= \mathcal{X}_{1,I}, \\
\widetilde{VaR}_\alpha(X_2) &= \mathcal{X}_{2,I}, \\
&\vdots \\
\widetilde{VaR}_\alpha(X_d) &= \mathcal{X}_{d,\tilde{I}}, \\
&\vdots \\
\widetilde{VaR}_\alpha(X_N) &= \mathcal{X}_{N,I}.
\end{aligned}$$

- Processing a view on expected shortfall $\widetilde{CVaR}_\alpha(X)$ amounts to choosing a new set of probability masses $\tilde{p}_j$ such as:

$$\begin{aligned}
\sum_{i=1}^{\tilde{s}} \tilde{p}_i &< \alpha\%, \\
\sum_{i=1}^{\tilde{s}+1} -\tilde{p}_i &< -\alpha\%, \\
\widetilde{CVaR}_\alpha(X_1) &= es_{\alpha,1}, \\
\widetilde{CVaR}_\alpha(X_2) &= es_{\alpha,2}, \\
&\vdots \\
\widetilde{CVaR}_\alpha(X_d) &= \tilde{es}_{\alpha,d}, \\
&\vdots \\
\widetilde{CVaR}_\alpha(X_N) &= es_{\alpha,N}.
\end{aligned}$$

- We keep the same scenarios $\mathcal{X}$ and replace $\mathbf{p}$ with $\tilde{\mathbf{p}}$, the views on the d-th risk factor scenarios $\mathcal{X}_d$ are now satisfied.

As you can see, processing views on a statistical feature of a single risk factor is not complicated but we have to fix the statistic value for the other risk factors. If we do not require that, their values will change as well since we have a multivariate specification of the scenarios. Actually, in order to obtain an accurate solution, all the moments and quantiles of the other risk factors should be fixed and this is impossible to do in practice. A simpler and more elegant way to process views on marginal distributions is presented in the copula section and it will reveal the power of that framework.

2.1.2 Location and dispersion views: portfolio management

In the univariate case, we processed views on expectation and variance. In the multivariate case, we will consider views on an expectation vector and a covariance or correlation matrix, since the co-movements between the different risk factors have to be taken into account and can be altered, either manually by a qualitative judgment or quantitatively with a model output. We can also stress-test the location and the co-movements. We present the optimization problem we need to solve in order to process these views. As an application, we perform a statistical procedure from [LR10] yielding robust estimates of expected returns and covariance matrix in order to build optimal portfolios. We compute and compare the performance of a trading strategy using the posterior estimates with a trading strategy built upon the sample estimates. This is another example where views are delivered by a model, in this case the robust procedure. We can, in all generality, modify the distribution's location and dispersion features with better predictive estimates instead of those that have been estimated beforehand.

Expectation and correlation

Suppose we have a multivariate market model X represented by an $J \times N$ matrix $\mathcal{X}$ of simulations, where N are the number of risk factors and J the number of scenarios. We want to process views on all expectations and the correlation matrix between the risk factors. As in the first chapter, we have programmed a Matlab function incorporating the matrix form for the expectation and the correlation.

Expectation and correlation views

- Recognize that the expectations in the scenario-probability representation are written as:

$$\begin{aligned} \sum_{j=1}^{J} p_j \mathcal{X}_{j,1} &= \hat{m}_1, \\ \sum_{j=1}^{J} p_j \mathcal{X}_{j,2} &= \hat{m}_2, \\ &\vdots \\ \sum_{j=1}^{J} p_j \mathcal{X}_{j,N} &= \hat{m}_N. \end{aligned}$$

- The correlation in the scenario-probability representation is written as:

$$\frac{\sum_{j=1}^{J} p_j \mathcal{X}_{j,k} \mathcal{X}_{j,l} - \hat{m}_k \hat{m}_l}{\hat{\sigma}_k \hat{\sigma}_l} = \hat{\mathbb{C}}_{k,l} \text{ for all } k,\, l = 1,...N$$

- Processing views on expectations $\tilde{m}$ and corelations $\tilde{\sigma}$ amounts to choosing a new set of probability masses $\tilde{p}_j$ such as:

$$\begin{aligned} \sum_{j=1}^{J} \tilde{p}_j \mathcal{X}_{j,k} &= \tilde{m}_k \text{ for all } k = 1,...N, \\ \frac{\sum_{j=1}^{J} \tilde{p}_j \mathcal{X}_{j,k} \mathcal{X}_{j,l} - \hat{m}_k \hat{m}_l}{\hat{\sigma}_k \hat{\sigma}_l} &= \tilde{\mathbb{C}}_{k,l} \text{ for all } k,\, l = 1,...N. \end{aligned}$$

- We can write the views as a linear equality system on the posterior probability masses $\tilde{p}_j$

$$A\tilde{\mathbf{p}} = b$$

where

$$A = \begin{pmatrix} \mathcal{X}_{1,1} & \mathcal{X}_{2,1} & \cdots & \mathcal{X}_{J,1} \\ \mathcal{X}_{1,2} & \mathcal{X}_{2,2} & \cdots & \mathcal{X}_{J,2} \\ \vdots & \vdots & \vdots & \vdots \\ \mathcal{X}_{1,N} & \mathcal{X}_{2,N} & \cdots & \mathcal{X}_{J,N} \\ \mathcal{X}_{1,k}\mathcal{X}_{1,l} & \mathcal{X}_{2,k}\mathcal{X}_{2,l} & \cdots & \mathcal{X}_{J,k}\mathcal{X}_{J,l} \end{pmatrix}$$

for all k, $l = 1,...N$, and

$$b = \begin{pmatrix} \tilde{m}_1 \\ \tilde{m}_2 \\ \vdots \\ \tilde{m}_N \\ \hat{m}_k \hat{m}_l + \hat{\sigma}_k \hat{\sigma}_l \tilde{\mathbb{C}}_{k,l} \end{pmatrix}$$

for all k, $l = 1,...N$.

- We solve the relative entropy minimization problem:

$$\tilde{\mathbf{p}} = \underset{A\tilde{\mathbf{p}}=b}{\operatorname{argmin}} \sum_{j=1}^{J} \tilde{p}_j \big[\ln(\tilde{p}_j) - \ln(p_j) \big]$$

- We keep the same scenarios $\mathcal{X}$ and replace $\mathbf{p}$ with $\tilde{\mathbf{p}}$, the expectation and correlation views are now satisfied.

Application to robust asset allocation

A legitimate question is why we would like to have views on expectations and correlation. In the first chapter, where we performed univariate scenario analysis on expectation and variance, the goal was to stress-test and validate the numerical model with analytical expressions. In the multivariate setting, there is another application that comes to mind: robust asset allocation. Expected returns and covariance must be forecasted in a Markowitz framework, and it is often the case that models produce estimates that are unstable with respect to errors in the data. There are several robust methods for estimating expected returns and covariance, so once we determine our method of choice, we would like to replace the prior unreliable estimates by new ones. We therefore start with sample estimates of expected returns and covariance, and implement a modification of the Levy-Roll procedure described in [MAK11], that uses relative entropy as an optimization function instead of the weighted Euclidean norm in the original paper [LR10]. The basic idea of the Levy-Roll procedure is to take a reverse approach to what we usually do: instead of checking if variations of return parameters are market efficient, Levy and Roll take the reverse approach and constrain the market efficiency of return parameters and then search for the nearest sample parameters. Since we are in a probabilistic framework, it makes sense to use a measure of distance between distributions. We are not going to take the relative entropy between any distribution, but we are going to specifically consider the distance between two normal distributions. The two parameters describing the normal distribution are: μ, a $N \times 1$ vector of expectations; and Σ, a $N \times N$ symmetric and positive definite covariance matrix. We take a rescaled and shifted version of the relative entropy between two normal distributions as the optimization function, without altering the end result. This function can be written as:

$$\mathcal{E}\left(\mu, \Sigma | \underline{\mu}, \underline{\Sigma^2}\right) = Tr(\Sigma \underline{\Sigma}^{-1}) - \ln |\sigma^2 \underline{\Sigma}^{-1}| + (\mu - \underline{\mu})' \underline{\Sigma}^{-1} (\mu - \underline{\mu}) \tag{2.1.11}$$

In large dimensional markets, we impose a structure on the covariance matrix with a factor analysis model from [LM62]:

$$\Sigma \equiv bb' + \delta^2 \tag{2.1.12}$$

where b is a $N \times K$ matrix, $K << N$ and δ is a $N \times N$ diagonal matrix. We have a new parametrization $\theta = (\mu, b, \delta)$ which ranges in the full space $\Theta = \mathbb{R}^N \times \mathbb{R}^{N \times K} \times \mathbb{R}^N$ as prescribed in [MAK11], and should improve the statistical efficiency of the estimates. The posterior estimates of the new parametrization are obtained by solving

$$(\overline{\mu}, \overline{b}, \overline{\delta}) \equiv \underset{\mu,b,\delta \in \Upsilon}{\operatorname{argmin}} \mathcal{E}\left(\mu, bb' + \delta^2 | \underline{\mu}, \underline{\Sigma}\right) \tag{2.1.13}$$

The Matlab program for the following algorithm was realized without off-the-shelf code. The posterior estimates for the mean and the covariance can be obtained as follows:

Entropy-based Levy-Roll procedure

- Compute the empirical prior parameter $\underline{\mu}$ as the sample mean.
- Compute the empirical prior parameter $\underline{\Sigma}$ as the sample covariance.
- Decompose the sample covariance $\underline{\Sigma}$ with the parametrization

$$\Sigma \equiv bb' + \delta^2.$$

- Compute the market price of risk γ

$$\gamma = \frac{R_M - r_f}{\sigma_M^2}$$

  where R_M is the market return, σ_M^2 the market squared volatility and r_f the monthly risk-free rate for the considered period. Also called Sharpe Ratio, see [Sha98], γ is the return on top of the risk-free rate that the market demands as a compensation for taking risk, represented here by the volatility.

- Find the posterior estimates $\overline{\mu}$, $\overline{b}$ and $\overline{\delta}$

$$\begin{aligned} (\overline{\mu}, \overline{b}, \overline{\delta}) &\equiv \underset{\mu,b,\delta \in \Upsilon}{\operatorname{argmin}} \mathcal{E}\left(\mu, bb' + \delta^2 | \underline{\mu}, \underline{\Sigma}\right) \\ \Upsilon &: \mu - \gamma(bb' + \delta^2) w_{MC} \equiv 0 \end{aligned}$$

  and w_{MC} is the market capitalization vector.

- The robust estimates $\overline{\mu}$ and $\overline{\Sigma}$ are finally used as views for our prior risk model $\mathcal{X}$:

$$\begin{aligned} \sum_{j=1}^{J} \tilde{p}_j \mathcal{X}_{j,k} &= \overline{\mu}_k \text{ for all } k = 1,...N, \\ \frac{\sum_{j=1}^{J} \tilde{p}_j \mathcal{X}_{j,k} \mathcal{X}_{j,l} - \hat{m}_k \hat{m}_l}{\hat{\sigma}_k \hat{\sigma}_l} &= \overline{\Sigma}_{k,l} \text{ for all } k,\, l = 1,...N. \end{aligned}$$

- Update the risk model probabilities by solving again a relative entropy minimization problem with the previous views: replace $\mathbf{p}$ with $\tilde{\mathbf{p}}$.

We will do a portfolio optimization with 12 diverse stocks, small and large caps: American Express Company, AT&T, Coca Cola, General Electric, Johnson and Johnson, JP Morgan, Microsoft, Boeing, Consolidated Edison, Exxon Mobil, Honeywell International and Forest Laboratories. We consider a period from 1977 to 2007, and compute the monthly log-returns of each stock. The data was taken from Yahoo Finance and exported into csv files. The first five years of the sample are used to estimate the sample mean and the sample covariance with a 3-factor decomposition. We use the entropy-based Levy-Roll procedure to get the adjusted parameters. We proceed as in [NMSW14] to build ex-ante portfolios and ex-post portfolios. We dynamically rollover the estimation and portfolio allocation, plot their returns and compare the portfolios produced by the sample-based and the adjusted parameters. The following portfolios come from Markowitz's work in [Mar52]. We first consider the minimum standard deviation portfolio MVP, which is the portfolio on the efficient frontier that minimizes the standard deviation without specifying any target return, i.e solving the optimization problem

$$\overrightarrow{W}_{MVP} = \min_{w} w^T \Sigma w \tag{2.1.14}$$

subject to

$$w^T \mathbf{1} = 1. \tag{2.1.15}$$

Second, we consider the tangent portfolio TGT. In this case, the investor chooses between a risk-free asset and risky assets, and this yields the capital market line, which is obtained by solving

$$\overrightarrow{W}_{CML} = \min_{w} w^T \Sigma w \tag{2.1.16}$$

subject to

$$\mu_0 = w^T \mu + \left(1 - w^T \mathbf{1}\right) \tag{2.1.17}$$

The tangential portfolio corresponds to the weights $\overrightarrow{W}_{TGT} = \overrightarrow{W}_{CML}$ such that $\overrightarrow{W}_{CML}^T \times \mathbf{1} = 1$, since we only hold risky assets. The asset weights W and ex-ante monthly returns R, and the weights $\overrightarrow{W}_{MVP}$, $\overrightarrow{W}_{TGT}$ are given by:

$$\overrightarrow{W}_{MVP} = \frac{\Sigma^{-1}\overrightarrow{1}}{\overrightarrow{1}^T\Sigma^{-1}\overrightarrow{1}}, \tag{2.1.18}$$

$$\overrightarrow{W}_{TGT} = \frac{\Sigma^{-1}(\mu - r_f\overrightarrow{1})}{\overrightarrow{1}^T\Sigma^{-1}(\mu - r_f\overrightarrow{1})}, \tag{2.1.19}$$

$$R_{MVP}(T) = \overrightarrow{W}_{MVP}(T-1)\overrightarrow{R}(T), \tag{2.1.20}$$

$$R_{TGT}(T) = \overrightarrow{W}_{TGT}(T-1)\overrightarrow{R}(T), \tag{2.1.21}$$

where $\overrightarrow{R}(T)$ are the realized monthly returns at the end of month T, $\overrightarrow{W}(T-1)$ the asset weights computed at the end of month $T-1$. As in [NMSW14], if the return in that period is smaller than the risk-free rate that month, we invest our capital in the risk-free rate.

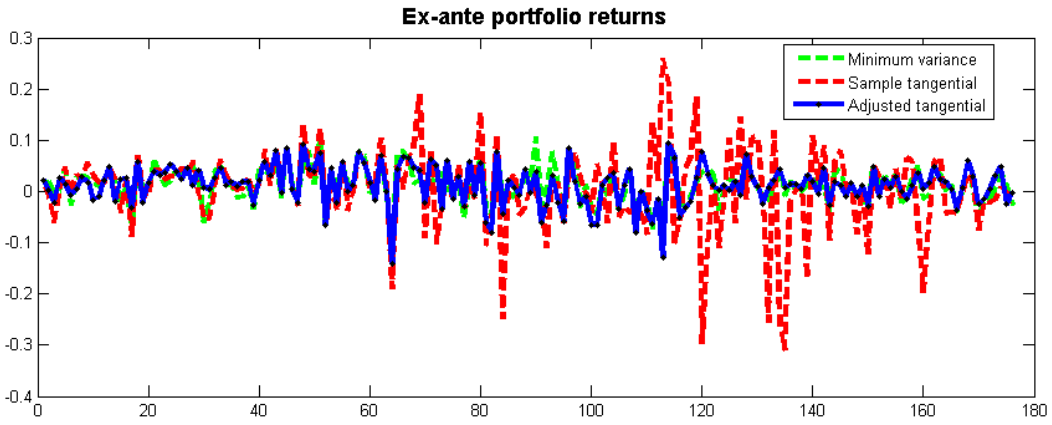

Figure 31: Forecasted portfolio returns with sample-based and enhanced estimates

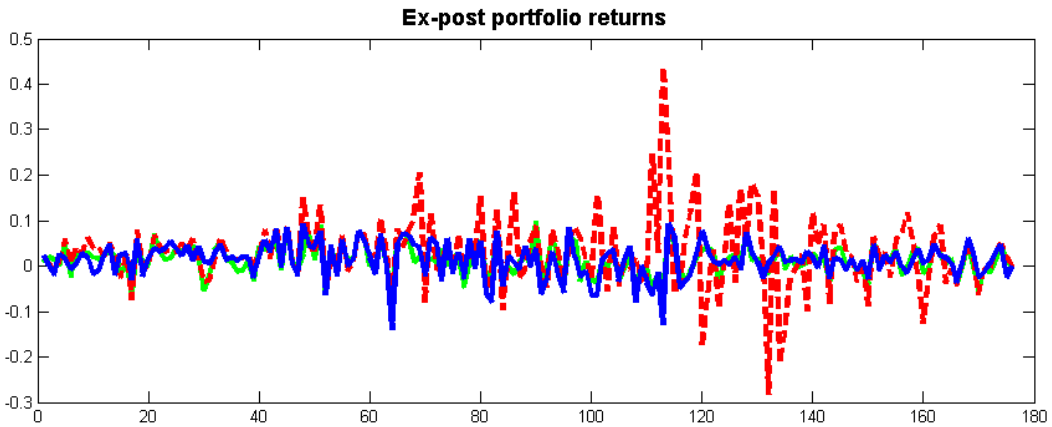

Figure 32: Realized portfolios returns with sample-based and enhanced estimates

	Return forecasts			
	Sample MVP	Enhanced MVP	Sample TGT	Enhanced TGT
Min(%)	-13.69	-13.69	-Â¨30.98	-14.10
Max(%)	10.66	10.59	25.98	9.44
Std (%)	3.62	3.62	8.31	3.89

Table 33: Ex-ante returns

	Return forecasts			
	Sample MVP	Enhanced MVP	Sample TGT	Enhanced TGT
Min (%)	-7.91	-7.89	-28.21	-14.07
Max (%)	9.98	9.98	-43.73	38.97
Std. (%)	3.2	-0.8	8.03	3.89

Table 34: Ex-post returns

We can see from Table 33 and Table 34 that the behaviour of the asset allocation with sample parameters is less stable since the range is much higher and the returns exhibit a higher volatility, whereas the asset allocation with the adjusted parameters shows an allocation with more consistent returns, as shown in Fig. 31 and Fig. 32. The tangential portfolio with sample-based parameters indeed displays a higher frequency of large, positive returns, though the negative returns are also more extreme than with the adjusted parameters and can threaten the asset manager's solvency. On a risk-adjusted basis, the enhanced estimates deliver a better performance.

2.2 Nonlinear dependence and extreme co-movements

Portfolio risk measures the uncertainty in the returns distribution of the portfolio or the P&L uncertainty over a time window. In the previous section, we have considered views on correlation or covariance, which are very commonly used but make the assumption that the returns are i.i.d and come from elliptical distributions. Since this is not satisfied in most cases, we need to work with the entire joint distribution. We do not have that much flexibility and choices with multivariate distributions, we will thus use copulas, a mathematical construct that allows us to form a joint distribution by specifiying the marginals and bind them with a dependence structure that varies from a copula to another. We can choose different marginals for each asset and different copulas. We present the theory of how we can extract the marginals and the copula from a joint distribution, but also how to combine the marginals and the copula into a joint distribution. We then review the algorithm which performs these separation and combination tasks very efficiently, the Copula-Marginal-Algorithm in [Meu11a]. We have plenty of copulas: the multivariate gaussian or Student copula, or the Archimedean family, which exhibit the dependence characteristics that have been observed in financial time series, such as lower tail dependence in equity markets for Clayton copulas. We are going to explain how we simulate them, but the problem with them is that they don't allow for different dependency models between pairs of variables. One parameter is definitely not enough to model the dependence of a high-dimension portfolio distribution, so we will study and

simulate vine copulas in [ACFB09]. Vine copulas allow us to decompose a multivariate distribution with a cascade of pair-copulae. Finally, we will introduce tail-dependence views and alter extreme co-movement probabilities at different threshold levels.

2.2.1 Copulas: separation and combination

Theoretical separation and combination

We review here the results for copula theory, following [Meu11a] and completing the derivations with [EFM05]. We start with a random variable X described by its probability density function f_X or equivalently its cumulative distribution function F_X. If we compose the random variable X with its own cumulative distribution function, we obtain a transformed random variable which is called the grade of X:

$$U \equiv F_X(X). \tag{2.2.1}$$

The grade is uniformly distributed on the unit interval regardless of what the distribution of X is, as we prove here:

$$\begin{aligned} F_U(u) &\equiv \mathbb{P}(U \leq u) = \mathbb{P}(F_X(X) \leq u) \\ &= \mathbb{P}\left(X \leq F_X^{-1}(u)\right) = F_X\left(F_X^{-1}(u)\right) = u. \end{aligned}$$

Reciprocally, if we compose a uniform random variable U with the inverse cumulative distribution F_X^{-1}, we obtain a random variable $X \equiv F_X^{-1}(U)$ with density f_X. So if we start with an arbitrary target distribution and a uniformly distributed random variable, we can transform the uniform into a new random variable with the desired target distribution.

In the multivariate case, if we have an N-dimensional random $\boldsymbol{X}$, we can compute the N marginal distributions $X_n \equiv f_{X_n}$ as follows:

$$f_{X_n}(x_n) = \int_{\mathbb{R}^{N-1}} f_{\boldsymbol{X}}(x_1, ..., x_N) dx_1 \dots dx_{n-1} dx_{n+1} \dots dx_N. \tag{2.2.2}$$

We then compute the grades of each marginal distribution function:

$$U_n \equiv F_{X_n}(x_n) \sim U_{\left[0,\ 1\right]}. \tag{2.2.3}$$

Collecting all these uniforms in a vector $U \equiv (U_1, ..., U_N)'$, the marginals are uniforms but the joint distribution is not uniform on the $N-$dimensional cube, and we call this joint distribution $f_{\boldsymbol{U}}$ the copula of the distribution $f_{\boldsymbol{X}}$.

We will show now that to get the joint distribution $f_{\boldsymbol{X}}$, we need the copula and the marginals.

$$
\begin{aligned}
F_{\boldsymbol{U}}(u) &\equiv \mathbb{P}\left(U_1 \leq u_1,\, ...,\, U_N \leq u_N\right) \\
&= \mathbb{P}\left(F_{X_1}(X_1) \leq u_1,\, ...,\, F_{X_N}(X_N) \leq u_N\right) \\
&= \mathbb{P}\left(X_1 \leq F_{X_1}^{-1}(u_1),\, ...,\, X_N \leq F_{X_N}^{-1}(u_N)\right) \\
&= F_{\boldsymbol{X}}\left(F_{X_1}^{-1}(u_1),\, ...,\, F_{X_N}^{-1}(u_N)\right) \qquad (2.2.4)
\end{aligned}
$$

Differentiating the expression, we get:

$$
\begin{aligned}
f_{\boldsymbol{U}}(u_1,\, ...,\, u_N) &= \partial u_1 \dots {u_N}^N F_{\boldsymbol{U}}(u_1,\, ...,\, u_N) = \partial u_1 \dots {u_N}^N F_{\boldsymbol{X}}\left(F_{X_1}^{-1}(u_1),\, ...,\, F_{X_N}^{-1}(u_N)\right) \\
&= \partial x_1 \dots {x_N}^N F_{\boldsymbol{X}}\left(F_{X_1}^{-1}(u_1),\, ...,\, F_{X_N}^{-1}(u_N)\right) d_{u_1} F_{X_1}^{-1}(u_1) \dots d_{u_N} F_{X_N}^{-1}(u_N) \\
&= \frac{\partial x_1 \dots {x_N}^N F_{\boldsymbol{X}}\left(F_{X_1}^{-1}(u_1),\, ...,\, F_{X_N}^{-1}(u_N)\right)}{d_{x_1} F_{X_1}\left(F_{X_1}^{-1}(u_1)\right) \dots d_{x_N} F_{X_N}\left(F_{X_N}^{-1}(u_N)\right)} \\
&= \frac{f_{\boldsymbol{X}}\left(F_{X_1}^{-1}(u_1),\, ...,\, F_{X_N}^{-1}(u_N)\right)}{f_{X_1}\left(F_{X_1}^{-1}(u_1)\right) \dots f_{X_N}\left(F_{X_N}^{-1}(u_N)\right)} \qquad (2.2.5)
\end{aligned}
$$

We call this result Sklar's theorem, see [Skl59], and it links the joint distribution $f_{\boldsymbol{X}}$, the copula $f_{\boldsymbol{U}}$ and the marginals f_{X_n}:

$$
f_{\boldsymbol{X}}\left(F_{X_1}^{-1}(u_1),\, ...,\, F_{X_N}^{-1}(u_N)\right) = f_{\boldsymbol{U}}(u_1,\, ...,\, u_N) \times f_{X_1}\left(F_{X_1}^{-1}(u_1)\right) \dots f_{X_N}\left(F_{X_N}^{-1}(u_N)\right)
$$

There are two distinct processes in a copula-marginal decomposition: separation and combination. The separation process $\mathcal{S}$ transforms an arbitrary distribution $f_{\boldsymbol{X}}$ into its marginals f_{X_N} and its copula $f_{\boldsymbol{U}}$, where U_n are the grades:

$$
\mathcal{S} : \left(X_1 \dots X_N\right) \sim f_{\boldsymbol{X}} \quad \mapsto \quad \left\{ \begin{array}{l} f_{X_1},\, ...,\, f_{X_N} \\ \left(U_1 \dots U_N\right) \sim f_{\boldsymbol{U}} \end{array} \right. \qquad (2.2.6)
$$

The separation process can be reverted by composing each grade U_n with the respective inverse cumulative distribution function $F_{X_n}^{-1}$, we obtain a random vector $\boldsymbol{X} \equiv \left(X_1 \dots X_N\right)$ whose joint distribution is exactly the original one $f_{\boldsymbol{X}}$. But we don't actually need to revert to the original distribution, but can glue copulas with any arbitrary marginal distributions, so if we start with with an arbitrary copula $\bar{f}_{\boldsymbol{U}}$ (grades U_i uniformly distributed and joint distribution structure) and arbitrary marginal distributions $\bar{f}_{\boldsymbol{X_n}}$, we can compute the marginal cumulative distributions and their inverses, which we can then

compose with the respective grades to get an $N-$dimensional random vector $\boldsymbol{X}$ with copula $\bar{f}_{\boldsymbol{U}}$ and marginals $\bar{f}_{\boldsymbol{X_n}}$.

The combination process $\mathcal{C}$ sticks arbitrary marginals $\bar{f}_{\boldsymbol{X_n}}$ and an arbitrary copula $\bar{f}_{\boldsymbol{U}}$ into a new joint distribution $\bar{f}_{\boldsymbol{X}}$:

$$\mathcal{C} \quad : \quad \left.\begin{array}{l} f_{X_1},\, ...,\, f_{X_N} \\ \left(U_1 \dots U_N\right) \sim f_{\boldsymbol{U}} \end{array}\right\} \mapsto \left(X_1 \dots X_N\right) \sim \bar{f}_{\boldsymbol{X}},\; X_n \equiv \bar{F}^{-1}_{X_n}(U_n) \tag{2.2.7}$$

To conclude this section, we have to note that practical applications all rely on numerical techniques, using Monte Carlo scenarios for instance. The separation and combination processes can be computationally challenging because we have to compute an inverse, perform univariate and multivariate integrations. Reliance on parametric copulas such as the Archimedean family to avoid such hurdles have been proposed, but even for large dimensions, it becomes tough to compute grade scenarios.

Numerical algorithm for copula separation and combination

Meucci proposes an algorithm to circumvent the previous issues. This scenario-based algorithm is called the Copula-Marginal Algorithm and presented in [Meu11a]. The scenario representation allows this algorithm to broaden the scope previously limited to a small set of parametric copulas. It is flexible because it does not force us to deliver the quantile function, potentially difficult to compute. Finally, Meucci claims it is computationally efficient in large dimensions compared to intensive parametric copula calibrations. The Copula-Marginal-Algorithm, in its full generality, displays the separation and combination processes. We can start with individual risk factors, described by marginal distributions and then glue a copula representing their tail behaviour to get a multivariate distribution: this is the combination part. We could also start with a historical multivariate distribution working with historical simulation only, and extract the dependence and the marginals: this is the separation part. This historical dependence could then be used along with other marginal models to be glued into a new multivariate distribution. We can therefore assemble and disassemble these bricks together, and reassemble disassembled bricks with a new glue.

Let us start with the separation $\mathcal{S}$, where we have a multivariate joint distribution $F_{\boldsymbol{X}}$ represented by scenario-probability vectors $\{x_{1,j} \dots x_{N,j}; p_j\}$ for $j = 1, \dots, J$. We compute the grade scenarios $u_{n,j}$ as the probability-weighted empirical grades

$$u_{n,j} \equiv \sum_{i=1}^{J} p_i 1_{x_{n,i} \leq x_{n,j}} \qquad n = 1, \dots, N; \qquad j = 1, \dots, J. \tag{2.2.8}$$

We can now separate the copula and the marginals from the joint distribution $F_{\boldsymbol{X}}$. The copula $F_{\boldsymbol{U}}$ is given by the joint distribution of the grades, so we represent it by $\{u_{1,j} \dots u_{N,j}; p_j\}$ for $j = 1, \dots, J$. For each scenario pair $\{x_{n,j},\, u_{n,j}\}$, we can inter or extrapolate those values with a function $I_{\{x_{n,j},\, u_{n,j}\}}$, which will be the marginal distribution of the n-th variable

$$F_{X_n}(x) \equiv I_{\{x_{n,j},\, u_{n,j}\}}(x), \qquad n = 1, \dots, N; \tag{2.2.9}$$

To resume the separation with a mapping:

$$\mathcal{S}_{CMA} : \{x_{1,j} \dots x_{N,j}; p_j\} \sim F_{\boldsymbol{X}} \quad \mapsto \quad \begin{cases} I_{\{x_{1,j},\ u_{1,j}\}},\ \dots,\ I_{\{x_{N,j},\ u_{N,j}\}} \\ \{u_{1,j} \dots u_{N,j}; p_j\} \sim F_{\boldsymbol{U}} \end{cases} \tag{2.2.10}$$

For the combination step, we start with any copula obtained with the separation step given by the specification $\{\bar{u}_{1,j} \dots \bar{u}_{N,j}; \bar{p}_j\}$ and marginal distributions $\bar{F}_{X_n}$. For each n we build the following grid $\{\tilde{x}_{n,k},\ \tilde{u}_{n,k}\}_{k=1,\dots,K_n}$ where $\tilde{u}_{n,k} \equiv \bar{F}_{X_n}(\tilde{x}_{n,k})$. We then map each grade scenario $\bar{u}_{n,j}$ to the scenarios $\bar{x}_{n,j}$ by interpolating the copula scenarios on the grid. The interpolating function I avoids the computation of the quantile function $\bar{F}^{-1}_{X_n}$.

$$\bar{x}_{n,j} \equiv I_{\{\tilde{x}_{n,k},\ \tilde{u}_{n,k}\}}(\bar{u}_{n,j}), \qquad n = 1, \dots, N; \qquad j = 1, \dots, J. \tag{2.2.11}$$

To resume the combination with a mapping:

$$\mathcal{C} \quad : \quad \left.\begin{array}{l} \bar{F}_{X_1},\ \dots,\ \bar{F}_{X_N} \\ \{\bar{u}_{1,j} \dots \bar{u}_{N,j}; \bar{p}_j\} \sim \bar{F}_{\boldsymbol{U}} \end{array}\right\} \mapsto \{\bar{x}_{1,j} \dots \bar{x}_{N,j}; \bar{p}_j\} \sim \bar{F}_{\boldsymbol{X}}. \tag{2.2.12}$$

In our structured setting, our input are the univariate distributions X_n of the risk drivers, and we use the combination step $\mathcal{C}$ with a suitable copula to obtain the joint distribution of the risk drivers.

Marginal views revisited

Previously, we had an encumbering algorithm to process views on a univariate distribution of a risk factor when we start with a joint scenario representation. We show here a simpler way to do it with the separation and combination algorithms:

Views on the marginal distribution of the d-th risk factor

- Sort the scenarios $\mathcal{X}_d$ in

$$\mathcal{X} = \begin{pmatrix} \mathcal{X}_{1,1} & \cdots & \mathcal{X}_{k,1} & \cdots & \mathcal{X}_{J,1} \\ \mathcal{X}_{1,2} & \cdots & \mathcal{X}_{k,2} & \cdots & \mathcal{X}_{J,2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathcal{X}_{1,N} & \cdots & \mathcal{X}_{k,N} & \cdots & \mathcal{X}_{J,N} \end{pmatrix}$$

  in increasing order.

- Apply the separation process from the Copula-Marginal Algorithm to extract the copula and the marginal distributions for each risk factor.

- From now on, we only work with the scenarios extracted from the marginal distribution of the d-th risk factor.

- Processing views on expectation $\tilde{m}_d$ and variance $\tilde{\sigma_d}^2$ amounts to choosing a new set of probability masses $\tilde{p}_j$ such as:

$$\begin{aligned} \sum_{j=1}^{J} \tilde{p}_j \mathcal{X}_{j,d} &= \tilde{m}_d, \\ \sum_{j=1}^{J} \tilde{p}_j \left(\mathcal{X}_{j,d} - \hat{m_d} \right)^2 &= \tilde{\sigma_d}^2. \end{aligned}$$

- Processing a view on value-at-risk $\widetilde{VaR}_\alpha(X_d)$ amounts to choosing a new set of probability masses $\tilde{p}_j$ such as:

$$\begin{aligned} \sum_{i=1}^{\tilde{I}} \tilde{p_i} &\leq \alpha\%, \\ \sum_{i=1}^{\tilde{I}+1} -\tilde{p_i} &< -\alpha\%, \\ \widetilde{VaR}_\alpha(X_d) &= \mathcal{X}_{d,\tilde{I}}. \end{aligned}$$

- Processing a view on expected shortfall $\widetilde{CVaR}_\alpha(X)$ amounts to choosing a new set of probability masses $\tilde{p}_j$ such as:

$$\begin{aligned} \sum_{i=1}^{\tilde{s}} \tilde{p_i} &< \alpha\%, \\ \sum_{i=1}^{\tilde{s}+1} -\tilde{p_i} &< -\alpha\%, \\ \widetilde{CVaR}_\alpha(X_d) &= \tilde{es}_{\alpha,d}. \end{aligned}$$

- We changed the specification of the marginal distribution of the d-th risk factor, so we keep the same scenarios and replace $\mathbf{p}$ with $\tilde{\mathbf{p}}$, the marginal views on the d-th risk factor scenarios $\mathcal{X}_d$ are now satisfied.

- Use the combination process from the Copula-Marginal Algorithm to glue the copula and all marginal distributions for each risk factor. All the marginals except the d-th one are the same as previously.

- We obtain a multivariate representation that satisfies the views on the d-th risk factor.

If we compare the first algorithm in the marginal views subsection and this algorithm, the latter solution is more elegant and straightforward and doesn't require ad-hoc manipulations on the other risk factors. Moreover, it fixes the issue we had with the former algorithm, i.e constraining all the statistics of the other risk factor dynamics. Only the code for the Copula-Marginal algorithm was provided by Meucci, the rest of the procedure was programmed by the author of this report.

2.2.2 Multidimensional copula simulations

Since we can generate as many scenarios as we like from the copula and the margins, we will now turn to some specific copulas and especially how we can simulate them, since simulated values will serve as inputs to Meucci's Copula-Marginal algorithm. We first present the standard procedures to simulate the following multivariate copulas: the gaussian, the Student and the Clayton copula, see [EFM05] for their definition and properties.

Gaussian copula simulation

1. Simulate N scenarios $\boldsymbol{x}$ coming from the multivariate normal distribution $\mathcal{N}_N\left(\mathbf{0}, \boldsymbol{R}\right)$.
2. Set $\boldsymbol{u} = \Phi(\boldsymbol{x})$ where Φ is the standard normal cumulative distribution function.
3. $\boldsymbol{u}$ are N realizations of the gaussian copula with linear correlation parameter R.

Student copula simulation

1. Simulate N scenarios $\boldsymbol{x}$ coming from the multivariate Student distribution $t_N\left(\mathbf{0}, \boldsymbol{R}, \nu\right)$.
2. Set $\boldsymbol{u} = t_\nu(\boldsymbol{x})$ where Φ is the standard normal cumulative distribution function.
3. $\boldsymbol{u}$ are N realizations of the Student copula with linear correlation parameter R and degrees of freedom ν.

Clayton copula simulation

1. Simulate a Gamma realization $x \sim Ga\left(\frac{1}{\delta},\ 1\right)$.
2. Simulate N independent and identically distributed scenarios $\boldsymbol{v}$ coming from the uniform distribution $U\left(0,\ 1\right)$.
3. Set $\boldsymbol{u} = \left((1 - \frac{\log(v_i)}{x})^{\frac{-1}{\delta}}\right)$ for $i = 1, ...N$.
4. $\boldsymbol{u}$ are N realizations of the Clayton copula with parameter δ.

However, these standard multivariate copulas have one parameter governing the dependence. If we consider all different pairs of variables, we have the same dependency model. However, this is not necessarily so, and even more when we have different asset classes. This is why we consider a class of copula models using a pair-copula construction, they are called vine copulas and have been developed by Aas et al. in [ACFB09]. The very basic idea idea of vine copulas is that we factorize the whole multivariate distribution (for example a portfolio of thousand assets) with pairs of assets, we then specify a base copula, it can be Gaussian, Clayton, Gumbel, etc, and finally obtain the model parameters for

each pair of assets (which are going to be different for each pair) with statistical estimation. Since we always have some sort of tail-dependence, we are only going to simulate vine copulas that take bivariate Student and bivariate Clayton copulas as building blocks.

Canonical vine copula, see [ACFB09]

We follow Aas' exposition but we only bring up the main points we need in order to simulate canonical vine copulas, and give more details than the paper when needed. Aas starts factorising the joint density $f(x_1, x_2, ...x_N)$ as

$$f(x_1, x_2, ...x_N) = f(x_N) \cdot f(x_{N-1}|x_N) \cdot f(x_{N-2}|x_{N-1}, x_N)... \cdot f(x_1|x_2,... x_N) \quad (2.2.13)$$

We remind that the copula density $c_{12...N}$ is linked to the joint density by:

$$f(x_1, x_2, ...x_N) = c_{12...N}\Big(F_1(x_1), ...F_N(x_N)\Big) \cdot f_1(x_1) \cdot ...f_N(x_N). \quad (2.2.14)$$

Let's work with $N = 3$. We have:

$$\begin{aligned} f(x_1, x_2, x_3) &= c_{123}\Big(F_1(x_1), F_2(x_2), F_3(x_3)\Big) \cdot f_1(x_1) \cdot f_2(x_2) \cdot f_3(x_3) \\ &= f(x_3) \cdot f(x_2|x_3) \cdot f(x_1|x_2, x_3). \end{aligned} \quad (2.2.15)$$

We can decompose $f(x_2|x_3)$, by writing the expression for $f(x_2, x_3)$, into

$$f(x_2|x_3) = c_{23}\Big(F_2(x_2), F_3(x_3)\Big) \cdot f_2(x_2). \quad (2.2.16)$$

In the same way, we can decompose $f(x_1|x_2, x_3)$ into

$$\begin{aligned} f(x_1|x_2, x_3) &= c_{12|3}\Big(F_{1|3}(x_1|x_3), F_{2|3}(x_2|x_3)\Big) \cdot f(x_1|x_3) \quad (2.2.17) \\ &= c_{12|3}\Big(F_{1|3}(x_1|x_3), F_{2|3}(x_2|x_3)\Big) \cdot c_{13}\Big(F_1(x_1), F_3(x_3)\Big) \cdot f_1(x_1) \end{aligned}$$

which yields:

$$\begin{aligned} f(x_1, x_2, x_3) &= f_3(x_3) \cdot c_{23}\Big(F_2(x_2), F_3(x_3)\Big) \cdot f_2(x_2) \cdot c_{12|3}\Big(F_{1|3}(x_1|x_3), F_{2|3}(x_2|x_3)\Big) \\ &\cdot c_{13}\Big(F_1(x_1), F_3(x_3)\Big) \cdot f_1(x_1) \quad (2.2.18) \\ &= f_1(x_1) \cdot f_2(x_2) \cdot f_3(x_3) \cdot c_{13}\Big(F_1(x_1), F_3(x_3)\Big) \\ &\cdot c_{23}\Big(F_2(x_2), F_3(x_3)\Big) \cdot c_{12|3}\Big(F_{1|3}(x_1|x_3), F_{2|3}(x_2|x_3)\Big). \end{aligned}$$

The general formula for the conditional densities is:

$$f(x|\boldsymbol{v}) = c_{xv_j|\boldsymbol{v}_{-j}}\Big(F(x|\boldsymbol{v}_{-j}),\, F(v_j|\boldsymbol{v}_{-j})\Big) \cdot f(x|\boldsymbol{v}_{-j}). \tag{2.2.19}$$

where $\boldsymbol{v}$ is a $d-$dimensional vector and $\boldsymbol{v}_{-j}$ is the vector $\boldsymbol{v}$ excluding the component v_j. We use the conditional distribution $F(x|\boldsymbol{v}) = h(x,\, \boldsymbol{v},\, \Theta)$ where Θ represent the copula parameters for the joint distribution of x and $\boldsymbol{v}$.

There are different factorisations with the pair-copula construction, and we are going to choose the canonical vine representation in [ACFB09], where we have one variable leading the interaction in the data set. Even with this representation, the number of factorisations grows with the number of variables. There is a tree structure induced by this representation, if we have N variables, we have $N-1$ trees. A unique node joins $N-j$ edge for the tree T_j, and since there are $N-1$ trees, we have an arithmetic sum that gives us $\frac{N(N-1)}{2}$ parameters to estimate.

We can generalize the canonical vine decomposition to N variables, we only state the result:

$$\begin{aligned} f(x_1,\, x_2,\ldots\, x_N) \;&=\; \prod_{j=1}^{N}\prod_{i=1}^{N-j} c_{j,\; j+i|1,\ldots j-1\boldsymbol{v}_{-j}}\Big(F(x_j|x_1,\ldots\, x_{j-1}),\, F(x_{j+i}|x_1,\ldots\, x_{j-1})\Big) \\ &\cdot\; \prod_{k=1}^{N} f(x_k). \end{aligned} \tag{2.2.20}$$

We observe intuitively that we multiply the N marginal densities, we form a product over the $N-1$ trees and for each tree, we have a product of $N-j$ vine copula parameters that arise from the $N-j$ edges.

We have used the dynamic copula toolbox in Matlab to fit the canonical vines. We want to simulate canonical vines and we implement the sampling procedure described in [ACFB09] and where the h-function in the algorithm is dependent on the base copula model chosen. However, the toolbox does not allow simulation of the vine copulas and we therefore implemented the sampling procedure in Matlab.

The h-function can be computed for every copula and it is proven in [Joe96] by computing the derivative with respect to the second variable of the copula distribution function.

$h-$ function for the Gaussian copula

$$h(u_1,\, u_2,\, \rho_{12}) = \Phi\left(\frac{\Phi^{-1}(u_1) - \rho_{12}\Phi^{-1}(u_2)}{\sqrt{1-\rho_{12}^2}}\right) \tag{2.2.21}$$

$h-$ function for the Student copula

$$h(u_1,\, u_2,\, \rho_{12}, \nu_{12}) = t_{\nu_{12}+1}\left(\frac{t_{\nu_{12}}^{-1}(u_1) - \rho_{12}t_{\nu_{12}}^{-1}(u_2)}{\frac{\left(\nu_{12}+\left(t_{\nu_{12}}^{-1}(u_2)\right)^2\right)\cdot\left(1-\rho_{12}^2\right)}{\nu_{12}+1}}\right) \tag{2.2.22}$$

Algorithm 2.1 Simulate M samples x_1, x_2,...x_N from a canonical vine

for $m \leftarrow 1, 2...M$ **do**
  Sample N independent standard uniforms w_i.
  $x_1 = v_{1,\,1} = w_1$
  for $i \leftarrow 2, 3...N$ **do**
    $v_{i,\,1} = w_i$
    for $k \leftarrow i-1, i-2...1$ **do**
      $v_{i,\,1} = h^{-1}(v_{i,\,1}, v_{k,\,k}, \Theta_{k,\,i-k})$
    end for
    $x_i = v_{i,\,1}$
    if $i == N$ **then**
      Stop
    end if
    for $j \leftarrow 1, 2...i-1$ **do**
      $v_{i,\,j+1} = h(v_{i,\,j}, v_{j,\,j}, \Theta_{j,\,i-j})$
    end for
  end for
end for

$h-$ function for the Clayton copula

$$h(u_1,\, u_2,\, \delta_{12}, \nu_{12}) = u_2^{-\delta_{12}-1}\left(u_1^{-\delta_{12}} + u_2^{-\delta_{12}} - 1\right)^{-1-\frac{1}{\delta_{12}}} \tag{2.2.23}$$

The purpose of this section was to simulate Gaussian, Student, Clayton and canonical Clayton vines. We want to compare and stress their different tail behaviours with identical marginal distributions, and to that intent, we require simulated copulas and the simulated scenarios from the marginals because they form the inputs to the Copula-Marginal algorithm. The following section illustrates how we visualize, compute and stress the tail-dependence.

2.2.3 Tail-dependence views

If we consider a bivariate distribution, the tail-dependence at a certain level is intuitively the proportion that one margin exceeds a level conditioned on the event that the other margin exceeds it as well. Upper-tail, lower-tail dependence and tail independence are defined in [EFM05] and reveal the clustering properties of the multivariate distribution in the extremes. The previous copulas we have simulated have different tail properties. The Gaussian copula is lower and upper tail-independent, the Student copula is both upper and lower tail-dependent whereas the Clayton copula is only lower tail-dependent. Empirically speaking, equity returns show more dependence in bear markets than in bull markets, so we want our returns to have both lower and upper tail-dependence to some extent, though preferably not in a symmetric manner. In light of these characteristics, we expect the Student copula to perform best when fitted to bivariate equity log-returns, and we will validate this assumption with statistical criterias.

In order to get an intuition about tail-dependence structures for these copulas, we fit gaussian, Student and Clayton copulas to the joint log-returns of IBM and JP Morgan, prices for these stocks are taken from Yahoo Finance and processed. We then create the resulting bivariate scenarios by using the combination process in the copula-marginal algorithm and visualize the empirical tail-dependence with a 2-dimensional scatter plot of historical log-returns using the Matlab function scatterhist.

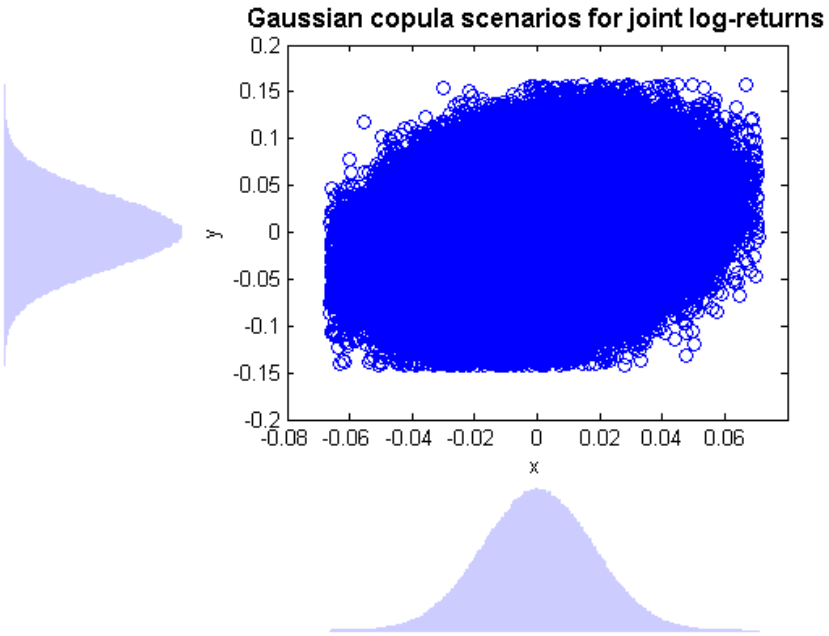

Figure 35: Scatterplot of bivariate normal scenarios

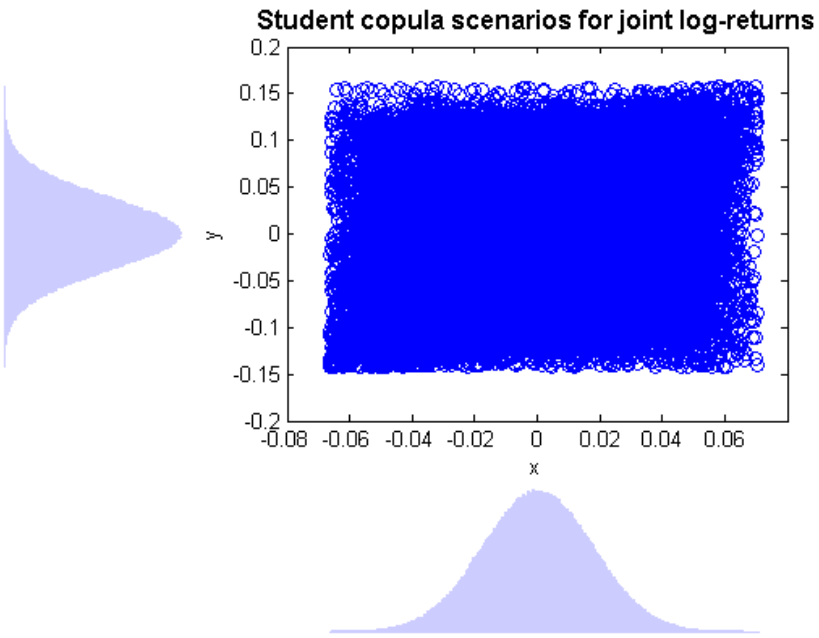

Figure 36: Scatterplot of bivariate Student scenarios

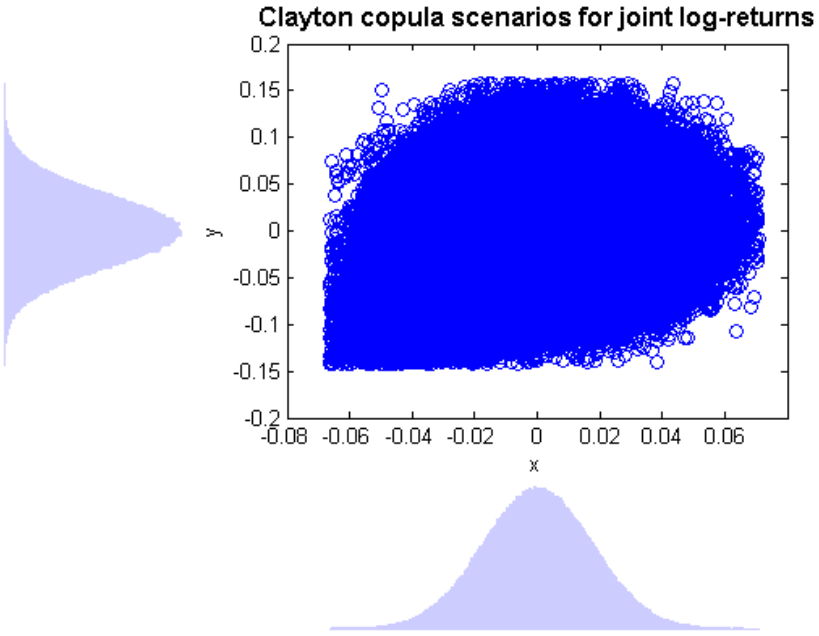

Figure 37: Scatterplot of bivariate Clayton scenarios

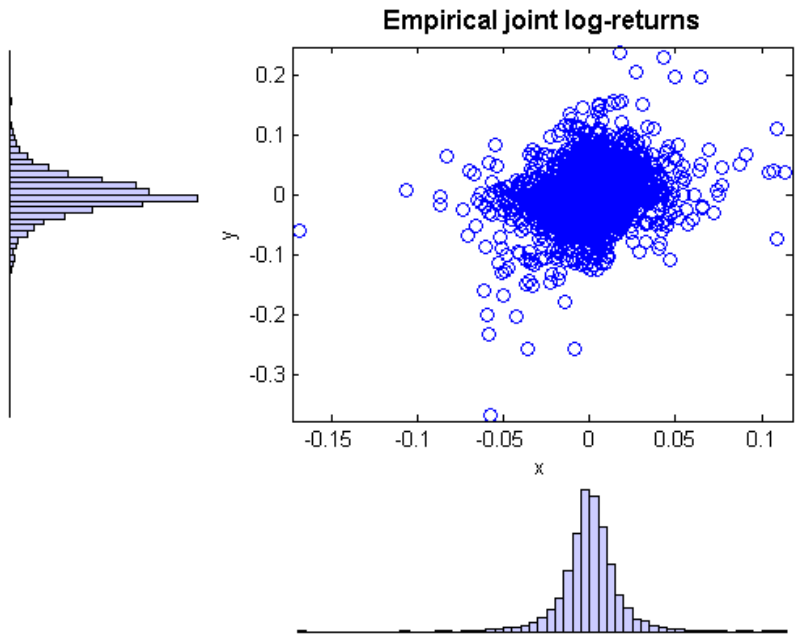

Figure 38: Scatterplot of bivariate empirical scenarios

Fig. 35 doesn't display any significant clustering, the simulated values are evenly distributed on the scatter plot. This hints at a lack of tail dependence, both in the upper and lower tails. Fig. 36, while not apparent, has a higher concentration in both the negative and positive joint returns, pointing to an upper and lower tail-dependence. Fig. 37 clearly has a clustering in the lower tail while it remains even in the upper tail, so we retrieve the lower tail-dependence and the upper-tail independence. The empirical scenarios in Fig. 38 show a much higher concentration of negative co-movements than positive ones, which suggests a lower tail-dependence structure but also an upper-tail dependence, although the latter is much weaker. The range of values is however quite different if we compare the simulated scenarios arising from copulas and the empirical scenarios. The empirical scenarios in Fig. 38 show extreme values for both IBM and JP Morgan that are not spanned by the copula scenarios. The most negative log-return for IBM is lower than -15%, while it doesn't get lower than -8% with the copula simulations. The lowest for JP Morgan is lower than -30%, and this observation is also coupled with a -6% log-return for IBM, whereas it only reaches -15% with the simulation approach. There is clearly a higher clustering of negative joint returns than positive joint returns.

We fit a 4-dimensional, canonical Clayton vine copula to the log-returns of IBM, JP Morgan, Quiksilver and Exxon Mobil, prices are again imported from Yahoo Finance. It requires writing the density for $N = 4$ in the previous section and the estimation of 6 copula parameters by maximum likelihood estimation. We then represent some scatter plots of the resulting pair dependences between: IBM and JPM, IBM and KWK, IBM and XOM, JPM and KWK, JPM and XOM. In all of them, the lower tail-dependence is observed while the number of parameters in the canonical vine decomposition allows a different representation of each coupled dependence.

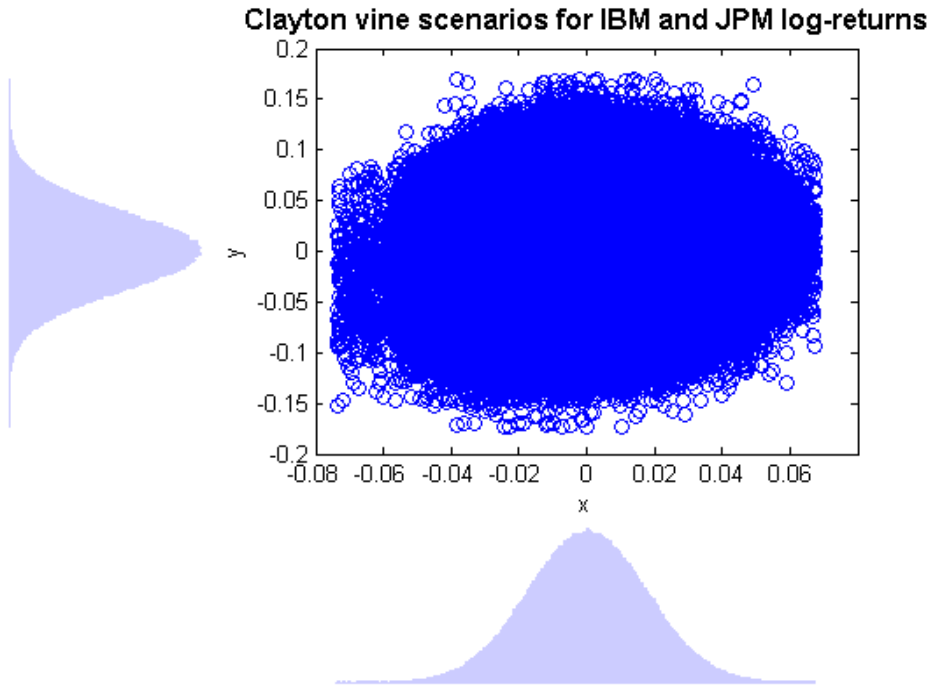

Figure 39: Scatterplot of bivariate Clayton vine scenarios: JP Morgan and IBM

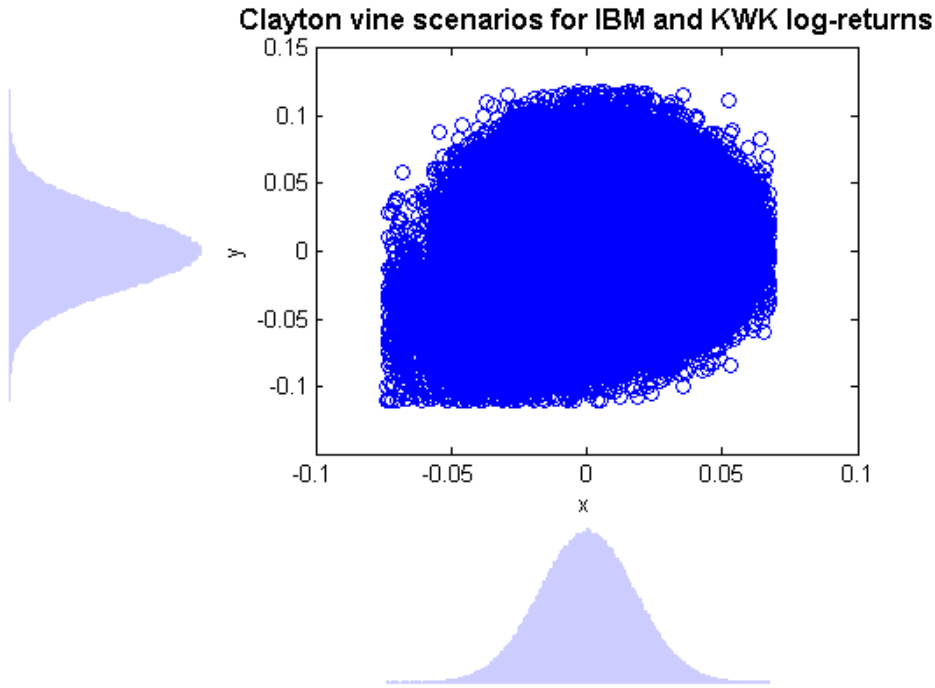

Figure 40: Scatterplot of bivariate Clayton vine scenarios: JP Morgan and IBM

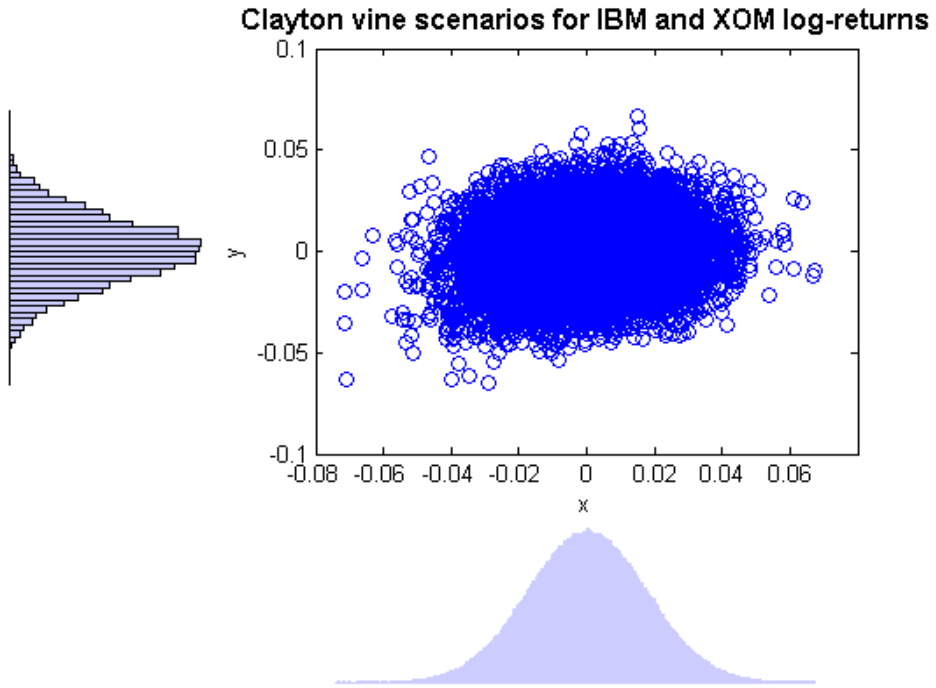

Figure 41: Scatterplot of bivariate Clayton vine scenarios: Exxon Mobil and IBM

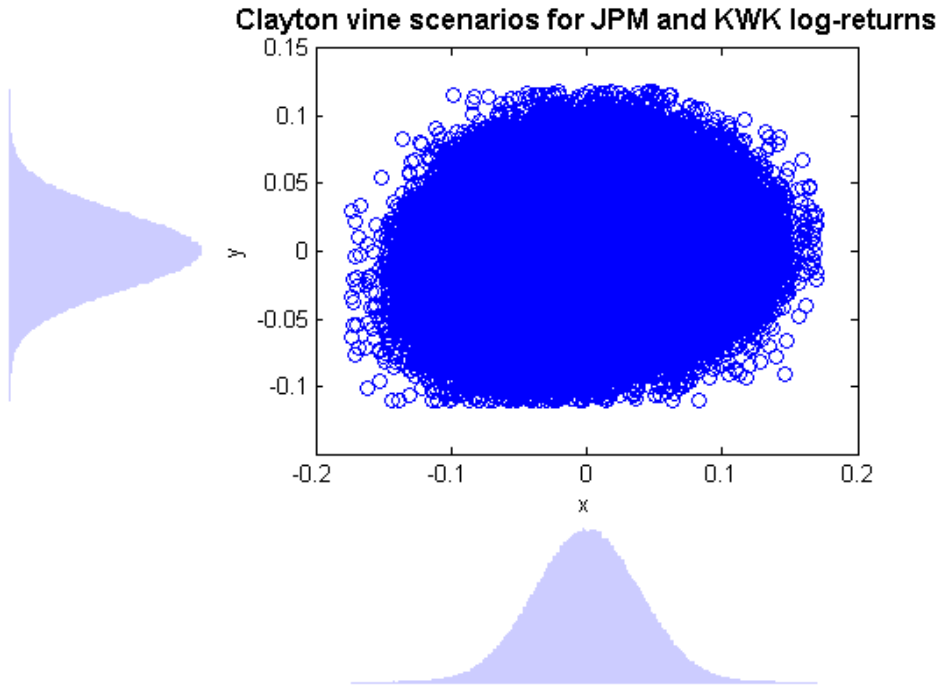

Figure 42: Scatterplot of bivariate Clayton vine scenarios: JP Morgan and Quiksilver

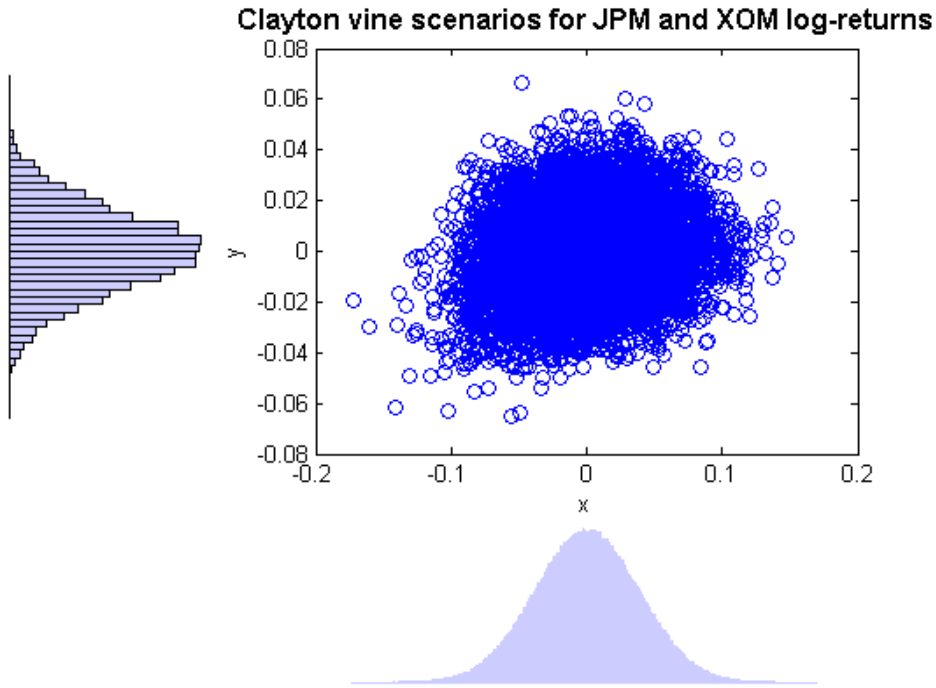

Figure 43: Scatterplot of bivariate Clayton vine scenarios: JP Morgan and Exxon Mobil

We can simulate scenarios of joint log-returns with different copulas and visualize their tail behaviours. Our previous work has made the multivariate scenario-probability representation available for tail-dependence stress-testing. We can now show, in the following algorithm taking advantage of Meucci's fully flexible probabilities framework and that we implemented in Matlab, how we compute joint exceedance probabilities for risk factor scenarios at thresholds $\boldsymbol{u}$, and how to alter the tail-dependence exogenously.

Tail-dependence views

- Build a matrix $\mathcal{R}$ that assigns to the elements of each column of $\mathcal{X}$ a ranking in ascending order.
- Extract the empirical copula by setting: $\mathcal{U}_{j,k} = \frac{\mathcal{R}_{j,k}}{J}$, where J is the number of simulations.
- If p_j are the prior probability masses attached to the scenarios $\nu_{j,\,k}$ or $\mathcal{X}_{j,\,k}$, the tail-dependence at threshold $\boldsymbol{u}$ is defined as:

$$\mathbb{P}\left(U_j \leq u_j \ \forall j = 1,...N\right) = \sum_{j \in \mathcal{I}_{\boldsymbol{u}}} p_j$$

where

$$\mathcal{I}_{\boldsymbol{u}} = \left\{ i \mid \mathcal{U}_{i,k} \leq u_k \ \forall k = 1,...N \right\}$$

- A view on tail dependence amounts to choosing the posterior probability mass $\tilde{p}_j$ such that

$$\sum_{j \in \mathcal{I}_{\boldsymbol{u}}} \tilde{p}_j = \tilde{C}$$

  where $\tilde{C}$ is set exogenously or comes from another model.

- We can write the view as a linear equality system on the posterior probability masses $\tilde{p}_j$

$$A\tilde{\mathbf{p}} = b$$

  where

$$A = \begin{array}{c} \begin{array}{ccccccc} 1 \in \mathcal{I}_{\boldsymbol{u}} & 2 \notin \mathcal{I}_{\boldsymbol{u}} & \cdots & M \notin\in \mathcal{I}_{\boldsymbol{u}} & M+1 \in \mathcal{I}_{\boldsymbol{u}} & \cdots & J \in \mathcal{I}_{\boldsymbol{u}} \end{array} \\ \begin{pmatrix} 1 & 0 & \cdots & 0 & 1 & \cdots & 1 \end{pmatrix} \end{array}$$

  has dimension $1 \times J$, and

$$b = \left(\tilde{C}\right)$$

- We solve the relative entropy minimization problem:

$$\begin{aligned} \tilde{\mathbf{p}} &= \underset{\sum_{j \in \mathcal{I}_{\boldsymbol{u}}} \tilde{p}_j = \tilde{C}}{\operatorname{argmin}} \sum_{j=1}^{J} \tilde{p}_j \left[\ln(\tilde{p}_j) - \ln(p_j) \right] \\ \mathcal{I}_{\boldsymbol{u}} &= \left\{ i \mid \mathcal{U}_{i,k} \leq u_k \ \forall k = 1,...N \right\} \end{aligned}$$

- We keep the same scenarios ν or $\mathcal{X}$ and replace $\mathbf{p}$ with $\tilde{\mathbf{p}}$, the tail-dependence view is now satisfied.

The algorithm for computing and stress-testing the tail-dependence is applied here for joint return losses of IBM and JP Morgan. We take a closer look at lower tail dependences for the Gaussian, the Student, the canonical Clayton vine, the empirical copula, and compute, for various threshold levels ranging from 10^{-3} to 10^{-1}, the associated probabilities for log-return values below each threshold. Fig. 44 shows the tail-dependences for a continuum of tail levels. We have looped over a part of the instructions described in the previous algorithm, computing therefore the tail-dependence at the specified tail levels and excluding only the views. The Gaussian copula is, as expected, displaying

thinner tails than the Student copula for all tail levels, whereas it is surprising that the canonical Clayton vine copula has lower probabilities than the Gaussian copula. However, when we zoom on the extreme tail levels in Fig. 45, we see that the tail-dependence curve for the canonical Clayton vine copula becomes greater and one should remember that the lower-tail dependence property means that asymptotically the probability does not go to 0, whereas it is the case for the Gaussian copula. Returning to Fig. 44, we see that the Student copula and the empirical copula are closely matched, which would indicate that the Student copula is suitable. The empirical copula can however wildly depend on the number of observations we consider, and what we typically see is that at very low thresholds, the empirical copula tail-dependence goes to 0 at a tail level of $4 * 10^{-3}$, as shown in the black line of Fig. 45. If we take the tail level at 10^{-3} in Fig. 45, the Student copula indicates this is an event happening a little more than once every ten thousand days, which is once every 27 years. In any case this probability shouldn't be 0 because big joint losses are very frequent in a financial crash. The empirical copula has a 0 probability associated to that tail level in Fig. 45, which is not only unmet in practice but also renders empirical copula stress-testing unrealistic in Meucci's framework since only the scenarios with positive probability can be stress-tested.

If we decide to stress-test the canonical Clayton vine copula, we can either do it exogenously or with a model. By model we mean that we could assign, for a given tail level, the tail dependence view forecasted by the Student copula for example, or any other copula, to our Clayton vine copula. We did it exogenously here, by specifying a stress-test intensity parameter. This parameter is multiplied by the prior tail-dependence probability and this product constitutes the view. We chose 5 for the stress-test intensity and a 10^{-2} level. We show the standard and the stressed Clayton vine copulas, along with the others, in Fig. 46. Obviously the tail-dependence curve for the stressed copula is shifted above because we have forced the probabilities to be 5 times higher, and we believe the stressed copula will actually produce a better goodness-of-fit than the prior one and the Gaussian copula because it follows the empirical copula's behaviour more closely.

We need to formally investigate which copula produces the best dependence structure with two model selection criterions, and we also add the canonical Student copula in the mix to see how it performs relative to the canonical Clayton vine and the Student copula. The model selection measures we use are the the Akaike Information Criterion and the Bayesian Information Criterion, and there are functions in Matlab to compute these quantities.

$$AIC = 2k - 2\log(L), \tag{2.2.24}$$
$$BIC = k\log(n) - \log(L). \tag{2.2.25}$$

where k is the number of parameters, L the maximized value for the likelihood function and n the number of obversations. AIC and BIC are computed for each copula, and the model chosen is the one which has the smallest value. Table 47 shows that the model with the smallest AIC and BIC values is the Student copula (-3274.7 and -3262.4), followed by the canonical Student vine copula (-2636.6988 and -2599.8674), with the canonical Clayton vine copula (-2018.8613 and -1982.0299) finishing last.

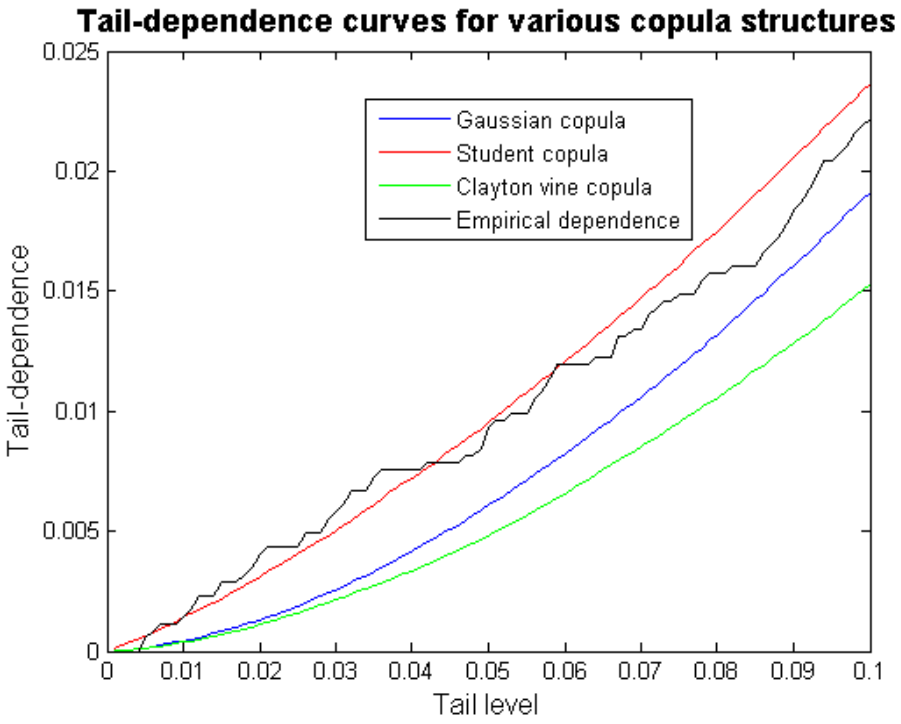

Figure 44: Joint exceedance probabilities for lower thresholds

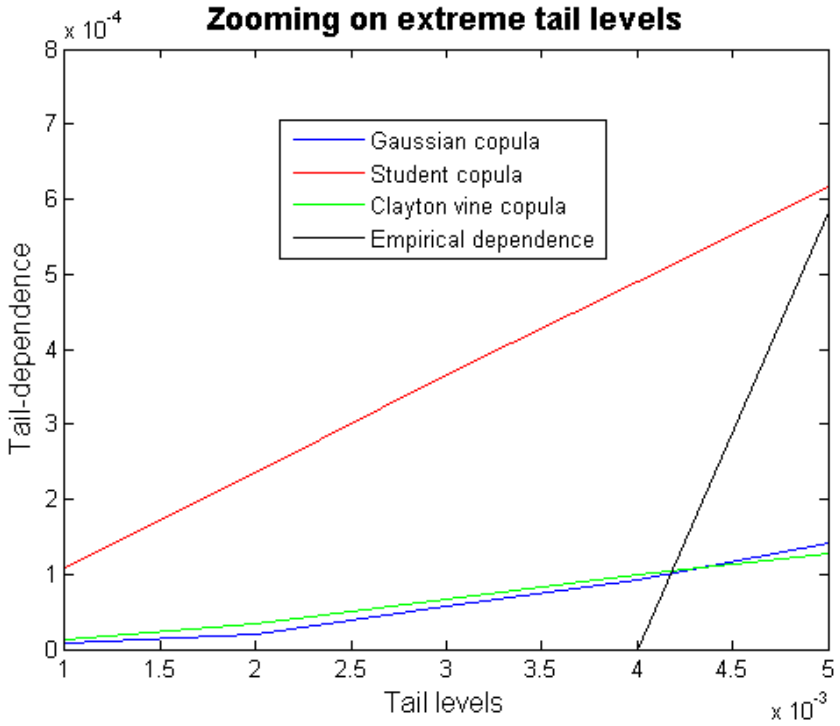

Figure 45: Joint exceedance probabilities for very low thresholds

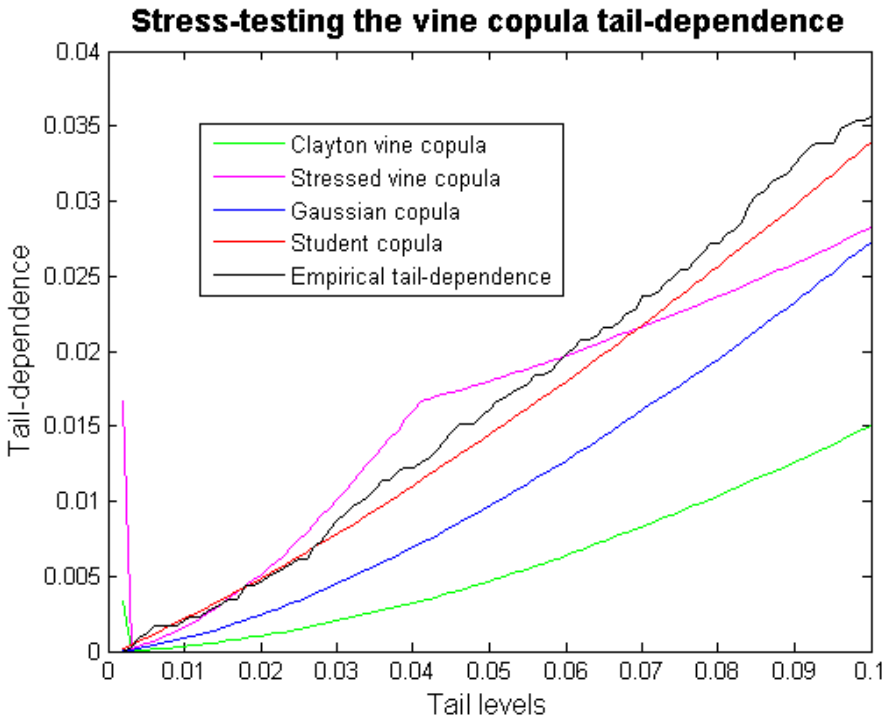

Figure 46: Tail-dependence views algorithm applied to original curves

		Model selection criterion	
		AIC	BIC
Dependence structure	Gaussian copula	-2605.5	-2599.3
	Student copula	-3274.7	-3262.4
	Clayton vine copula	-2018.8613	-1982.0299
	Student vine copula	-2636.6988	-2599.8674

Table 47: Model selection with various dependence structures

We conclude our section on copulas. Going further than the correlation which is suitable for elliptic distributions, copulas helped us capture the nonlinearity in the dependence structure and were especially useful to process marginal views by isolating the marginals from the dependence, hence providing an additional layer of flexibility. Meucci's Copula-Marginal algorithm extracts the copula from multivariate distributions and combines marginal distributions with the copula. These both tasks are achieved in a scenario-based framework, which makes for boundless possibilities since we can work with parametric copulas but also nonparametric copulas of our own creation, as long as we are able to simulate. Furthermore, the scenario generation of the joint distribution with the Copula-Marginal algorithm, along with the reuse of the fully flexible probabilities machinery, permits a straightforward quantification and stress-testing of extreme co-movement occurrences, where the stress-testing view comes from a user or another model. We visualized and compared the tail-dependence curves when we stressed exogenously joint losses. Finally, we assessed the goodness of fit of various copula models, in

order to select a suitable model. We could even go further and search for views that improve on the model selection criteria. Having suitable tools at our disposition for stress-testing market risk in large portfolios, we will at last explore liquidity risk and its effects on risk assessments according to various trading scenarios.

2.3 Liquidity scenarios

2.3.1 Integrated risk model and liquidity score

We finally exploit a new framework in [Meu12b] blending liquidity, funding and market risk, where a liquidity distribution is associated to each future market-risk scenario, and the liquidity uncertainty is scenario-dependent on the liquidation or funding policy. We obtain a liquidity-adjusted P&L distribution, and decompose the total risk according to the market or liquidity contributions as well as compute a monetary measure of portfolio liquidity. There are many advantages to this framework advertised by Meucci:

- full-impact modelling of the amount liquidated at the future horizon or liquidation schedule, the impact uncertainty, correlations and speed of trading impact,
- adverse developments in the market lead to worse liquidation schedules,
- exogenous and endogenous liquidity risk modelling,
- novel decomposition into a market and liquidity component and definition of the liquidity score,
- fast distributional stress-testing.

We will first talk about pricing and aggregation. If we have N securities with mark-to-market P&Ls $\bar{\Pi}_n$ and D risk drivers $\boldsymbol{X} \equiv (X_1, \ldots, X_D)'$, the profit and loss for each security is a deterministic function π_n of the risk drivers:

$$\bar{\Pi}_n = \pi_n(\boldsymbol{X}) \qquad n = 1, \ldots, N. \tag{2.3.1}$$

We now define the holdings of a portfolio, i.e the number of units of the securities, represented by the vector $\boldsymbol{h} \equiv (h_1, \ldots, h_N)'$. The mark-to-market P&L of the portfolio is thus:

$$\bar{\Pi} = \sum_{n=1}^{N} h_n \pi_n(\boldsymbol{X}). \tag{2.3.2}$$

Previously, we modeled the joint distribution of the risk drivers with a scenario-based approach, so we have the following representation:

$$\boldsymbol{X} \sim \{\boldsymbol{x}_j; p_j\}_{j=1,\ldots,J} \tag{2.3.3}$$

The conservation law of money allows us to compute the portfolio P&L scenario-by-scenario, with the probabilities remaining untouched:

$$\bar{\mathbf{\Pi}} \sim \{\bar{\boldsymbol{\pi}}_j; p_j\}_{j=1,\dots,J} \tag{2.3.4}$$

where

$$\bar{\boldsymbol{\pi}}_j = \sum_{n=1}^{N} h_n \pi_n(\boldsymbol{x}_j). \tag{2.3.5}$$

A realization of the risk drivers is always surrounded by a liquidity-related uncertainty, noted $\Delta\Pi_n$, in the $n-$th security's P&L. Three factors affect $\Delta\Pi_n$, which are:

- the liquidity schedule $\Delta\boldsymbol{h} \equiv (\Delta h_1, \dots, \Delta h_N)'$, causing a price impact decomposed in a linear, permanent component and a temporary component negatively impacting the portfolio on average,
- the execution horizons $\boldsymbol{\tau} \equiv (\tau_1, \dots, \tau_N)'$ for the liquidations, with longer executions causing less impact but a bigger uncertainty, and conversely for shorter executions.

We suppose, even though we can generalize it to elliptical distributions, that the liquidity uncertainty is modeled by a normal random variable:

$$\Delta\Pi_n \sim N(\mu_n, \sigma_n^2). \tag{2.3.6}$$

The mean μ_n and the volatility σ_n of the liquidity uncertainty, derived from the volume-weighted-average-price execution in the appendix of [MAK12], are:

$$\mu_n = -\alpha_n e_n |\Delta h_n| - \beta_n e_n \bar{\sigma}_n \frac{|\Delta h_n|^{\frac{3}{2}}}{\sqrt{v_n}}, \tag{2.3.7}$$

$$\sigma_n = \delta_n \sqrt{v_n} e_n |\Delta h_n|. \tag{2.3.8}$$

Let's detail what all these terms mean. α_n estimates the commissions added to half the bid-ask spread, and this is given as a fraction of the exposure e_n of one unit of security n. In our case, to simplify things, we only take half the bid-ask spread. The bid-ask spread for each security is computed from the high and low prices in the Yahoo Finance exported csv files, using the estimator for the spread in [CS12]. Meucci assumed in his code that all bid-ask spreads are 1% (again in fraction of the exposure e_n), which we believe is a simplistic assumption. β_n is very similar for securities of the same asset class, and δ_n is a constant coefficient as well, both are provided by Meucci in his Matlab code and we therefore take his values $\delta_n = 0.5$ and $\beta_n = 10^5$. We wrote a Matlab function in order to specify the trading horizon and the liquidation amount in a proper way. $\bar{\sigma}_n$ gives a best estimate of the average annualized P&L volatility, also given as a percentage similarly to α_n. v_n is the approximate number of units of security n traded by the whole market over the execution horizon τ_n. When the execution horizon τ_n is longer, it means that there are more shares of security n traded on the market, so as we can observe from

their relation to v_n, the average impact is lower while the impact volatility is higher. The more we liquidate, the more negative the average impact and the higher the volatility impact become. If we have two positions with liquidity uncertainties $\Delta\Pi_n$ and $\Delta\Pi_m$, their correlations $\rho_{n,m}$ are higher than the respective market correlations $\hat{\rho}_{n,m}$, liquidity risk lacking diversification potential, so we shrink the market correlations with a parameter γ close to 1, in our examples we took $\gamma = 0.9$:

$$\rho_{n,m} = \gamma + (1-\gamma)\hat{\rho}_{n,m}. \tag{2.3.9}$$

We aggregate the parameters α_n, β_n, δ_n and γ to get the liquidity distribution for the whole portfolio:

$$\begin{aligned} \Delta\Pi &= \sum_{n=1}^{N} \Delta\Pi_n, && (2.3.10)\\ \Delta\Pi &\sim N(\mu, \sigma^2), && (2.3.11)\\ \mu = \sum_{n=1}^{N} \mu_n \quad &\text{and} \quad \sigma^2 = \sum_{n,m=1}^{N} \sigma_n \sigma_m \rho_{n,m}. && (2.3.12) \end{aligned}$$

To arrive at the total portfolio P&L, we just sum-up the mark-to-market P&L $\bar{\Pi}$ and the liquidity-adjusted P&L $\Delta\Pi$. We compute the form of the portfolio density of the P&L $\Pi = \bar{\Pi} + \Delta\Pi$, the proof adding to the comprehension and so we include it here. If we take the conditional distribution of the liquidity adjustment

$$\Delta\Pi|\boldsymbol{x} \sim N\Big(\mu(\boldsymbol{x}, \sigma^2(\boldsymbol{x})\Big) \tag{2.3.13}$$

where the liquidation policy is dependent on the market scenario. Writing this distribution as a linear combination of a standard normal variable, we can retrieve the conditional probability density function:

$$f_{\Delta\Pi|\boldsymbol{x}} = \frac{1}{\sigma(\boldsymbol{x})}\varphi\bigg(\frac{y-\mu(\boldsymbol{x})}{\sigma(\boldsymbol{x})}\bigg). \tag{2.3.14}$$

To get the unconditional distribution of the total P&L, we use:

$$\begin{aligned} f_{\Pi}(y) &= \int f_{\Pi|\boldsymbol{x}}(y) f_{\boldsymbol{X}}(\boldsymbol{x}) d\boldsymbol{x}\\ &= \int f_{\bar{\Pi}+\Delta\Pi|\boldsymbol{x}}(y) f_{\boldsymbol{X}}(\boldsymbol{x}) d\boldsymbol{x}\\ &= \int f_{\Delta\Pi|\boldsymbol{x}}\big(y - \sum_{n=1}^{N} h_n \pi_n(\boldsymbol{x})\big) f_{\boldsymbol{X}}(\boldsymbol{x}) d\boldsymbol{x}. \end{aligned} \tag{2.3.15}$$

Since we have a generalized empirical distribution representation of the risk drivers from earlier, we can write:

$$f_{\boldsymbol{X}}(\boldsymbol{x}) = \sum_{j=1}^{J} p_j \delta^{\boldsymbol{x}_j}(\boldsymbol{x}). \tag{2.3.16}$$

Thus:

$$\begin{aligned} f_{\Pi}(y) &= \int f_{\Delta\Pi|\boldsymbol{x}}\big(y - \sum_{n=1}^{N} h_n \pi_n(\boldsymbol{x})\big) \sum_{j=1}^{J} p_j \delta^{\boldsymbol{x}_j}(\boldsymbol{x}) d\boldsymbol{x} \\ &= \sum_{j=1}^{J} p_j f_{\Delta\Pi|x_j}\big(y - \bar{\pi}_j\big) \end{aligned} \tag{2.3.17}$$

where $\bar{\pi}_j$ is the pure market P&L in the $j-$th scenario. By substituting the formula for the conditional density, we get the portfolio P&L density:

$$f_{\Pi}(y) = \sum_{j=1}^{J} \frac{p_j}{\sigma_{(j)}} \varphi\bigg(\frac{y - \bar{\pi}_j - \mu_{(j)}}{\sigma_{(j)}}\bigg). \tag{2.3.18}$$

where $\mu_{(j)}$ and $\sigma_{(j)}$ are the liquidity-adjusted parameters in the $j-$th scenario. Making the liquidation schedule and the execution horizons depend on the scenarios can be easily done, allowing us to stress test funding risk.

Since the total P&L $\Pi = \bar{\Pi} + \Delta\Pi$ is a sum, we can decompose all standard measures of risk into their marginal contributions. With the conditional value-at-risk, we have:

$$CVaR(\Pi) = \partial_{\bar{\Pi}} CVaR(\Pi) + \partial_{\Delta\Pi} CVaR(\Pi). \tag{2.3.19}$$

where, for any statistic σ, we defined the operator:

$$\partial_X \sigma(X + Y) \equiv \frac{d\sigma(uX + Y)}{du}\bigg|_{u=1} \tag{2.3.20}$$

Here again, the proof of the marginal $CVaR$ computations is helpful for the general comprehension and we will present it. The above decomposition follows from the Euler principle. We can compute the cumulative distribution function for the P&L from the density and get:

$$F_{\Pi}(y) = \sum_{j=1}^{J} \frac{p_j}{\sigma_{(j)}} \phi\bigg(\frac{y - \bar{\pi}_j - \mu_{(j)}}{\sigma_{(j)}}\bigg). \tag{2.3.21}$$

As we explained for the copula-marginal algorithm, we can recover the inverse cumulative distribution function F_{Π}^{-1} by linear interpolation, and particularly the $\alpha-$quantile

$$z \equiv F_{\Pi}^{-1}(\alpha). \tag{2.3.22}$$

We also compute the $CVaR$:

$$CVaR(\Pi) = \frac{1}{\alpha} \int_0^{\alpha} F_{\Pi}^{-1}(u) du. \tag{2.3.23}$$

We now derive the marginal contribution of the liquidity:

$$\begin{aligned}\partial_{\Delta\Pi} CVaR(\Pi) &= \mathbb{E}\left[\Delta\Pi|\bar{\Pi}+\Delta\Pi\leq z\right] \\ &= \mathbb{E}\left[\Delta\Pi|\Delta\Pi\leq z-\bar{\Pi}\right] \\ &= \int \mathbb{E}\left[\Delta\Pi|\bar{\Pi}=y,\Delta\Pi\leq z-y\right] f_{\bar{\Pi}}(y)dy \\ &= \sum_{j=1}^{J} p_j \,\mathbb{E}\left[\Delta\Pi|\bar{\Pi}=\bar{\pi}_j,\Delta\Pi\leq z-\bar{\pi}_j\right] \end{aligned} \tag{2.3.24}$$

For normal random variables X, the expected shortfall is:

$$\mathbb{E}\left[X|X\leq z\right] = \mu - \sigma\frac{\varphi\left(\frac{(z-\mu}{\sigma}\right)}{\phi\left(\frac{(z-\mu}{\sigma}\right)}. \tag{2.3.25}$$

Since

$$\Delta\Pi|\bar{\Pi}=\bar{\pi}_j \sim N(\mu_{(j)},\sigma^2_{(j)}), \tag{2.3.26}$$

then

$$\partial_{\Delta\Pi} CVaR(\Pi) = \sum_{j=1}^{J} p_j\left(\mu_{(j)} - \sigma_{(j)}\frac{\varphi\left(\frac{(z-\bar{\pi}_j-\mu_{(j)}}{\sigma_{(j)}}\right)}{\phi\left(\frac{(z-\bar{\pi}_j-\mu_{(j)}}{\sigma_{(j)}}\right)}\right) \tag{2.3.27}$$

To get the marginal market contribution $\partial_{\bar{\Pi}} CVaR(\Pi)$, we subtract the total conditional value-at-risk and the liquidity contribution:

$$\begin{aligned}\partial_{\bar{\Pi}} CVaR(\Pi) &= CVaR(\Pi) - \partial_{\Delta\Pi} CVaR(\Pi) \\ &= \frac{1}{\alpha}\int_0^{\alpha} F_{\Pi}^{-1}(u)du - \sum_{j=1}^{J} p_j\left(\mu_{(j)} - \sigma_{(j)}\frac{\varphi\left(\frac{(z-\bar{\pi}_j-\mu_{(j)}}{\sigma_{(j)}}\right)}{\phi\left(\frac{(z-\bar{\pi}_j-\mu_{(j)}}{\sigma_{(j)}}\right)}\right).\end{aligned} \tag{2.3.28}$$

Finally, we define the liquidity score LS for the portfolio. We developed a Matlab program for computing the liqudity score, since it was not provided by Meucci. When the liquidity adjustment barely affects the total portfolio Π, it means the portfolio is liquid and thus the liquidity score should be high. The liquidity adjustment reduces the P&L, so the liquidity score will measured as a high-confidence level (90%) difference in the expected shortfalls in the left tail between the pure market risk and the total risk:

$$0 \leq LS = \frac{CVaR(\bar{\Pi})}{CVaR(\Pi)} \leq 1. \tag{2.3.29}$$

2.3.2 Liquidity adjustments for equity portfolios

We consider equity portfolios from the S&P 500 index, and we vary the number of stocks in each portfolio. We form a portfolio of 1, 4 and 20 stocks. The same capital is allocated to each portfolio, and the stocks are equally weighted. Our initial capital Cap is 1 billion dollars. We then compute the holdings with:

$$h = \frac{Cap}{N * e_n}; \tag{2.3.30}$$

In the portfolio of 20 stocks, each stock will thus have a capital 20 times smaller than the capital for a portfolio with only one stock. We compute the pure market risk P&L of the portfolios but also integrate the liquidity risk to the pure market component, and look at the impact when we vary the liquidation schedule and the execution horizon. We also chose a different number of stocks for each portfolio to investigate and compare the diversification effects between pure market risk P&L and liquidity-adjusted P&L. We want to ensure that what happens with the expected impact and the impact uncertainty is consistent with the formulas and also makes sense from an intuitive point of view. Finally we compute the liquidity score in each case and explain these changes.

In the 1-stock portfolio in Fig. 48, the liquidity score is 38%, it increases to 44% for the portfolio of 4 stocks in Fig. 49 and even further to 48% for the portfolio of 20 stocks in Fig. 51. The market risk diversification, as can be observed with the histogram widths and the scales on the x-axis in the three previous plots, is significant as we move from 1 to 4 to 20 stocks. The liquidity risk diversification however, measured by the liquidity scores, is much smaller. The expected impact, visualized by the shift towards the negative realizations, is much more pronounced for highly concentrated portfolios. Obviously the bid-ask spread is much higher when we liquidate one stock compared to twenty stocks, assuming a same capital, so the expected impact is also more negative. The impact uncertainty is also higher the less stocks we have, as we can see we go from a scale of dozen million dollars to less than 10 million dollars.

For the 4-stock portfolio, we can compare the liquidity score of a full liquidation in Fig. 49 and a partial (20%) liquidation in Fig. 50. The liquidity score for the partial liquidation is 88%, which is significantly higher than for a full liquidation. The expected impact is a lot lower in a partial liquidation, since it is an increasing function of the liquidation schedule, i.e the fraction of the holdings we liquidate.

Finally, we want to vary the execution horizon of the liquidation, and for that we compare the 1-month full liquidation in Fig. 52 to the 1-day full liquidation in Fig. 48 of the 1-stock portfolio. The liquidity score for the execution horizon of one month is 79%, which is what we expected since the impact should be a lot lower than a 1-day liquidation yielding a score of 38%. The impact uncertainty is however much higher for a 1-month liquidation than for a 1-day liquidation. The expected impact is much lower since the number of shares traded in one month is much higher, and the formula suggests that the expectation is a decreasing function of the volume generated by this execution horizon. For the impact uncertainty, it is an increasing function of the volume, so it is clearly higher for one month than for one day. The trend is the same if we look at Fig. 54 and Fig. 51.

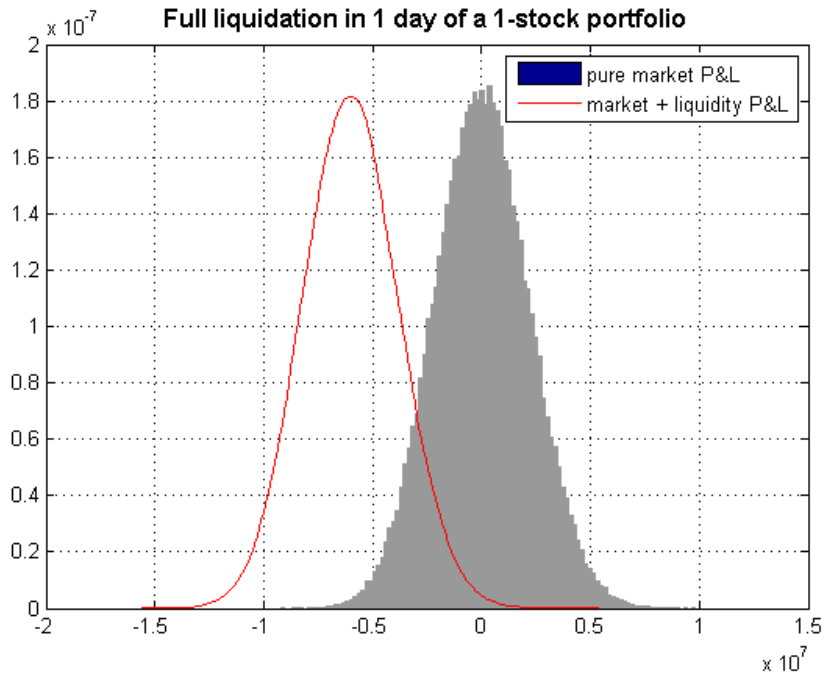

Figure 48: P&L adjustment with a full, one day asset liquidation

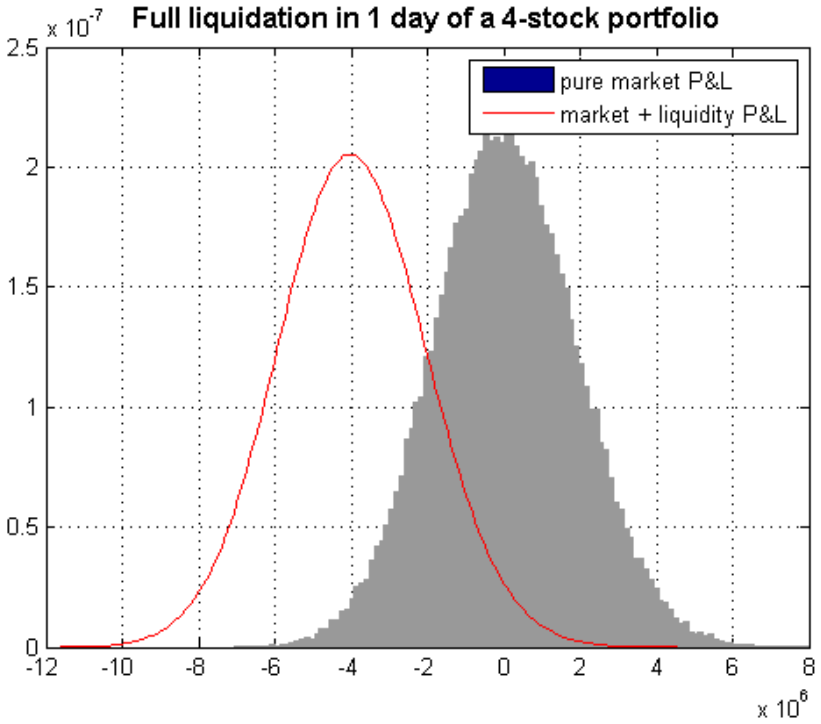

Figure 49: P&L adjustment with a full, one day liquidation of a 4-stocks portfolio

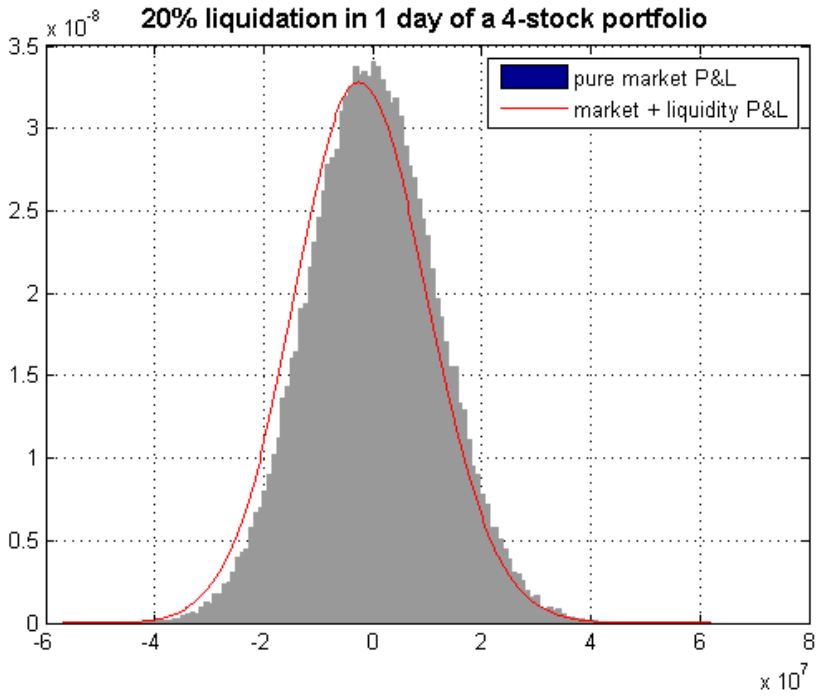

Figure 50: P&L adjustment with a 20%, one day liquidation of a 4-stocks portfolio

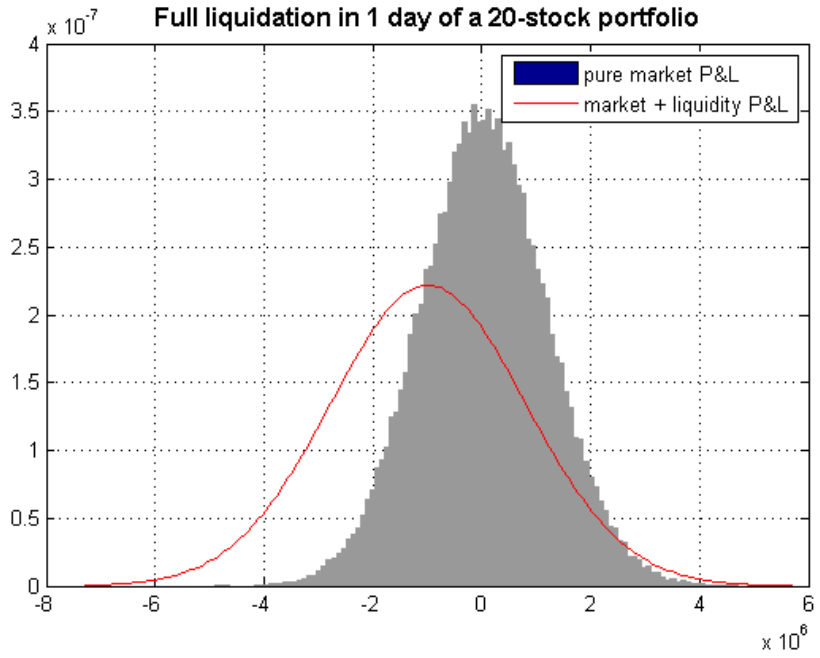

Figure 51: P&L adjustment with a full, one day liquidation of a 20-stocks portfolio

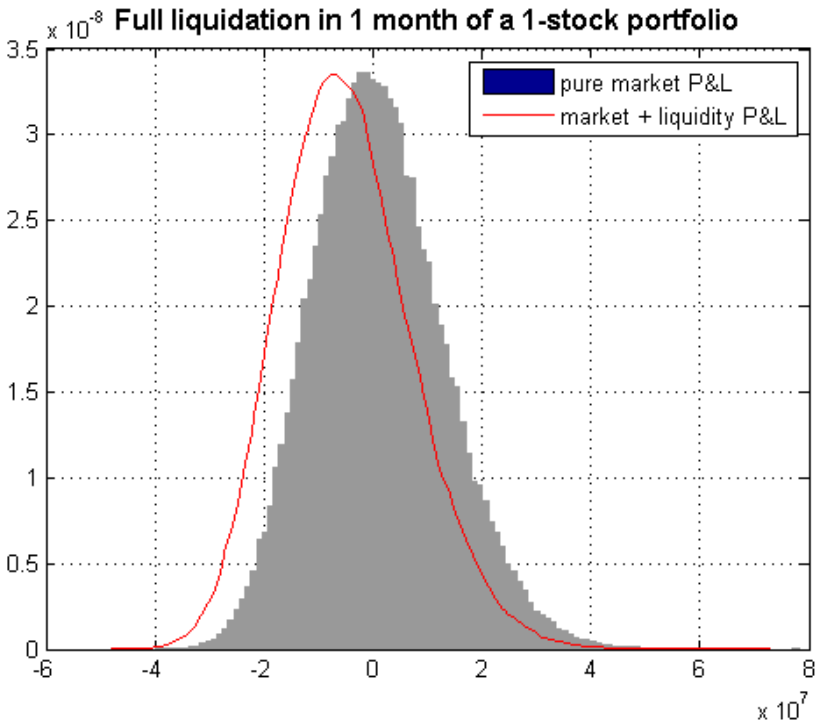

Figure 52: P&L adjustment with a full, monthly, asset liquidation

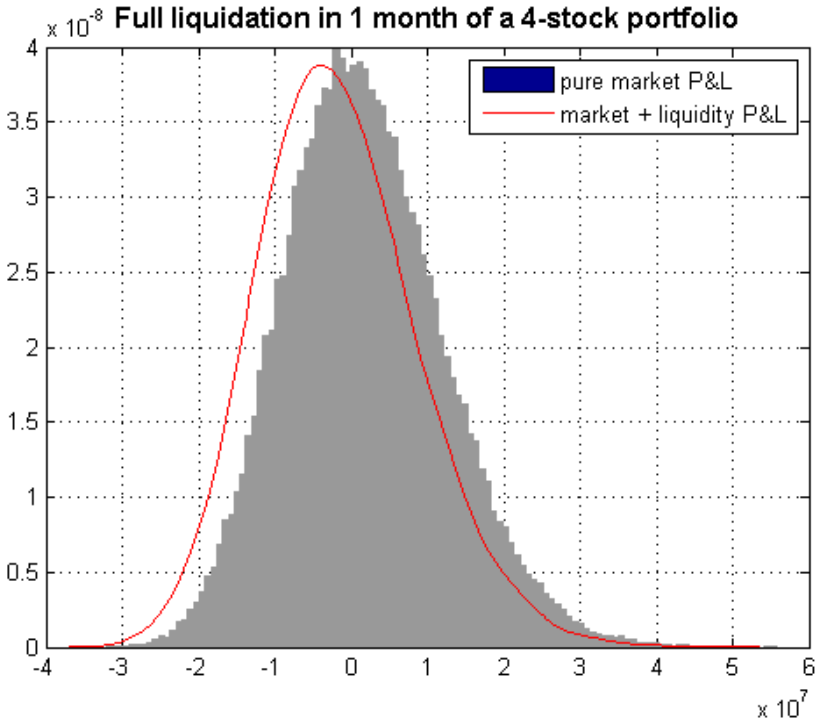

Figure 53: P&L adjustment with a full, one day liquidation of a 4-stocks portfolio

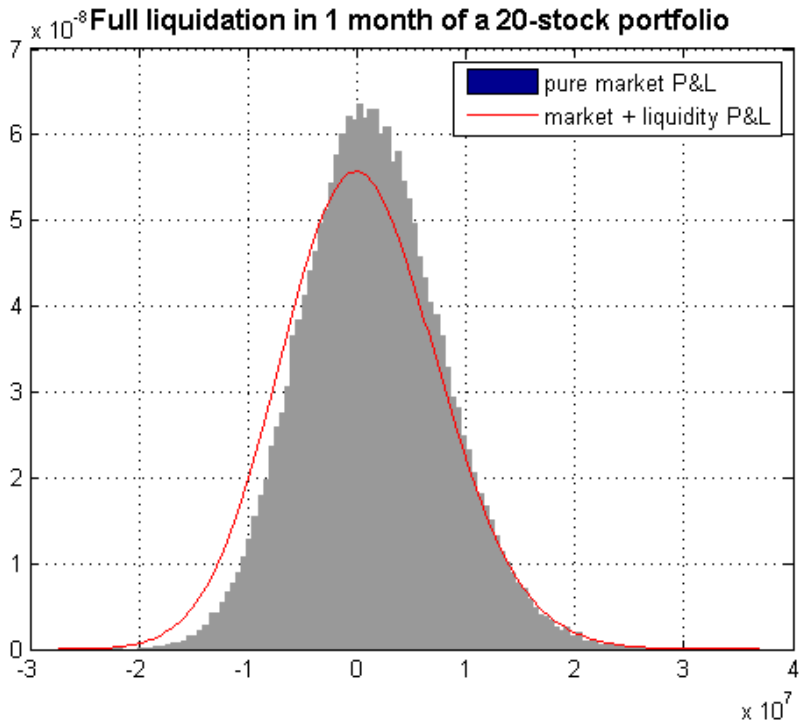

Figure 54: P&L adjustment with a full, monthly liquidation of a 20-stocks portfolio

We finally sum up this section on liquidity. Starting with a portfolio embedding market risk only, an adjustment was made by convoluting the pure market distribution with another distribution representing the liquidity risk component. The liquidity risk component is generally an elliptic distribution, whose components depend on bid-ask spreads, asset prices and volatility, the holding proportions we wish to liquidate as well as the horizon with which this liquidation takes place. A convolution is most easily computed in a scenario-based framework, and this is indeed ideal since all our previous work can be leveraged. Once we have computed the liquidity-adjusted profit and loss distribution for the portfolio, we quantified the impact of diversification, different liquidation amounts and horizons, by computing a risk statistic measuring the portfolio's illiquidity expressed as a ratio of expected shortfalls in the left tail. Hence, we can not only stress-test all the previous features such as location, dispersion, tail-dependence, VaR and expected shortfall, but also various liquidation policies. The result is that basically diversifying the portfolio reduces the liquidity component versus the market component. Shorter trading horizons have a higher expected impact but a smaller impact uncertainty, and a partial liquidation has a lower expected impact. In a further direction that we did not explore here, we could condition the liquidity adjustment parameters on the market state, such as a decrease in liquidity when the VIX rises, or consider the Morgan Stanley Liquidity Factor.

Conclusion

We can, at the end of our journey, fully stress-test a risk factor or a portfolio's distributional properties and select the stress type but also the stress value, both of which determine the view. As we have seen before, we have three types of views: expert views arising from a qualitative research or a manager's opinion about future developments; risk-correcting views, usually given by other risk models for a specific purpose of providing a more accurate estimation of risk measures; predictive views coming from statistical models to choose robust portfolios, better quantify investment opportunities or exploit arbitrage opportunities, in short to give a forward-looking perspective to the prior risk factor specification. The power of this framework is to combine these three types of views to provide a unified picture of investment upsides and downsides with more accurate risk estimates and external perspectives reflecting personal or data mining insights. Instead of adding more variables and making a model very complex, we use multiple models who add value for specific problems.

We have showed that risk and asset management can be merged by incorporating views from both areas in the representation of the risk factors or the portfolio prices or P&Ls. This gives a more holistic view of risks, where the word "risks" encompasses the threats as much as the opportunities. This ultimately leads to better decision-making and a simpler grasp on the whole information, seized with only two data structures: the scenario-probability representation. We can extend this framework to a dynamic one, we could have time-varying volatilities for example with a GARCH model, or mean-reverting stochastic processes, for which we can write the distribution at each timestep. This would require simulating the process over time and concatenating its realizations in a matrix where each column in the matrix would represent a date and there would be a probability vector indexed by time as well. The choice of the Kullback-Leibler distance should be discussed, and other measures could be considered. In a future work, we would focus a lot more on evaluating the benefits brought by the views by comparing prior and posterior models with performance measures, test statistics, out-of-sample backtests or portfolio strategies under each model. An interesting question is why we focus on the whole distribution and not only some select features of it such as the two first moments; the reason is that for multi-asset class investing or when we have derivatives in the portfolio, stylized facts become important and having only some statistics is not sufficient. Refining the distribution as much as possible, we can perform a fairly recent method called full-scale portfolio optimization in [AK07], which takes the simulated distribution as input and can process all kinds of utility function. Full scale optimization has been shown to achieve significantly better results than mean-variance optimization, see [AK07], when the investors do not have quadratic utility, for example when the utility is kinked or S-shaped.

Bibliography

[ACFB09] Kjersti Aas, Claudia Czado, Arnoldo Frigessi, and Henrik Bakken. Pair-copula constructions of multiple dependence. *Insurance: Mathematics and economics*, 44(2):182–198, 2009.

[ADEH99] Philippe Artzner, Freddy Delbaen, Jean-Marc Eber, and David Heath. Coherent measures of risk. *Mathematical finance*, 9(3):203–228, 1999.

[AK07] Timothy Adler and Mark Kritzman. Meanvariance versus full-scale optimisation: In and out of sample. *Journal of Asset Management*, 7(5):302–311, 2007.

[Ale09] Carol Alexander. *Market Risk Analysis, Value at Risk Models*, volume 4. John Wiley & Sons, 2009.

[Bre01] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[CS12] Shane A Corwin and Paul Schultz. A simple way to estimate bid-ask spreads from daily high and low prices. *The Journal of Finance*, 67(2):719–760, 2012.

[EFM05] Paul Embrechts, Rdiger Frey, and Alexander McNeil. Quantitative risk management. *Princeton Series in Finance, Princeton*, 10, 2005.

[FS99] Vladimir Filimonov and Didier Sornette. A stable and robust calibration scheme of the log-periodic power law model. *Physica A: Statistical Mechanics and its Applications*, 392(17):3698–3707, 2013, `http://arxiv.org/abs/1108.0099`.

[JLS71] Anders Johansen, Olivier Ledoit, and Didier Sornette. Crashes as critical points. *International Journal of Theoretical and Applied Finance*, 3(02):219–255, 2000, `http://xxx.lanl.gov/abs/cond-mat/9810071`.

[Joe96] Harry Joe. Families of m-variate distributions with given margins and m (m-1)/2 bivariate dependence parameters. *Lecture Notes-Monograph Series*, pages 120–141, 1996.

[LM62] DN Lawley and AE Maxwell. Factor analysis as a statistical method. *Journal of the Royal Statistical Society. Series D (The Statistician)*, 12(3):209–229, 1962.

[LR10] Moshe Levy and Richard Roll. The market portfolio may be mean/variance efficient after all. *Review of Financial Studies*, 23(6):2464–2491, 2010.

[MAK11] Attilio Meucci, David Ardia, and Simon Keel. Fully flexible views in multivariate normal markets. 2011.

[MAK12] Attilio Meucci, David Ardia, and Simon Keel. Fully flexible extreme views. *Journal of Risk*, 14(2):39, 2012.

[Mar52] Harry Markowitz. Portfolio selection*. *The journal of finance*, 7(1):77–91, 1952.

[Meu08] Attilio Meucci. Fully flexible views: Theory and practice. *Fully Flexible Views: Theory and Practice, Risk*, 21(10):97–102, 2008.

[Meu10] Attilio Meucci. Historical scenarios with fully flexible probabilities. *GARP Risk Professional*, pages 47–51, 2010.

[Meu11a] Attilio Meucci. A new breed of copulas for risk and portfolio management. *Risk*, 24(9):122–126, 2011.

[Meu11b] Attilio Meucci. The prayer: Ten-step checklist for advanced risk and portfolio management. *Ten-Step Checklist for Advanced Risk and Portfolio Management (February 2, 2011)*, 2011.

[Meu12a] Attilio Meucci. Effective number of scenarios in fully flexible probabilities. *GARP Risk Professional*, pages 32–35, 2012.

[Meu12b] Attilio Meucci. A fully integrated liquidity and market risk model. *Financial Analysts Journal*, 68(6):94, 2012.

[NMSW14] Xiaohui Ni, Yannick Malevergne, Didier Sornette, and Peter Woehrmann. Robust reverse engineering of cross-sectional returns and improved portfolio allocation performance using the capm. *The Journal of Portfolio Management*, 37(4):76–85, 2011, `http://ssrn.com/abstract=1753014`.

[Sha98] William F Sharpe. The sharpe ratio. *Streetwise–the Best of the Journal of Portfolio Management*, pages 169–185, 1998.

[Skl59] M Sklar. *Fonctions de répartition à n dimensions et leurs marges*. Université Paris 8, 1959.

[Sor09] Didier Sornette. *Why stock markets crash: critical events in complex financial systems*. Princeton University Press, 2009.

[SWYZ71] Didier Sornette, Ryan Woodard, Wanfeng Yan, and Wei-Xing Zhou. Clarifications to questions and criticisms on the johansen–ledoit–sornette financial bubble model. *Physica A: Statistical Mechanics and its Applications*, 392(19):4417–4428, 2013, `http://arxiv.org/abs/1107.3171`.

[YRWS12] Wanfeng Yan, Reda Rebib, Ryan Woodard, and Didier Sornette. Detection of crashes and rebounds in major equity markets. *International Journal of Portfolio Analysis and Management*, 1(1):59–79, 2012.

[Zho12] Zhi-Hua Zhou. *Ensemble methods: foundations and algorithms*. CRC Press, 2012.