
Marius Hofert, Amir Memartoluie, David Saunders, Tony Wirjanto

2016-06-22

Abstract

Numerical challenges inherent in algorithms for computing worst Value-at-Risk in homogeneous portfolios are identified and solutions as well as words of warning concerning their implementation are provided. Furthermore, both conceptual and computational improvements to the Rearrangement Algorithm for approximating worst Value-at-Risk for portfolios with arbitrary marginal loss distributions are given. In particular, a novel Adaptive Rearrangement Algorithm is introduced and investigated. These algorithms are implemented using the R package qrmtools and may be of interest in any context in which it is required to find columnwise permutations of a matrix such that the minimal (maximal) row sum is maximized (minimized).

Keywords
Risk aggregation, model uncertainty, Value-at-Risk, Rearrangement Algorithm.

MSC2010
65C60, 62P05

1 Introduction

An integral part of Quantitative Risk Management is to analyze the one-period ahead vector of losses \( L = (L_1, \ldots, L_d) \), where \( L_j \) represents the loss (a random variable) associated with a given business line or risk type with counterparty \( j, j \in \{1, \ldots, d\} \), over a fixed time horizon. For financial institutions, the aggregated loss

\[ L^+ = \sum_{j=1}^{d} L_j \]

is of particular interest. Under Pillar I of the Basel II Accords, financial institutions are required to set aside capital to manage market, credit and operational risk. To this end a risk measure \( \rho(\cdot) \) is used to map the aggregate position \( L^+ \) to \( \rho(L^+) \in \mathbb{R} \) for obtaining the amount of capital required to account for future losses over a predetermined time period. As a risk measure, Value-at-Risk (VaR\(_\alpha\)) has been widely adopted by the financial industry since the mid nineties. It is defined as the \( \alpha \)-quantile of the distribution function \( F_{L^+} \) of \( L^+ \), i.e.,

\[ \text{VaR}_\alpha(L^+) = F_{L^+}^{-}(\alpha) = \inf\{x \in \mathbb{R} : F_{L^+}(x) \geq \alpha\}, \]

where \( F_{L^+}^{-} \) denotes the quantile function of \( F_{L^+} \); see Embrechts and Hofert [17] for more details. There are various methods for estimating the marginal loss distributions \( F_1, \ldots, F_d \) of \( L_1, \ldots, L_d \), respectively, but capturing the \( d \)-variate dependence structure (i.e., the underlying copula \( C \)) of \( L \) is often more difficult. This is due to the fact that typically not much is known about \( C \) and estimation is often not feasible (e.g., for rare-event losses occurring in different geographic regions).
In this work we focus on the case where $C$ is unknown and study the problem of computing $\text{VaR}_\alpha(L^+)$ bounds $\text{VaR}_\alpha(L^+)$ and $\text{VaR}_\alpha(L^+)$, where
\[
    \text{VaR}_\alpha(L^+) = \inf\{\text{VaR}_\alpha(L^+) : L_1 \sim F_1, \ldots, L_d \sim F_d\}, \\
    \text{VaR}_\alpha(L^+) = \sup\{\text{VaR}_\alpha(L^+) : L_1 \sim F_1, \ldots, L_d \sim F_d\},
\]
i.e., $\text{VaR}_\alpha(L^+)$ and $\text{VaR}_\alpha(L^+)$ denote the best and the worst $\text{VaR}_\alpha(L^+)$ over all distributions of $L$ with marginals $F_1, \ldots, F_d$, respectively. This problem is non-trivial as $\text{VaR}_\alpha(L^+)$ can be superadditive, so, in particular, computing $\text{VaR}_\alpha(L^+)$ in the comonotonic case (of all losses being equal in distribution to $F_1(U), \ldots, F_d(U)$ for $U \sim U[0, 1]$) may not lead to $\text{VaR}_\alpha(L^+)$.

The dependence uncertainty interval $[\text{VaR}_\alpha(L^+), \text{VaR}_\alpha(L^+)]$ can be wide, but financial firms are interested in computing it (often in a high dimensional $d$) to determine their risk capital for $L^+$ within this range. As we show in this work, even for small value of $d$ (and other moderate parameter choices) the required computations can be challenging. The case of partial information about $C$ (and thus a possibly smaller dependence uncertainty spread $\text{VaR}_\alpha(L^+) - \text{VaR}_\alpha(L^+)$), is studied by Bernard et al. [4], Bernard and Vanduffel [2], Bernard and Vanduffel [3], Bignozzi et al. [8] and Puccetti et al. [26], where the RA also has been shown to be a useful tool.

We investigate solutions in the homogeneous case (i.e., $F_1 = \cdots = F_d$) which are considered to be “explicit”; see Embrechts et al. [15]. In the general, inhomogeneous case (i.e., not all $F_j$’s necessarily being equal), we consider and improve the Rearrangement Algorithm (RA) of Embrechts et al. [16] for computing $\text{VaR}_\alpha(L^+)$ and $\text{VaR}_\alpha(L^+)$. All presented algorithms have been implemented in the R package qrmtools; see also the accompanying vignette VaR_bounds which provides further results, numerical investigations, diagnostic checks and an application. The results presented in this paper can be reproduced with the package and vignette (and, obviously, other parameter values can be chosen).

Another direction to improve the RA is by considering so-called block rearrangements (thus leading to the so-called block RA), where blocks of columns (instead of single columns) are rearranged at a time. This idea was introduced in Bernard et al. [6] and further applied in Bernard et al. [4]. It is related to the notion of $\Sigma$-countermonotonicity studied in Puccetti and Wang [25]. In addition, Bernard and McLeish [1] also address the issue of non-convergence of the block RA to a global optimum by applying Markov Chain Monte Carlo methods.

The remaining parts of this paper are organized as follows. In Section 2 we highlight and solve numerical challenges that practitioners may face when implementing theoretical solutions for $\text{VaR}_\alpha(L^+)$ in the homogeneous case $F_1 = \cdots = F_d$. Section 3 presents the main concept underlying the Rearrangement Algorithm for computing $\text{VaR}_\alpha(L^+)$ and $\text{VaR}_\alpha(L^+)$, levies criticism on its tuning parameters and investigates its empirical performance under various scenarios. Section 4 then presents a conceptually and numerically improved version of the RA, which we call the Adaptive Rearrangement Algorithm (ARA), for calculating $\text{VaR}_\alpha(L^+)$ and $\text{VaR}_\alpha(L^+)$. Section 5 concludes. Proofs of results stated in the main body are relegated to an appendix.

### 2 Known optimal solutions in the homogeneous case and their tractability

In order to assess the quality of general algorithms such as the RA, we need to know (at least some) optimal solutions with which we can compare such algorithms. Embrechts et al. [16, Proposition 4] and Embrechts et al. [15, Proposition 1] present formulas for obtaining $\text{VaR}_\alpha(L^+)$ in the homogeneous case. In this section, we address the corresponding numerical aspects and algorithmic improvements.

1Note that the formulas for both $\text{VaR}_\alpha(L^+)$ and $\text{VaR}_\alpha(L^+)$ have typographical errors which we correct below.
Improved Algorithms for Computing Worst VaR

We assume $d \geq 3$ throughout; for $d = 2$, Embrechts et al. [16, Proposition 2] provide an explicit solution for computing $\nabla \text{VaR}_\alpha(L^+)$ (under certain weak assumptions).

### 2.1 Crude bounds for any $\text{VaR}_\alpha(L^+)$

The following lemma provides (crude) bounds for $\text{VaR}_\alpha(L^+)$ which are useful for computing initial intervals (see Section 2.2) and conducting sanity checks. Note that we do not make any (moment or other) assumptions on the involved marginal loss distributions; in fact, we do not even require them to be equal.

**Lemma 2.1 (Crude bounds for $\text{VaR}_\alpha(L^+)$)**

Let $L_j \sim F_j$, $j \in \{1, \ldots, d\}$. For any $\alpha \in (0, 1)$,

$$d \min_j F_j^{-}(\alpha/d) \leq \text{VaR}_\alpha(L^+) \leq d \max_j F_j^{-}\left(1 - \frac{1 - \alpha}{d}\right),$$

where $F_j^{-}$ denotes the quantile function of $F_j$.

Note that the bounds (1) can be computed with the function `crude_VaR_bounds()` in the R package `qrmtools`.

### 2.2 The dual bound approach for computing $\nabla \text{VaR}_\alpha(L^+)$

This approach for computing $\nabla \text{VaR}_\alpha(L^+)$ in the homogeneous case with margin(s) $F$ is presented in Embrechts et al. [16, Proposition 4] and termed a dual bound approach in what follows; note that there is no corresponding algorithm for computing $\text{VaR}_\alpha(L^+)$ with this approach. In the remaining part of this subsection we assume that $F(0) = 0$, $F(x) < 1$ for all $x \in [0, \infty)$ and that $F$ is absolutely continuous with an ultimately decreasing density. Let

$$D(s, t) = \frac{d}{s - dt} \int_t^{s-(d-1)t} \tilde{F}(x) \, dx \quad \text{and} \quad D(s) = \min_{t \in [0,s/d]} D(s, t),$$

where $\tilde{F}$ denotes the survival function of $F$, i.e., $\tilde{F}(x) = 1 - F(x)$. In comparison to Embrechts et al. [16, Proposition 4], the dual bound $D$ here uses a compact interval for $t$ (and thus $\min\{\cdot\}$) by our requirement that $F(0) = 0$ and since $\lim_{t \to s/d} D(s, t) = d\tilde{F}(s/d)$ by l’Hôpital’s Rule. The procedure for computing $\nabla \text{VaR}_\alpha(L^+)$ according to Embrechts et al. [16, Proposition 4] can now be given as follows.

**Algorithm 2.2 (Computing $\nabla \text{VaR}_\alpha(L^+)$ according to the dual bound approach)**

1. Specify initial intervals $[s_l, s_u]$ and $[t_l, t_u]$.
2. Inner root-finding in $t$: For each considered $s \in [s_l, s_u]$, compute $D(s)$ by iterating over $t \in [t_l, t_u]$ until a $t^*$ is found for which $h(s, t^*) = 0$, where
   
   $$h(s, t) := D(s, t) - (\tilde{F}(t) + (d - 1)\tilde{F}(s - (d - 1)t)).$$

   Then $D(s) = \tilde{F}(t^*) + (d - 1)\tilde{F}(s - (d - 1)t^*)$.
3. Outer root-finding in $s$: Iterate Step 2) over $s \in [s_l, s_u]$ until an $s^*$ is found for which $D(s^*) = 1 - \alpha$. Then return $s^* = \nabla \text{VaR}_\alpha(L^+)$.

Algorithm 2.2 is implemented in the function `worst_VaR_hom(\ldots, method="dual")` in the R package `qrmtools`; the dual bound $D$ is available via `dual_bound()`. It requires a one-dimensional numerical integration (unless $F$ can be integrated explicitly) within two nested calls of a root-finding algorithm (`uniroot()` in R). Note that Algorithm 2.2 requires a specification of the two initial
Algorithm 2.2 will not be able to locate a root. There is currently no (good) solution known on how to
locate a root. The approach mentioned in Embrechts et al. [15, Proposition 1] is termed
Wang’s approach for computing \( \text{VaR}_\alpha(L^+) \)

Theory

The approach mentioned in Embrechts et al. [15, Proposition 1] is termed Wang’s approach here. It originates from Wang et al. [33, Corollary 3.7] and, thus, strictly speaking, precedes the dual bound approach of Embrechts et al. [16, Proposition 4]. It is conceptually simpler and numerically more stable than the dual bound approach, yet it is still not straightforward to apply. For notational simplicity, let us introduce

\[
    a_c = \alpha + (d-1)c, \quad b_c = 1 - c, \tag{3}
\]

for \( c \in [0, (1-\alpha)/d] \) (so that \( a_c \in [\alpha, 1-(1-\alpha)/d] \) and \( b_c \in [1-(1-\alpha)/d, 1] \)) and

\[
    \bar{I}(c) := \frac{1}{b_c - a_c} \int_{a_c}^{b_c} F^-(y) \, dy, \quad c \in (0, (1 - \alpha) / d),
\]

with the assumption that \( F \) admits a density which is positive and decreasing on \([\beta, \infty)\) for some \( \beta \leq F^-(\alpha) \). Then, for \( L \sim F \),

\[
    \text{VaR}_\alpha(L^+) = \mathbb{E}[L \mid L \in [F^-(a_c), F^-(b_c)]], \quad \alpha \in [F(\beta), 1), \tag{4}
\]

where \( c \) (typically depending on \( d, \alpha \)) is the smallest number in \((0, (1 - \alpha)/d)\) such that

\[
    \bar{I}(c) \geq \frac{d - 1}{d} F^-(a_c) + \frac{1}{d} F^-(b_c). \tag{5}
\]

In contrast to what is given in Embrechts et al. [15], note that \((0, (1 - \alpha)/d)\) has to exclude 0 since otherwise, for \( \text{Par}(\theta) \) margins with \( \theta \in (0, 1) \), \( c \) equals 0 and thus, erroneously, \( \text{VaR}_\alpha(L^+) = \infty \). If
Algorithm 2.4 (Computing $\text{worst}_\alpha(L^+)$ according to Wang’s approach)

1) Specify an initial interval $[c_l, c_u]$ with $0 \leq c_l < c_u < (1 - \alpha)/d$.
2) Root-finding in $c$: Iterate over $c \in [c_l, c_u]$ until a $c^*$ is found for which $h(c^*) = 0$, where

$$h(c) := I(c) - \left(\frac{d - 1}{d} F^-(a_c) + \frac{1}{d} F^-(b_c)\right), \quad c \in (0, (1 - \alpha)/d].$$

Then return $(d - 1)F^-(a_c) + F^-(b_c)$.

This procedure is implemented in the function `worst_Var_hom(..., method="Wang")` in the R package `qrmtools` with numerical integration via R’s `integrate()` for computing $I(c)$; the function `worst_Var_hom(..., method="Wang.Par")` makes use of (6).

The following proposition shows that the root-finding problem in Step 2) of Algorithm 2.4 is well-defined in the case of Pareto margins for all $\theta > 0$ (including the infinite-mean case); for other distributions under more restrictive assumptions, see Bernard et al. [5].

**Proposition 2.5**

Let $F(x) = 1 - (1 + x)^{-\theta}$, $\theta > 0$, be the distribution function of the Par($\theta$) distribution. Then $h$ in Step 2) of Algorithm 2.4 has a unique root on $(0, (1 - \alpha)/d)$, for all $\alpha \in (0, 1)$ and $d > 2$.

**Practice**

Let us now focus on the case of Par($\theta$) margins (see `worst_Var_hom(..., method="Wang.Par")`) and, in particular, how to choose the initial interval $[c_l, c_u]$ in Algorithm 2.4. We first consider $c_l$. $I$ satisfies

$I(0) = \frac{1}{1-\theta} \int_0^1 F^-(y) \, dy = \text{ES}_a(L)$, i.e., $I(0)$ is the expected shortfall of $L \sim F$ at confidence level $\alpha$. If $L$ has a finite first moment, then $I(0)$ is finite. Therefore, $h(0)$ is finite (if $F^-(1) < \infty$) or $-\infty$ (if $F^-(1) = \infty$). Either way, one can take $c_l = 0$. However, if $L \sim F$ has an infinite first moment (see, e.g., Hofert and Wüthrich [21] or Chavez-Demoulin et al. [9] for situations in which this can happen), then $I(0) = \infty$ and $F^-(1) = \infty$, so $h(0)$ is not well defined; this happens, e.g., if $F$ is Par($\theta$) with $\theta \in (0, 1]$. In such a case, we are forced to choose $c_l \in (0, (1 - \alpha)/d)$; see the following proposition for how this can be done theoretically. Concerning $c_u$, note that l’Hôpital’s Rule implies that $I(c_u) = F^-(1 - (1 - \alpha)/d)$ and thus $h((1 - \alpha)/d) = 0$. We thus have a similar problem (a root at the upper endpoint of the initial interval) to the computation of the dual bound. However, here we can construct a suitable $c_u < (1 - \alpha)/d$; see the following proposition.
Proposition 2.6 (Computing $c_l, c_u$ for Par($\theta$) risks)

Let $F$ be the distribution function of a Par($\theta$) distribution, $\theta > 0$. Then $c_l$ and $c_u$ in Algorithm 2.4 can be chosen as

$$c_l = \begin{cases} 
\frac{(1-\theta)(1-\alpha)}{d}, & \text{if } \theta \in (0, 1), \\
\frac{1-\alpha}{d(1+\theta)+d-1}, & \text{if } \theta = 1, \\
\frac{1-\alpha}{(d/	heta-1)+1} + d-1, & \text{if } \theta \in (1, \infty),
\end{cases}$$

$$c_u = \begin{cases} 
\frac{(1-\alpha)(d-1+\theta)}{(d-1)(2\theta+d)}, & \text{if } \theta \neq 1, \\
\frac{1-\alpha}{3\delta/2-1}, & \text{if } \theta = 1.
\end{cases}$$

In the following example, we briefly address several numerical hurdles we had to overcome when implementing `worst_VaR_hom(..., method="Wang.Par")`; see the vignette `VaR_bounds` for more details.

Example 2.7 (Comparison of the approaches for Par($\theta$) risks)

For obtaining numerically reliable results, one has to be careful when computing the root of $h$ for $c \in (0, (1-\alpha)/d)$. As an example, consider Par($\theta$) risks and the confidence level $\alpha = 0.99$. Figure 1 compares Wang’s approach (using numerical integration; see `worst_VaR_hom(..., method="Wang")`), Wang’s approach (with an analytical formula for the integral $I(c)$ but `uniroot()`’s default tolerance; see the vignette `VaR_bounds`), Wang’s approach (with an analytical formula for the integral $I(c)$ and auxiliary function $h$ transformed to $(1, \infty)$; see the vignette `VaR_bounds`), Wang’s approach (with analytical formula for the integral $I(c)$, smaller `uniroot()` tolerance and adjusted initial interval; see `worst_VaR_hom(..., method="Wang.Par")`), and the lower and upper bounds obtained from the RA (with absolute tolerance 0); see Section 3 and RA(). All of the results are divided by the values obtained from the dual bound approach to facilitate comparison.

As can be seen, choosing a smaller root-finding tolerance is crucial. Figure 1 shows what could occur if this is not done (our procedure chooses MATLAB’s default 2.2204·10^{-16} instead of the much larger `uniroot()` default `Machine$double.eps^0.25`). Furthermore, it turned out to be required to adjust the theoretically valid initial interval described in Proposition 2.6 in order to guarantee that $h$ is numerically of opposite sign at the interval end points. In particular, `worst_VaR_hom(..., method="Wang.Par")` chooses $c_l/2$ as a lower end point (with $c_l$ as in Proposition 2.6) in the case $\theta \neq 1$.

These problems are described in detail in Section 1.4 of the vignette `VaR_bounds`, where we also show that transforming the auxiliary function $h$ to a root-finding problem on $(1, \infty)$ as described in the proof of Proposition 2.6 does not only require a smaller root-finding tolerance but also an extended initial interval and, furthermore, it faces a cancellation problem (which can be solved, though); see also the left-hand side of Figure 3, where we compare this approach to `worst_VaR_hom(..., method="Wang.Par")` after fixing these numerical issues.

In short, one should exercise caution in implementing supposedly “explicit solutions” for computing $\text{VaR}_\alpha(L^+)$ or $\overline{\text{VaR}}_\alpha(L^+)$ in the homogeneous case with Par($\theta$) (and most likely also other) margins.

Let us again stress how important the initial interval $[c_l, c_u]$ is. One could be tempted to simply choose $c_u = (1-\alpha)/d$ and force the auxiliary function $h$ to be of opposite sign at $c_u$, e.g., by setting $h(c_u)$ to `Machine$double.xmin`, a positive but small number. Figure 2 (left-hand side) shows how Figure 1 looks like in this case (here it is standardized with respect to the upper bound obtained from the RA). Figure 2 (right-hand side) shows the implied $\text{VaR}_\alpha(L^+)$. In particular, the computed $\text{VaR}_\alpha(L^+)$ values are not monotone in $\alpha$ anymore (and thus not correct).

After carefully addressing the numerical issues, we can now consider $\text{VaR}_\alpha(L^+)$ and $\overline{\text{VaR}}_\alpha(L^+)$ from a different perspective. The right-hand side of Figure 3 shows $\text{VaR}_\alpha(L^+)$ and $\overline{\text{VaR}}_\alpha(L^+)$ as functions of the dimension $d$. The linearity of $\overline{\text{VaR}}_\alpha(L^+)$ in the log-log scale suggests that $\overline{\text{VaR}}_\alpha(L^+)$ is actually a power function in $d$. To the best of our knowledge, this is not known (nor theoretically justified) yet.
Figure 1 Comparisons of Wang’s approach (using numerical integration; see `worst_VaR_hom(..., method="Wang")`), Wang’s approach (with an analytical formula for the integral $\tilde{I}(c)$ but `uniroot()`’s default tolerance; see the vignette `VaR_bounds`), Wang’s approach (with an analytical formula for the integral $\tilde{I}(c)$ and auxiliary function $h$ transformed to $(1, \infty)$; see the vignette `VaR_bounds`), Wang’s approach (with analytical formula for the integral $\tilde{I}(c)$, smaller `uniroot()` tolerance and adjusted initial interval; see `worst_VaR_hom(..., method="Wang.Par")`), and the lower and upper bounds obtained from the RA; all of the results are divided by the values obtained from the dual bound approach to facilitate comparison. The left-hand side shows the case with $d = 8$, the right-hand side with $d = 100$.

Figure 2 Comparison of various methods for computing $\text{VaR}_{0.99}(L^+)$ (left-hand side) and $\text{VaR}_\alpha(L^+)$ as a function of $\alpha$ (right-hand side) for $\text{Par}(\theta)$ risks with $h((1-\alpha)/d)$ being naively adjusted to `.Machine$double.xmin`. 
The RA can be applied to approximate the best Value-at-Risk $\text{VaR}_\alpha(L^+)$ or the worst Value-at-Risk $\overline{\text{VaR}}_\alpha(L^+)$ for any set of marginals $F_j$, $j \in \{1, \ldots, d\}$. In what follows our focus is mainly on $\overline{\text{VaR}}_\alpha(L^+)$; our implementation RA() in the R package qrmtools also addresses $\text{VaR}_\alpha(X^+)$ and $\overline{\text{VaR}}_\alpha(X^+)$, respectively.

To understand the algorithm, note that two columns $a, b \in \mathbb{R}^N$ are called *oppositely ordered* if for all $i, j \in \{1, \ldots, N\}$ we have $(a_i - a_j)(b_i - b_j) \leq 0$. Given a number $N$ of discretization points of the marginal quantile functions $F_1^{-}, \ldots, F_d^{-}$ above $\alpha$ (see Steps 2.1 and 3.1) of Algorithm 3.1 below), the RA constructs two $(N, d)$-matrices, denoted by $X^{\alpha}$ and $\overline{X}^{\alpha}$, respectively; the first matrix aims at constructing an approximation of $\overline{\text{VaR}}_\alpha(L^+)$ from below, the second matrix is used to construct an approximation of $\text{VaR}_\alpha(L^+)$ from above. Separately for each of these matrices, the RA iterates over its columns and rearranges each of them so that it is oppositely ordered to the sum of all other columns. This is repeated until the minimal row sum

$$s(X) = \min_{1 \leq i \leq N} \sum_{1 \leq j \leq d} x_{ij}$$
(for $X = (x_{ij})$ being one of the said $(N,d)$-matrices) changes by less than a given (convergence) tolerance $\varepsilon \geq 0$. The RA for $\text{VaR}_\alpha(L^+)$ thus aims at solving the maximin problem. The intuition behind this approach is to minimize the variance of the conditional distribution of $L^+ | L^+ > F_{L^+}^{-1}(\alpha)$ so that it can concentrate more of the $1 - \alpha$ mass of $F_{L^+}$ in its tail. This pushes $\text{VaR}_\alpha(L^+)$ further up. As Embrechts et al. [16] state, one then typically ends up with two matrices whose minimal row sums are close to each other and roughly equal to $\text{VaR}_\alpha(L^+)$. Note that if one iteration over all columns of one of the matrices does not lead to any change in that matrix, then each column of the matrix is oppositely ordered to the sum of all others and thus there is also no change in the minimal row sum (but the converse is not necessarily true; see below).

The version of the RA given below contains slightly more information than the one stated in Embrechts et al. [16]; e.g., how infinite quantiles are dealt with. For more features of the actual implementation which is an improved version of the one given below (see Section 3.2), see the underlying workhorse `rearrange()`.

Algorithm 3.1 (RA for computing $\text{VaR}_\alpha(L^+)$)

1) Fix a confidence level $\alpha \in (0, 1)$, marginal quantile functions $F_1^-, \ldots, F_d^-$, an integer $N \in \mathbb{N}$ and the desired (absolute) convergence tolerance $\varepsilon \geq 0$.

2) Compute the lower bound:

2.1) Define the matrix $X^\alpha = (x_{ij}^\alpha)$ for $x_{ij}^\alpha = F_j^{-1}(\alpha + \frac{(1-\alpha)(i-1)}{N})$, $i \in \{1, \ldots, N\}$, $j \in \{1, \ldots, d\}$.

2.2) Permute randomly the elements in each column of $X^\alpha$.

2.3) Set $Y^\alpha = X^\alpha$. For $1 \leq j \leq d$, rearrange the $j$th column of the matrix $Y^\alpha$ so that it becomes oppositely ordered to the sum of all other columns.

2.4) While $s(Y^\alpha) - s(X^\alpha) > \varepsilon$, set $X^\alpha$ to $Y^\alpha$ and repeat Step 2.3.

2.5) Set $\underline{x}_N = s(Y^\alpha)$.

3) Compute the upper bound:

3.1) Define the matrix $\overline{X}^\alpha = (\overline{x}_{ij}^\alpha)$ for $\overline{x}_{ij}^\alpha = F_j^{-1}(\alpha + \frac{(1-\alpha)i}{N})$, $i \in \{1, \ldots, N\}$, $j \in \{1, \ldots, d\}$. If (for $i = N$ and) for any $j \in \{1, \ldots, d\}$, $F_j^{-1}(1) = \infty$, adjust it to $F_j^{-1}(\alpha + \frac{(1-\alpha)(N-1/2)}{N})$.

3.2) Permute randomly the elements in each column of $\overline{X}^\alpha$.

3.3) Set $\overline{Y}^\alpha = \overline{X}^\alpha$. For $1 \leq j \leq d$, rearrange the $j$th column of the matrix $\overline{Y}^\alpha$ so that it becomes oppositely ordered to the sum of all other columns.

3.4) While $s(\overline{Y}^\alpha) - s(\overline{X}^\alpha) > \varepsilon$, set $\overline{X}^\alpha$ to $\overline{Y}^\alpha$ and repeat Step 3.3.

3.5) Set $\overline{x}_N = s(\overline{Y}^\alpha)$.

4) Return $(\underline{x}_N, \overline{x}_N)$.

As mentioned earlier, the main feature of the RA is to iterate over all columns and oppositely order each of them with respect to the sum of all others (see Steps 2.3 and 3.3). This procedure aims at reducing the variance of the row sums with each rearrangement. Note that it does not necessarily reach an optimal solution of the maximin problem (see, e.g., Haus [20, Lemma 6] for a counter-example) and thus the convergence $|\overline{x}_N - \underline{x}_N| \to 0$ is not guaranteed in this approach. In order to reduce the possibility of this happening in practice, the randomization of the initial input in Steps 2.2) and 3.2) has been put in place; however, see our study in Section 2.3 of the vignette.
**VaR_bounds** concerning the possible rearranged output matrices and the influence of the underlying sorting algorithm on the outcome.

### 3.2 Conceptual and numerical improvements

Some words of warning are in order here. Besides the confidence level \( \alpha \) and the marginal quantile functions \( F_1^{-1}, \ldots, F_d^{-1} \), RA relies on two sources of input, namely \( N \in \mathbb{N} \) and \( \varepsilon \geq 0 \), for which Embrechts et al. [16] do not provide guidance on reasonable defaults. Concerning \( N \), it obviously needs to be “sufficiently large”, but a practitioner is left to make a choice about this value. Another issue is the use of the absolute tolerance \( \varepsilon \) in the algorithm. There are two problems. The first problem is that it is more natural to use a relative instead of an absolute tolerance in this context. Without (roughly) knowing the minimal row sum in Steps 2.4) and 3.4), a pre-specified absolute tolerance does not guarantee that the change in the minimal row sum from \( X^\alpha \) to \( Y^\alpha \) is of the right order (and this order depends at least on \( d \) and the chosen quantile functions). If \( \varepsilon \) is chosen to be too large, the computed bounds \( s_N \) and \( \bar{s}_N \) would carry too much uncertainty, whereas if it is selected to be too small, an unnecessarily long run time results; the latter seems to be the case for Embrechts et al. [16, Table 3], where the chosen \( \varepsilon = 0.1 \) is roughly 0.000004% of the computed \( \text{VaR}_{0.99}(L^+) \).

The second problem is that the absolute tolerance \( \varepsilon \) is only used to check individual “convergence” of \( s_N \) and of \( \bar{s}_N \). It does not guarantee that \( s_N \) and \( \bar{s}_N \) are sufficiently close to provide a reasonable approximation to \( \text{VaR}_\alpha(L^+) \). We are aware of the theoretical hurdles underlying the algorithm which are still open questions at this point (e.g., the probability of convergence of \( s_N \) and \( \bar{s}_N \) to \( \text{VaR}_\alpha(L^+) \) or that \( \text{VaR}_\alpha(L^+) \leq \bar{s}_N \) for sufficiently large \( N \)), but from a computational point of view one should still check that \( s_N \) and \( \bar{s}_N \) are close to each other. Also, the algorithm should return convergence and other useful information, e.g., the relative rearrangement range \( |(\bar{s}_N - s_N)/\bar{s}_N| \), the actual individual absolute tolerances reached when computing \( s_N \) and \( \bar{s}_N \), the number of column rearrangements used, logical variables indicating whether the individual absolute tolerances have been reached, the number of column rearrangements for \( s_N \) and \( \bar{s}_N \), the row sums computed after each column rearrangement, the constructed input matrices \( X^\alpha, \bar{X}^\alpha \) and the corresponding rearranged, final matrices \( Y^\alpha, \bar{Y}^\alpha \); see RA() in the R package qrmtools for such information.

Another suboptimal design of the RA is that it iterates over all \( d \) columns before checking the termination conditions; see Steps 2.3) and 3.3) of Algorithm 3.1. Our underlying workhorse rearrange() keeps track of the column rearrangements of the last \( d \) considered columns and can thus terminate after rearranging any column (not only the last one); see also Algorithm 4.1 below. This saves run time (despite the “tracking” overhead). We advise the interested reader to have a look at the source code of rearrange() for further numerical and run-time improvements (fast accessing of columns via lists; avoiding having to compute the row sums over all but the current column; an extended tracing feature), some of which are mentioned in the vignette VaR_bounds.

### 3.3 Empirical performance under various scenarios

In order to empirically investigate the performance of the RA, we consider two studies, each of which addresses four cases; we thus consider eight scenarios. In terms of studies, we consider the following:

- **Study 1** (\( N \) running, \( d \) fixed): \( N \in \{2^7, 2^8, \ldots, 2^{17}\} \) and \( d = 20 \);
- **Study 2** (\( N \) fixed, \( d \) running): \( N = 2^8 = 256 \) and \( d \in \{2^2, 2^3, \ldots, 2^{10}\} \).

These choices allow us to investigate the impact of the upper tail discretization parameter \( N \) (in Study 1) and the impact of the number of risk factors \( d \) (in Study 2) on the performance of the RA. In terms of cases, we consider the following different marginal tail behaviors based on the Pareto distribution function \( F_j(x) = 1 - (1 + x)^{-\theta_j} \):

\[ F_1(x) = \frac{x^{2\theta}}{\theta_1^2}, \quad F_2(x) = \frac{1 - (1 + x)^{-\theta_2}}{1 - (1 + x)^{-\theta_3}}, \quad F_3(x) = \frac{1 - (1 + x)^{-\theta_4}}{1 - (1 + x)^{-\theta_5}}, \quad F_4(x) = \frac{1 - (1 + x)^{-\theta_6}}{1 - (1 + x)^{-\theta_7}}. \]
Case HH : $\theta_1, \ldots, \theta_d$ form an equidistant sequence from 0.6 to 0.4; this case represents a portfolio with all marginal loss distributions being heavy-tailed (slightly increasing in heaviness).

Case LH : $\theta_1, \ldots, \theta_d$ form an equidistant sequence from 1.5 to 0.5; this case represents a portfolio with marginals with different tail behaviors ranging from comparably light-tailed to very heavy-tailed distributions.

Case LL : $\theta_1, \ldots, \theta_d$ form an equidistant sequence from 1.6 to 1.4; this case represents a portfolio with all marginal loss distributions being comparably light-tailed.

Case LH_1 : $\theta_1, \ldots, \theta_{d-1}$ are chosen as in Case LL and $\theta_d = 0.5$; this case represents a portfolio all marginal loss distributions being light-tailed except for the last.

To keep the studies tractable, we focus on the confidence level $\alpha = 0.99$ and the absolute convergence tolerance $\varepsilon = 0 \times 10^7$ in all scenarios. Furthermore, we consider $B = 100$ replicated simulation runs in order to provide empirical 95% confidence intervals for the estimated quantities; note that some of them are so tight that they are barely visible in the figures presented below. The $B$ replications only differ due to different permutations of the columns in Steps 2.2) and 3.2) of Algorithm 3.1, everything else is deterministic; this allows us to study the effect of these (initial) randomization steps on the (convergence) results of the RA. Concerning the hardware used, all results were produced on an AMD 3.2 GHz Phenom II X4 955 processor with 8 GB RAM. Note that we only present detailed figures for the results of Study 1; the figures related to Study 2 can be obtained from the authors upon request.

**Results of Study 1 (N running, d fixed)**

The simulation results for Study 1 can be summarized as follows:

- As can be seen in Figure 4, the means over all $B$ computed $\bar{\xi}_N$ and $\bar{\sigma}_N$ converge as $N$ increases.
- Figure 5 indicates that as $N$ increases, so does the mean of the elapsed time (as to be expected). Overall, run time does not drastically depend on the case for our choices of Pareto margins, which is a good feature.
- Figure 6 shows that the maximum number of column rearrangements rarely exceeds $10d$ as $N$ increases; this will be used as a default maximum number of column rearrangements required in the ARA.
- Finally, Figure 7 indicates that the rate at which the number of oppositely ordered columns (based on the final rearranged $Y^{\alpha}$ and $Y^{\alpha^*}$) decreases depends on the characteristics of the marginal distributions involved, i.e., the input matrix $X$. The number of oppositely ordered columns seems particularly small (for large $N$) in Case LL, where essentially only the last column is oppositely ordered to the sum of all others.

![Figure 4](image-url) Study 1: $\text{VaR}_{0.99}$ bounds $\bar{\xi}_N$ and $\bar{\sigma}_N$ for the Cases HH, LH, LL and LH_1 (from left to right).
Figure 5 Study 1: Run times (in s) for the Cases HH, LH, LL and LH$_1$ (from left to right).

Figure 6 Study 1: Number of rearranged columns for the Cases HH, LH, LL and LH$_1$ (from left to right).

Figure 7 Study 1: Number of oppositely ordered columns (of $Y^a$ and $Y'^a$) for the Cases HH, LH, LL and LH$_1$ (from left to right).
Results of Study 2 ($N$ fixed, $d$ running)

In Study 2 we are interested in analyzing the impact of the number of risk factors $d$ on portfolios which exhibit different marginal tail behaviors. The simulation results can be summarized as follows:

- The means over all computed $s_N$ and $\pi_N$ as functions of $d$ diverge from one another; especially in the case where more marginal distributions are heavy-tailed. This is due to the fact that we have kept $N$ the same for all cases in Study 2.
- Similar to what we have seen in Study 1, the mean of the run time of the RA increases as the number of risk factors increases, with Case LL in Study 2 having the smallest run time on average.
- As in Study 1, the mean of the number of rearranged columns is typically below $10^d$.
- The number of oppositely ordered columns (of $Y^\alpha$ and $Y'^\alpha$) increases as $d$ increases; note that this finding does not contradict what we have seen in Study 1 as we have kept $N$ the same here.

4 The Adaptive Rearrangement Algorithm

4.1 How the Adaptive Rearrangement Algorithm works

In this section we present an adaptive version of the RA, termed the Adaptive Rearrangement Algorithm (ARA). This algorithm for computing the bounds $s_N$ and $\pi_N$ for $\text{VaR}_\alpha(L^+)$ or $\text{VaR}_\alpha(L^+)$ (as before, we focus on the latter) provides an algorithmically improved version of the RA, has more meaningful tuning parameters and adaptively chooses the number of discretization points $N$. The ARA is implemented in the R package qrmtools, see the function ARA(). Similar to our RA() implementation, ARA() also relies on the workhorse rearrange() and returns much more information (and can also compute $\text{VaR}_\alpha(L^+)$), but the essential part of the algorithm is given below.

Algorithm 4.1 (ARA for computing $\text{VaR}_\alpha(L^+)$)

1) Fix a confidence level $\alpha \in (0, 1)$, marginal quantile functions $F_1^-, \ldots, F_d^-$, an integer vector $K \in \mathbb{N}^d$, $l \in \mathbb{N}$, (containing the numbers of discretization points which are adaptively used), a bivariate vector of relative convergence tolerances $\varepsilon = (\varepsilon_1, \varepsilon_2)$ (containing the individual relative tolerance $\varepsilon_1$ and the joint relative tolerance $\varepsilon_2$; see below) and the maximal number of iterations used for each $k \in K$.

2) For $N = 2^k$, $k \in K$, do:

2.1) Compute the lower bound:

2.1.1) Define the matrix $X^\alpha = (x^\alpha_{ij})$ for $x^\alpha_{ij} = F_j^-(\alpha + \frac{(1-\alpha)(i-1)}{N})$, $i \in \{1, \ldots, N\}$, $j \in \{1, \ldots, d\}$.

2.1.2) Permute randomly the elements in each column of $X^\alpha$.

2.1.3) Set $Y^\alpha = X^\alpha$. For $j \in \{1, 2, \ldots, d, 1, 2, \ldots, d, \ldots\}$, rearrange the $j$th column of the matrix $Y^\alpha$ so that it becomes oppositely ordered to the sum of all other columns. After having rearranged at least $d+1$ columns, set, after every column rearrangement, the matrix $X^\alpha$ to the matrix $Y^\alpha$ from $d$ rearrangement steps earlier and stop if the maximal number of column rearrangements is reached or if

$$\frac{|s(Y^\alpha) - s(X^\alpha)|}{s(X^\alpha)} \leq \varepsilon_1.$$  \hfill (9)

2.1.4) Set $s_N = s(Y^\alpha)$. 
2.2) Compute the upper bound:

2.2.1) Define the matrix $\mathbf{X}^\alpha = (x^\alpha_{ij})$ for $x^\alpha_{ij} = F_j^{-\alpha + (1-\alpha) \frac{i}{N}}$, $i \in \{1, \ldots, N\}$, $j \in \{1, \ldots, d\}$. If (for $i = N$ and) for any $j \in \{1, \ldots, d\}$, $F_j^{-\alpha + (1-\alpha) \frac{N-1}{2N}} = \infty$, adjust it to $F_j^{-\alpha + (1-\alpha) \frac{N-1}{2N}}$.

2.2.2) Permute randomly the elements in each column of $\mathbf{X}^\alpha$.

2.2.3) Set $\mathbf{Y}^\alpha = \mathbf{X}^\alpha$. For $j \in \{1, 2, \ldots, d, 1, 2, \ldots, d, \ldots\}$, rearrange the $j$th column of the matrix $\mathbf{Y}^\alpha$ so that it becomes oppositely ordered to the sum of all other columns. After having rearranged at least $d+1$ columns, set, after every column rearrangement, the matrix $\mathbf{X}^\alpha$ to the matrix $\mathbf{Y}^\alpha$ from $d$ rearrangement steps earlier and stop if the maximal number of column rearrangements is reached or if

$$\left| \frac{s(\mathbf{Y}^\alpha) - s(\mathbf{X}^\alpha)}{s(\mathbf{X}^\alpha)} \right| \leq \varepsilon_1. \quad (10)$$

2.2.4) Set $\bar{s}_N = s(\mathbf{Y}^\alpha)$.

2.3) Determine convergence based on both the individual and the joint relative convergence tolerances:

If (9) and (10) hold, and if $\left| \frac{\bar{s}_N - \bar{s}_N}{\bar{s}_N} \right| \leq \varepsilon_2$, break.

3) Return $(\bar{s}_N, \bar{s}_N)$.

Concerning the choices of tuning parameters in Algorithm 4.1, note that if $\mathbf{K} = (k = \log_2 N)$, that is, if we have a single number of discretization points, then the ARA reduces to the RA but uses the more meaningful relative instead of absolute tolerances and not only checks what we termed the individual (convergence) tolerance, i.e., the tolerance $\varepsilon_1$ for checking “convergence” of $\bar{s}_N$ and of $\bar{s}_N$ individually, but also the joint (convergence tolerance), i.e., the relative tolerance $\varepsilon_2$ between $\bar{s}_N$ and $\bar{s}_N$; furthermore, termination is checked after each column rearrangement (which is also done by our implementation RA() but not the version given in Embrechts et al. [16]). As our simulation studies in Section 3.3 suggest, useful (conservative) defaults for $\mathbf{K}$ and the maximal number of column rearrangements are $\mathbf{K} = (8, 9, \ldots, 19)$ and $10d$, respectively. Given the ubiquitous model risk and the (often) rather large values of $\text{VaR}_\alpha(L^+)$ (especially in heavy-tailed of test cases), a conservative choice for the relative tolerance $\varepsilon$ may be $\varepsilon = (0, 0.01)$; obviously, all these values can be freely chosen in the actual implementation of ARA().

4.2 Empirical performance under various scenarios

In this section, we empirically investigate the performance of the ARA. To this end, we consider $d \in \{20, 100\}$ risk factors, paired with the Cases HH, LH, LL, LH as in Section 3.3. The considered relative joint tolerances are 0.5%, 1% and 2% and we investigate the results for the individual relative tolerances 0% and 1%. Therefore the performance of ARA is investigated in 48 different scenarios. As before, the results shown are based on $B = 100$ simulations and we investigate the $\text{VaR}_\alpha(\mathcal{L}^+)$ bounds $\bar{s}_N$ and $\bar{s}_N$, the $N$ used on the final column rearrangement of ARA (i.e., the $N = 2^k$, $k \in \mathbf{K}$ for which the algorithm terminates), the run time (in s), the number of column rearrangements (measured for the $N$ used on termination of the algorithm) and the number of oppositely ordered columns after termination of the algorithm.
Results

We first consider the results for the individual relative tolerance fixed at $\varepsilon_1 = 0$.

Our findings (see Figures 8–11 and computed numbers; the latter are omitted) indicate that:

- Although for both $d = 20$ and $d = 100$ the length of the confidence intervals for $\text{VaR}_{0.99}(L^+)$ can be checked to be increasing as the joint relative tolerance $\varepsilon_2$ gets larger, the mean and lower and upper confidence bounds remain fairly close to each other. More importantly for a fixed individual relative tolerance, as $\varepsilon_2$ increases, we do not observe a drastic shift in both lower and upper bounds for the mean across different examples.

- An important observation about the $N$ in the ARA is that in virtually all examples, the 95% confidence interval remains the same; this fact can be leveraged in practice for portfolios which exhibit the same marginal tail behavior to reduce the run time of the ARA.

- Across all of the 24 scenarios, doubling the joint relative tolerance reduces the run time (measured in s) by more than 50%.

- The number of column rearrangements for the $N$ used remains below 10$d$.

- Finally, as Figures 8–11 reveal, randomizing the input matrix $X$ has a minimal impact on various outputs of the ARA. However, this randomization has an interesting effect on the run time in that it seems to avoid the worst case in which sorting a lot of numbers is required when oppositely ordering the columns causing the algorithm to take quite a bit longer time to run. For this behavior (as well as dependence of run time on the order of the columns, convergence was faster with heavier-tailed columns being slower) was clearly visible during our testing phase and leads to the following open research question: How can the rows and columns of the initial matrices be sorted such that the run time of the ARA is minimal?

The effect of a different choice of $\varepsilon_1$ can be seen from Figures 13–16. Overall, they are very similar; note, however, Figure 18 (based on the 20 constituents of the SMI from 2011-09-12 to 2012-03-28) which involves a too large choice of $\varepsilon_1$. As we can see, both bounds can be fairly close (according to $\varepsilon_2$) but the individual “convergence” did not take place yet; hence our default $\varepsilon_1 = 0$ of RA(0) and ARA(0).

![Figure 8 Boxplots of the lower and upper $\text{VaR}_{0.99}(L^+)$ bounds $\underline{\pi}_N$ (left-hand side) and $\overline{\pi}_N$ (right-hand side) computed with the ARA for $\text{itol}=0$ based on $B = 100$ replications.](image)

4.3 A comparison with an asymptotic result for large $d$

Another interesting practical question not addressed so far is whether, given existence of the first moments of all marginal distributions, asymptotic sharp bounds by worst expected shortfall $\mathbb{E}\mathcal{S}_\alpha(L^+)$ can replace the need of a computational tool such as the (A)RA. Since expected shortfall
Figure 9 Boxplots of the actual $N = 2^k$ used for computing the lower and upper $\text{VaR}_{0.99}(L^+)$ bounds $\underline{\sigma}_N$ (left-hand side) and $\bar{\sigma}_N$ (right-hand side) with the ARA for $\text{itol}=0$ based on $B = 100$ replications.

Figure 10 Boxplots of the run time (in s) for computing the lower and upper $\text{VaR}_{0.99}(L^+)$ bounds $\underline{\sigma}_N$ (left-hand side) and $\bar{\sigma}_N$ (right-hand side) with the ARA for $\text{itol}=0$ based on $B = 100$ replications.
Improved Algorithms for Computing Worst VaR

Figure 11 Boxplots of the number of column rearrangements (measured for the $N$ used) for computing the lower and upper $\text{VaR}_{0.99}(L^+)$ bounds $\underline{s}_N$ (left-hand side) and $\overline{s}_N$ (right-hand side) with the ARA for $\text{itol}=0$ based on $B = 100$ replications.

Figure 12 Boxplots of the number of oppositely ordered columns for computing the lower and upper $\text{VaR}_{0.99}(L^+)$ bounds $\underline{s}_N$ (left-hand side) and $\overline{s}_N$ (right-hand side) with the ARA for $\text{itol}=0$ based on $B = 100$ replications.

Figure 13 Boxplots of the lower and upper $\text{VaR}_{0.99}(L^+)$ bounds $\underline{s}_N$ (left-hand side) and $\overline{s}_N$ (right-hand side) computed with the ARA for $\text{itol}=0.001$ based on $B = 100$ replications.
Figure 14 Boxplots of the actual $N = 2^k$ used for computing the lower and upper $\text{VaR}_{0.99}(L^+)$ bounds $\underline{s}_N$ (left-hand side) and $\overline{s}_N$ (right-hand side) with the ARA for $\text{itol}=0.001$ based on $B = 100$ replications.

Figure 15 Boxplots of the run time (in s) for computing the lower and upper $\text{VaR}_{0.99}(L^+)$ bounds $\underline{s}_N$ (left-hand side) and $\overline{s}_N$ (right-hand side) with the ARA for $\text{itol}=0.001$ based on $B = 100$ replications.
Figure 16 Boxplots of the number of column rearrangements (measured for the $N$ used) for computing the lower and upper $\text{VaR}_{0.99}(L^+)$ bounds $\underline{s}_N$ (left-hand side) and $\overline{s}_N$ (right-hand side) with the ARA for $\text{itol}=0.001$ based on $B=100$ replications.

Figure 17 Boxplots of the number of oppositely ordered columns for computing the lower and upper $\text{VaR}_{0.99}(L^+)$ bounds $\underline{s}_N$ (left-hand side) and $\overline{s}_N$ (right-hand side) with the ARA for $\text{itol}=0.001$ based on $B=100$ replications.
Figure 18 Computed $\text{VaR}_{0.99}(L^+)$ bounds $\xi_N$ and $\pi_N$ (with rearrange()) depending on the chosen relative tolerance $\varepsilon_1$ (the cross “$+$” indicating the values for $\varepsilon_1 = \text{NULL}$) for an application to SMI constituents data from 2011-09-12 to 2012-03-28; see the vignette VaR_bounds for more details.

Figure 19 Relative speed-up (in %) of the actual implementation of ARA() over a basic version as given in the vignette VaR_bounds.

is subadditive and comonotone additive, the worst expected shortfall $\text{ES}_\alpha(L^+)$ is attained for the sum of the marginal expected shortfalls. Under suitable conditions, it is known that

$$\lim_{d \to \infty} \frac{\text{ES}_\alpha(L^+)}{\text{VaR}_\alpha(L^+)} = 1;$$

see Bernard et al. [7] or McNeil et al. [23, Proposition 8.36].

Figure 20 shows $d \mapsto \frac{\text{ES}_\alpha(L^+)}{\text{VaR}_\alpha(L^+)}$ for different confidence levels $\alpha$ under Cases $\text{HH}^{\text{shifted}}$, $\text{LH}^{\text{shifted}}$, $\text{LL}^{\text{shifted}}$ and $\text{LH}_1^{\text{shifted}}$ which simply correspond to Cases HH, LH, LL and LH$_1$ where each marginal $\theta_j$ is increased by one; this is done to ensure that the first moment of all considered marginal distributions (and thus expected shortfall) exists. Worst Value-at-Risk $\text{VaR}_\alpha(L^+)$ is computed as the mean of the bounds $\xi_N$ and $\pi_N$ obtained from the ARA (with default values). Figure 20 also displays 2% bounds around 1.

As can be seen from Figure 20, the convergence rate of $\text{VaR}_\alpha(L^+)$ to $\text{ES}_\alpha(L^+)$ for large $d$ seems to largely depend on the chosen confidence level $\alpha$. Only for very large $\alpha \in (0, 1)$ (as required for credit risk and operational risk within the Basel II framework), the convergence rate seems to be acceptable.

5 Conclusion

This paper presents two major contributions to the computation of Value-at-Risk bounds for a sum of losses with given marginals in the context of Quantitative Risk Management.

First, we considered the homogeneous case (i.e., all margins being equal) and addressed the dual bound approach based on Embrechts et al. [16, Proposition 4] and Wang’s approach based on
Improved Algorithms for Computing Worst VaR

Figure 20 Worst expected shortfall $\mathbb{E}S_\alpha(L^+)$ over worst Value-at-Risk $\text{VaR}_\alpha(L^+)$ as functions of the dimension $d$ for different confidence levels $\alpha$. The considered cases $\text{HH}_\text{shifted}$, $\text{LH}_\text{shifted}$, $\text{LI}_\text{shifted}$ and $\text{LH}_\text{shifted}$ (from left to right) are as before, but each parameter is increased by 1 so that $\mathbb{E}S_\alpha(L^+)$ exists.

Embretschs et al. [15, Proposition 1] for computing Value-at-Risk bounds. Although both of these approaches are mathematically “explicit”, care has to be exercised when computing Value-at-Risk bounds with these algorithms in practice. We identified and overcame several numerical and computational hurdles in their implementation and addressed them using the R package qrmtools including the vignette VaR bounds. We covered several numerical aspects such as how to compute initial intervals for the root-finding procedure involved or that care has to be taken when choosing the tolerance of the root-finding procedure; a particular example which highlights the numerical challenges when computing Value-at-Risk bounds in general is the case of equal Pareto margins for which we also showed uniqueness of the root even in the infinite-mean case.

Overall, the reader should keep in mind that there is an important difference between implementing a specific model (say, with Par(2) margins) where initial intervals can be guessed or chosen “sufficiently large” and the proper implementation of a result such as Embretschs et al. [16, Proposition 4] in the form of a black-box algorithm; see the source code of qrmtools for more details and the work required to go in this direction.

Second, we considered the inhomogeneous case. We first investigated the Rearrangement Algorithm presented by Embretschs et al. [16]. Due to its simplicity, this algorithm by now has been widely adopted by the industry (see also https://sites.google.com/site/rearrangementalgorithm/). Nevertheless, the original algorithm leaves questions unanswered concerning the concrete choice of two of its tuning parameters. These parameters were shown to have a substantial impact on the algorithm’s performance and thus need to be chosen with care. We therefore presented an improved version of the Rearrangement Algorithm termed the Adaptive Rearrangement Algorithm. The latter improves the former in that it addresses the aforementioned two tuning parameters and improves on the underlying algorithmic design. The number of discretization points is chosen automatically in an adaptive way (hence the name of the algorithm). The absolute convergence tolerance is replaced by two relative convergence tolerances. Since they are relative tolerances, their choice is much more intuitive. Furthermore, the first relative tolerance is used to determine the individual convergence of each of the lower bound $\bar{S}_N$ and the upper bound $\bar{S}_N$ for worst (or best) Value-at-Risk and the second relative tolerance is used to control how far apart $\bar{S}_N$ and $\bar{S}_N$ are; the original version of the Rearrangement Algorithm does not allow for such a control. The Adaptive Rearrangement algorithm has been implemented in the R package qrmtools, together with conservative defaults. The implementation contains several other improvements as well (e.g., fast accessing of columns via lists; avoiding having to compute the row sums over all but the current column; an extended tracing feature). Both the algorithmic and the numerical improvements may be of interest in any context in which it is required to find columnwise permutations of a matrix such that the minimal (maximal) row sum is maximized (minimized).
There are still several interesting questions left to be investigated. First of all, as for the Rearrangement Algorithm, the theoretical convergence properties of the Adaptive Rearrangement Algorithm remain an open problem. Also, as mentioned above, it remains unclear how the rows and columns of the input matrices for the RA or ARA can be set up in an initialization such that the run time is minimal. Another interesting question is by how much we can reduce run time when using the rearranged matrix from the case where \( N = 2^{k-1} \) to construct the matrix for the case where \( N = 2^k \) (which is why we used powers of 2 here); it remains an open question whether the overhead of building that matrix outperforms (the additional) sorting.

Acknowledgments

We would like to thank Giovanni Puccetti (University of Milano), Ruodu Wang (University of Waterloo), Kurt Hornik (Vienna University of Economics and Business) and the reviewer for valuable comments and feedback on the paper. We would also like to thank Robert Stelzer (Editor-in-Chief) for handling our paper. All remaining errors and omissions are ours.

References

Improved Algorithms for Computing Worst VaR


A Appendix

Proof of Lemma 2.1. Consider the lower bound for VaR_\alpha(L^+). By De Morgan’s Law and Boole’s
inequality, the distribution function F_{L^+} of L^+ satisfies

\[ F_{L^+}(x) = \mathbb{P}\left( \sum_{j=1}^{d} L_j \leq x \right) \leq \mathbb{P}(\min L_j \leq x/d) = \mathbb{P}\left( \bigcup_{j=1}^{d} \{ L_j \leq x/d \} \right) \leq \sum_{j=1}^{d} \mathbb{P}(L_j \leq x/d) \]

\[ \leq d \max_{j} F_j(x/d). \]

Now \( d \max_{j} F_j(x/d) \leq \alpha \) if and only if \( x \leq d \min_{j} F_j^{-}(\alpha/d) \) and thus \( \text{VaR}_\alpha(L^+) = F_{L^+}^{-}(\alpha) \geq d \min_{j} F_j^{-}(\alpha/d) \).

Similarly, for the upper bound for \( \text{VaR}_\alpha(L^+) \), we have that

\[ F_{L^+}(x) = \mathbb{P}\left( \sum_{j=1}^{d} L_j \leq x \right) \geq \mathbb{P}(\max L_j \leq x/d) = \mathbb{P}(L_1 \leq x/d, \ldots, L_d \leq x/d) \]

\[ = 1 - \mathbb{P}\left( \bigcup_{j=1}^{d} \{ L_j > x/d \} \right) \geq \max\left\{ 1 - \sum_{j=1}^{d} \mathbb{P}(L_j > x/d), 0 \right\} \]

\[ = \max\left\{ \sum_{j=1}^{d} F_j(x/d) - d + 1, 0 \right\} \geq \max\{ d \min_{j} F_j(x/d) - d + 1, 0 \}. \]

Now \( d \min_{j} F_j(x/d) - d + 1 \geq \alpha \) if and only if \( x \geq d \max_{j} F_j^{-}((d-1+\alpha)/d) \) and thus \( \text{VaR}_\alpha(L^+) = F_{L^+}^{-}(\alpha) \leq d \max_{j} F_j^{-}((d-1+\alpha)/d) \).

\( \square \)

Proof of Proposition 2.3.

1) Let \( s \geq s' \) and \( t' \in [0, \frac{s'}{d}] \) such that \( D(s', t') = D(s') \). Define

\[ t = \frac{s - (s' - t'd)}{d} = \frac{s - s'}{d} + t' \]

so that \( 0 \leq t' \leq t \leq \frac{s}{d} \). Let \( \kappa = s' - t'd = s - td \). If \( \kappa > 0 \), noting that \( \bar{F} \) is decreasing and that \( t' \leq t \), we obtain

\[ D(s', t') - D(s, t) = \frac{d}{\kappa} \left( \int_{t'}^{t' + \kappa} \bar{F}(x)dx - \int_{t}^{t + \kappa} \bar{F}(x)dx \right) \geq 0, \]

so that \( D(s) \leq D(s, t) \leq D(s', t') = D(s') \). If \( \kappa = 0 \) then \( D(s') = D(s', \frac{s'}{d}) = d\bar{F}(\frac{s'}{d}) \geq d\bar{F}(\frac{s}{d}) \geq D(s) \).

2) Recall that \( D(s, t) = \frac{d}{s-td} \int_{t}^{t+(s-td)} \bar{F}(x)dx \). Using the transformation \( z = (x-t)/(s-td) \), we have

\[ D(s, t) = d \int_{0}^{1} \bar{F}(sz + t(1-zd))dz \]

Define \( C = \{ (s, t) | 0 \leq s < \infty, \ 0 \leq t \leq \frac{s}{d} \} \), and note that \( C \) is convex. Furthermore, if \( \bar{F} \) is
convex, then $D(s, t)$ is jointly convex in $s$ and $t$ on $C$ since for $\lambda \in (0, 1)$,

$$D(\lambda s_1 + (1 - \lambda)s_2, \lambda t_1 + (1 - \lambda)t_2)$$

$$= d \int_0^1 \tilde{F}((\lambda s_1 + (1 - \lambda)s_2)z + (\lambda t_1 + (1 - \lambda)t_2)(1 - zd)) dz$$

$$= d \int_0^1 \tilde{F}(\lambda(s_1z + t_1(1 - zd)) + (1 - \lambda)(s_2z + t_2(1 - zd))) dz$$

$$\leq \int_0^1 \lambda \tilde{F}(((s_1 + t_1(1 - zd))) + (1 - \lambda)\tilde{F}((s_2 + t_2(1 - zd)))) dz$$

$$= \lambda D(s_1, t_1) + (1 - \lambda)D(s_2, t_2) \quad \square$$

**Proof of Proposition 2.5.** First consider $\theta \neq 1$. Using (6), one can rewrite $h(c)$ as

$$h(c) = \frac{c^{-1/\theta+1} \theta (1 - (1 - \alpha/d - (d - 1))^{-1/\theta+1})}{1 - \alpha - dc} - \frac{(d - 1)(1 - \alpha/c - (d - 1)^{-1/\theta} + 1)}{c^{1/\theta}d}.$$  

Multiplying with $c^{1/\theta}d$ and rewriting the expression, one sees that $h(c) = 0$ is equivalent to $h_1(x_c) = 0$ where

$$x_c = \frac{1 - \alpha}{c} - (d - 1)$$  

(11)

(which is in $(1, \infty)$ for $c \in (0, (1 - \alpha)/d)$) and $h_1(x) = d^{-\theta} \frac{1 - x^{-1/\theta+1}}{x - 1} - (d - 1)x^{-1/\theta} + 1$. It is easy to see that $h_1(x) = 0$ if and only if $h_2(x) = 0$, where

$$h_2(x) = (d/(1 - \theta) - 1)x^{-1/\theta+1} - (d - 1)x^{-1/\theta} + x - (d\theta/(1 - \theta) + 1), \quad x \in (1, \infty).$$  

(12)

We are done for $\theta \neq 1$ if we show that $h_2$ has a unique root on $(1, \infty)$. To this end, note that $h_2(1) = 0$ and $\lim_{x \uparrow \infty} h_2(x) = \infty$. Furthermore,

$$h_2'(x) = (1 - 1/\theta)(d/(1 - \theta) - 1)x^{-1/\theta} + (d - 1)\theta x^{-1/\theta - 1} + 1,$$

$$h_2''(x) = (d + \theta - 1)/\theta^2 x^{-1/\theta - 1} - (1/\theta + 1)(d - 1)/\theta x^{-1/\theta - 2}.$$  

It is not difficult to check that $h_2''(x) = 0$ if and only if $x = (d - 1)/(d + \theta - 1)$ (which is greater than 1 for $d > 2$). Hence, $h_2$ can have at most one root. We are done if we find an $x_0 \in (1, \infty)$ such that $h_2(x_0) < 0$, but this is guaranteed by the fact that $\lim_{x \uparrow 1} h_2'(x) = 0$ and $\lim_{x \uparrow 1} h_2''(x) = -(d - 2)/\theta < 0$ for $d > 2$.

The proof for $\theta = 1$ works similarly; in this case, $h_2$ is given by

$$h_2(x) = x^2 + x(-d \log(x) + d - 2) - (d - 1), \quad x \in (1, \infty),$$  

(13)

and the unique point of inflection of $h_2$ is $x = d/2$. \quad \square

**Proof of Proposition 2.6.** First consider $c_1$ and $\theta \neq 1$. Instead of $h$, (11) and (12) allow us to study

$$h_2(x) = (d/(1 - \theta) - 1)x^{-1/\theta+1} - (d - 1)x^{-1/\theta} + x - (d\theta/(1 - \theta) + 1), \quad x \in [1, \infty).$$

Consider the two cases $\theta \in (0, 1)$ and $\theta \in (1, \infty)$ separately. If $\theta \in (0, 1)$, then $d/(1 - \theta) - 1 > d - 1 \geq 0$ and $x^{-1/\theta+1} \leq x^{-1/\theta}$ for all $x \geq 1$, so $h_2(x) \geq ((d/(1 - \theta) - 1) - (d - 1))x^{-1/\theta} + x - (d\theta/(1 - \theta) + 1) \geq x - (d\theta/(1 - \theta) + 1)$ which is 0 if and only if $x = d\theta/(1 - \theta) + 1$. Setting this equal to $x_c$ (defined in (11)) and solving for $c$ leads to the $c_1$ as provided. If $\theta \in (1, \infty)$, then using $x^{-1/\theta} \leq 1$ leads to $h_2(x) \geq (d/(1 - \theta) - 1)x^{-1/\theta+1} + x$ which is 0 for $x \geq 1$ if and only if $x = (d/(\theta - 1) + 1)^\theta$. Setting this equal to $x_c$ and solving for $c$ leads to the $c_1$ as required.
Now consider $\theta = 1$. As before, we can consider (11) and (13). By using the fact that $\log x \leq x^{1/e}$ and $x \geq -x^{1+1/e}$ for $x \in [1, \infty)$, we obtain $h_2(x) \geq x^2 + x(-dx^{1/e} + d - 2) - (d-1) \geq x^2 - (d+1)x^{1+1/e}$ which is 0 if and only if $x = (d+1)e/(e-1)$. Setting this equal to $x_c$ and solving for $c$ leads to the $c_l$ as required.

Lastly consider $c_u$. It is easy to see that the inflection point of $h_2$ provides a lower bound $x_c$ on the root of $h_2$. As derived in the proof of Proposition 2.5, the point of inflection is $x = x_c := (d-1)(1+\theta)/d-1$ for $\theta \neq 1$ and $x = d/2$ for $\theta = 1$. Solving $x_c = (1-\alpha)/c - (d-1)$ for $c$ then leads to $c_u$ as stated. □