An Algorithm for Sequential Tail Value at Risk for path-independent payoffs in a binomial tree

Berend Roorda*

April 28, 2009

Abstract

We present an algorithm that determines Sequential Tail Value at Risk (STVaR) for path-independent payoffs in a binomial tree. STVaR is a dynamic version of Tail-Value-at-Risk (TVaR) characterized by the property that risk levels at any moment must be in the range of risk levels later on. The algorithm consists of a finite sequence of backward recursions that is guaranteed to arrive at the solution of the corresponding dynamic optimization problem. The algorithm makes concrete how STVaR differs from TVaR over the remaining horizon, and from recursive TVaR, which amounts to Dynamic Programming. Time consistency and comonotonicity properties are illustrated by elementary examples, using the algorithm.

Keywords: Value at Risk, Tail Value at Risk, Dynamic Risk Measures, Time Consistency, Dynamic Programming

1 Introduction

A wide range of problems in applied science involve the optimization of a performance criterion under risk limits that guarantee a desired or required level of safety. In finance, the dominant approach to express risk limits is in terms of Value-at-Risk (VaR) (Morgan J. P. Inc 1996; Jorion 1997; Duffie and Pan 1997), being the maximum loss over a give time horizon at a certain confidence level. The key to its success is that it expresses risk as a monetary value with a transparent interpretation, which is very helpful in comparing and aggregating risks originating from different sources. The dominance of VaR in the financial industry is apparent from its central role in the world-wide regulation of banks for all risk

*B. Roorda, FELab and School of Management and Governance, University of Twente, P.O. Box 217, 7500 AE, Enschede, the Netherlands. Phone: +3153-4894383. E-mail: b.roorda@utwente.nl.

categories, including operational risk (BIS 2006). Although VaR inherently has a financial flavor, already in the name itself, it is applicable in a non-financial context as well, if all types of risk under consideration can be quantified in a common unit of value loss, see e.g. Tapiero (2003) for an application in inventory control.

A well-known shortcoming of using VaR as risk limit is that it stimulates concentration of risk, because it is insensitive for the actual level of the worst losses that can be ignored under a given confidence level. In the financial industry, with an abundance of opportunities to exploit any loophole at large scale, this aspect can be really harmful, and has led to considerable interest in TVaR as an alternative, also called Average VaR, Conditional VaR, or Expected Shortfall (Artner et al. 1999; Szegö 2002; Föllmer and Schied 2004; McNeil et al. 2005; Pflug 2007). TVaR measures the expected loss on the probability mass that is ignored in VaR, thus avoiding the anomalies in VaR as risk limit. We refer to Rockafellar and Uryasev (2002) for a fundamental result on optimizing performance under TVaR-constraints.

Anyhow, due to the intuitive appeal of VaR and VaR-like measures, for modelers as well as managers and regulators, it may be expected that they will remain the main standards in the financial industry for the coming years, if not decades.

The aim of this paper, however, is not primarily related to the controversy VaR vs. TVaR (we take our starting point in TVaR, and indicate how to derive a corresponding VaR-version), nor to optimizing performance under VaR-like restrictions (we only compute the outcome of these constraints for a given position).

Our primary concern is *the dynamics of risk measurement itself*, which brings us to the heart of the matter: our findings suggest that the evaluation of dynamic risk measures requires a new class of algorithms, more complex than Dynamic Programming, yet with sufficient structure to maintain some weaker, iterative form of backward recursive evaluation.

In fact, it is surprisingly difficult to extend a static notion of risk to a multiperiod setting, without violating certain compelling rules for the consistency of risk levels over time. The literature on dynamic risk measures and their time consistency properties is rapidly growing, but here we just briefly sketch the situation for VaR and TVaR. Straightforward extensions, such as TVaR over the remaining horizon, are severely time-inconsistent, in the sense that initial risk levels may decrease with probability one in the next period (Artzner et al. 2007). An obvious way to avoid time inconsistency is to adhere to a backward recursive definition, corresponding to so-called (strongly) time consistent risk measures, satisfying (8.1), but for TVaR this leads to accumulation of conservatism, as explained in Roorda and Schumacher (2007), henceforth RS07. In continuous time such strongly time consistent versions don't even exist, cf. Kupper and Schachermayer (2009).

Sequential Tail-Value-at-Risk (STVaR) has been introduced in RS07 as a weakly time consistent dynamic version of static TVaR. On the one hand, it avoids the type of time inconsistency as indicated above, by imposing so-called *sequential consistency*, which is the property that risk levels should never increase or decrease *for sure*, as expressed in (8.2). In fact, STVaR is the most conservative risk measure with this property that is dominated by TVaR over the entire (and remaining) horizon. On the other hand, accumulation of conservatism is avoided by deliberately giving up the backward recursive structure, corresponding to strong time consistency. We remark that STVaR does not involve any extra parameters, besides the confidence level, unlike the proposal for multiperiod TVaR in Pflug (2007).

We present an algorithm for computing STVaR in a binomial tree model, for a path-independent payoff. In RS07 it already has been shown that the optimization related to STVaR amounts to a Linear Programming problem. The algorithm presented here exploits more specific features of the problem, that allow for a solution by a finite sequence of backward recursions, despite the fact that it is, for reasons indicated above, *not* strongly time consistent and hence does not follow the standard backward recursive scheme of Dynamic Programming.

From this perspective, the algorithm not only is a computational tool for solving the STVaR optimization problem, in a perhaps overly simple setting, but also provides an illustration of how exactly the weakly time consistent dynamics of risk processes can deviate from the certainty equivalence principles for value processes. This may serve as a blueprint for computing weakly time consistent risk measures in more advanced settings, e.g. in continuous time.

The paper is organized as follows. In Section 2 we repeat the definition of STVaR, and reformulate it as an optimization problem over admissible weighting functions. The algorithm is described in Section 3 (outline) and 4 (implementation). The proof of correctness can be found in Section 6, after an explanation how the output of the algorithm should be interpreted in terms of weighting functions. In Section 7 the working of the algorithm is further explained by an example. Time consistency aspects are discussed in Section 8, and conclusions follow in Section 9.

1.1 Notation

For the notation, we use a simplified version of that in RS07. We consider a binary tree over T periods. N denotes the set of all nodes, N' the set of all nodes except the final ones, and N_t is the set of nodes at time t, so $N = N' \cup N_T$. The root of the tree is denoted as $\mathbf{0}$. N_τ consists of the nodes at stopping time τ, so that τ is the first time a path reaches N_τ. For a subset $S \subseteq N$ the stopping time of reaching S for the first time is denoted as $\tau(S)$. Notice that $N_{\tau(S)}$ is the minimal set of stopping nodes for $\tau(S)$, e.g. $N_{\tau(N)} = \mathbf{0}$. For each node ν in N' the tree has an up-branch and a down-branch, to nodes indicated by respectively νu,

νd. The subtree with root ν is indicated as $F(\nu)$.

The outcome space Ω is identified with full paths in the tree, so $\omega \in \Omega$ can be represented as $(\mathbf{0}, \nu_1, \ldots, \nu_T)$ with $\nu_t \in N_t$ and all nodes connected by branches; ω_t is the event corresponding to the path up to and including t. Ω_τ is the outcome space of all paths that are stopped in τ.

The probability for an up-branch is p. $E[\cdot]$ denotes expected value in the binomial tree; $E_t[\cdot]$ is the expected value conditioned at t. For a stopping time τ, E_τ denotes the corresponding conditional expectation. We also make use of the notation $E_\nu[\cdot]$, which is the expected value conditioned on node ν, seen as an operator on $F(\nu)$, and it will be independent of the path to ν unless explicitly mentioned.

For a function $h : \Omega \to \mathbb{R}$, $|h|_t$ denotes the maximum of h conditioned at $\mathcal{F}_t$; $|h|_\nu$ is the maximum of h on the subtree $F(\nu)$. Further, with a function $h : N \to \mathbb{R}$, we associate the function $h^\tau : \Omega \to \mathbb{R}$ defined by $h^\tau = \{\omega \mapsto h(\nu) \,|\, \nu \text{ the end point of } \omega_\tau\}$.

2 Sequentially consistent TVaR

Let be given a binomial tree model over T periods with probability p for an up-branch, and a path-independent payoff $X : N_T \to \mathbb{R}$. STVaR at level α is defined as (cf. RS07)

$$\mathrm{STVaR}_\alpha(X) = \inf_{Z \in \mathcal{Z}} E[ZX] \tag{2.1}$$

with

$$\mathcal{Z} = \{Z : \Omega \to \mathbb{R} \,|\, E[Z] = 1 \text{ and } 0 \le Z \le \alpha^{-1} Z_t \ \text{ for } t = 0, \ldots, T\} \tag{2.2}$$

writing Z_t for $E_t[Z]$. In particular, $0 \le Z \le \alpha^{-1}$ in $\mathcal{Z}$. It turns out to be convenient to rewrite this as

$$\mathrm{STVaR}_\alpha(X) = \inf_{W \in \mathcal{W}} E[WX]/E[W] \tag{2.3}$$

with $\mathcal{W}$ the *set of admissible weighting functions*, given by

$$\mathcal{W} = \{W : \Omega \to [0,1] \,|\, |W| = 1 \text{ and } E_t[W] \ge \alpha |W|_t \ \text{ for } t = 0, \ldots, T\}. \tag{2.4}$$

The equivalence of both formulations follows readily from taking $W = Z/|Z|$ for a given Z in $\mathcal{Z}$, or, conversely, $Z = W/E[W]$ for a given $W \in \mathcal{W}$.Proof: W and Z differ only in a scalar (that is always nonzero). After rewriting the second inequality in (2.2) as $|Z|_t \le \alpha^{-1} Z_t$, the result follows immediately.

Notice that Z and W only differ in scaling. Where Z represents a relative density, having unit expected value, W is a scaled version of Z so that its maximum is 1.Also notice that if $Z < \alpha^{-1}$, still $|W| = 1$, yet $E[W] > \alpha$ for $W = Z/|Z|$. Similarly, Z/Z_t is the conditional relative density on $F(\nu)$, while $W^t := W/|W|_t$ is the same object, down-scaled to maximum value 1. We let W^ν denote the weighting function on $F(\nu)$ with values $W/|W|_\nu$; note that this may depend on the path to ν. [1]

Intuitively, $W(\omega)$ can be seen as the survival probability of a path ω, in the sense that the contribution of a path ω to the expectation in the numerator of (2.3) equals $E[1_\omega X(\omega)]$ times the 'probability' $W(\omega)$ to 'survive' the selection of contributing paths; W^ν has a similar interpretation on the subtree $F(\nu)$. As we will see, worst case paths will always certainly contribute to the STVaR outcome, regardless the value of $\alpha > 0$, hence get weight 1 in a solution of (2.3).[2] Best-case paths get zero weight, unless α is too close to 1 to allow for this. The optimal weight for paths in between, however, turns out to be heavily path-dependent, and optimal weights for different paths to the same end node may actually vary from 0 to 1.

We will refer to the conditional expected survival probability $E_\nu[W^\nu]$ as the *probability mass* in ν (under W), and to $E_\nu[W^\nu X]/E_\nu[W^\nu]$ as the *level* of node ν (under W). Note that both quantities are functions of paths to ν, in principle. We will, however, manage to work only with weighting functions for which W^ν is (essentially) independent of the path to ν, so that the probability mass and level of a node ν are constants in resp. [0,1] and $\mathbb{R}$. For such weighting functions W we define the triple of functions on N:

$$\begin{aligned} y(\nu) &= E_\nu[W^\nu] && \text{'the probability mass in } \nu \text{ (under } W\text{)'} \\ g(\nu) &= E_\nu[W^\nu X] && \text{'the raw level in } \nu \text{ (under } W\text{)'} \\ f(\nu) &= g(\nu)/y(\nu) && \text{'the level in } \nu \text{ (under } W\text{)'} \end{aligned} \tag{2.5}$$

which play a basic role in the implementation of the algorithm. The STVaR condition requiring that probability mass along paths never falls below α,[3] can now be expressed as

$$y(\nu) \geq \alpha \text{ for all } \nu \in N. \tag{2.6}$$

For the intuition we remark that ordinary Tail Value at Risk, over [0,T] as a single period, would only impose this restriction for $\nu = \mathbf{0}$, which always allows for a path independent

[1] Some care has to be taken with regard to 'trivial' subtrees $F(\nu)$ on which W and Z vanish, to make W^ν (and $W^t, Z/Z_t$) well defined for any $\nu \in N'$. We could adopt the convention in Delbaen (2006) to set W^ν (and Z/Z_ν) equal to 1 on $F(\nu)$ in that case, as this will never cause a violation of the STVaR conditions, even for $\alpha = 1$. In principal this will do, but this may cause artificial path dependencies in weighting functions, in case $|W|_\nu$ is not zero for all paths to ν; alternatively, we then can also 'paste' such a non-trivial value of W^ν, as will be explained later on.

[2] To be precise, only in the trivial case with STVaR$_\alpha$ coinciding with the worst case outcome, having probability exceeding α, solutions exist with weight below 1 for (some) worst-case paths.

[3] This does not exclude that W can become zero on $F(\nu)$: even then the conditional version W^ν is always (artificially) defined in such a way that probability mass is not below α.

optimal weighting function. However, it lacks the property of sequential consistency, as is illustrated by Example 8.1.

3 Outline of the algorithm

The algorithm determines a finite sequence of decreasing admissible weighting functions $W_{(0)}, \dots W_{(K)} =: W^*$ so that W^* is a solution of (2.3). It starts with taking $W = W_{(0)} \equiv 1$, corresponding to initial probability mass one in all nodes, and computing $g(\mathbf{0}) = f(\mathbf{0}) = E[WX] = E[X]$. If $\alpha = 1$, the algorithm is already finished, so we assume that $\alpha < 1$.

The main idea behind the algorithm is simple: in each loop it maximally reduces weights of paths leading to nodes with maximum level, and it stops, roughly speaking, when the probability mass at the root has been decreased to α.

In the first step it maximally reduces the probability mass of nodes with maximum level $M := \max_{\nu \in N_T} X(\nu)$. Such nodes are called M-nodes. Notice that also nodes ν before T can be M-nodes (under $W \equiv 1$), namely if $X = M$ on the entire subtree $F(\nu)$, cf. (2.5).

If in every node $\nu \in N'$, $P(X = M|\nu) \leq 1 - \alpha$, we can simply annihilate all weights for paths to M-nodes, i.e., set $W = 1_{X<M}$.Note: The rule does not hold per node separately, i.e., it is not true that every node allows for full reduction of probability mass by $q(\nu)$, as soon as $q(\nu) = P(X = M \,|\, \nu) \leq 1 - \alpha$. Think of early nodes with $q(\nu)$ small, with a link to a node where (2.6) is binding. If not, we construct a weighting function $W \in \mathcal{W}$ that corresponds to maximal reduction at rate M in each node, respecting the STVaR condition (2.6). This typically involves weights between 0 and 1 for some paths to M-nodes, as is illustrated by the example below. Nodes with probability mass at minimum level α are called STVaR-nodes.

The algorithm can be stopped at this point if $y(\mathbf{0})(= E[W]) = \alpha$, i.e., if $\mathbf{0}$ itself has become an STVaR node. Then $f(\mathbf{0})(= E[WX]/E[X]) = \mathrm{STVaR}_\alpha(X)$. This also holds if the root itself has become an M-node, so if $f(\mathbf{0}) = M$, which can only happen in the first loop if X is the constant M.

For the next loop, all nodes with minimal probability mass α (if any) are collected in the set $\mathcal{S}$, and all M-nodes in Ex. These are considered as stopping nodes, and the corresponding stopping time τ replaces the role of T.Three alternatives for stopping time: (i) $\tau(\mathcal{S})$, so no stop in Ex: possible, but proofs get more complicated. (ii): $\tau(\mathrm{Ex} \cup \mathcal{S} \cup \mathcal{C})$ with $\mathcal{C}$ the set of nodes with an outgoing branch cut: this does not work, because in subtree of $\nu \in \mathcal{C}$ still reduction possible. (iii) $\tau(\mathrm{Ex})$?? Then we again perform maximal reduction, similarly as before, but now in Ω_τ, of probability mass at the maximum possible rate M, which is now decreased to $M = \max\{f(\nu) \,|\, \nu \in N_\tau \setminus \mathrm{Ex}\}$, because all nodes in Ex already have zero weight. New nodes with $f(\nu) = M$ and with $y(\nu) = \alpha$ are added to resp. Ex and $\mathcal{S}$.

This construction is repeated until $\tau = 0$. Then $\mathbf{0} \in \mathcal{S}$ and/or $\mathbf{0} \in \mathrm{Ex}$, and in both cases $f(\mathbf{0}) = \mathrm{STVaR}_\alpha(X)$. In case $\mathbf{0} \in \mathrm{Ex}$, this coincides with the worst case outcome of X.

4 The algorithm

The basic variables in the algorithm are the triple (y, g, f) as defined in (2.5). They are initialized according to the weighting function $W \equiv 1$:

- $y(\nu) := 1$ for all $\nu \in N$.
- $g(\nu) := X(\nu)$ for $\nu \in N_T$, and, backward recursively, $g(\nu) := pg(\nu u) + (1-p)g(\nu d)$.
- $f(\nu) := g(\nu)/y(\nu) = g(\nu)$

If $\alpha = 1$, $f(\mathbf{0}) = \mathrm{STVaR}_\alpha(X)$, and the algorithm stops. Assume from now on that $\alpha < 1$.

The other basic variables are $\mathcal{S} = \{\nu \in N' \,|\, y(\nu) = \alpha\}$, $\mathrm{Ex} = \{\nu \in N \,|\, \nu$ has been M-node$\}$, and $\tau = \tau(\mathrm{Ex} \cup \mathcal{S})$. Initially, $\mathcal{S} := \emptyset$, $\mathrm{Ex} := \emptyset$, $\tau := T$.

In addition to the notation N_τ, the set of end nodes in Ω_τ, we also need notation for the set of all nodes occurring in Ω_τ as non-final nodes, $N_{<\tau} := \{\nu \in N' \,|\, \omega_\tau$ crosses ν for some $\omega \in \Omega\}$. Further, $N_{\leq\tau} = N_\tau \cup N_{<\tau}$.

Now repeat the following loop as long as $\tau > 0$, or, equivalently, $y(\mathbf{0}) > \alpha$ and $\mathbf{0} \notin \mathrm{Ex}$. The previous version had as second condition $\mathrm{Ex} \neq N$, but this is not correct. Nodes can remain outside Ex if they become unreachable in Ω_τ. Some other changes: the definition of τ was $\tau(\mathcal{S} \cup N_T)$, now also stopping as soon as Ex is reached. I removed some ineffective assignments in nodes outside $N_{\leq\tau}$.

1. Define $M := \max\{f(\nu) | \nu \in N_\tau \setminus \mathrm{Ex}\}$, and $N^M := \{\nu \in N_{\leq\tau} \,|\, f(\nu) = M\}$, the set of (new) M-nodes in the loop.
2. For $t = T-1$ down to 0, for all nodes $\nu \in (N_t \cap N_{<\tau}) \setminus N^M$ (the non-final nodes in Ω_τ at t with $y > \alpha$ and $f < M$):
   - If $f(\nu u) = M > f(\nu d)$ (ν is a pre-M-node),

$$yred := (1-p)y(\nu d) \text{ (the probability mass after a cut)} \tag{4.1}$$

     If $yred < \alpha$, define $w := (\alpha - (1-p)y(\nu d))/(py(\nu u))$, and

$$y(\nu) := \alpha \tag{4.2}$$
$$g(\nu) := wpg(\nu u) + (1-p)g(\nu d) \tag{4.3}$$
$$f(\nu) := g(\nu)/\alpha, \tag{4.4}$$

else set

$$y(\nu) := yred \tag{4.5}$$
$$f(\nu) := f(\nu d) \tag{4.6}$$
$$g(\nu) := y(\nu)f(\nu) \tag{4.7}$$

- If $f(\nu d) = M > f(\nu u)$ (ν is a pre-M-node),

  $$yred := py(\nu u) \text{ (the probability mass after a cut)} \tag{4.8}$$

  If $yred < \alpha$, define $w := (\alpha - py(\nu u))/((1-p)y(\nu d))$ and set

  $$y(\nu) := \alpha \tag{4.9}$$
  $$g(\nu) := pg(\nu u) + w(1-p)g(\nu d) \tag{4.10}$$
  $$f(\nu) := g(\nu)/\alpha, \tag{4.11}$$

  else set

  $$y(\nu) := yred \tag{4.12}$$
  $$f(\nu) := f(\nu u) \tag{4.13}$$
  $$g(\nu) := y(\nu)f(\nu) \tag{4.14}$$

- otherwise (no branch to M-node):

  $$\begin{aligned} y(\nu) &:= py(\nu u) + (1-p)y(\nu d) \\ g(\nu) &:= pg(\nu u) + (1-p)g(\nu d) \\ f(\nu) &:= g(\nu)/y(\nu) \end{aligned} \tag{4.15}$$

3. Adjust bookkeeping variables

   - $\mathrm{Ex} := \mathrm{Ex} \cup N^M$
   - $\mathcal{S} := \mathcal{S} \cup \{\nu \in N_{<\tau} \,|\, y(\nu) = \alpha\}$
   - $\tau := \tau(\mathrm{Ex} \cup \mathcal{S} \cup N_T)$ (then $N_\tau \subseteq \mathcal{S}$ are the 'reachable' stopping nodes).

Here ends the loop.

After the last loop, $f(\mathbf{0}) = \mathrm{STVaR}_\alpha(X)$.

5 The weighting function determined by the algorithm

In this section we explain how to interpret the algorithm in terms of admissible weighting functions, and describe the structural properties that are preserved after each loop. Optimality properties are addressed in the next section.

Notation requires some extra attention in carefully discriminating between values of variable at the beginning and the end of a loop. To suppress indices, we simply write W for the weighting function corresponding to the end of the loop under consideration, and use M,$\mathcal{S}$,τ, etc. for the value of the other variables in that stage. By a subscript prev we indicate the values of variables as determined by the *previous* loop, so we write $W_{\text{prev}}, M_{\text{prev}}, \text{Ex}_{\text{prev}}, \mathcal{S}_{\text{prev}}$ etc. For convenience, we write $\overline{\tau}$ for τ_{prev}, and $\overline{y}, \overline{g}, \overline{f}$ for the value of the triple as determined by the previous loop. These are hence the initial values for the loop under consideration.

It is clear that after a loop the triple (y, g, f) consists of functions on N with $\alpha \leq y(\nu) \leq 1$, $g(\nu) = y(\nu)f(\nu)$, and for $\nu \in N_T$, $y(\nu) = 1$ and $f(\nu) = X(\nu)$.

The triple $(y(\nu), g(\nu), f(\nu))$ in any node $\nu \in N_{<\overline{\tau}}$ is redefined in the loop by an assignment of the form

$$\begin{aligned} y(\nu) &= w_u(\nu)py(\nu u) + w_d(\nu)(1-p)y(\nu u) \\ g(\nu) &= w_u(\nu)pg(\nu u) + w_d(\nu)(1-p)g(\nu u) \\ f(\nu) &= g(\nu)/y(\nu) \end{aligned} \tag{5.1}$$

with *branch weights* $w_u(\nu), w_d(\nu)$ in $[0, 1]$.

The weighting function W corresponding to the end state of a loop is partly determined by these branch weights, partly by pasting the previously determined weighting function on subtrees as soon as the path reaches, before T, an STVaR-node $\nu \in \mathcal{S}_{\text{prev}}$ or a former M-node in Ex_{prev}. This pasting reflects the fact that in all nodes outside $N_{<\overline{\tau}}$, in particular on $N_{\overline{\tau}}$, Ex_{prev}, as well as $\mathcal{S}_{\text{prev}}$, no changes are made anymore. In fact this also holds for new M-nodes, hence for Ex.

The loop only affects the weight for a branch from a non-M-node ν to an M-node ν' (such ν must have exactly one branch to an M-node, and is called a *pre-M-node*). Clearly every path in $\Omega_{\overline{\tau}}$ can have at most one such a transition, because either the M-node ν' itself is in $N_{\overline{\tau}}$, or the entire subtree $F(\nu')$ stopped at $\overline{\tau}$ must consist of M-nodes, and then the remainder of the path in $\Omega_{\overline{\tau}}$ does not contain other pre-M-nodes. Of course, full paths in Ω can contain several pre-M-nodes.

Let $\Lambda : \Omega \to [0, 1]$ be the function that assigns to all paths $\omega \in \Omega$ the branch weight from the first pre-M-node to M-node, as determined by the loop, or the value 1 if it does not contain any such transition. Then the weighting scheme W determined by the loop can be expressed recursively as

$$W = |W|_{\overline{\tau}} W^{\overline{\tau}} \text{ with } W^{\overline{\tau}} = W^{\overline{\tau}}_{\text{prev}} \text{ and } |W|_{\overline{\tau}} = \Lambda \tag{5.2}$$

The identity $W^{\overline{\tau}} = W^{\overline{\tau}}_{\text{prev}}$ reflects the pasting at $\overline{\tau}$.

It is important to notice that all branch weights in Ω_τ are crisp, i.e., either zero or one, which can be proved as follows. In the loop intermediate weights only are assigned to branches if the corresponding pre-M-node becomes a new stopping node in $\mathcal{S}$, so that such branch weights apply *after* τ (and before $\overline{\tau}$). Assuming, as induction hypothesis, that before the loop also all branch weights in $\Omega_{\overline{\tau}}$ were crisp, then this hence also holds after the loop on Ω_τ.

In a similar way, it is easily derived that all branch weights in Ω_τ are one, except for the last transition of paths in this space that end up in Ex, which must be a cut link with zero weight. Indeed, paths stopped at τ that end at τ in $\mathrm{Ex}_{\mathrm{prev}}$ or in an M-node have their final branch cut (in resp. a previous and the current loop), while other paths in Ω_τ keep unit branch weights.

Consequently, $\tau(\omega)$ is the first moment that a path $\omega \in \Omega$ reaches a node ν with (i) $y(\nu) = \alpha$ (iff $\nu \in \mathcal{S}$) and/or (ii) $f(\nu) \geq M$ (iff $\nu \in \mathrm{Ex}$) and/or (iii) $\nu \in N_T$.

Returning to the level of weighting functions, it is now clear that W^ν is path independent, if we restrict the attention to Ω_τ, and it is easily verified that W^ν is linked to the triple (y, g, f) by (2.5)

To see that this is indeed an admissible weighting scheme, i.e., that $W \in \mathcal{W}$, observe that by construction (2.6) holds. Moreover, $|W| = 1$ follows directly from the fact that in all nodes, at least one of the branch weights is 1. From an obvious inductive argument it then follows that W is admissible.

The structure sketched above can be translated to the following properties of W. Define

$$1_{\geq M} : \text{the indicator function of } \Omega_\tau^{\geq M} := \{\omega \in \Omega \,|\, \omega_\tau \text{ ends in Ex}\} \tag{5.3}$$

$$1_{<M} : \text{the indicator function of } \Omega_\tau^{<M} := \{\omega \in \Omega \,|\, \omega_\tau \text{ ends outside Ex}\} \tag{5.4}$$

This notation is motivated by the fact that paths end in Ex precisely when their end level is larger or equal to M, as explained above.

Lemma 5.1 *The weighting function W determined at the end of the loop has the following structure.*

1. *$W = 1_{<M} W^\tau$, or, equivalently, $|W|_\tau = 1_{<M}$*

2. *W^τ is path independent on Ω_τ, with $E_\tau[W^\tau] = y^\tau$ and $E_\tau[W^\tau X] = g^\tau$.*

3. *For a stopping time $\sigma \leq \tau$,*

$$E_\sigma[W] = y^\sigma = E_\sigma[1_{<M} y^\tau], \text{ in particular } y(\mathbf{0}) = E[W] = E[1_{<M} y^\tau] \tag{5.5}$$

$$E_\sigma[WX] = g^\sigma = E_\sigma[1_{<M} g^\tau], \text{ in particular } g(\mathbf{0}) = E[WX] = E[1_{<M} g^\tau] \tag{5.6}$$

6 Proof of correctness

We have to show that if the algorithm stops, $f(\mathbf{0}) = \mathrm{STVaR}_\alpha(X)$, and furthermore that the number of loops is finite. The claim on correctness will be derived mainly from two optimality properties that hold after each loop,

- *semi-optimality* in Ω_τ, i.e., in all nodes $\nu \in N_{\le\tau}$, for any triple $(\tilde{y}, \tilde{g}, \tilde{f})$ corresponding to some $V \in \mathcal{W}$ for some $\omega \in \Omega$ with ω_τ ending in ν, with $\tilde{y} = y(\nu)$, it holds that $\tilde{g} \ge g(\nu)$ and, equivalently, $\tilde{f} \ge f(\nu)$
- *(full) optimality* at τ, i.e., in the same notation, for all $\nu \in N_\tau$ and $V \in \mathcal{W}$, not necessarily with $\tilde{y} = y(\nu)$, $\tilde{f} \ge f(\nu)$.

Then it is obvious that the algorithm correctly stops when $\tau = 0$. For the proof we need a somewhat stronger formulation.

LEMMA 6.1 *At the end of each loop, semi-optimality in Ω_τ and optimality at τ holds. Furthermore, for any $V \in \mathcal{W}$, with $(\tilde{y}, \tilde{g}, \tilde{f})$ the corresponding triple in a node ν and a given path to ν in Ω_τ,*

$$\textit{for } \nu \in N_{<\tau} : \tilde{g} \ge g(\nu) + M[\tilde{y} - y(\nu)]^+ - M_{\mathrm{next}}[\tilde{y} - y(\nu)]^- \tag{6.1}$$

$$\textit{for } \nu \in \mathrm{Ex} : \tilde{g} \ge g(\nu) + K_\nu[\tilde{y} - y(\nu)]^+ - f(\nu)[\tilde{y} - y(\nu)]^- \tag{6.2}$$

$$\textit{for } \nu \in \mathcal{S} : \tilde{g} \ge g(\nu) + M_\nu(\tilde{y} - y(\nu)) \textit{ and } \tilde{y} \ge y(\nu) \tag{6.3}$$

with $M_\nu > f(\nu)$ the reduction rate in the loop that made ν belong to $\mathcal{S}$, and $K_\nu > f(\nu)$ the reduction rate in the loop before ν became an M-node.

PROOF It is straightforwardly verified that the three inequalities imply semi-optimality in $\nu \in N_\tau$, by substituting $\tilde{y} = y(\nu)$ in (6.1), as well as (full) optimality for $\nu \in N_\tau$, by using $K_\nu, M_\nu > f(\nu) = g(\nu)/y(\nu)$ in (6.2) and (6.3). So it suffices to prove the inequalities.

Consider $V \in \mathcal{W}$, and assume as induction hypothesis that the inequalities hold at the end of the previous loop, so on $\Omega_{\overline{\tau}}$ we have

$$\text{for } \nu \in N_{<\overline{\tau}} : \tilde{g} \ge \overline{g}(\nu) + M_{\mathrm{prev}}[\tilde{y} - \overline{y}(\nu)]^+ - M[\tilde{y} - \overline{y}(\nu)]^- \tag{6.4}$$

$$\text{for } \nu \in \mathrm{Ex}_{\mathrm{prev}} : \tilde{g} \ge \overline{g}(\nu) + K_\nu[\tilde{y} - \overline{y}(\nu)]^+ - f(\nu)[\tilde{y} - \overline{y}(\nu)]^- \tag{6.5}$$

$$\text{for } \nu \in \mathcal{S}_{\mathrm{prev}} : \tilde{g} \ge \overline{g}(\nu) + M_\nu(\tilde{y} - \overline{y}(\nu)]) \text{ and } \tilde{y} \ge \overline{y}(\nu) \tag{6.6}$$

We first concentrate on (6.2) and (6.3). The loop does not affect nodes in $\mathrm{Ex}_{\mathrm{prev}}$ and $\mathcal{S}_{\mathrm{prev}}$, so for $\nu \in (\mathrm{Ex} \cup \mathcal{S}) \cap (\mathrm{Ex}_{\mathrm{prev}} \cup \mathcal{S}_{\mathrm{prev}})$, $(y(\nu), g(\nu), f(\nu)) = (\overline{y}(\nu), \overline{g}(\nu), \overline{f}(\nu))$, and for these ν the lemma follows immediately. All other nodes in $\mathrm{Ex} \cup \mathcal{S}$ must satisfy (6.4). Now if such ν belongs to Ex, it must be an M-node, with $f(\nu) = M$, and (6.4) implies (6.2) directly. Otherwise, such ν is a new STVaR-node in $\mathcal{S}$. Because the reduction in the loop (if any) is at rate exactly M in $\Omega_{\overline{\tau}}$, it follows that

$$\overline{g}(\nu) - g(\nu) = M(\overline{y}(\nu) - y(\nu)) \tag{6.7}$$

and together with (6.4) this implies (6.3).

So we have proved (6.2) and (6.3), and we already showed that this implies that f^τ is minimal, i.e., in obvious yet incorrect notation, $\tilde{f}^\tau \geq f^\tau$. Another consequence is that

$$\tilde{g}^\tau - g^\tau \geq M(\tilde{y}^\tau - y^\tau). \tag{6.8}$$

We now prove (6.1) for $\nu = \mathbf{0}$, the proof for other nodes in $N_{<\tau}$ is entirely similar. Decompose $\tilde{y} - y(0) = E[V - W] = E[E_\tau[V - W]]$ into three terms $\delta_1 + \delta_2 - \delta_3$, given by

$$E[1_{\geq M}|V|_\tau \tilde{y}^\tau] + E[1_{<M}|V|_\tau(\tilde{y}^\tau - y^\tau)] - E[1_{<M}(1 - |V|_\tau)y^\tau].$$

Similarly, decompose $\tilde{g} - g(0)$ into three terms $g_1 + g_2 - g_3$, given by

$$E[1_{\geq M}|V|_\tau \tilde{g}^\tau] + E[1_{<M}|V|_\tau(\tilde{g}^\tau - g^\tau)] - E[1_{<M}(1 - |V|_\tau)g^\tau].$$

Now (6.1) directly follows from $g_1 \geq M\delta_1$, $g_2 \geq M\delta_2$, and $g_3 \leq M_{\text{next}}\delta_3$. Here the first two inequalities are obtained from (6.8), and the third one from the fact that on the support of $1_{<M}$, $f \leq M_{\text{next}}$ and hence $g \leq M_{\text{next}}y$. □

Lemma 6.2 *The algorithm stops within* $1/2(T + 1)(T + 2)$ *loops.*

Proof The algorithm starts with Ex $= \emptyset$, and in each loop this set is extended by at least one M-node. If the algorithm would not have terminated before the $1/2(T + 1)(T + 2)$-th loop, then after that loop Ex $= N$, hence $\mathbf{0} \in$ Ex, and the algorithm stops. In fact $\mathbf{0} \in$ Ex already one loop earlier, because it cannot be the case that the root is the only element outside Ex. □

So we have proved the following result.

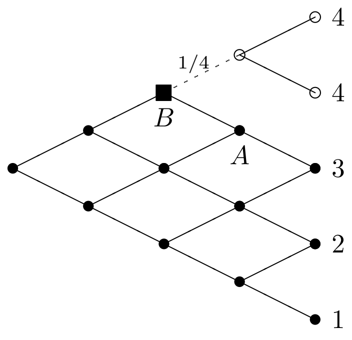
Theorem 6.3 *The algorithm terminates within* $1/2(T + 1)(T + 2)$ *loops (the number of nodes in the tree) and then* $f(\mathbf{0})$ *equals* $\mathrm{STVaR}_\alpha(X)$.

7 Example

We illustrate the working of the algorithm by an example. Meanwhile we discuss discuss some aspects of it that may be less obvious. The example is also used to describe the resulting optimal weighting scheme.

7.1 First step

We consider a binary tree with $p = 1/2$, $T = 4$, and payoff X as indicated in the picture below, with maximum value $M = 4$. $E[X] = 2\frac{15}{16}$. We apply the first loop of the algorithm for STVaR level $\alpha = 3/8$. The end result of the first step can be visualized as follows.



The M-nodes are indicated by open circles. The pre-M-nodes are nodes A and B, they have exactly one branch to an M-node. Starting in node A, the last one, we see that the branch to its M-node can be cut completely. Formally, we set $W^A(Au) = 0$, and keep $W^A(Ad) = 1$, with Au, Ad the nodes reaches from A after resp and up- or down-branch. This leaves node A with probability mass 1/2, which is not below α, as required. Obviously, the level in node A after this cut is given by $f(A) = 3$.

For node B the branch to the M-node cannot be cut completely, taking into account that node A has not full probability mass anymore: this would yield probability mass 1/4 in B, which is below α. Setting the transition weight equal to $w = 1/4$, as depicted above, the probability mass is reduced to α exactly. This brings the value $f(B)$ down to $wp4 + (1-p)^2 3 = 3\frac{1}{3}$. This is actually the STVaR_α value of X on $F(B)$, and therefore B is called an STVaR-node.

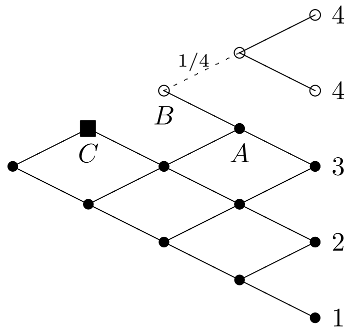
The probabilities and levels in the other nodes follow (4.15), corresponding to unit transition weights. This gives $y(\mathbf{0}) = \frac{23}{32}$, $f(\mathbf{0}) = 2\frac{12}{23}$.

At the end of the loop, the stopping time τ equals $\tau(B \cup N_T)$. Intuitively speaking, in the backward recursions of later loops, node B will pass its STVaR_α value to earlier nodes, regardless of any further reductions of nodes in $F(B)$. So, in forward perspective, for paths through B there is no reason to 'look behind' node B. Formally, the outcome space will now be restricted to Ω_τ. Notice that the path $duuu$ still belongs to Ω_τ. It is the only path that arrives at τ in Ex, hence the only one with zero weight in Ω_τ.

We remark that it is not crucial that the reduction in node A is passed through to node B. Alternatively, one could determine in node B, regardless of the reduction in node A, one weight $w' \in [0,1]$ for all paths to M-nodes through B, which would then be given by $w'P(X = M|B) + P(X < M|B) = w'\frac{3}{4} + \frac{1}{4} = \alpha$, so $w' = \frac{1}{6}$. This is conceptually straightforward, but there are several disadvantages. The reduction as computed in A only would apply to paths that do not first cross B, introducing an (unnecessary) path-dependency for W^A. Secondly, in node B all (multiperiod) paths to M-nodes have to be determined, while the original method can do with backward recursion over single steps. In addition, it is not clear that assigning the same weight to all paths to M-nodes is a valid construction in general.

7.2 Second step

Taking starting point in the end result of the first step, now the maximum reduction rate M has become $3\frac{1}{3}$, and B becomes the new M-node. Reduction now takes place in the only pre-M-node C, and it turns out that cutting its branch to B reduces its probability mass exactly to $y(C) = \alpha$. The corresponding level is $f(C) = 2\frac{2}{3}$, copied from the node below B. So the picture now becomes



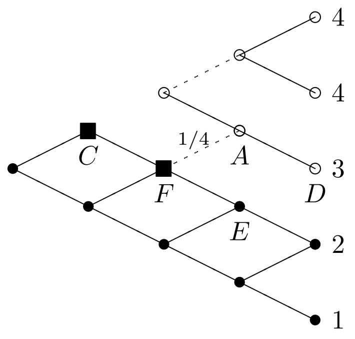
This updates the stopping time τ to $\tau(C \cup N_T)$. Notice that B, which was already a stopping node in $\mathcal{S}$, now also belongs to Ex. At the root now $y(\mathbf{0}) = \frac{5}{8}$ and $f(\mathbf{0}) = 2\frac{2}{5}$.

Notice that not all paths from C to $X < M$ have unit weight after reduction at rate M. For example, the path $\omega = uudd$ leads to $X(\omega) = 3$, yet $W(\omega) = 0$. By giving this path some extra weight in W^C the pair $y(C), g(C)$ would increase to $y(C) + \delta, g(C) + 3\delta$. This seems to contradict Lemma 6.1, which states that the increase rate in g is at least $M = 3\frac{1}{3}$, hence cannot be 3. However, this extra weight would make W^C (and hence W) an inadmissible weighting scheme, violating (2.6) in node B. Instead, the minimum rate of increase is exactly $M = 3\frac{1}{3}$, corresponding to giving back the cut branch to B some positive weight.

This is the crucial difference with considering TVaR over the remaining period, as discussed in Section 8. It also illustrates that $\mathcal{W}$ is *not* closed under increasing weighting schemes bounded by 0 and 1, if it is allowed to increase the support of $\mathcal{W}$.

7.3 Remaining steps

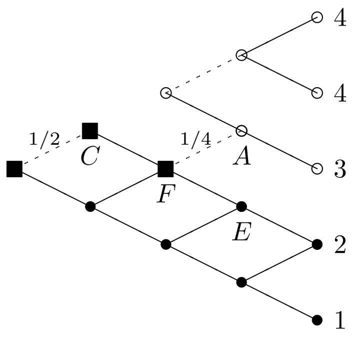
In the third loop, the reduction rate is $M = 3$, and there are two M-nodes, D and A. Reduction at this rate takes place in the pre-M-nodes E (first) and F. This leads to the following situation after the third loop.



The value in F is now $2\frac{1}{3}$. The stopping nodes in N_τ are now C and D. Notice that the reduction in F is not affecting the level of the STVaR-node C. A subtle point arises here: on the original outcome space Ω, W^F is path dependent, as its value really depends on whether the path addresses C or not. However, the path via C to F is excluded in Ω_τ, and on this space path independency of W^F holds, albeit in a trivial way in this example, leaving just one path to F.

At the root, $y(\mathbf{0}) = \frac{15}{32}$, still beyond α, and $f(\mathbf{0}) = 2\frac{1}{5}$.

Finally, node C becomes M-node with level $M = 2\frac{2}{3}$. Maximum reduction at this rate corresponds to weight $w = 1/2$ for the first up-branch, and level $2\frac{1}{12}$ for the root, which is the outcome of $\text{STVaR}_\alpha(X)$. The end situation is depicted below.



7.4 Final state

The final state in the example is typical: the root has one branch to a node in $\mathcal{S}$ (u in this case), with level equal to the last reduction rate, i.e., $f(u) = M$. The other node, d in this case, has level $f(d) < M$, and probability $y(d) > \alpha$. Optimality of the final weighting scheme is reflected by the fact that in node d, probability mass can only be increased at increase rate beyond M, and decreased at rate below M. Intuitively, instead of taking the STVaR value in d, the node is filled with extra probability mass until the increase rate begins to exceed the level $f(u)$.

It is clear that in the final situation, $\mathrm{Ex} = \{\nu \in N \mid \mathrm{STVaR}_\alpha(X|\nu) \geq M\}$.Nodes in Ex have been M-node in a loop, and then their level was minimal and equal to a reduction rate $> M$. Any node with minimal level $< M$ cannot have been an M-node. So $\tau(\mathrm{Ex} \cup N_T)$ is the stopping time of reaching STVaR_α level M for the first time. There are three types of paths in $\Omega_{\tau(\mathrm{Ex}\cup N_T)}$: those ending in T without reaching Ex, hence in $X < M$, having weight 1, those crossing $\mathcal{S}$ just before stopping in Ex, having level below M, and weight α, and those having in the last step a cut link to Ex, while the probability mass is still above α, having weight zero.

So the working of the algorithm can be summarized as follows. It determines the region Ex of nodes where the STVaR_α level exceeds a certain level M, and maximally reduces the last branch weight of paths arriving at Ex, giving priority to branches to Ex-nodes with higher level, while respecting (2.6). The level M (the final reduction rate) is determined as the highest level for which then the root gets probability mass α.[4]

8 Time consistency aspects

The *raison d'être* of STVaR is that it avoids time inconsistency problems in defining Value-at-Risk in a multi-period setting. We briefly illustrate this aspect in the context of the algorithm. First we will summarize some basic notions, see RS07 for a more extensive description. This involves the extension of a risk measure ϕ_0, such as STVaR_α, to a sequence of refined risk measures $\{\phi_t\}_{t=0,\ldots,T}$, also called updates of ϕ_0, that take t as initial time.

A sequence of updates $\{\phi_t\}_{t=0,\ldots,T}$ is called strongly time consistent if the 'risk-equivalence' principle

$$\phi_0(\phi_t(X)) = \phi_0(X) \tag{8.1}$$

holds, and sequentially consistent if

$$\phi_t(X) \geq 0 \Rightarrow \phi_u(X) \not< 0 \text{ and } \phi_t(X) \leq 0 \Rightarrow \phi_u(X) \not> 0 \text{ for } t < u. \tag{8.2}$$

Intuitively, this means that $\phi_t(X)$ is in the range of $\phi_u(X)$, or, for binomial trees, that $\phi_\nu(X) \in \{\lambda\phi_{\nu u} + (1-\lambda)\phi_{\nu d} \mid \lambda \in [0,1]\}$.

[4]In the trivial case where the algorithm stops because of $\mathbf{0} \in \mathrm{Ex}$, M is the worst-case outcome of X, and the algorithm possibly leaves the root with more than minimal probability mass, but even then it could be reduced to α, afterwards.

A fundamental observation is that initial risk measures ϕ_0 admit only one update at t that has a chance of being sequentially and/or strongly time consistent, so (weak) time consistency can also be seen as *a property of the initial risk measure itself.*

For a coherent risk measures ϕ_0, which is representable as the worst expected value operator over a set of probability measures, this update ϕ_t amounts to conditioning expected values on the information at t, cf. RS07. We refer to Roorda and Schumacher (2009) for a much more general result, for convex and even non-convex risk measures and an even weaker type of time consistency.

Applying this to $\phi_0 = \text{STVaR}_\alpha[\cdot]$, as defined in (2.3), we have

$$\phi_t(X) = \inf_{W \in \mathcal{W}} E_t[WX]/E_t[W] \tag{8.3}$$

which is nothing else than STVaR_α over subtrees with roots in N_t.

It may be illuminating to compare this to taking the initial measure equal to TVaR over the entire horizon, cf. the remark after (2.6). This is not sequentially consistent, as is shown by the following small example.[5] Other examples, involving a path dependent payoff, can be found in RS07 and in Artzner et al. (2007).

Example 8.1 Consider a binary tree with two steps, $\Omega = \{uu, ud, du, dd\}$, and $X(uu) = 0$, $X(ud) = X(du) = 1$, and $X(dd) = -1$. Assume probability 3/4 for up, 1/4 for down. Consider TVaR_α with $\alpha = 1/2$. Then in u and d, the outcome is 0, while for the entire period $[0, T]$ the algorithm yields outcome -1/8.

STVaR is in fact the most conservative risk measure dominated by TVaR that is sequentially consistent. So, combining the restriction that risk should not be seen as higher than TVaR_α over the entire horizon, with the natural requirement that risk levels should never increase or decrease for sure, automatically leads to the STVaR concept. This also holds for each subtree separately.

The first moment that the gap between STVaR and TVaR becomes visible in the original example is after the last step of the algorithm, in the root node **0**. The STVaR level in **0** has been determined at $2\frac{1}{12}$, while the corresponding TVaR level is 2. This is obtained by giving full weight to paths to end value 1 and 2, and the remaining probability mass of 1/16 to some path ending in value 3, say a path crossing C. The conflict with (2.6) now arises in node C, which has probability mass $5/16 < \alpha = 3/8$. This illustrates that TVaR allows a degree of conservatism in future states, that will be considered as too excessive when that state actually materializes. It is exactly this type of inconsistency that STVaR avoids. We remark that in other examples this difference can be much more pronounced.

[5] The example in Section 7 is not suitable for this, because it has a monotone payoff.

On the other hand, STVaR is *not* strongly time consistent, and we argued in RS07 why this can be a desirable property, in particular in the context of capital requirements (as opposed to *pricing* measures). Examples of strongly time consistent risk measures dominated by TVaR over the entire period correspond to taking TVaR_{α_t} over period $[t, t+1]$, e.g. $\alpha_t = \alpha^{1/T}$. These can be computed backward recursively according to the dynamic programming principles. This prescribes the level of TVaR in each period *a priori*, as if the time profile of most adverse events is time homogeneous or predetermined, regardless the position. Moreover, for large T there is in fact no reasonable approximation of TVaR that is strongly time consistent, and for continuous time the whole concept fails, as already indicated in the introduction. In contrast, STVaR lets the induced level of conservatism depend on the position X, without making any *ad hoc* choice on the timing of risk. It is the position X that determines whether nodes contribute at most conservative level (the ones in $\mathcal{S}$ and Ex), or less conservatively, just as it turns out to be most adverse.

This clearly indicates that STVaR is fundamentally different from strongly time consistent versions of TVaR per period, on the one hand, and TVaR over the remaining horizon, on the other. This discrepancy is further underlined by the fact that STVaR is not comonotonically additive, and hence can also not be represented as a mixture of TVaR with different confidence levels (Kusuoka 2001; Föllmer and Schied 2004, Thm 4.87). We conclude this section by a counter example for comonotonic additivity.

Example 8.2 In the same setting as the previous example, consider now the comonotone family of positions $\{X_\mu\}_{\mu\in(0,1)}$ with $X_\mu(uu) = 1$, $X_\mu(ud) = X_\mu(du) = 0$, and $X_\mu(dd) = \mu$. Take $\alpha = 3/4$ and $p = 1/2$. The STVaR algorithm starts with creating an STVaR node in the node u (in obvious notation) with corresponding level $f(u) = 1/3$. If $\mu < 1/3$, the algorithm proceeds with reducing the branch weight to u, and results in $\text{STVaR}(X_\mu) = \frac{1}{9} + \frac{1}{3}\mu$. If $\mu \geq 3$, the second loop reduces the branch weight to dd, which creates first an STVaR-node in d of level $\frac{1}{3}\mu$, and then, in the same loop, determines the STVaR value of X_μ in the root as $\frac{1}{6} + \frac{1}{6}\mu$. Now it is easily verified that for a pair of positions $X_\mu, X_{\mu'}$ with $0 < \mu < 1/3 < 1$, $\text{STVaR}(X_\mu) + \text{STVaR}(X_{\mu'}) < \text{STVaR}(X_\mu + X_{\mu'}) = 2\,\text{STVaR}(X_{(\mu+\mu')/2})$. For instance, for $\mu = 1/6$and $\mu' = 1/2$, the left-hand side equals $\frac{1}{6} + \frac{1}{4}$, while the right-hand side is $\frac{5}{12}$.

9 Conclusions

We showed how STVaR can be computed by a sequence of backward recursions, and used the algorithm to illustrate and motivate the difference with other versions of multiperiod TVaR.

This has been done in an extremely simple setting, for path independent positions on a binomial tree, so that attention can be focussed on conceptual aspects entirely. It is straightforward to generalize the algorithm to multinomial trees, and mild forms of path dependency. Time steps can be refined in order to approximate continuous time STVaR. The interpretation of τ as the first time that conditional STVaR hits a level M suggests a link with optimal stopping problems in the spirit of American option pricing. This could be further exploited in order to develop stochastic calculus for STVaR itself as a continuous time process.

Perhaps more crucial for the acceptance of a risk measure in the industry is an absolutely transparent interpretation. In this respect VaR over a single period still sets the standard, despite its shortcomings that have been widely addressed in the literature. Now VaR can be reconstructed from TVaR by the rule

$$\mathrm{VaR}_\alpha := \lim_{\delta \searrow 0} \frac{(\alpha+\delta)\,\mathrm{TVaR}_{\alpha+\delta}}{\alpha+\delta},$$

and, inspired by this rule, one could develop a 'sequential' version of Value-at-Risk at confidence level $1-\alpha$. Such a notion can be helpful in further developing sequentially consistent risk measures that can compete with VaR in terms of transparency.

References

Artzner, Ph., F. Delbaen, J.-M. Eber, and D. Heath, 1999. Coherent measures of risk. *Mathematical Finance*, **9**, 203–228.

Artzner, Ph., F. Delbaen, J.-M. Eber, D. Heath, and H. Ku, 2007. Coherent multiperiod risk adjusted values and Bellman's principle. *Annals of Operations Research*, **152-1**, pp. 5-22.

Bank of International Settlements, 2006. Basel II: International Convergence of Capital Measurement and Capital Standards: A Revised Framework", *www.bis.org.*

Delbaen, F., 2006. The structure of m-stable sets and in particular of the set of risk neutral measures, in Memoriam Paul-And Meyer, Lecture Notes in Mathematics 1874, 215-258.

Duffie, D. and Pan, J., 1997. An overview of Value at Risk. Journal of Derivatives 4, pp. 749.

Föllmer, H., and A. Schied, 2004. *Stochastic Finance. An introduction in discrete time* (2nd ed.). De Gruyter Studies in Mathematics, Vol. 27, de Gruyter, Berlin / New York.

Jorion, P., 1997. Value at risk: the new benchmark for controlling market risk. New York: McGraw-Hill.

Kupper, M. and W. Schachermayer, 2009. Representation results for law invariant time consistent functions, *Preprint.*

Kusuoka, S., 2001. On law invariant coherent risk measures. *Advances in Mathematical Economics*, **3**, 83–95.

McNeil, A.J., R. Frey and P. Embrechts, 2005. *Q*uantitative Risk Management: Concepts, Techniques, Tools. Princeton University Press.

Morgan J.P., Inc., 1996. RiskMetrics$^{\text{TM}}$: technical document (4th edn.). New York: Morgan.

Pflug, G., 2007. *Modeling, Measuring, and Managing Risk.* World Scientific.

Rockafellar, R. T. and S. Uryasev, 2002. Conditional value-at-risk for general loss distributions. *Journal of Banking and Finance*, **26**, 1443–1471.

Roorda, B., and J. M. Schumacher, 2007. Time consistency conditions for acceptability measures, with an application to tail value at risk, to appear in *Insurance: Mathematics and Economics.*

Roorda, B., and J. M. Schumacher, 2009. Time Consistency of Nonconvex Risk Measures, 2009. Netspar Discussion Paper DP01/ 2009 006. Submitted

Szegö, G.P., 2002. No more VaR (this is not a typo), Editorial in *Journal of Banking and Finance*, **26**, 1247–1251

Charles S. Tapiero, 2005. Value at risk and inventory control. *European Journal of Operational Research*, **163-3**, 769–775.