CIRRELT

Centre interuniversitaire de recherche
sur les réseaux d'entreprise, la logistique et le transport

Interuniversity Research Centre
on Enterprise Networks, Logistics and Transportation

Quality Evaluation of Scenario-Tree Generation Methods for Solving High-Dimensional Stochastic Programs

Julien Keutchayan
Michel Gendreau
Antoine Saucier

September 2016

CIRRELT-2016-46

Bureaux de Montréal :
Université de Montréal
Pavillon André-Aisenstadt
C.P. 6128, succursale Centre-ville
Montréal (Québec)
Canada H3C 3J7
Téléphone : 514 343-7575
Télécopie : 514 343-7121

Bureaux de Québec :
Université Laval
Pavillon Palasis-Prince
2325, de la Terrasse, bureau 2642
Québec (Québec)
Canada G1V 0A6
Téléphone : 418 656-2073
Télécopie : 418 656-2624

www.cirrelt.ca

Quality Evaluation of Scenario-Tree Generation Methods for Solving High-Dimensional Stochastic Programs

Julien Keutchayan[1,2,*], Michel Gendreau[1,2], Antoine Saucier[2]

[1] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT)

[2] Department of Mathematics and Industrial Engineering, Polytechnique Montréal, P.O. Box 6079, Station Centre-Ville, Montréal, Canada H3C 3A7

Abstract. This paper addresses the generation of scenario trees to solve two-stage or multistage stochastic programs that have a large number of possible outcomes for the random parameters (possibly infinitely many). For the sake of the computational efficiency, the scenario trees cannot include all outcomes and therefore the approximate solution provides decisions for some, but not all, random values. To overcome the resulting loss of information, we propose to introduce an *extension procedure*. This is a systematic approach to interpolate and extrapolate the optimal solutions of the scenario tree to provide the decision-maker with decisions for any value of the random parameters, at little computational cost. To assess the overall quality of the scenario-tree generation method and the extension procedure, we introduce three generic quality parameters that focus on the quality of the decisions. We use these quality parameters to develop a framework that will help the decision-maker to select the most suitable pair of scenario-tree generation method and extension procedure for a given stochastic programming problem. We perform numerical experiments on two case studies. We apply the quality parameters to compare three scenario-tree generation methods and three extension procedures. We show that it is possible to single out the best methods and that some extension procedures provide decisions of good quality at little computational cost. We also show that a method selection merely based on the scenario-tree optimal value is not reliable.

Keywords: Stochastic programming, scenario-tree generation method, decision policy extension, out-of-sample evaluation.

Acknowledgements. Financial support for this work was provided by the Natural Sciences and Engineering Research Council of Canada (NSERC) under grants RGPIN-2015-04696 and PIN-115965. This support is gratefully acknowledged.

---

* Corresponding author: Julien.Keutchayan@cirrelt.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2016

1 Introduction

Stochastic programming is a mathematical framework used to formulate and solve sequential decision-making problems under uncertainty. It relies on the assumption that the probability distribution of the random parameters is known, or at least can be estimated, and that the information regarding those parameters becomes available step by step. If the probability distribution is supported by a finite number of points, small enough for the computation to be tractable, then all the random outcomes can be represented in a so-called *scenario tree* and solving the stochastic program on the scenario tree provides the optimal decisions for every outcome. In this context, stochastic programming has proved to be a powerful framework to solve problems in energy, transportation, logistic and finance, etc.; see Wallace and Fleten (2003), Yu et al. (2003), Louveaux (1998), Powell and Topaloglu (2003) and the references therein. The situation becomes more complicated if the random parameters take a large number of values, since stochastic programs are now high-dimensional optimization problems (possibly infinite-dimensional) that are typically too complex to be solved analytically, or computationally in a reasonable time. Several authors have proposed to bypass this issue by building a scenario-tree with a selected subset of random values only. This is referred to as the discretization scheme of the stochastic process. Many discretization schemes have been developed in the literature; we will cite some important references in Section 1.2. The discretization scheme bypasses the intractability issue but introduces the question of the implementation of the tree optimal solutions, since they provide decisions only for some outcomes of the random parameters. Another important question for a decision-maker is how to tell which discretization scheme (or more generally which scenario-tree generation method) is more suitable for a given stochastic programming problem. A suitable method should produce a scenario tree with optimal solutions as close as possible from the optimal solutions of the original problem. In this paper, we propose a first mathematical framework that answers both questions.

The remainder of this paper is organized as follows: in Section 1.1 and 1.2 we introduce the notation to describe a stochastic programming problem and its approximation on a scenario tree; this notation is summarized in Appendix A. In Section 1.3 we describe with more details the motivation of our approach and we provide the general definition of an extension procedure. Section 2 introduces the quality parameters along with their statistical estimators. Section 3 provides a confidence interval for the quality parameters, along with an effective computational scheme for estimating them. Section 4 presents actual extension procedures, which are used to examine two case studies in Section 5. Section 6 concludes the paper and proposes future works.

1.1 Stochastic programming problem formulation

We consider a stochastic programming problem with an integer time horizon T and integer time stages t ranging from 0 to T. At each time period $(t-1, t)$ some random parameters are revealed. All the random information is contained in a stochastic process $\boldsymbol{\xi} = (\boldsymbol{\xi}_1, ..., \boldsymbol{\xi}_T)$, where $\boldsymbol{\xi}_t$ is a random vector with support $\Xi_t \subseteq \mathbb{R}^{d_t}$. Thus, $\boldsymbol{\xi}_t$ has d_t components, each of which represents a random parameter revealed at period $(t-1, t)$. We denote by $\boldsymbol{\xi}_{..t}$ the partial stochastic process $(\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_t)$ up to stage t. The supports of $\boldsymbol{\xi}$ and $\boldsymbol{\xi}_{..t}$ are denoted by Ξ and $\Xi_{..t}$, respectively, and the realizations of $\boldsymbol{\xi}$, $\boldsymbol{\xi}_t$ and $\boldsymbol{\xi}_{..t}$ are denoted by ξ, ξ_t and $\xi_{..t}$, respectively.

At each stage $t \in \{1, \ldots, T\}$, the decisions must be based only on the information available at this stage in order to be *non-anticipative*, which means that the stage-t decision function, denoted

by x_t, is defined as

$$\begin{aligned} x_t : \Xi_{..t} &\to \mathbb{R}^{s_t} \\ \xi_{..t} &\mapsto x_t(\xi_{..t}), \end{aligned} \tag{1}$$

where s_t is the number of decisions to be made at stage t; for the sake of clarity we consider that $s_t = s$ and $d_t = d$ for every t. The decision at stage 0 is a vector $x_0 \in \mathbb{R}^s$. We assume no particular property for the decision function (measurability is always implicity assumed throughout this paper). The *decision policy* (or simply *policy*) is denoted by x and is the collection of all the decision vector/functions from stage 0 to stage T, i.e., $x := (x_0, x_1, \ldots, x_T)$ or equivalently

$$x(\boldsymbol{\xi}) := (x_0, x_1(\boldsymbol{\xi}_1), \ldots, x_t(\boldsymbol{\xi}_{..t}), \ldots, x_T(\boldsymbol{\xi})). \tag{2}$$

The set of feasible decision vectors at stage 0 is X_0. At stage $t \in \{1, \ldots, T\}$, this set is denoted by $X_t(x_{..t-1}(\xi_{..t-1}); \xi_{..t})$ to emphasize that it may depend on the realization $\xi_{..t} \in \Xi_{..t}$ and on the previous decisions $x_{..t-1}(\xi_{..t-1}) := (x_0, ..., x_{t-1}(\xi_{..t-1}))$. A decision policy is *feasible* if it yields a feasible decision vector at every stage and for every realization of the stochastic process. We assume that the set of feasible policies is nonempty and that the decisions made at each stage do not alter the probability distribution of the stochastic process.

We introduce a revenue function q that takes a realization ξ and a complete sequence of decisions $x(\xi)$ and yields the *total* revenues $q(x(\xi); \xi)$ obtained over all the stages. We are interested in the dependence of the first moment of $q(x(\boldsymbol{\xi}); \boldsymbol{\xi})$ with respect to the decision policy, i.e., in the functional $Q(x) := \mathbb{E}[q(x(\boldsymbol{\xi}); \boldsymbol{\xi})]$, which we suppose is well-defined for any feasible policy x. The stochastic program that we consider consists in finding a feasible and non-anticipative policy that maximizes $Q(\cdot)$, which means finding x^* of the form (2) satisfying

$$Q(x^*) = \max_{x=(x_0,\ldots,x_T)} \mathbb{E}[q(x(\boldsymbol{\xi}); \boldsymbol{\xi})] \tag{3}$$

$$s.t. \quad x_0 \in X_0; \tag{4}$$

$$x_1(\boldsymbol{\xi}_1) \in X_1\big(x_0; \boldsymbol{\xi}_1\big), \ w.p.1; \tag{5}$$

$$x_t(\boldsymbol{\xi}_{..t}) \in X_t\big(x_{..t-1}(\boldsymbol{\xi}_{..t-1}); \boldsymbol{\xi}_{..t}\big), \ w.p.1, \ t = 2, \ldots, T. \tag{6}$$

Constraints (5) and (6) are asked to hold with probability one (w.p.1). The policy x^* is called an *optimal decision policy* and $Q(x^*)$ is the *optimal value* of the stochastic program. Some additional conditions should be added to ensure that there exists at least one optimal decision policy for the stochastic program (3)-(6); see, e.g., Rockafellar and Wets (1974).

1.2 Scenario tree and deterministic program formulation

The stochastic program (3)-(6) can be solved exactly or approximately by representing the stochastic process $\boldsymbol{\xi}$ with a scenario tree. We refer the reader to Birge and Louveaux (1997), Ruszczyński and Shapiro (2003) and Defourny et al. (2011) for a general presentation on the scenario-tree approach. This solution approach yields exact x^* and $Q(x^*)$ provided the stochastic process has a finite number of possible realizations and if all of them are included in the scenario tree. However, if $\boldsymbol{\xi}$ has a large number of realizations, then including all of them would lead to an intractable problem and therefore the scenario tree must be built via a discretization of $\boldsymbol{\xi}$ with a smaller number of scenarios. The deterministic program resulting from this discretization is no longer equivalent to the stochastic program (3)-(6) and the optimal decisions and the optimal value of the deterministic program are approximations of their original counterparts; see, e.g., Pennanen (2005) and references therein for convergence results.

Several methods exist to discretize the stochastic process. Among the most popular ones are: Monte Carlo sampling (Shapiro, 2003, 2006), integration quadrature and quasi-Monte Carlo (Pennanen and Koivu, 2002), moment-matching (Høyland and Wallace, 2001; Høyland et al., 2003), optimal quantization (Pflug, 2001; Pflug and Pichler, 2015), scenario reduction (Dupačová et al., 2003; Heitsch and Römisch, 2009), bound-based approximations (Frauendorfer, 1996; Edirisinghe, 1999), sparse grid quadrature rules (Chen and Mehrotra, 2008; Chen et al., 2015). All of them yield a procedure to generate scenarios from $\boldsymbol{\xi}$, but only a few also provide a systematic approach to generate a *tree structure* (defined below). Throughout this article, we use the expression *scenario-tree generation method* to name either a method that generates the scenarios along with a tree structure or a method that generates only scenarios and for which the tree structure has to be chosen beforehand by the decision-maker. Either way, the scenario-tree generation methods must be subject to a thorough evaluation in order to enable the decision-maker to make informed choices. The remainder of this section introduces the notation for the scenario tree and the deterministic program.

A scenario tree is composed of a rooted tree structure with node set $\mathcal{N}$, edge set $\mathcal{E}$, and root node n_0 (the node set minus the root is $\mathcal{N}^*$). The structure is such that T edges separate the root from any of the leaves. We denote by $\mathcal{N}_t$, for every $t \in \{1, \ldots, T\}$, the set of nodes corresponding to stage t, i.e., which are separated from the root by t edges. The children nodes $C(n)$ of n are the nodes linked to n at the next stage. The ancestor node $a(n)$ of n is the node linked to n at the previous stage. In addition to the tree structure, each node $n \in \mathcal{N}_t$ carries a discretization point ζ^n of $\boldsymbol{\xi}_t$. For $t = 1$, ζ^n is a discretization point of $\boldsymbol{\xi}_1$; for $t \in \{2, \ldots, T\}$, ζ^n is a discretization point of $\boldsymbol{\xi}_t$ conditional to $\boldsymbol{\xi}_{..t-1} = \zeta^{..a(n)}$, where $\zeta^{..a(n)}$ is the discretization sequence on the path from n_0 to $a(n)$. Each node $n \in \mathcal{N}^*$ also carries a positive value w^n that represents its weight with respect to its sibling nodes.

The usage of scenario-tree is twofold. On the one hand, if the scenario tree is the exact representation of the stochastic process, then the weight w^n is the conditional probability given by

$$w^n = \begin{cases} \mathbb{P}(\boldsymbol{\xi}_1 = \zeta^n) & \text{if } n \in \mathcal{N}_1; \\ \mathbb{P}(\boldsymbol{\xi}_t = \zeta^n \mid \boldsymbol{\xi}_{..t-1} = \zeta^{..a(n)}), & \text{if } n \in \mathcal{N}_t,\ t \in \{2, \ldots, T\}. \end{cases} \tag{7}$$

It follows that the weights satisfy the condition

$$\sum_{m \in C(n)} w^m = 1,\ \forall n \in \mathcal{N} \setminus \mathcal{N}_T. \tag{8}$$

On the other hand, if the scenario tree aims at being an approximation of $\boldsymbol{\xi}$, then the tree structure $(\mathcal{N}, \mathcal{E})$ and each couple (ζ^n, w^n) are given by the scenario-tree generation method, in most cases with the goal to approximate in some sense the stochastic process; see Pflug and Pichler (2011) for a presentation on probability distribution distances. In that case, (7)-(8) no longer have to hold.

No matter which usage is considered, the scenario-tree solution method proceeds as follows: we associate to each node $n \in \mathcal{N}$ a decision vector $\widehat{x}^n \in \mathbb{R}^s$. For $n \in \mathcal{N}^*$, we also denote $\widehat{x}^n$ by $\widehat{x}(\zeta^{..n})$ to emphasize that it is associated with the sequence $\zeta^{..n}$ that leads to n. From the standpoint of the first scenario-tree usage described above, we express the deterministic program as

$$\widehat{Q}^* := \max_{\{\widehat{x}^n\,:\,n \in \mathcal{N}\}} \sum_{l \in \mathcal{N}_T} W^l\, q(\widehat{x}^{..l}; \zeta^{..l}) \tag{9}$$

$$s.t. \quad \widehat{x}^{n_0} \in X_0; \tag{10}$$

$$\widehat{x}^n \in X_1\big(\widehat{x}^{n_0}; \zeta^n\big),\ \forall n \in \mathcal{N}_1; \tag{11}$$

$$\widehat{x}^n \in X_t\big(\widehat{x}^{..a(n)}; \zeta^{..n}\big),\ \forall n \in \mathcal{N}_t,\ \forall t \in \{2, \ldots, T\}, \tag{12}$$

where W^l is the product of the weights w^n on the scenario leading to the leaf l and $\widehat{x}^{\cdot\cdot l}$ is the collection of all the $\widehat{x}^n$ on this scenario. In the first scenario-tree interpretation, W^l is the probability of the scenario leading to l, i.e., $W^l = \mathbb{P}(\boldsymbol{\xi} = \zeta^{\cdot\cdot l})$. The optimal value of the deterministic program is denoted by $\widehat{Q}^*$ and is referred to as the *scenario-tree optimal value*. The optimal decision policy for the deterministic program is

$$\widehat{x}^* := \{\widehat{x}^{n_0 *}\} \cup \{\widehat{x}^*(\zeta^{\cdot\cdot n}) : n \in \mathcal{N}^*\}, \tag{13}$$

and is referred to as the *scenario-tree optimal policy*. In the second scenario-tree usage, which is the one that we consider in this paper, the outputs $\widehat{x}^*$ and $\widehat{Q}^*$ of the scenario-tree solution method are approximations of x^* and $Q(x^*)$, respectively.

1.3 The problem of evaluating scenario-tree generation methods

Scenario-tree solution methods have proved to be a useful solution method for a wide class of optimization problems (see references in Introduction). However, as pointed out in Ben-Tal et al. (2009), this approach fails to provide decisions for any values of the random parameters. Using the notation we introduced, this means that the decisions at stage t are available on the set $\{\zeta^{\cdot\cdot n} : n \in \mathcal{N}_t\}$ that is a proper subset of $\Xi_{\cdot\cdot t}$. This limits the usage of scenario-tree methods in real-word applications since the former set has probability zero if $\boldsymbol{\xi}$ has a continuous distribution. A classical attempt to bypass this issue, that was proposed for instance in Kouwenberg (2001), Chiralaksanakul and Morton (2004) and Hilli and Pennanen (2008), consists in solving the deterministic program dynamically on a shrinking horizon and to implement the stage-0 decision at every stage. This procedure may be computationally costly since it requires as many resolutions as the total number of stages and the procedure must be carried out all over again for each new realization ξ. In that framework, it can be computationally costly to perform an out-of-sample testing, which is an evaluation of the tree decisions on a set of scenarios directly sampled from $\boldsymbol{\xi}$, since it is typically required to test the decisions on thousands of realizations for a reliable accuracy. In addition, as noted in Ben-Tal et al. (2009), if the stochastic programming problem does not have relatively complete recourse, then the implementation of the stage-0 decision does not guarantee the existence of a feasible decision for the next stages. The other reason that motivates this paper is the fact that the stochastic programming community has developed many methods to generate scenario trees, but as far as we know, little attention has been paid to develop relevant ways of comparing methods, especially for a comparison based on the quality of the decisions. Indeed, all too often the comparison relies solely on the scenario-tree optimal value $\widehat{Q}^*$. However, the fact that $\widehat{Q}^*$ is large does not necessarily imply good quality decisions for the original stochastic program. The reason for this is that the value of $\widehat{Q}^*$ is an indicator of the quality of the decision within the framework of the deterministic program only. Less obviously, a value of $\widehat{Q}^*$ close to $Q(x^*)$ may also not be an indicator of good decisions quality, as we will show in the case studies.

In this paper, we completely depart from the view of the references above. From the tree optimal policy (13), we intend to recover a decision policy of the form (2) in order to treat the decisions of the scenario tree as a candidate solution of the original stochastic program. We do so as follows: after solving the approximate problem (9)-(12) on the scenario tree, we extrapolate and interpolate the tree optimal decisions outside the set of the scenario tree realizations. We refer to this technique as an *extension procedure* and to the resulting policy as an *extended tree policy*. An extended tree policy (formalized in Definition 1.1) is defined over all the possible realizations of the stochastic process and it coincides with the tree optimal policy on the discretization points of the scenario tree. Definition 1.1 is purposely generic and does not provide an actual extension procedure, but in Section 4 we propose practical ways to do so.

Definition 1.1. *Let* $\widehat{x}^* = \{\widehat{x}^{n_0*}\} \cup \{\widehat{x}^*(\zeta^{\cdot\cdot n}) : n \in \mathcal{N}^*\}$ *be a tree optimal policy. An* extended tree policy *for* $\widehat{x}^*$ *is a decision policy* $\widetilde{x} = (\widetilde{x}_0, \ldots, \widetilde{x}_T)$*, with* $\widetilde{x}_0 = \widehat{x}^{n_0*}$ *and for every* $t \in \{1, \ldots, T\}$*,* $\widetilde{x}_t$ *is defined as in (1) and satisfies*

$$\widetilde{x}_t(\zeta^{\cdot\cdot n}) = \widehat{x}^*(\zeta^{\cdot\cdot n}), \quad \forall n \in \mathcal{N}_t. \tag{14}$$

The extended tree policy allows for an accurate out-of-sample testing since the decisions are available simply through the evaluation of a function at a particular point, which can be carried out many times at little cost. In addition, this feature turns out to be essential for problems for which the time between the decision at two consecutive stages is smaller than the time required to reoptimize in the dynamical shrinking horizon approach. If $\boldsymbol{\xi}$ has a continuous distribution, then the extension procedure alone specifies the values of the policy on a set of probability one. Thus, different extension procedures can lead to very different policies and we expect the approximation quality to be as dependent on the extension procedures as it is on the scenario-tree generation methods. This leads to define several quality parameters that will enable the decision-maker to select the best combination of methods. We refer the reader to Mak et al. (1999) and Kaut and Wallace (2003) for some works on comparison techniques for two-stage programs. Perhaps the closest work to our approach is done by Defourny et al. (2013), where the authors extend the tree policy using regression techniques from machine learning. Our approach and theirs differ since they intend to select a good decision policy, while we want to select a good scenario-tree generation method and extension procedure. More details on this difference will be provided in Section 2.4.

This paper provides a mathematical framework to compare pairs of scenario-tree generation method and extension procedure. An important point is that the decision-makers do not need to know, nor to estimate, the optimal value of the stochastic program (3)-(6) in order to conclude about which methods are the best for a particular problem. The quality parameters introduced in this paper provide the criterion needed to reach this conclusion. The quality parameters of Section 2.1 and 2.2 are developed without any assumption apart from the existence of an optimal solution to the deterministic program (9)-(12). As for the quality parameter in Section 2.4, we assume that the stochastic program has relatively complete recourse.

Remark 1.1. The scenario-tree generation methods that perform the discretization procedure via a random sampling of the stochastic process produce different tree optimal policies every time that they are carried out. These methods are said to be *stochastic* in contrast to the other ones that are *deterministic*. Throughout this paper, it is important to keep in mind that a stochastic method yields a *random* extended policy $\widetilde{\boldsymbol{x}}$, that is denoted in bold font. Every time the decision-makers use a stochastic method, they obtain a realization $\widetilde{x}$ of $\widetilde{\boldsymbol{x}}$. In this paper, we develop a unified framework for both stochastic and deterministic methods. However, for the sake of conciseness, all the mathematical developments are done as if the policy was random. The equivalent results for deterministic methods are easy to deduce.

2 Quality parameters for scenario-tree generation methods and extension procedures

2.1 Probability of the policy feasibility

It follows from Definition 1.1 that the extended tree policy yields feasible decisions at the root node and for any realization $\xi_{\cdot\cdot t}$ that coincides with a discretization sequence $\zeta^{\cdot\cdot n}$ in the scenario tree. For any other realization, the feasibility depends on the considered extension procedure. The first feature that we want to assess is the probability of feasibility of the extended policy. In particular,

we want to know how this probability evolves across the stages. To this end, consider a scenario-tree generation method and an extension procedure that together yield a random extended policy $\widetilde{\boldsymbol{x}}$, and let $\widetilde{x}$ be a realization of $\widetilde{\boldsymbol{x}}$. We define the subset $\widetilde{\Xi}_{..t}(\widetilde{x})$ of $\Xi_{..t}$ on which $\widetilde{x}$ yields feasible decisions up to stage t, i.e.,

$$\widetilde{\Xi}_1(\widetilde{x}) = \big\{\xi_1 \in \Xi_1 \,|\, \widetilde{x}_1(\xi_1) \in X_1(\widetilde{x}_0;\xi_1)\big\}, \tag{15}$$

and for each $t \in \{2,\dots,T\}$,

$$\widetilde{\Xi}_{..t}(\widetilde{x}) = \big\{\xi_{..t} \in \Xi_{..t} \,\big|\, \xi_{..t-1} \in \widetilde{\Xi}_{..t-1}(\widetilde{x}),\ \widetilde{x}_t(\xi_{..t}) \in X_t\big(\widetilde{x}_{..t-1}(\xi_{..t-1});\xi_{..t}\big)\big\}. \tag{16}$$

The probability that $\widetilde{x}$ yields feasible decisions up to stage t is $\mathbb{P}(\boldsymbol{\xi}_{..t} \in \widetilde{\Xi}_{..t}(\widetilde{x}))$. This leads to the following definition working for both stochastic and deterministic methods.

Definition 2.1. *The values $p(t)$, for $t \in \{0,\dots,T\}$, are the probabilities that a random extended tree policy $\widetilde{\boldsymbol{x}}$ yields feasible decisions up to stage t. They are given by $p(0) = 1$ and*

$$p(t) = \mathbb{P}_{(\boldsymbol{\xi},\widetilde{\boldsymbol{x}})}(\boldsymbol{\xi}_{..t} \in \widetilde{\Xi}_{..t}(\widetilde{\boldsymbol{x}})), \tag{17}$$

for $t \in \{1,\dots,T\}$.

The sequence $(p(t))_{t\in\{0,\dots,T\}}$ is non-increasing by definition of $\widetilde{\Xi}_{..t}$. The notation $\mathbb{P}_{(\boldsymbol{\xi},\widetilde{\boldsymbol{x}})}$ in (17) emphasizes that the probability is taken with respect to the probability distribution of $\boldsymbol{\xi}$ and $\widetilde{\boldsymbol{x}}$. The probability distribution of $\widetilde{\boldsymbol{x}}$ is not known, but it can be sampled by generating several scenario trees and by solving each corresponding deterministic program. The probability distribution of $\boldsymbol{\xi}$ is known but typically too complex to enable an analytic derivation. Thus, to estimate $p(t)$ we propose the following unbiased and consistent estimator (these properties are proved in Section 3.1):

$$\widehat{p}_{K,M}(t) = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{M}\sum_{m=1}^{M}\mathbf{1}_{\xi_{..t}^{k,m} \in \widetilde{\Xi}_{..t}(\widetilde{x}^k)}, \tag{18}$$

where $\widetilde{x}^k$ is the extended policy obtained with the k-th scenario tree and the set $\{\xi^{k,m} : m = 1,\dots,M; k = 1,\dots,K\}$ is a collection of $K \times M$ independent sample points of $\boldsymbol{\xi}$ and $\xi_{..t}^{k,m}$ denotes the first-t components of $\xi^{k,m}$. The notation $\mathbf{1}_{\xi_{..t}^{k,m} \in \widetilde{\Xi}_{..t}(\widetilde{x}^k)}$ is the indicator function that yields 1 if $\xi_{..t}^{k,m} \in \widetilde{\Xi}_{..t}(\widetilde{x}^k)$ and 0 otherwise. Plotting the sequence $(p(t))_{t\in\{0,\dots,T\}}$, or in practice its estimator $(\widehat{p}_{K,M}(t))_{t\in\{0,\dots,T\}}$, provides information about the evolution of the size of the stage-t feasible region $\widetilde{\Xi}_{..t}(\widetilde{\boldsymbol{x}})$ (as a number ranging from 0 to 1) embedded in the support $\Xi_{..t}$ of $\boldsymbol{\xi}_{..t}$.

2.2 Expected revenues conditional to the policy feasibility

Another important information about the extended tree policy is the average revenues that it yields when it is feasible.

Definition 2.2. *The conditional revenues* CR *obtained with a random extended tree policy $\widetilde{\boldsymbol{x}}$ when it yields feasible decisions up to the end of the optimization horizon is given by*

$$\mathrm{CR} = \mathbb{E}_{(\boldsymbol{\xi},\widetilde{\boldsymbol{x}})}\big[q(\widetilde{\boldsymbol{x}}(\boldsymbol{\xi});\boldsymbol{\xi}) \,\big|\, \boldsymbol{\xi} \in \widetilde{\Xi}_{..T}(\widetilde{\boldsymbol{x}})\big]. \tag{19}$$

The conditional revenues CR is a conditional expectation that is well-defined provided that $p(T) > 0$. The value CR is a reliable quality parameter provided $p(T)$ is close to one. Indeed, if $p(T)$ is much smaller than one, then CR provides the conditional revenues on a small subset of realizations and its value can be even much greater than $Q(x^*)$.

It follows from the definition of the conditional expectation with respect to a probability event that a consistent estimator $\widehat{\mathrm{CR}}_{K,M}$ of CR is given by

$$\widehat{\mathrm{CR}}_{K,M} = \left(\frac{1}{K} \sum_{k=1}^{K} \frac{1}{M} \sum_{m=1}^{M} \mathbf{1}_{\xi^{k,m} \in \widetilde{\Xi}_{..T}(\widetilde{x}^k)} \right)^{-1} \frac{1}{K} \sum_{k=1}^{K} \frac{1}{M} \sum_{m=1}^{M} q(\widetilde{x}^k(\xi^{k,m}); \xi^{k,m}) \mathbf{1}_{\xi^{k,m} \in \widetilde{\Xi}_{..T}(\widetilde{x}^k)}. \quad (20)$$

The estimator $\widehat{\mathrm{CR}}_{K,M}$ is well-defined provided $\widehat{p}_{K,M}(T) > 0$, which is ensured if $p(T) > 0$ and K is large enough as a consequence of the law of large numbers.

The values $(p(t))_{t \in \{0,\dots,T\}}$ and CR provide complementary information about the quality of the extended tree policy. They have to be used together to avoid making erroneous conclusions when facing the situation that $p(T)$ is close to one but CR is much smaller than $Q(x^*)$, or conversely, when CR is close to $Q(x^*)$ but $p(T)$ is close to zero. In both cases, the value $p(T) \times \mathrm{CR}$ is much smaller than $Q(x^*)$, which means that the scenario-tree generation method and/or the extension procedure is of poor quality. In applications $Q(x^*)$ is typically not known, therefore the above analysis can be done with $\widehat{Q}^*$ or an upper bound on $Q(x^*)$.

When considering two methods (a) and (b), a decision-maker will prefer method (a) if the respective quality parameters satisfy: $p_a(T) > p_b(T)$ and $\mathrm{CR}_a > \mathrm{CR}_b$, with $p_a(T) \geq \alpha$, where α is a probability threshold that the decision-maker considers as satisfactory (typically $\alpha = 0.95$). As we will see in the numerical experiments, this selection criterion allows to put aside methods of poor quality. However, it usually does not allow to single out the best method out of a group of good methods. The reason for this is that smaller values of $p(T)$ allows for larger values of CR, which prevents the above selection criterion from being always conclusive. In Section 2.4, we introduce a third quality parameter that leads to a more robust selection criterion. To this end, we first address the problem of the infeasibility of the extended policy.

2.3 Feasibility restoration

In a real-world application, an extended tree policy $\widetilde{x}$ can be used all the way to the end of the optimization horizon provided the real-word realization ξ satisfies $\xi \in \widetilde{\Xi}_{..T}(\widetilde{x})$. If this condition does not hold, then there exists a stage t^* such that $\xi_{..t^*} \in \widetilde{\Xi}_{..t^*}(\widetilde{x})$ but $\xi_{..t} \notin \widetilde{\Xi}_{..t}(\widetilde{x})$ for every $t > t^*$. The decision-maker has to find alternative feasible decisions from stage t^*+1 to stage T. This is known in the literature as the *feasibility restoration problem*. The common approach to restore the feasibility is to consider the projection of the infeasible decision on the set of feasible decision vectors; see, e.g., Küchler and Vigerske (2010) and Defourny et al. (2013). This approach has the advantage of being systematic, however, it requires the resolution of a non-linear optimization problem that may be computationally costly. In this paper, in order to proceed with the idea that the decisions must be available at little cost, we investigate the possibility that the decision-makers have the ability to fix the infeasibility by their own empirical knowledge on the optimization problem. In other words, we assume that the decision-maker possesses a *recourse policy*, obtained empirically, that always provides feasible decisions. To formalize this approach, we assume that the stochastic programming problem has relatively complete recourse, so that any feasible decisions made in the past lead to possible feasible decisions in the future.

We model the recourse policy as a sequence $r = (r_1, \dots, r_T)$ of T functions, where r_t takes a realization $\xi_{..t}$ and a sequence of decisions $x_{..t-1}$ and yields a feasible decision vector, i.e., $r_t(x_{..t-1}; \xi_{..t}) \in X_t(x_{..t-1}; \xi_{..t})$. We emphasize that the definition of the recourse functions differs from the definition of the decision functions in (1), since the former may depend on the previous decisions. The implementation of $\widetilde{x}$, whenever it is feasible, and r yields a new decision policy, called the *fully-feasible extended tree policy*, that provides feasible decisions at every stage and for every realization of the stochastic process. Definition 2.3 below provides its explicit construction.

Definition 2.3. *The* fully-feasible extended tree policy *resulting from* $\widetilde{x}$ *and* r*, denoted by* $\overline{x}$*, is defined as in (1) and is given by* $\overline{x}_0 = \widetilde{x}_0$ *and recursively from* $t = 1$ *to* $t = T$*,*

$$\overline{x}_t(\xi_{..t}) = \begin{cases} \widetilde{x}_t(\xi_{..t}) & \text{if } \xi_{..t} \in \widetilde{\Xi}_{..t}(\widetilde{x}), \\ r_t(\overline{x}_{..t-1}(\xi_{..t-1}); \xi_{..t}) & \text{otherwise,} \end{cases} \tag{21}$$

where for $t = 1$ *the term* $\overline{x}_{..0}(\xi_{..0})$ *corresponds to* $\overline{x}_0$*.*

2.4 Selection criterion

A stochastic scenario-tree generation method yields a fully-feasible extended tree policy that is random and is denoted by $\overline{\boldsymbol{x}}$ (cf. Remark 1.1). The expected revenue earned when implementing $\overline{\boldsymbol{x}}$ is given by the random variable $Q(\overline{\boldsymbol{x}}) = \mathbb{E}_{\boldsymbol{\xi}}[q(\overline{\boldsymbol{x}}(\boldsymbol{\xi}); \boldsymbol{\xi})]$, where the notation $\mathbb{E}_{\boldsymbol{\xi}}$ emphasizes that the expectation is taken with respect to the probability distribution of $\boldsymbol{\xi}$ only. The random policy $\overline{\boldsymbol{x}}$ is feasible and non-anticipative, therefore the following relation holds:

$$Q(\overline{\boldsymbol{x}}) \leq Q(x^*), \quad w.p.1. \tag{22}$$

This inequality is useful for practical and theoretical reasons:

- Suppose that we do not know the optimal value $Q(x^*)$. In this case, each realization $\overline{x}$ of $\overline{\boldsymbol{x}}$ provides a lower bound $Q(\overline{x})$ of $Q(x^*)$. This lower bound is obtained by solving the deterministic program and estimating numerically the expected revenue $Q(\overline{x})$.
- Suppose that we know $Q(x^*)$ exactly or approximately. Then it is possible to estimate the proximity between $\overline{\boldsymbol{x}}$ and x^* by estimating the numerical difference between $Q(\overline{\boldsymbol{x}})$ and $Q(x^*)$. This difference, which is random, represents the missing revenues resulting from the implementation of the suboptimal random policy $\overline{\boldsymbol{x}}$.

Relation (22) holds if we take the expectation $\mathbb{E}_{\overline{\boldsymbol{x}}}[\cdot]$ on each side of the inequality (this expectation is with respect to the probability distribution of $\overline{\boldsymbol{x}}$ only). This leads to the definition of a distance between the fully-feasible extended policy $\overline{\boldsymbol{x}}$ and the optimal policy x^*.

Definition 2.4. *The distance* $d(\overline{\boldsymbol{x}}, x^*)$ *between the fully-feasible extended tree policy* $\overline{\boldsymbol{x}}$ *and the optimal policy* x^* *of the stochastic program is given by*

$$d(\overline{\boldsymbol{x}}, x^*) = Q(x^*) - \mathbb{E}_{\overline{\boldsymbol{x}}}\big[Q(\overline{\boldsymbol{x}})\big] \geq 0. \tag{23}$$

This distance characterizes an optimality for the choice of the scenario-tree generation method and the extension procedure, in the sense of the following proposition.

Proposition 2.5. *A scenario-tree generation method together with an extension procedure satisfying Definitions 1.1 and 2.3 yield a random policy* $\overline{\boldsymbol{x}}$ *that is optimal with probability one for the stochastic program if and only if* $d(\overline{\boldsymbol{x}}, x^*) = 0$*.*

Thus, the quantity $d(\overline{\boldsymbol{x}}, x^*)$ can be viewed as a distance between the considered methods and the ideal method that always provides the optimal solution. In applications, a value $d(\overline{\boldsymbol{x}}, x^*) > 0$ represents the expected missing revenue resulting from the repeated implementation of $\overline{\boldsymbol{x}}$.

Proof. The proof is straightforward: we have that $d(\overline{\boldsymbol{x}}, x^*) = 0$ if and only if $\mathbb{E}_{\overline{\boldsymbol{x}}}[Q(x^*) - Q(\overline{\boldsymbol{x}})] = 0$, and by the inequality (22), this is equivalent to $Q(x^*) = Q(\overline{\boldsymbol{x}})$, with probability one. By construction, the random policy $\overline{\boldsymbol{x}}$ is feasible and non-anticipative with probability one, which completes the proof. □

On the same model as the estimators (18) and (20), we propose the following unbiased and consistent estimator for $\mathbb{E}_{\overline{\boldsymbol{x}}}[Q(\overline{\boldsymbol{x}})]$:

$$\widehat{\mathbb{E}_{\overline{\boldsymbol{x}}}[Q(\overline{\boldsymbol{x}})]}_{K,M} = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{M}\sum_{m=1}^{M} q(\overline{x}^k(\xi^{k,m});\xi^{k,m}). \tag{24}$$

It follows from Proposition 2.5 that a decision-maker will select a scenario-tree generation method and the extension procedure that provide the smallest value of $d(\overline{\boldsymbol{x}}, x^*)$. This selection criterion assesses the absolute quality of a method, i.e., in comparison to the ideal method. In applications, $Q(x^*)$ is typically not known, hence the decision-maker will rather compare the relative efficiency of two methods, as shown in Definition 2.6.

Definition 2.6 (Selection criterion)**.** *Let (a) and (b) be two pairs of scenario-tree generation method and extension procedure that yield random policies $\overline{\boldsymbol{x}}_a$ and $\overline{\boldsymbol{x}}_b$, respectively. We say that (a) is better than (b) for the considered stochastic programming problem if*

$$\mathbb{E}_{\overline{\boldsymbol{x}}_a}[Q(\overline{\boldsymbol{x}}_a)] > \mathbb{E}_{\overline{\boldsymbol{x}}_b}[Q(\overline{\boldsymbol{x}}_b)]. \tag{25}$$

Criterion (25) is one relevant way, among others, to define and select the best method. An alternative definition could be to select the method on the basis of the *best* policy $\overline{x}$ that it yields in the long run, i.e., to decide that (a) is better than (b) if $\sup Q(\overline{\boldsymbol{x}}_a) > \sup Q(\overline{\boldsymbol{x}}_b)$. This criterion ensures that, when carrying out many times (a) and (b), (a) will eventually yield a policy with higher revenues than (b). An estimator for $\sup Q(\overline{\boldsymbol{x}})$ is $\max_{k=1,\dots,K} Q(\overline{x}^k)$ and this links this criterion with the policy selection technique developed by Defourny et al. (2013).

The selection criterion (25) can be slightly modified to suit a risk-averse decision-maker; see Shapiro et al. (2014) for a presentation on risk-averse optimization. For instance, the expectation $\mathbb{E}_{\overline{\boldsymbol{x}}}[Q(\overline{\boldsymbol{x}})]$ can be substituted with the quantity $\alpha\,\mathbb{E}_{\overline{\boldsymbol{x}}}[Q(\overline{\boldsymbol{x}})] - (1-\alpha)\mathrm{Var}_{\overline{\boldsymbol{x}}}[Q(\overline{\boldsymbol{x}})]$, where $\alpha \in (0,1)$ is the risk-averse coefficient. The term $\mathrm{Var}_{\overline{\boldsymbol{x}}}[Q(\overline{\boldsymbol{x}})]$ measures the robustness of the scenario-tree generation method and the extension procedure with regard to their repeated use. Its estimator is given by

$$\frac{1}{K}\sum_{k=1}^{K}\left(\frac{1}{M}\sum_{m=1}^{M} q(\overline{x}^k(\xi^{k,m});\xi^{k,m})\right)^2 - \left(\frac{1}{K}\sum_{k=1}^{K}\frac{1}{M}\sum_{m=1}^{M} q(\overline{x}^k(\xi^{k,m});\xi^{k,m})\right)^2. \tag{26}$$

Note that deterministic scenario-tree generation methods satisfy $\mathrm{Var}_{\overline{\boldsymbol{x}}}[Q(\overline{\boldsymbol{x}})] = 0$.

In the next section, we show how to compute efficiently estimators of the form (18) and (24).

3 Statistical study of the quality parameters

Quality parameters $p(t)$ and $\mathbb{E}_{\overline{\boldsymbol{x}}}[Q(\overline{\boldsymbol{x}})]$ can be written in a general form $\mathbb{E}[\boldsymbol{F}(\boldsymbol{\xi})]$, where $\boldsymbol{\xi}$ is the stochastic process introduced in Section 1.1 and $\boldsymbol{F}$ denotes a random *function*, independent of $\boldsymbol{\xi}$, that takes values in a function space. Realizations of $\boldsymbol{F}$ are denoted by F and are functions from $\mathbb{R}^{d\times T}$ to $\mathbb{R}$ (recall that $\xi \in \Xi \subseteq \mathbb{R}^{d\times T}$). The expectation is taken with respect to the probability distributions of $\boldsymbol{\xi}$ and $\boldsymbol{F}$. For $p(t)$ we have

$$\boldsymbol{F}(\boldsymbol{\xi}) = \mathbf{1}_{\boldsymbol{\xi}_{..t}\in\widetilde{\Xi}_{..t}(\widetilde{\boldsymbol{x}})}, \tag{27}$$

and for $\mathbb{E}_{\overline{\boldsymbol{x}}}[Q(\overline{\boldsymbol{x}})]$ we have

$$\boldsymbol{F}(\boldsymbol{\xi}) = q(\overline{\boldsymbol{x}}(\boldsymbol{\xi});\boldsymbol{\xi}). \tag{28}$$

As for CR, it is the ratio of two expectations of the form $\mathbb{E}[\boldsymbol{F}(\boldsymbol{\xi})]$.

In this section, we show how to estimate an expectation of the form $\mathbb{E}[\boldsymbol{F}(\boldsymbol{\xi})]$ with a good accuracy and at minimum computational cost by choosing the sample sizes of $\boldsymbol{\xi}$ and $\boldsymbol{F}$ in an optimal way. To this end, we assume that the second moment of $\boldsymbol{F}(\boldsymbol{\xi})$ is finite,

$$\mathbb{E}[|\boldsymbol{F}(\boldsymbol{\xi})|^2] < +\infty. \tag{29}$$

This assumption is evidently satisfied if $\boldsymbol{F}(\boldsymbol{\xi})$ is given by (27), while for (28) it depends on the stochastic process, the extended policy and the revenue function. Some sufficient conditions on the revenue function such as its domination by a square-integrable function of $\boldsymbol{\xi}$ for any value of the decisions can ensure that (29) holds.

3.1 Confidence interval

We refer to $\mathbb{E}[\boldsymbol{F}(\boldsymbol{\xi})]$ by the shorthand θ. The probability distribution of $\boldsymbol{F}$ is not known but it is possible to sample it, although this may take some time since every sample point is obtained through the resolution of the deterministic program (9)-(12). In the notation of this section, the estimators (18) and (24) take the form

$$\widehat{\theta}_{K,M} = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{M}\sum_{m=1}^{M} F^k(\xi^{k,m}), \tag{30}$$

where the set $\{F^k : k = 1, ..., K\}$ is a sample of K independent and identically distributed (i.i.d) copies of $\boldsymbol{F}$ and the set $\{\xi^{k,m} : m = 1, ..., M;\ k = 1, ..., K\}$ is a sample of $K \times M$ i.i.d copies of $\boldsymbol{\xi}$, which are independent of $\boldsymbol{F}$. The estimator (30) samples $\boldsymbol{\xi}$ M-times more than $\boldsymbol{F}$, which is justified by the fact that the realizations of $\boldsymbol{\xi}$ are faster to generate than the realizations of $\boldsymbol{F}$. Additionally, we expect their respective variances to differ greatly, possibly by several orders of magnitude; note that for deterministic methods $\boldsymbol{F}$ has zero variance.

It follows from the above definitions of $\{F^k\}$ and $\{\xi^{k,m}\}$ and the linearity of the expectation that $\widehat{\theta}_{K,M}$ is an unbiased estimator of θ. Consistency is also straightforward: assume condition (29) holds, then the random variables $U_k := \frac{1}{M}\sum_{m=1}^{M} F^k(\xi^{k,m})$, for all $k \in \{1, \dots, K\}$, are integrable and i.i.d., hence by the law of large numbers we have that $\widehat{\theta}_{K,M} = \frac{1}{K}\sum_{k=1}^{K} U_k$ converges in probability to $\mathbb{E}[U_1] = \theta$, as K goes to infinity. Proposition 3.1 below provides a confidence interval for θ.

Proposition 3.1. *Assume condition* (29) *holds. A 100(1 - α)% asymptotic confidence interval for θ is*

$$\mathcal{I}_{K,M}^{1-\alpha} := \left[\widehat{\theta}_{K,M} \pm z_{1-\alpha/2}\left(\frac{\beta + \gamma(M-1)}{KM}\right)^{1/2}\right], \tag{31}$$

where z_α denotes the α-level quantile of a standard normal distribution and β and γ are given by

$$\beta = \mathrm{Var}\left(\boldsymbol{F}(\boldsymbol{\xi})\right), \tag{32}$$

$$\gamma = \mathrm{Cov}(\boldsymbol{F}(\boldsymbol{\xi}^1), \boldsymbol{F}(\boldsymbol{\xi}^2)), \tag{33}$$

with $\boldsymbol{\xi}^1$ and $\boldsymbol{\xi}^2$ two i.i.d copies of $\boldsymbol{\xi}$.

Proof. Condition (29) implies $\mathbb{E}[U_1^2] < +\infty$, which is a sufficient condition for using the central limit theorem on the i.i.d random variables U_k, for all $k \in \{1, \dots, K\}$. It follows from the central limit theorem that

$$\mathbb{P}\left(\left|\frac{\sqrt{K}}{\sqrt{\mathrm{Var}(U_1)}}\left(\frac{1}{K}\sum_{k=1}^{K} U_k - \theta\right)\right| \le z_{1-\alpha/2}\right) \underset{K\to+\infty}{\longrightarrow} \mathbb{P}(|Z| \le z_{1-\alpha/2}) = 1 - \alpha, \tag{34}$$

where Z follows a standard normal distribution and z_α denotes the α-level quantile of Z. Thus, a 100(1 - α)% asymptotic confidence interval for θ is

$$\mathcal{I}_{K,M}^{1-\alpha} = \Big[\frac{1}{K}\sum_{k=1}^{K} U_k \pm \frac{z_{1-\alpha/2}}{\sqrt{K}}\sqrt{\mathrm{Var}(U_1)}\Big] \tag{35}$$

$$= \Big[\widehat{\theta}_{K,M} \pm \frac{z_{1-\alpha/2}}{\sqrt{K}}\Big(\mathrm{Var}\Big(\frac{1}{M}\sum_{m=1}^{M} F^1(\xi^{1,m})\Big)\Big)^{1/2}\Big]. \tag{36}$$

The above variance can be simplified using the fact that the random variables $\{F^1(\xi^{1,m})\}_m$ are i.i.d:

$$\mathrm{Var}\Big(\frac{1}{M}\sum_{m=1}^{M} F^1(\xi^{1,m})\Big) = \frac{1}{M^2}\Big[\sum_{m=1}^{M}\mathrm{Var}\big(F^1(\xi^{1,m})\big) + 2\sum_{i=1}^{M}\sum_{j=i+1}^{M}\mathrm{Cov}\big(F^1(\xi^{1,i}), F^1(\xi^{1,j})\big)\Big] \tag{37}$$

$$= \frac{1}{M}\mathrm{Var}(F^1(\xi^{1,1})) + \frac{M-1}{M}\mathrm{Cov}(F^1(\xi^{1,1}), F^1(\xi^{1,2})). \tag{38}$$

Finally, combining (36) and (38) yields exactly the interval (31). □

The exact values of the parameters β and γ are not available for the same reasons as for θ. We can estimate them using the sample sets $\{F^k\}_k$ and $\{\xi^{k,m}\}_{k,m}$. The coefficient β can be written as $\beta = \mathbb{E}[\boldsymbol{F}(\boldsymbol{\xi})^2] - \mathbb{E}[\boldsymbol{F}(\boldsymbol{\xi})]^2$, hence an estimator $\widehat{\beta}_{K,M}$ is given by

$$\widehat{\beta}_{K,M} = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{M}\sum_{m=1}^{M} F^k(\xi^{k,m})^2 - (\widehat{\theta}_{K,M})^2. \tag{39}$$

As for γ, we can decompose the covariance (33) as follows:

$$\gamma = \mathbb{E}\big[\mathrm{Cov}\big(\boldsymbol{F}(\boldsymbol{\xi}^1), \boldsymbol{F}(\boldsymbol{\xi}^2) \,|\, \boldsymbol{F}\big)\big] + \mathrm{Cov}\big[\mathbb{E}\big(\boldsymbol{F}(\boldsymbol{\xi}^1) \,|\, \boldsymbol{F}\big), \mathbb{E}\big(\boldsymbol{F}(\boldsymbol{\xi}^2) \,|\, \boldsymbol{F}\big)\big]. \tag{40}$$

The covariance in the first term above boils down to zero because the random variables $\boldsymbol{F}(\boldsymbol{\xi}^1)$ and $\boldsymbol{F}(\boldsymbol{\xi}^2)$ are independent conditionally to $\boldsymbol{F}$. Moreover, it follows from the i.i.d condition on $\boldsymbol{\xi}^1$, $\boldsymbol{\xi}^2$ and $\boldsymbol{\xi}$ that $\mathbb{E}[\boldsymbol{F}(\boldsymbol{\xi}^1) \,|\, \boldsymbol{F}) = \mathbb{E}[\boldsymbol{F}(\boldsymbol{\xi}^2) \,|\, \boldsymbol{F}] = \mathbb{E}[\boldsymbol{F}(\boldsymbol{\xi}) \,|\, \boldsymbol{F}]$. Thus, γ can be simplified as

$$\gamma = \mathrm{Var}\big[\mathbb{E}[\boldsymbol{F}(\boldsymbol{\xi}) \,|\, \boldsymbol{F}]\big] = \mathbb{E}\big[\mathbb{E}[\boldsymbol{F}(\boldsymbol{\xi}) \,|\, \boldsymbol{F}]^2\big] - \theta^2, \tag{41}$$

hence an estimator $\widehat{\gamma}_{K,M}$ is

$$\widehat{\gamma}_{K,M} = \frac{1}{K}\sum_{k=1}^{K} U_k^2 - (\widehat{\theta}_{K,M})^2. \tag{42}$$

In particular, it follows from (41) that $\gamma = 0$ if and only if $\boldsymbol{F}$ is a deterministic function, i.e., $\gamma = 0$ if and only if the scenario-tree generation method is deterministic. By using the same argument as above, the estimators $\widehat{\beta}_{K,M}$ and $\widehat{\gamma}_{K,M}$ are consistent and unbiased. Consequently, Slutsky's theorem allows to substitute β and γ with $\widehat{\beta}_{K,M}$ and $\widehat{\gamma}_{K,M}$ in the confidence interval $\mathcal{I}_{K,M}^{1-\alpha}$ of Proposition 3.1 and obtain a fully practical 100(1 - α)% asymptotic confidence interval for θ.

3.2 Optimal sample sizes selection

The bounds of the confidence interval $\mathcal{I}_{K,M}^{1-\alpha}$ depend on the sample sizes K and M. There is a degree of freedom in choosing how to balance the relative sample sizes and this choice clearly affects the approximation quality of $\theta = \mathbb{E}[\boldsymbol{F}(\boldsymbol{\xi})]$. The following algorithmic procedure aims at choosing K and M in an optimal way. By optimal we mean that K and M minimize the confidence interval bound for a given running time limit, or conversely, they minimize the running time for a given confidence interval bound target. To this end, we denote by t_0 and t_1 the generation time of a realization of $\boldsymbol{F}$ and $\boldsymbol{\xi}$, respectively, and by t_2 the evaluation time of the quantity $F^k(\xi^{k,m})$. Thus, one realization of $\widehat{\theta}_{K,M}$ is obtained in $Kt_0 + KM(t_1 + t_2)$ units of time. Algorithm 1 below provides an effective computational scheme for approximating θ, β and γ. It is based on the resolution of the two following non-linear integer optimization problems:

- To minimize the running time for an approximation error smaller than $v > 0$, we choose K and M as the optimal solutions of

$$P_v(\beta,\gamma): \quad \min_{K,M} \quad Kt_0 + KM(t_1+t_2) \tag{43}$$

$$s.t. \quad \frac{\beta + \gamma(M-1)}{KM} \leq v, \tag{44}$$

$$K, M \text{ integers.} \tag{45}$$

- To minimize the approximation error for a running time limit τ, we choose K and M as the optimal solutions of

$$P_\tau(\beta,\gamma): \quad \min_{K,M} \quad \frac{\beta + \gamma(M-1)}{KM} \tag{46}$$

$$s.t. \quad Kt_0 + KM(t_1+t_2) \leq \tau, \tag{47}$$

$$K, M \text{ integers.} \tag{48}$$

Note that $P_\tau(\beta,\gamma)$ has a nonempty feasible set provided $\tau > t_0 + t_1 + t_2$. In practice, we solve the continuous relaxation of P_v and P_τ, i.e., we remove the constraints (45) and (48), and we set (K^*, M^*) to the closest integers that satisfy (44) and (47).

4 Proposed methods to extend the tree policy

4.1 Piecewise-constant extension with the nearest neighbor

The piecewise-constant extension procedure (PC) consists in interpolating and extrapolating each tree optimal decision in a way that $\widetilde{x}_t(\cdot)$ is piecewise constant and, for an argument $\xi_{..t}$, it yields the closest tree optimal decision in the scenario tree. Different metrics will produce different piecewise-constant functions. In this paper, for a stage-1-to-t decision vector $u = (u_1, \ldots, u_t) \in \mathbb{R}^{d\times t}$, we consider the stage-$t$ Euclidean metric

$$\|u\|_t := \Big(\sum_{i=1}^{t} \sum_{j=1}^{d} (u_i)_j^2 \Big)^{1/2}, \tag{49}$$

where $(u_i)_j$ denotes the j-th component of u_i. Two piecewise-constant extended tree policies can be defined, depending on the set of nodes in which we search the closest discretization point. The first

Algorithm 1 Estimation of θ, β et γ.
Inputs: variance target v ; computational time limit τ (large enough).
Outputs: $\widehat{\theta}_{K,M}$, $\widehat{\beta}_{K,M}$ and $\widehat{\gamma}_{K,M}$.

Choose empirically $K_0 \geq 2$ and $M_0 \geq 2$ to have a fast but fairly accurate estimation of t_0, t_1, t_2, $\widehat{\theta}_{K_0,M_0}$, $\widehat{\beta}_{K_0,M_0}$ and $\widehat{\gamma}_{K_0,M_0}$.
if $\widehat{\beta}_{K_0,M_0} + (M_0 - 1)\widehat{\gamma}_{K_0,M_0} \leq vK_0M_0$ **or** $K_0t_0 + K_0M_0(t_1 + t_2) \geq \tau$ **then**
  Return $\widehat{\theta}_{K_0,M_0}$, $\widehat{\beta}_{K_0,M_0}$, $\widehat{\gamma}_{K_0,M_0}$.
else
  Solve $P_v(\widehat{\beta}_{K_0,M_0}, \widehat{\gamma}_{K_0,M_0})$ and retrieve (K^*, M^*).
  if $K^*t_0 + K^*M^*(t_1 + t_2) \leq \tau$ **then**
    Compute and return $\widehat{\theta}_{K^*,M^*}$, $\widehat{\beta}_{K^*,M^*}$, $\widehat{\gamma}_{K^*,M^*}$.
  else
    Solve $P_\tau(\widehat{\beta}_{K_0,M_0}, \widehat{\gamma}_{K_0,M_0})$ and retrieve (K^*, M^*).
    Compute and return $\widehat{\theta}_{K^*,M^*}$, $\widehat{\beta}_{K^*,M^*}$, $\widehat{\gamma}_{K^*,M^*}$.
  end if
end if

choice is to search in the whole set $\mathcal{N}_t$, which we refer to as PC-AT (AT standing for *across tree*). The second choice is to search solely among the children nodes $C(m)$, where $m \in \mathcal{N}_{t-1}$ is the node corresponding to the decision made at stage $t-1$. We refer to the latter as PC-AC (AC standing for *across children*). PC-AC selects the next decisions in the subtree rooted at the current node decision, hence it produces a sequence of decisions that coincides with some sequence of decisions in the scenario tree. Conversely, PC-AT allows to switch among the branches of the tree, which is an appreciable feature to avoid that an unfortunate decision at a previous stage impacts negatively all subsequent decisions. Note that for two-stage programs the two extensions coincide.

Let us now give a formal definition for these extensions. Let $t \in \{1, \dots, T\}$ be a given stage and $m \in \mathcal{N}_{t-1}$ be the stage-$(t-1)$ node of the previous decision. The extended decision functions $\widetilde{x}_t^{\mathrm{AT}}(\cdot)$ and $\widetilde{x}_t^{\mathrm{AC}}(\cdot)$ are piecewise-constant on the Voronoï cells, which are given by

$$V_t^{\mathrm{AT}}(\zeta^{\cdot\cdot n}) = \big\{\xi_{\cdot\cdot t} \in \Xi_{\cdot\cdot t} \;\big|\; \forall r \in \mathcal{N}_t \setminus \{n\},\ \|\xi_{\cdot\cdot t} - \zeta^{\cdot\cdot n}\|_t < \|\xi_{\cdot\cdot t} - \zeta^{\cdot\cdot r}\|_t\big\}, \tag{50}$$

for each $n \in \mathcal{N}_t$ for PC-AT, and

$$V_t^{\mathrm{AC}}(\zeta^{\cdot\cdot n}) = \big\{\xi_{\cdot\cdot t} \in \Xi_{\cdot\cdot t} \;\big|\; \xi_{\cdot\cdot t} = (\zeta^{\cdot\cdot m}, \xi_t),\ \forall r \in C(m) \setminus \{n\},\ \|\xi_{\cdot\cdot t} - \zeta^{\cdot\cdot n}\|_t < \|\xi_{\cdot\cdot t} - \zeta^{\cdot\cdot r}\|_t\big\}, \tag{51}$$

for each $n \in C(m)$ for PC-AC. The sequence $\zeta^{\cdot\cdot n}$ is called the *site* of the cell. The stage-t decision functions of PC-AT and PC-AC are given by

$$\widetilde{x}_t^{\mathrm{AT}}(\xi_{\cdot\cdot t}) = \sum_{n \in \mathcal{N}_t} \widehat{x}^*(\zeta^{\cdot\cdot n})\, \mathbf{1}_{V_t^{\mathrm{AT}}(\zeta^{\cdot\cdot n})}(\xi_{\cdot\cdot t}), \tag{52}$$

and

$$\widetilde{x}_t^{\mathrm{AC}}(\xi_{\cdot\cdot t}) = \sum_{n \in C(m)} \widehat{x}^*(\zeta^{\cdot\cdot n})\, \mathbf{1}_{V_t^{\mathrm{AC}}(\zeta^{\cdot\cdot n})}(\xi_{\cdot\cdot t}), \tag{53}$$

respectively. Simple illustrations of PC-AT, PC-AC and the Voronoï cells are displayed in Figure 2 (a) and Figure 3 (a)-(b) (the Voronoï cells are plotted using the *Voronoï* function in Matlab). They correspond to the scenario tree in Figure 1. An application of these extensions on true stochastic programming problems will be done in the case studies.

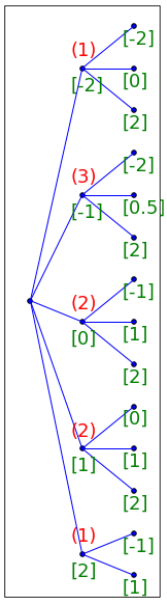


Figure 1: A 3-stage scenario tree. The values in bracket are the discretization points. The values in parenthesis are the tree optimal decisions (only shown at stage 1).

4.2 N-nearest-neigbhor-weighted extension

The extension of the previous section can be generalized to a weighted average over the N-nearest discretization point in the whole scenario tree (NNNW-AT). To define it formally, let us denote by $\mathcal{V}_N(\xi_{..t}) \subseteq \mathcal{N}_t$, for $N \geq 2$, the set of the N closest discretization points to $\xi_{..t}$ in the scenario tree for the Euclidean metric (49). For a given realization $\xi_{..t}$, the stage-t component of the N-nearest-neighbor-weighted policy across tree is given by

$$\widetilde{x}_t(\xi_{..t}) = \sum_{n \in \mathcal{V}_N(\xi_{..t})} \lambda^n(\xi_{..t})\, \widehat{x}^*(\zeta^{..n}), \tag{54}$$

where $\lambda^n(\cdot)$ is a weight function that we define as

$$\lambda^n(\xi_{..t}) = \left[\sum_{l \in \mathcal{V}_N(\xi_{..t})} \prod_{m \in \mathcal{V}_N(\xi_{..t}) \setminus \{l\}} \|\xi_{..t} - \zeta^{..m}\|_t \right]^{-1} \prod_{m \in \mathcal{V}_N(\xi_{..t}) \setminus \{n\}} \|\xi_{..t} - \zeta^{..m}\|_t, \tag{55}$$

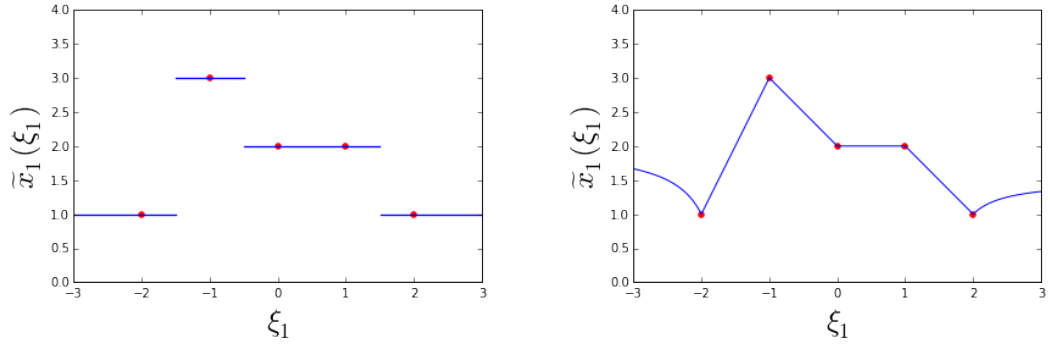
for every $n \in \mathcal{V}_N(\xi_{..t})$. This definition is justified by the fact that $\lambda^n(\cdot)$ satisfies:

(i) $\lambda^n(\cdot) \geq 0$;

(ii) $\sum_{n \in \mathcal{V}_N(\xi_{..t})} \lambda^n(\xi_{..t}) = 1$ for every $\xi_{..t} \in \Xi_{..t}$;

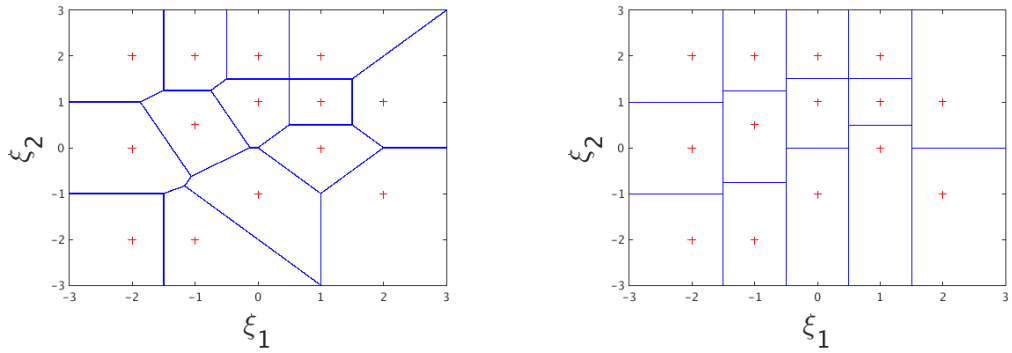
(iii) $\lambda^n(\zeta^{..n}) = 1$ and $\lambda^m(\zeta^{..n}) = 0$ for every $m \in \mathcal{V}_N(\zeta^{..n}) \setminus \{n\}$.

Items (i) and (ii) imply that $\widetilde{x}_t(\xi_{..t})$ is a convex combination of the tree decisions $\widehat{x}^*(\zeta^{..n})$ for $n \in \mathcal{V}_N(\xi_{..t})$. Item (iii) implies that $\widetilde{x}_t(\cdot)$ satisfies $\widetilde{x}_t(\zeta^{..n}) = \widehat{x}^*(\zeta^{..n})$, which is a requirement of Definition 1.1. A simple illustration of this extension is displayed in Figure 2 (b), for the scenario tree in Figure 1. In the case studies it is performed on true stochastic programming problems.



(a) PC (b) 2NNW

Figure 2: Stage-1 extended decision functions of PC (a) and 2NNW (b), for the scenario tree in Figure 1.



(a) PC-AT (b) PC-AC

Figure 3: Voronoï cells (50) and (51) in the support of $(\boldsymbol{\xi}_1, \boldsymbol{\xi}_2)$ for PC-AT (a) and PC-AC (b), and for the scenario tree in Figure 1. The +-markers are the sites of the cells.

5 Numerical experiments

5.1 Scenario-tree generation methods and extension procedures

In this section, we apply the mathematical framework developed in Section 2, 3 and 4 on two case studies. The scenario-tree generation methods and extension procedures that we consider are given in Table 1. All the resolutions are done on symmetric trees (ST) with a constant branching coefficient across the stages, i.e., $|C(n)|$ is constant for all $n \in \mathcal{N} \setminus \mathcal{N}_T$. We choose this structure because Monte Carlo sampling (MC) and randomized quasi-Monte Carlo sampling (RQMC) are methods that do not provide a procedure to generate a tree structure. The numerical results of this section may be further improved by choosing a structure that is suitable for the considered stochastic problem and scenario-tree generation method. In particular, for the optimal quantization method (OQ) this can be done using the algorithms in Pflug and Pichler (2015). In our experiments, the difference between the extended policies arises from the discretization method of the stochastic process (OQ, RQMC or MC) and from the extension procedure (PC-AC, PC-AT or 2NNW-AT).

The resolution of both case studies relies on the discretization of a standard normal distribution. The OQ method is done with Algorithm 2 in Pflug and Pichler (2015). This algorithm minimizes the Wasserstein distance of order 2 between the standard normal distribution and its approxima-

Notation	Description
PC-AC	Piecewise-constant extension across children
PC-AT	Piecewise-constant extension across tree
2NNW-AT	2-nearest-neighbor-weighted extension across tree

(a) Extension procedures

Notation	Description
ST-OQ	Symmetric tree (ST) with optimal quantization method (OQ)
ST-RQMC	Symmetric tree with randomized quasi-Monte Carlo sampling (RQMC)
ST-MC	Symmetric tree with Monte Carlo sampling (MC)

(b) Scenario-tree generation methods

Table 1: Methods considered in the numerical experiments.

tion sitting on finitely many points. The output is a set of discretization points along with their corresponding weights. The RQMC sampling is done by randomizing a low discrepancy set of N points in $[0, 1)$ and by transforming it with the inverse Gaussian cumulative ϕ^{-1} (approximated using Moro's transformation (Moro, 1995)), so that the points cover the real line. For an input u that is a realization of a continuous uniform distribution in $[0, 1]$, the output set of points is

$$\big\{\phi^{-1}(\{i/N + u\}) \,:\, i = 0, \ldots, N-1\big\}, \tag{56}$$

where $\{\cdot\}$ is the fractional part function. We refer the reader to Koivu (2005) for the use of RQMC in stochastic programming. Note that different realizations u yield different sets of points, therefore ST-RQMC is a stochastic scenario-tree generation method. The last discretization method is standard MC sampling, which provides the so-called Sample Average Approximation (SAA) of the stochastic program. MC sampling is performed without variance reduction techniques; see Shapiro (2003) for the SAA properties and a description on variance reduction techniques. For simplicity, the weights w^n for the methods ST-RQMC and ST-MC are set to the inverse of the branching coefficient at node n, i.e., $w^n = 1/|C(a(n))|$.

The first case study is the newsvendor problem. This two-stage program is a classic case study in stochastic programming, since it is one of the few problems for which closed-form optimal solutions are available. The second case study is a four-stage multi-product assembly problem, presented in Shapiro et al. (2014). In our experiments, we use the numerical parameters and the stochastic process introduced in Defourny et al. (2013). We assess the efficiency of each scenario-tree generation method and extension procedure with the help of the quality parameters introduced in Section 2 and also by considering the running time of each method. The numerical experiments are implemented in Python 2.7.4 on a Linux machine with Intel Xeon X5472 @ 3.00GHz. We use CPLEX 12.6.1.0 with default setting to solve the optimization programs.

5.2 Case study 1: the newsvendor problem

A newsvendor buys to a supplier a quantity x_0 of newspapers at stage 0 at a fixed price a. At stage 1, the newsvendor sells $x_{1,1}$ newspapers at price b and returns $x_{1,2}$ to the supplier, who pays c for each newspaper returned. Demand for newspapers is given by a positive random variable $\boldsymbol{\xi}_1$. For simplicity, we consider that the decision variables and the random variable can take continuous

values. The two-stage stochastic program is

$$\max_{(x_0,x_{1,1},x_{1,2})} \quad -a\,x_0 + \mathbb{E}[b\,x_{1,1}(\boldsymbol{\xi}_1) + c\,x_{1,2}(\boldsymbol{\xi}_1)] \tag{57}$$

$$\text{s.t.} \quad x_{1,1}(\boldsymbol{\xi}_1) \leq \boldsymbol{\xi}_1; \tag{58}$$

$$x_{1,1}(\boldsymbol{\xi}_1) + x_{1,2}(\boldsymbol{\xi}_1) \leq x_0; \tag{59}$$

$$x_0,\, x_{1,1}(\cdot),\, x_{1,2}(\cdot) \geq 0. \tag{60}$$

A complete presentation of the newsvendor problem can be found in Birge and Louveaux (1997) and Shapiro et al. (2014). In our experiments, we use the parameters $a = 2$, $b = 5$, $c = 1$ and the demand follows a log-normal distribution, i.e., $\log(\boldsymbol{\xi}_1)$ follows a normal distribution with mean $\mu = \log(200)$ and standard deviation $\sigma = 1/\sqrt{2}$. The optimal value and the optimal 1st stage solution rounded off to the second decimal are $Q(x^*) = 500.25$ and $x_0^* = 322.23$ (a detailed derivation can be found in Proulx (2014), Chapter 5). The optimal 2nd stage recourse decisions are $x_{1,1}^*(\boldsymbol{\xi}_1) = \min(x_0^*, \boldsymbol{\xi}_1)$ and $x_{1,2}^*(\boldsymbol{\xi}_1) = \max(x_0^* - \boldsymbol{\xi}_1, 0)$.

Quantitative analysis

The numerical experiments are performed with the methods in Table 1 and for scenario trees with 5, 20, 40 and 80 scenarios (which are the branching coefficient at the root node). We recall that for two-stage programs the extensions *across tree* or *across children* coincide, thus we remove the suffix -AT and -AC. The quality parameters are computed with the sample sizes given in Table 2. The sample sizes are obtained by applying Algorithm 1 on the quality parameter (64) with a 95%-confidence-interval-bound target of $0.1\% \times Q(x^*)$ and a computational time limit of 1h for each method.

	5 scen.		20 scen.		40 scen.		80 scen.	
	K^*	M^*	K^*	M^*	K^*	M^*	K^*	M^*
ST-OQ	1	2,186,936	1	2,118,314	1	2,274,600	1	2,188,189
ST-RQMC	186,336	22	80,006	23	43,390	26	19,616	34
ST-MC	160,366	20	74,776	25	40,717	29	21,867	24

Table 2: Optimal sample sizes K^* and M^*.

Table 3 displays the probability of feasibility $p(1)$ and the conditional revenues CR for each random extended policy $\widetilde{\boldsymbol{x}} = (\widetilde{\boldsymbol{x}}_0, \widetilde{\boldsymbol{x}}_{1,1}, \widetilde{\boldsymbol{x}}_{1,2})$. The numerical values are rounded off to the first non-statistically significant decimal. A direct application of Definitions 2.1 and 2.2 yields

$$p(1) = \mathbb{P}(\boldsymbol{\xi}_1 \in \widetilde{\Xi}_1(\widetilde{\boldsymbol{x}})), \tag{61}$$

and

$$\text{CR} = \mathbb{E}[-a\,\widetilde{\boldsymbol{x}}_0 + b\,\widetilde{\boldsymbol{x}}_{1,1}(\boldsymbol{\xi}_1) + c\,\widetilde{\boldsymbol{x}}_{1,2}(\boldsymbol{\xi}_1) \,|\, \boldsymbol{\xi}_1 \in \widetilde{\Xi}_1(\widetilde{\boldsymbol{x}})], \tag{62}$$

where $\widetilde{\Xi}_1(\widetilde{x})$ is given by

$$\widetilde{\Xi}_1(\widetilde{x}) = \{\xi_1 \in \Xi_1 \,|\, \widetilde{x}_0, \widetilde{x}_{1,1}(\cdot), \widetilde{x}_{1,2}(\cdot) \text{ satisfy } (58), (59), (60)\}. \tag{63}$$

Recall that the probability and the expectation above are taken with respect to the probability distribution of $\boldsymbol{\xi}$ and $\widetilde{\boldsymbol{x}}$. Table 3 reveals that the extension 2NNW yields policies of better quality than PC. This is true in particular when it is combined with the scenario-tree generation methods

ST-OQ and ST-RQMC, for which $p(1)$ is greater than 0.98 and CR is almost at optimality for 80 scenarios. ST-OQ suits particularly well the extension 2NNW, since its probability of feasibility is already at 95.7% for only 5 scenarios, in contrast to 89.5% and 75.6% for ST-RQMC/2NNW and ST-MC/2NNW, respectively. As expected, we observe than the probability of feasibility increases as the number of scenarios increases.

	5 scen.		20 scen.		40 scen.		80 scen.	
	$p(1)$	CR	$p(1)$	CR	$p(1)$	CR	$p(1)$	CR
ST-OQ/PC	0.618	102.1	0.620	114.3	0.622	116.9	0.622	117.2
ST-OQ/2NNW	**0.957**	**101.8**	**0.998**	**100.1**	**0.999**	**100.1**	**0.997**	**100.4**
ST-RQMC/PC	0.612	112.7	0.603	118.7	0.612	119.2	0.618	119.0
ST-RQMC/2NNW	0.895	109.0	**0.961**	**104.7**	**0.976**	**103.1**	**0.986**	**101.8**
ST-MC/PC	0.609	100.1	0.612	113.5	0.617	116.5	0.619	117.9
ST-MC/2NNW	0.756	102.0	0.790	106.3	0.799	107.1	0.803	107.5

Table 3: Quality parameters $p(1)$ and CR (CR is in percentage of $Q(x^*)$). Data in bold font single out the methods that satisfy $p(1) \geq 0.95$ and CR $\geq 99\% \times Q(x^*)$, which can be considered as satisfactory.

Table 4 displays the estimates of $\mathbb{E}_{\overline{\boldsymbol{x}}}[Q(\overline{\boldsymbol{x}})]$, where $\overline{\boldsymbol{x}} = (\widetilde{\boldsymbol{x}}_0, r_{1,1}, r_{1,2})$. They correspond to the expected revenues from the implementation of the stage-0 decision $\widetilde{\boldsymbol{x}}_0$ of the scenario tree and the stage-1 optimal recourse $r_{1,1}(x_0;\xi_1) := \min(x_0, \xi_1)$ and $r_{1,2}(x_0;\xi_1) := \max(x_0 - \xi_1; 0)$, i.e.,

$$\mathbb{E}_{\overline{\boldsymbol{x}}}[Q(\overline{\boldsymbol{x}})] = \mathbb{E}[-a\,\widetilde{\boldsymbol{x}}_0 + b\,r_{1,1}(\widetilde{\boldsymbol{x}}_0;\boldsymbol{\xi}_1) + c\,r_{1,2}(\widetilde{\boldsymbol{x}}_0;\boldsymbol{\xi}_1)]. \tag{64}$$

Each revenue is displayed with its corresponding 95% confidence interval bound (column $\pm\text{CI}_{95\%}$). The values in Table 4 measure how close the (random) stage-0 decision of each scenario-tree generation method is from the ideal method that would yield x_0^* with probability one. We see that the method ST-OQ is statistically significantly better than ST-RQMC and ST-MC for 5 scenarios. As the number of scenarios increases, the methods ST-OQ and ST-RQMC become indistinguishable and both provide stage-0 decision extremely close to optimality. ST-MC yields the poorest quality decision and the difference is statistically significant for all scenario sizes.

	5 scen.		20 scen.		40 scen.		80 scen.	
	Est.	$\pm\text{CI}_{95\%}$	Est.	$\pm\text{CI}_{95\%}$	Est.	$\pm\text{CI}_{95\%}$	Est.	$\pm\text{CI}_{95\%}$
ST-OQ	99.78	0.11	100.02	0.10	100.06	0.10	99.89	0.10
ST-RQMC	98.69	0.09	99.83	0.16	99.95	0.20	99.96	0.25
ST-MC	91.44	0.12	97.22	0.16	98.68	0.19	99.27	0.28

Table 4: $\mathbb{E}_{\overline{\boldsymbol{x}}}[Q(\overline{\boldsymbol{x}})]$ with $\overline{\boldsymbol{x}} = (\widetilde{\boldsymbol{x}}_0, r_{1,1}, r_{1,2})$ (in percentage of $Q(x^*)$).

To push further the quality analysis, we provide in Table 5 the estimates of $\mathbb{E}[\widehat{Q}^*]$ and $\mathbb{E}[|\widetilde{\boldsymbol{x}}_0 - x_0^*|]$ obtained from 10,000 resolutions for each scenario-tree generation method. We see in Table 5 (a) that the value of $\mathbb{E}[\widehat{Q}^*]$ is the highest for ST-MC and the closest to $Q(x^*)$ for ST-RQMC. Thus, if we compare the methods based on the value of $\mathbb{E}[\widehat{Q}^*]$ only, then we would state that either ST-MC or ST-RQMC is the highest quality method, which would contradict the results of Table 4. The estimates in Table 5 (b) can solved this alleged contradiction. They confirm that the average stage-0 error $\mathbb{E}[|\widetilde{\boldsymbol{x}}_0 - x_0^*|]$ is much larger for ST-MC and ST-RQMC than for ST-OQ. In particular, for

ST-MC this error ranges from 5 to 10 times the one of ST-OQ, while for ST-RQMC it is about twice as large as for ST-OQ. These numerical results demonstrate what we discussed in Section 1.3, i.e., that using $\mathbb{E}[\widehat{Q}^*]$ to compare scenario-tree generation methods is not reliable with regard to the quality of the decisions. In this case study, we see that selecting the method with the highest value of $\mathbb{E}[\widehat{Q}^*]$ turns out to be the worst choice. Likewise, choosing the method that best approximates $Q(x^*)$ is not the right choice either, although the error is not as significant. We emphasize that the estimates in Table 5 (b) cannot be computed in a real-world problem because x_0^* is not known. This demonstrates the importance and reliability of the selection criterion (25) to single out the best scenario-tree generation method.

	5 scen.		20 scen.		40 scen.		80 scen.	
	Est.	$\pm\text{CI}_{95\%}$	Est.	$\pm\text{CI}_{95\%}$	Est.	$\pm\text{CI}_{95\%}$	Est.	$\pm\text{CI}_{95\%}$
ST-OQ	103.19	-	100.20	-	100.05	-	100.011	-
ST-RQMC	102.12	0.26	99.993	0.067	100.012	0.034	99.992	0.017
ST-MC	111.09	0.74	102.75	0.33	101.25	0.24	100.75	0.16

(a) $\mathbb{E}[\widehat{Q}^*]$ (in percentage of $Q(x^*)$).

	5 scen.		20 scen.		40 scen.		80 scen.	
	Est.	$\pm\text{CI}_{95\%}$	Est.	$\pm\text{CI}_{95\%}$	Est.	$\pm\text{CI}_{95\%}$	Est.	$\pm\text{CI}_{95\%}$
ST-OQ	6.47	-	3.11	-	1.45	-	0.89	-
ST-RQMC	11.85	0.15	5.96	0.07	2.89	0.03	1.40	0.02
ST-MC	31.25	0.51	17.66	0.28	12.47	0.19	8.59	0.13

(b) $\mathbb{E}[|\widetilde{\boldsymbol{x}}_0 - x_0^*|]$ (in percentage of x_0^*).

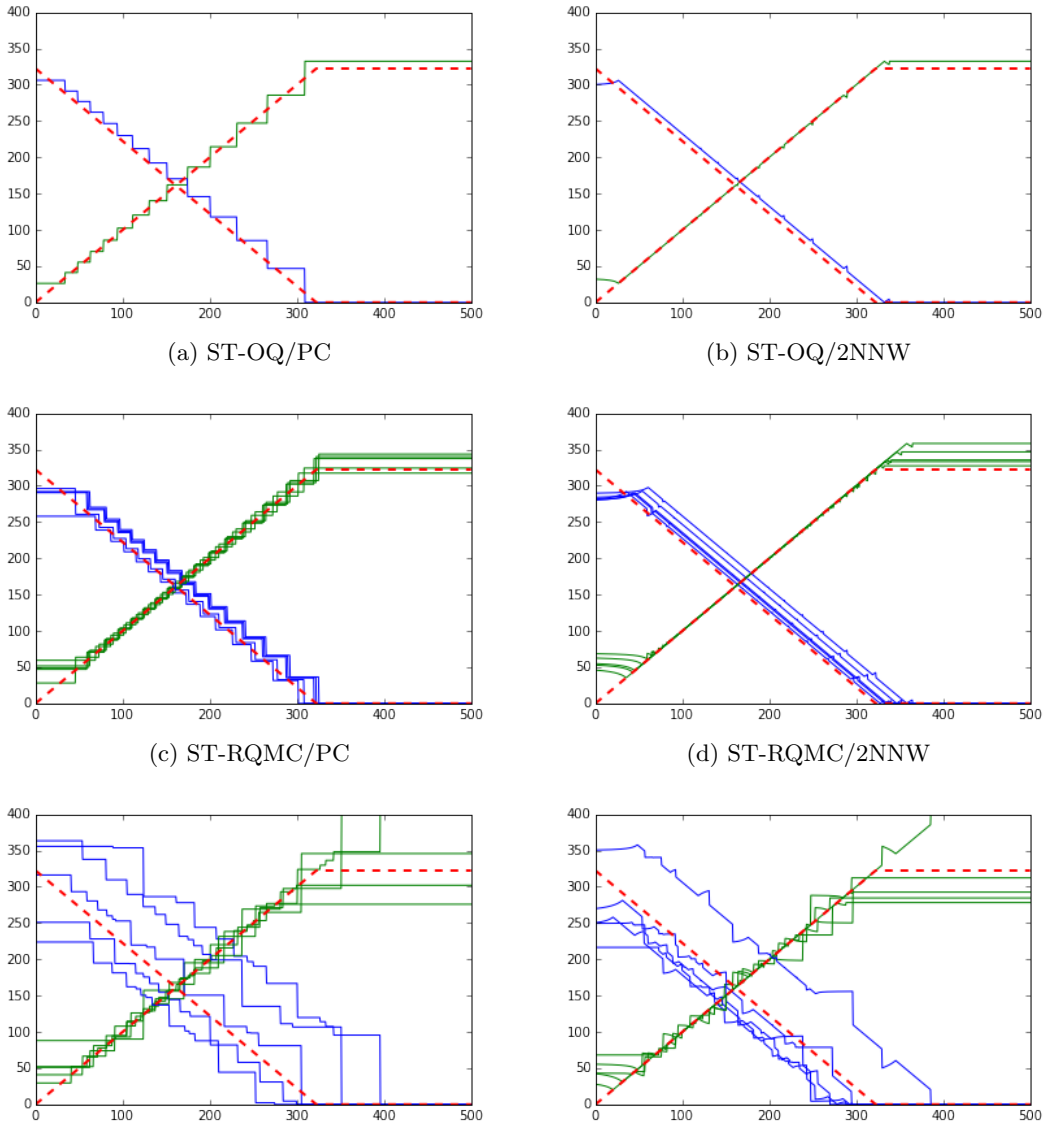
Table 5: Estimates of $\mathbb{E}[\widehat{Q}^*]$ and $\mathbb{E}[|\widetilde{\boldsymbol{x}}_0 - x_0^*|]$. For ST-RQMC and ST-MC the estimates are obtained from 10,000 resolutions.

Qualitative analysis

The simplicity of the newsvendor problem allows us to compare graphically the stage-1 extended decision functions $\widetilde{x}_{1,1}$ and $\widetilde{x}_{1,2}$ with their optimal counterparts. This is done for 20 scenarios in Figure 4. As expected from the above quantitative analysis, the combination ST-OQ/2NNW in (b) yields stage-1 decisions that fit very well the optimal ones. The approximation quality is also quite good for ST-RQMC/2NNW, as we see in (d) where five realizations of $\widetilde{\boldsymbol{x}}_{1,1}$ and $\widetilde{\boldsymbol{x}}_{1,2}$ are displayed. As for ST-MC in (e) and (f), we see that $\widetilde{\boldsymbol{x}}_{1,1}$ and $\widetilde{\boldsymbol{x}}_{1,2}$ have a large variability due to the absence of variance reduction techniques and an erratic behavior due to a discretization scheme that covers the support of $\boldsymbol{\xi}_1$ less uniformly than ST-RQMC and ST-OQ. The same plots for 40 and 80 scenarios (not shown here) reveal that an increase of the scenario size improves the quality of the extended decision functions, which seem to converge to their optimal counterparts. For ST-RQMC and ST-MC, the convergence implies that the spread reduces. Empirically, we observe that the convergence is the fastest for ST-OQ and the slowest for ST-MC.

Conclusion

It follows from the above analyses that the scenario-tree generation method ST-OQ and the extension procedure 2NNW yield the highest quality decision policy overall. This is true in particular for a small number of scenarios, typically less than 20, for which the difference can be substantial



(a) ST-OQ/PC (b) ST-OQ/2NNW

(c) ST-RQMC/PC (d) ST-RQMC/2NNW

(e) ST-MC/PC (f) ST-MC/2NNW

Figure 4: $\widetilde{\boldsymbol{x}}_{1,1}$ and $\widetilde{\boldsymbol{x}}_{1,2}$ (solid lines) compared with $x^*_{1,1}$ and $x^*_{1,2}$ (dashed lines) for 20 scenarios (for ST-RQMC and ST-MC five realizations of $\widetilde{\boldsymbol{x}}_{1,1}$ and $\widetilde{\boldsymbol{x}}_{1,2}$ are displayed on the same plot). The x-axis is the demand ξ_1.

compared to the other methods. For more than 20 scenarios, however, the method ST-RQMC together with 2NNW yield good quality decisions as well. We observed that the running time of each method is about the same.

5.3 Case study 2: the multi-product assembly problem

We consider the four-stage multi-product assembly problem presented in Defourny et al. (2013) with the same numerical parameters and stochastic process. The stochastic program is

$$\max_{(x_0,\dots,x_3)} \quad -c_0^T x_0 + \mathbb{E}\big[- c_1^T x_1(\boldsymbol{\xi}_1) - c_2^T x_2(\boldsymbol{\xi}_{..2}) + c_3^T x_3(\boldsymbol{\xi}_{..3})\big] \tag{65}$$

$$\text{s.t.} \quad \sum_{j=1}^{8} A_{i,j}\, x_{1,j}(\boldsymbol{\xi}_1) \le x_{0,i},\ i = 1,\dots,12; \tag{66}$$

$$\sum_{j=1}^{5} B_{i,j}\, x_{2,j}(\boldsymbol{\xi}_{..2}) \le x_{1,i}(\boldsymbol{\xi}_1),\ i = 1,\dots,8; \tag{67}$$

$$x_{3,i}(\boldsymbol{\xi}_{..3}) \le \max(0, \boldsymbol{\xi}_{3,i}),\ i = 1,\dots,5; \tag{68}$$

$$x_{3,i}(\boldsymbol{\xi}_{..3}) \le x_{2,i}(\boldsymbol{\xi}_{..2}),\ i = 1,\dots,5; \tag{69}$$

$$x_0,\ x_1(\cdot),\ x_2(\cdot),\ x_3(\cdot) \ge 0. \tag{70}$$

The decision variables x_0, x_1, x_2, x_3 have 12, 8, 5, 5 components, respectively. The variable $x_{0,i}$ is the quantity of product i purchased at stage 0 at a price $c_{0,i}$. The variables $x_{1,i}$ and $x_{2,i}$ are the quantities of product i produced at stages 1 and 2, at costs $c_{1,i}$ and $c_{2,i}$. The production is done using the quantities of products available at the previous stage and in proportions given by the matrices A and B in (66)-(67). The variable $x_{3,i}$ is the quantity of product i sold at the end of the horizon. The optimal stage-3 recourse decisions are readily deducible from the constraints (68)-(69): $r_{3,i}(x_{..2}; \xi_{..3}) = \max(\xi_{3,i}, x_{2,i})$ for each $i \in \{1,\dots,5\}$. The random vector $\boldsymbol{\xi}_3$ has 5 components that correspond to the random demand for each product. The components are linearly dependent from each other and each one has a marginal distribution that is a normal distribution with its specific mean and variance. The optimal value of this program is estimated by Defourny et al. (2013) to be about 375. We refer the reader to this paper for all details. The numerical experiments are performed with the methods in Table 1 and for scenario trees with 125, 512 and 1000 scenarios (corresponding to the branching coefficients 5, 8 and 10, respectively). The sample sizes are obtained by applying Algorithm 1 on the quality parameter $\mathbb{E}_{\overline{\boldsymbol{x}}}[Q(\overline{\boldsymbol{x}})]$ with a 95%-confidence-interval-bound target of 1.0 and a computational time limit of 4h for each method. We display in Table 6 the sample sizes for the methods that are considered in Table 9.

	125 scen.		512 scen.		1000 scen.	
	K^*	M^*	K^*	M^*	K^*	M^*
ST-OQ/PC-AC	1	1,482,773	1	1,674,287	1	1,546,985
ST-RQMC/PC-AC	9,168	218	1776	177	579	429
ST-MC/PC-AC	10,232	83	1815	98	611	278

Table 6: Optimal sample sizes K^* and M^*.

Table 7 displays the stagewise evolution of the probability of feasibility of each extended policy. The numerical values are rounded off to the first non-statistically significant decimal. Table 7

shows that the stage-1 decision functions are always feasible up to the rounded-off approximation. This implies that the specification of a stage-1 recourse function is not required for the feasibility restoration. At stage 2, the results are different depending on the extension procedure. The extension PC-AC keeps yielding feasible decisions with probability one, however, PC-AT and 2NNW-AT start yielding infeasible decisions with positive probability. In particular for PC-AT, the value $p(2)$ remains above 95% when the method is combined with ST-OQ or ST-RQMC, which is satisfactory. This is not the case for 2NNW-AT, whose value of $p(2)$ is below 50% for any scenario-tree generation method. At stage 3, the probability of feasibility is much smaller than one for every method. This is due to the fact that the random vector appears in the stage-3 constraint (68). For this reason, the estimates of CR in Table 8 must be taken with a grain of salt. Overall, the probability of feasibility reveals a hierarchy between the scenario-tree generation methods. The method ST-OQ performs better than ST-RQMC, which in turn performs better than ST-MC. A hierarchy also exists between the extension procedures. The extension PC-AC performs better than PC-AT, which in turn performs better than 2NNW-AT.

	125 scen.			512 scen.			1000 scen.		
	$p(1)$	$p(2)$	$p(3)$	$p(1)$	$p(2)$	$p(3)$	$p(1)$	$p(2)$	$p(3)$
ST-OQ/PC-AC	**1**	**1**	**0.637**	**1**	**1**	**0.669**	**1**	**1**	**0.680**
ST-OQ/PC-AT	**1**	**0.986**	**0.622**	**1**	**0.986**	**0.655**	**1**	**0.986**	**0.663**
ST-OQ/2NNW-AT	1	0.401	0.257	1	0.395	0.269	1	0.395	0.253
ST-RQMC/PC-AC	**1**	**1**	**0.618**	**1**	**1**	**0.657**	**1**	**1**	**0.670**
ST-RQMC/PC-AT	**1**	**0.958**	**0.577**	**1**	**0.958**	**0.612**	**1**	**0.957**	**0.625**
ST-RQMC/2NNW-AT	1	0.330	0.129	1	0.414	0.236	1	0.417	0.265
ST-MC/PC-AC	**1**	**1**	**0.570**	**1**	**1**	**0.612**	**1**	**1**	**0.631**
ST-MC/PC-AT	1	0.871	0.446	1	0.879	0.498	1	0.872	0.509
ST-MC/2NNW-AT	1	0.356	0.121	1	0.448	0.225	1	0.434	0.240

Table 7: Quality parameters $p(1)$, $p(2)$, $p(3)$. Data in bold font single out the methods that satisfy $p(2) \geq 0.95$, which can be considered as satisfactory.

	125 scen.	512 scen.	1000 scen.
ST-OQ/PC-AC	521	514	521
ST-OQ/PC-AT	516	516	523
ST-OQ/2NNW-AT	597	637	563
ST-RQMC/PC-AC	546	541	533
ST-RQMC/PC-AT	534	539	527
ST-RQMC/2NNW-AT	265	530	562
ST-MC/PC-AC	519	529	518
ST-MC/PC-AT	525	534	550
ST-MC/2NNW-AT	418	600	628

Table 8: Quality parameter CR.

In the multi-product assembly problem, the optimal stage-3 recourse function r_3 is known, hence we can build a fully-feasible extended policy of the form $\overline{\boldsymbol{x}} = (\widetilde{\boldsymbol{x}}_0, \widetilde{\boldsymbol{x}}_1, \widetilde{\boldsymbol{x}}_2, r_3)$ for any method that satisfy $p(2) = 1$ in Table 7. This condition is satisfied for all scenario-tree generation methods

combined with the extension PC-AC. In Table 9, we display the expected revenues $\mathbb{E}_{\overline{\boldsymbol{x}}}[Q(\overline{\boldsymbol{x}})]$ corresponding to implementation of $\overline{\boldsymbol{x}}$ for these methods. The estimates reveal that the method ST-OQ outperforms ST-RQMC and ST-MC for 125 scenarios. However, as the number of scenarios increases, the difference between ST-RQMC and ST-OQ reduces and both methods yield high quality policies for 1000 scenarios (recall that the optimal value is estimated at 375). The quality of the method ST-MC remains far below the other methods and the difference is statistically significant.

	125 scen.		512 scen.		1000 scen.	
	Est.	$\pm\text{CI}_{95\%}$	Est.	$\pm\text{CI}_{95\%}$	Est.	$\pm\text{CI}_{95\%}$
ST-OQ/PC-AC	366.6	1.1	369.5	1.1	371.9	1.1
ST-RQMC/PC-AC	349.3	1.7	367.9	3.4	371.9	4.1
ST-MC/PC-AC	297.1	2.1	330.3	3.9	339.1	5.3

Table 9: $\mathbb{E}_{\overline{\boldsymbol{x}}}[Q(\overline{\boldsymbol{x}})]$ where $\overline{\boldsymbol{x}} = (\widetilde{\boldsymbol{x}}_0, \widetilde{\boldsymbol{x}}_1, \widetilde{\boldsymbol{x}}_2, r_3)$,. The computation is done only for the methods that satisfy $p(2) = 1$ in Table 7.

	125 scen.		512 scen.		1000 scen.	
	Est.	$\pm\text{CI}_{95\%}$	Est.	$\pm\text{CI}_{95\%}$	Est.	$\pm\text{CI}_{95\%}$
ST-OQ	383.3	-	377.2	-	376.5	-
ST-RQMC	385.5	5.3	377.5	3.1	376.5	2.5
ST-MC	422.5	11.7	415.0	9.4	396.9	8.7

Table 10: Estimates of $\mathbb{E}[\widehat{Q}^*]$. For ST-RQMC and ST-MC the values are obtained from 1,000 resolutions.

The runtime for each scenario-tree generation method and extension procedure is displayed in Table 11. We observe in (a) that all methods solve the problem in a similar time. We observe in (b) that the extension PC-AC yields the decisions in the shortest time and does not suffer of the exponential growth of the scenario tree, unlike PC-AT and 2NNW-AT.

	125 scen.	512 scen.	1000 scen.
ST-OQ	1.3	7.7	23.8
ST-RQMC	1.3	7.8	23.9
ST-MC	1.3	7.8	24.3

(a) t_0 (in sec.).

	125 scen.	512 scen.	1000 scen.
PC-AC	1.4	1.7	2
PC-AT	7.1	24	46
2NNW-AT	8.2	26	47

(b) $t_1 + t_2$ (in millisec.).

Table 11: Computational time for the multi-product assembly problem. The value t_0 is the time to generate and solve the deterministic program. The value t_1 and t_2 are the times to generate a realization of the demand and to obtain the corresponding decisions, respectively (cf. Section 3.2).

Overall, the above analysis demonstrates that high quality decisions are available at little cost for the multi-product assembly problem. Specifically, a decision policy obtained from scenario trees with 1000 scenarios generated by ST-OQ or ST-RQMC and extended by PC-AC provide revenues that are about 1% away from optimality on average (assuming the optimal value is 375). Note that for a small number of scenarios, ST-OQ outperforms the other methods as in the previous case study. Once again, in this case study the estimates of $\mathbb{E}[\widehat{Q}^*]$ in Table 10 are not reliable, since ST-MC has the highest value for 125, 512 and 1000 scenarios.

We end this section by showing that the four-stage multi-product assembly problem should not be solved by a deterministic approach consisting in substituting $\boldsymbol{\xi}$ by its expectation in the stochastic program. Indeed, we observe that a policy obtained by this method provides revenues of value 263 ± 1, with 95% confidence. This value is much smaller than the revenues of the best scenario-tree policies in Table 9.

6 Conclusion

In this paper, we proposed a framework to assess the quality of the scenario-tree generation methods to solve high-dimensional stochastic programs. This framework is based on the concept of extension procedure, that takes a tree optimal policy and extends the decisions so that the policy is defined for every value of the random parameters. We introduced three quality parameters to assess any scenario-tree generation method and extension procedure. The quality parameters can be estimated numerically to provide the decision-maker with a way to select the highest quality method for the considered stochastic programming problem.

The whole framework of this paper is applied on a two-stage newsvendor problem and a four-stage multi-product assembly problem, with a random demand given by a log-normal and normal distribution, respectively. We demonstrate that a simple extension technique that set the decisions as a weighted average over the nearest node(s) provides good quality decisions at little computational cost. The analysis of the quality parameters demonstrates that, among Monte Carlo sampling, randomized quasi-Monte Carlo sampling and optimal quantization method, the latter yields the best scenario trees regarding the quality of the decisions in both case studies. We show that this conclusion can be reached using the quality parameters introduced in this paper but not by looking only at the optimal values of the approximate problems on the scenario trees, which can be misleading. This last statement should be kept in mind when using scenario trees to solve real-world stochastic programming problems.

Future work will be directed toward the applicability of the quality parameters in the context of a rolling horizon solution method, where only the stage-0 decision is implemented and new scenario trees are generated at each stage. Future work will also be directed toward the evaluation of the decision policies on *rare events*, which are low probability subsets of outcomes for which the revenue function takes extremely low or high values. The scenario-tree generation method may not always incorporate those events in the tree, which may result in overoptimistic policies that would provide decisions with disastrous consequences should one of these events occur. Such a context may require the investigation of other extension procedures.

A Appendix: Notation

Notation	Description
T	optimization horizon
$\boldsymbol{\xi}_t$	random vector at stage t
$\boldsymbol{\xi}$ or $(\boldsymbol{\xi}_1, ..., \boldsymbol{\xi}_T)$	stochastic process
$\boldsymbol{\xi}_{..t}$	components from stage 0 to stage t of $\boldsymbol{\xi}$
ξ_t (resp. ξ; resp. $\xi_{..t}$)	a realization of $\boldsymbol{\xi}_t$ (resp. $\boldsymbol{\xi}$; resp. $\boldsymbol{\xi}_{..t}$)
Ξ_t (resp. Ξ; resp. $\Xi_{..t}$)	support of $\boldsymbol{\xi}_t$ (resp. $\boldsymbol{\xi}$; resp. $\boldsymbol{\xi}_{..t}$)
d	number of random parameters revealed at each period
s	number of decisions made at each stage
$q(\cdot;\cdot)$	revenue function
$Q(\cdot)$	expected revenues
$Q(x^*)$	optimal value of the stochastic program
$x_t(\cdot)$	decision function at stage t
x_0	decision vector at stage 0
x or $(x_0, ..., x_T)$	decision policy
$x_{..t}$	components from stage 0 to stage t of x
x^*	optimal policy of the stochastic program
X_t	feasible decision set at stage t
$\mathcal{N}$	node set of the scenario tree
$\mathcal{E}$	edge set of the scenario tree
$\mathcal{N}^*$	set of nodes minus the root
$\mathcal{N}_t$	node set at stage t
n_0	root node
$a(n)$	ancestor node of n
$C(n)$	set of children nodes of n
ζ^n	discretization vector at node n
$\zeta^{..n}$	discretization sequence leading to node n
w^n	weight of node n
W^n	product of the weights leading to node n
$\widehat{Q}^*$	optimal value of a deterministic program
$\widehat{x}^n$ or $\widehat{x}(\zeta^{..n})$	decision vector at node n
$\widehat{x}^{..n}$	sequence of decision vectors leading to node n
$\widehat{x}^{*n}$ or $\widehat{x}^*(\zeta^{..n})$	optimal decision vector at node n

Table 12: Notation introduced in Section 1.1 and 1.2.

References

Ben-Tal, A., El Ghaoui, L., and Nemirovski, A. (2009). *Robust optimization*. Princeton University Press.

Birge, J. R. and Louveaux, F. (1997). *Introduction to stochastic programming*. Springer, New York.

Chen, M. and Mehrotra, S. (2008). Epi-convergent scenario generation method for stochastic problems via sparse grid. *Stochastic Programming E-Print Series*, (7).

Chen, M., Mehrotra, S., and Papp, D. (2015). Scenario generation for stochastic optimization problems via the sparse grid method. *Computational Optimization and Applications*, 62(3):669–692.

Chiralaksanakul, A. and Morton, D. P. (2004). Assessing policy quality in multi-stage stochastic programming. *Stochastic Programming E-Print Series*, (12).

Defourny, B., Ernst, D., and Wehenkel, L. (2011). Multistage stochastic programming: A scenario tree based approach. *Decision Theory Models for Applications in Artificial Intelligence: Concepts and Solutions: Concepts and Solutions*, page 97.

Defourny, B., Ernst, D., and Wehenkel, L. (2013). Scenario trees and policy selection for multistage stochastic programming using machine learning. *INFORMS Journal on Computing*, 25(3):488–501.

Dupačová, J., Gröwe-Kuska, N., and Römisch, W. (2003). Scenario reduction in stochastic programming. *Mathematical programming*, 95(3):493–511.

Edirisinghe, N. (1999). Bound-based approximations in multistage stochastic programming: Nonanticipativity aggregation. *Annals of Operations Research*, 85:103–127.

Frauendorfer, K. (1996). Barycentric scenario trees in convex multistage stochastic programming. *Mathematical Programming*, 75(2):277–293.

Heitsch, H. and Römisch, W. (2009). Scenario tree modeling for multistage stochastic programs. *Mathematical Programming*, 118(2):371–406.

Hilli, P. and Pennanen, T. (2008). Numerical study of discretizations of multistage stochastic programs. *Kybernetika*, 44(2):185–204.

Høyland, K., Kaut, M., and Wallace, S. W. (2003). A heuristic for moment-matching scenario generation. *Computational Optimization and Applications*, 24(2-3):169–185.

Høyland, K. and Wallace, S. W. (2001). Generating scenario trees for multistage decision problems. *Management Science*, 47(2):295–307.

Kaut, M. and Wallace, S. W. (2003). Evaluation of scenario-generation methods for stochastic programming. *Stochastic Programming E-Print Series*, (14).

Koivu, M. (2005). Variance reduction in sample approximations of stochastic programs. *Mathematical programming*, 103(3):463–485.

Kouwenberg, R. (2001). Scenario generation and stochastic programming models for asset liability management. *European Journal of Operational Research*, 134(2):279–292.

Küchler, C. and Vigerske, S. (2010). Numerical evaluation of approximation methods in stochastic programming. *Optimization*, 59(3):401–415.

Louveaux, F. (1998). An introduction to stochastic transportation models. In *Operations Research and Decision Aid Methodologies in Traffic and Transportation Management*, pages 244–263. Springer.

Mak, W.-K., Morton, D. P., and Wood, R. K. (1999). Monte carlo bounding techniques for determining solution quality in stochastic programs. *Operations Research Letters*, 24(1):47–56.

Moro, B. (1995). The full Monte. *Risk*, 8(2):57–58.

Pennanen, T. (2005). Epi-convergent discretizations of multistage stochastic programs. *Mathematics of Operations Research*, 30(1):245–256.

Pennanen, T. and Koivu, M. (2002). Integration quadratures in discretization of stochastic programs. *Stochastic Programming E-Print Series*, (11).

Pflug, G. C. (2001). Scenario tree generation for multiperiod financial optimization by optimal discretization. *Mathematical programming*, 89(2):251–271.

Pflug, G. C. and Pichler, A. (2011). Approximations for probability distributions and stochastic optimization problems. In Bertocchi, M., Consigli, G., and Dempster, M. A. H., editors, *Stochastic Optimization Methods in Finance and Energy*, pages 343–387. Springer.

Pflug, G. C. and Pichler, A. (2015). Dynamic generation of scenario trees. *Computational Optimization and Applications*, 62(3):641–668.

Powell, W. B. and Topaloglu, H. (2003). Stochastic programming in transportation and logistics. In Ruszczyński, A. and Shapiro, A., editors, *Handbooks in operations research and management science: Stochastic Programming*, volume 10, pages 555–635. Elsevier.

Proulx, S. (2014). Génération de scénarios par quantification optimale en dimension élevée. Master's thesis, École Polytechnique de Montréal.

Rockafellar, R. T. and Wets, R. J. (1974). Continuous versus measurable recourse in n-stage stochastic programming. *Journal of Mathematical Analysis and Applications*, 48(3):836–859.

Ruszczyński, A. and Shapiro, A. (2003). Stochastic programming models. In Ruszczyński, A. and Shapiro, A., editors, *Handbooks in operations research and management science: Stochastic Programming*, volume 10, pages 1–64. Elsevier.

Shapiro, A. (2003). Monte Carlo sampling methods. In Ruszczyński, A. and Shapiro, A., editors, *Handbooks in operations research and management science: Stochastic Programming*, volume 10, pages 353–425. Elsevier.

Shapiro, A. (2006). On complexity of multistage stochastic programs. *Operations Research Letters*, 34(1):1–8.

Shapiro, A., Dentcheva, D., and Ruszczyński, A. (2014). *Lectures on stochastic programming: modeling and theory*, volume 16. SIAM.

Wallace, S. W. and Fleten, S.-E. (2003). Stochastic programming models in energy. In Ruszczyński, A. and Shapiro, A., editors, *Handbooks in operations research and management science: Stochastic Programming*, volume 10, pages 637–677. Elsevier.

Yu, L.-Y., Ji, X.-D., and Wang, S.-Y. (2003). Stochastic programming models in financial optimization: A survey. *AMO - Advanced Modeling and Optimization*, 5(1).