Scenarios for multistage stochastic programs

Jitka Dupačová[1],Giorgio Consigli[2] and Stein W. Wallace[3]

[1] *Department of Probability and Mathematical Statistics, Charles University*
Sokolovská 83, 186 75 Prague, Czech Republic. E-mail: dupacova@karlin.mff.cuni.cz

[2] *UniCredito Italiano, Piazza Cordusio (C.P. 1242), I-20102 Milano, Italy.*
E-mail: consigli@GruppoCredit.it

[3] *Department of Industrial Economics and Technology Management*
Norwegian University of Science and Technology
N-7491 Trondheim, Norway. E-mail: sww@iot.ntnu.no

A major issue in any application of multistage stochastic programming is the representation of the underlying random data process. We discuss the case when enough data paths can be generated according to an accepted parametric or nonparametric stochastic model. No assumptions on convexity with respect to the random parameters are required. We emphasize the notion of representative scenarios (or a representative scenario tree) relative to the problem being modeled.

Keywords: : Scenarios and scenario trees, clustering, importance sampling, matching moments, problem oriented requirements, inference and bounds.

1 Multistage stochastic programs

In the general T -*stage stochastic program* we think of a stochastic data process

$$\omega = \{\omega_1, \dots, \omega_{T-1}\} \quad \text{or} \quad \omega = \{\omega_1, \dots, \omega_T\}$$

whose realizations are (multidimensional) data trajectories in $(\Omega, \mathcal{F}, P)$ and of a vector decision process

$$\boldsymbol{x} = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_T\},$$

a measurable function of ω. The sequence of decisions and observations is

$$\boldsymbol{x}_1, \omega_1, \boldsymbol{x}_2(\boldsymbol{x}_1, \omega_1), \omega_2, \dots, \boldsymbol{x}_T(\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_{T-1}, \omega_1, \dots, \omega_{T-1}) = \boldsymbol{x}_T(\boldsymbol{x}_1, \omega_1, \dots, \omega_{T-1}). \tag{1}$$

In addition, ω_T may contribute to the overall costs. The decision process is *nonanticipative* in the sense that decisions taken at any stage of the process do not depend on future *realizations* of the random parameters or on future decisions, whereas it is the past information and the given *probabilistic specification* $(\Omega, \mathcal{F}, P)$ of the process ω which are exploited. The dependence of the decisions solely on the history and on the probabilistic specification can be expressed as follows: Denote by $\mathcal{F}_{t-1} \subseteq \ \mathcal{F}$ the σ-field generated by the observations $\omega^{t-1,\bullet} := \{\omega_1, \dots, \omega_{t-1}\}$ of the part of the stochastic data process that precedes stage t. The dependence of the t-th stage decision $\boldsymbol{x}_t$ only on these past observations means that $\boldsymbol{x}_t$ is $\mathcal{F}_{t-1}$-adapted or, in other words, that $\boldsymbol{x}_t$ is measurable with respect to $\mathcal{F}_{t-1}$. In each of the stages, the decision is limited by constraints that may depend on the previous decisions and observations.

Two formulations of multistage stochastic programming problems can be used. For general results concerning their equivalence see, e.g., [64], for an introductory survey see [26].

Let $\mathcal{X}_t$ be given nonempty sets in $R^{n_t}, t = 1, \ldots, T$, and denote by

$$\mathcal{X}_t(\omega) = \left\{\boldsymbol{x}^{t\bullet} \in \mathcal{X}_1 \times \mathcal{X}_2 \times \ldots \times \mathcal{X}_t : f_{ti}(\boldsymbol{x}^{t\bullet}, \omega^{t-1,\bullet}) \leq 0, i = 1, \ldots, m_t\right\} \tag{2}$$

the t-th stage constraints, $t = 2, \ldots, T$, and by $f_0(\boldsymbol{x}, \omega)$ the overall cost connected with the decision process (1). The form of (2) reflects the requirement that the choice of decisions from $\mathcal{X}_t(\omega)$ is not explicitly constrained by future decisions and observations. However, this does not, in general, exclude the presence of *induced constraints* which must be fulfilled to guarantee the existence of a feasible nonanticipative decision process $\boldsymbol{x}$ for almost all ω.

The *T-stage stochastic program* is to find

$$\boldsymbol{x}_1 \in \mathcal{X}_1 \text{ and } \boldsymbol{x}_t \quad \mathcal{F}_{t-1}\text{–measurable, } \boldsymbol{x}^{t\bullet} \in \mathcal{X}_t(\omega), t = 2, \ldots, T \text{ a.s.,} \tag{3}$$

$$\text{that minimizes } E\left\{f_0(\boldsymbol{x}_1, \boldsymbol{x}_2(\omega), \ldots, \boldsymbol{x}_T(\omega), \omega)\right\}.$$

The special choice of the function f_0 in (3), as an indicator function of a certain interval, leads to the *probability objective function* of the form

$$P\{g_0(\boldsymbol{x}_1, \boldsymbol{x}_2(\omega), \ldots, \boldsymbol{x}_T(\omega), \omega) \notin \mathcal{I}\}$$

where $\mathcal{I}$ is a given interval of desirable values of g_0. Similarly, the replacement of the constraints $\boldsymbol{x}^{t\bullet} \in \mathcal{X}_t(\omega)$, a.s., $t = 1, \ldots, T$, by the requirement that $\boldsymbol{x}^{t\bullet} \in \mathcal{X}_t(\omega)$, $t = 1, \ldots, T$, holds true with a prescribed probability, provides stochastic programs with *probabilistic* or *chance constraints*.

The *second formulation of the T-stage stochastic program* is based on a recursive evaluation of the overall objective function which allows us to write the multistage stochastic program as a sequence of nested two-stage programs and spells out the nonanticipativity conditions in an explicit way:

$$\min E\left\{f_0(\boldsymbol{x}, \omega)\right\} := f_{10}(\boldsymbol{x}_1) + E_{\omega_1}\left\{\varphi_{10}(\boldsymbol{x}_1, \omega_1)\right\} \tag{4}$$

subject to

$$\boldsymbol{x}_1 \in \mathcal{X}_1 \text{ and } f_{1i}(\boldsymbol{x}_1) \leq 0, \quad i = 1, \ldots, m_1,$$

where for $t = 2, \ldots, T$, $\varphi_{t-1,0}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{t-1}, \omega_1, \ldots, \omega_{t-1})$ is the optimal value of the stochastic program

$$\min f_{t0}(\boldsymbol{x}_t) + E_{\omega_t|\omega^{t-1,\bullet}}\left\{\varphi_{t0}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t, \omega_1, \ldots, \omega_{t-1}, \omega_t)\right\} \tag{5}$$

with respect to $\boldsymbol{x}_t \in \mathcal{X}_t$ and subject to

$$f_{ti}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{t-1}, \boldsymbol{x}_t, \omega_1, \ldots, \omega_{t-1}) \leq 0, i = 1, \ldots, m_t.$$

Here, $\varphi_{T,0} \equiv 0$ or is an explictly given function of $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T, \omega_1, \ldots, \omega_T$. All constraints involving random parameters hold almost surely.

A special case is the following multistage stochastic *linear* program with recourse

$$\min \boldsymbol{c}_1^\top \boldsymbol{x}_1 + E_{\omega_1} \{\varphi_1(\boldsymbol{x}_1, \omega_1)\} \tag{6}$$

subject to

$$\begin{aligned} \boldsymbol{A}_1 \boldsymbol{x}_1 &= \boldsymbol{b}_1 \\ \boldsymbol{l}_1 \leq \ \boldsymbol{x}_1 &\leq \boldsymbol{u}_1, \end{aligned}$$

where the functions $\varphi_{t-1}, t = 2, \ldots, T$, are defined recursively as

$$\varphi_{t-1}(\boldsymbol{x}_{t-1}, \omega_{t-1}) = \inf_{\boldsymbol{x}_t} \left[\boldsymbol{c}_t(\omega_{t-1})^\top \boldsymbol{x}_t + E_{\omega_t|\omega_{t-1}} \{\varphi_t(\boldsymbol{x}_t, \omega_t)\}\right] \tag{7}$$

subject to

$$\begin{aligned} \boldsymbol{B}_t(\omega_{t-1})\, \boldsymbol{x}_{t-1} + \boldsymbol{A}_t(\omega_{t-1})\, \boldsymbol{x}_t = \boldsymbol{b}_t(\omega_{t-1}), &\quad \text{a.s.}, \\ \boldsymbol{l}_t \leq \boldsymbol{x}_t \leq \boldsymbol{u}_t, & \end{aligned}$$

and $\varphi_T \equiv \ 0$.

For simplicity, we denote by ω_{t-1} the random vector that generates the coefficients $\boldsymbol{b}_t, \boldsymbol{c}_t$ and matrices $\boldsymbol{A}_t, \boldsymbol{B}_t$ in the decision problem of the t-th stage, $t = 2, \ldots, T$. Both the individual stages' objectives and the constraints introduced have a certain Markov structure (compare with the scheme (5)). We assume that the $\boldsymbol{A}_t$ are (m_t, n_t) matrices and that the remaining vectors and matrices are of consistent dimensions. Furthermore, we suppose that the corresponding expectations E are well defined. For the first stage, known values of all elements of $\boldsymbol{b}_1, \boldsymbol{c}_1, \boldsymbol{A}_1$ are assumed. The main decision variable is $\boldsymbol{x}_1$ that corresponds to the first stage.

Finally, the *T-stage stochastic linear program with recourse and with a finite number of scenarios* written in the arborescent form (compare with (2), (3)) is

$$\min \boldsymbol{c}_1^\top \boldsymbol{x}_1 + \sum_{k_2=2}^{K_2} p_{k_2} \boldsymbol{c}_{k_2}^\top \boldsymbol{x}_{k_2} + \sum_{k_3=K_2+1}^{K_3} p_{k_3} \boldsymbol{c}_{k_3}^\top \boldsymbol{x}_{k_3} + \ldots + \sum_{k_T=K_{T-1}+1}^{K_T} p_{k_T} \boldsymbol{c}_{k_T}^\top \boldsymbol{x}_{k_T} \tag{8}$$

subject to constraints

$$\begin{array}{llllll} \boldsymbol{A}_1 \boldsymbol{x}_1 & & & & = & \boldsymbol{b}_1, \\ \boldsymbol{B}_{k_2} \boldsymbol{x}_1 & +\boldsymbol{A}_{k_2} \boldsymbol{x}_{k_2} & & & = & \boldsymbol{b}_{k_2},\ k_2 \in \mathcal{K}_2, \\ & \boldsymbol{B}_{k_3} \boldsymbol{x}_{a(k_3)} & +\boldsymbol{A}_{k_3} \boldsymbol{x}_{k_3} & & = & \boldsymbol{b}_{k_3},\ k_3 \in \mathcal{K}_3, \\ & \ddots & \ddots & & & \vdots \\ & & \boldsymbol{B}_{k_T} \boldsymbol{x}_{a(k_T)} & +\boldsymbol{A}_{k_T} \boldsymbol{x}_{k_T} & = & \boldsymbol{b}_{k_T},\ k_T \in \mathcal{K}_T, \end{array}$$

$$\boldsymbol{l}_t \leq \boldsymbol{x}_{k_t} \leq \boldsymbol{u}_t, \qquad k_t = K_{t-1} + 1, \ldots, K_t, \quad t = 1, \ldots, T, \tag{9}$$

with $K_1 = 1$, $\mathcal{K}_t = \{K_{t-1} + 1, \ldots, K_t\}$, $t = 2, \ldots, T$.

We denote here by $a(k_t)$ the immediate ancestor of k_t, so that $a(k_2) = 1, \quad k_2 = 2, \ldots, K_2$. Furthermore, $p_{k_t} > 0$ for all k_t, with $\sum_{k_t=K_{t-1}+1}^{K_t} p_{k_t} = 1, t = 2, \ldots, T$, are *path* *probabilities* of the corresponding subsequences of realizations $(\boldsymbol{c}_{k_\tau}, \boldsymbol{A}_{k_\tau}, \boldsymbol{B}_{k_\tau}, \boldsymbol{b}_{k_\tau}), \tau \leq t$. Finally, $p_{k_T} \equiv p_s$ are probabilities of the individual scenarios ω^s, that can be obtained by multiplication of the (conditional) *arc probabilities*. The nonanticipativity constraints are implicit in this formulation.

The size of the linear program (8), (9) can be very large; for instance, consider just the two-stage problem with random right hand sides $\boldsymbol{b}_{k_2}$ for $k_2 = 2, \ldots, K_2$, each consisting of m_2 independent random components with probability distributions approximated by the alternative ones: it gives $K_2 = 2^{m_2} + 1$, hence, $m_1 + m_2(2^{m_2} - 1)$ equations in (9). The usefulness of special numerical techniques based on decomposition, aggregation, sampling and parallelization is obvious, see e.g. [19], [21], [35], [41], [54], [60], [65], [66]. Many solvers (CPLEX, MSLiP-OSL, OSL-SP, etc.) are currently available for the solution of multistage problems with linear constraints and nonlinear objectives. Hence, in applications, it is the modeling part of the problem, including a meaningful generation of scenarios, which has become the most demanding part.

The stages do not necessarily refer to time periods, they correspond to steps in the decision process. The main interest lies in the first-stage decisions which consist of all decisions that have to be selected before the information is revealed, whereas the second-stage decisions are allowed to adapt to this information, etc.

2 Scenarios, scenario trees and their generation

For scenario-based multistage stochastic programs one assumes that the probability distribution P of ω is discrete, and concentrated on a finite number of points, say, $\omega^1, \ldots, \omega^S$. Accordingly, the supports $\mathcal{S}_t(\omega^{t-1,\bullet})$ of the conditional probability distributions of ω_t conditioned by past realizations $\omega^{t-1,\bullet} = \{\omega_1, \ldots, \omega_{t-1}\}$ are finite sets. The sequences of realizations $\omega^{t,\bullet} = \{\omega_1, \ldots, \omega_t\}$ are called *scenarios at stage* t and the condition on sensible scenarios at stage t is

$$\omega_\tau \in \mathcal{S}_\tau(\omega^{\tau-1,\bullet})\, \forall \tau > 1. \tag{10}$$

Hence, the set of all considered scenarios is

$$\mathcal{S} := \{\omega^1, \ldots, \omega^S\} = \{\omega | \omega_t \in \mathcal{S}_t(\omega^{t-1,\bullet})\, \forall t > 1\}. \tag{11}$$

Denote by $\mathcal{S}^t$ the (finite) sets of all different scenarios $\omega^{t,\bullet}$ at stage t which satisfy (10), i.e., $\mathcal{S} := \mathcal{S}^T$, and by $\mathcal{S}_t$ the (finite) supports of the marginal probability distributions of $\omega_t, t = 1, \ldots, T$.

The associated conditional probabilities $P(\omega_t | \omega^{t-1,\bullet})$ on $\mathcal{S}_t(\omega^{t-1,\bullet})$ for $t > 1$ and the marginal probabilities $P(\omega_1)$ on $\mathcal{S}_1$ are called the *arc probabilities.* Their products $P(\omega^{t-1,\bullet}) = P(\omega_1) \prod_{\tau=2}^{t-1} P(\omega_\tau | \omega^{\tau-1,\bullet})$ are the *path probabilities* and the probability p_s of scenario $\omega^s = \{\omega_1^s, \ldots, \omega_T^s\} \in \mathcal{S}$ is

$$p_s = P(\omega^s) = P(\omega_1^s) \prod_{\tau=2}^{T} P(\omega_\tau^s | \omega_1^s, \ldots, \omega_{\tau-1}^s).$$

The decisions at stage $t = 1, \ldots, T$ depend on the sequence of observed realizations of the random variables in the preceding stages, i.e., on the section $(\omega_1^s, \ldots, \omega_{t-1}^s)$ in case of the s-th scenario. The first-stage decision variables are scenario independent and the last stage decisions depend on the sections $\omega^{T-1,\bullet}$. The last component ω_T of ω (observable after the last decision $\boldsymbol{x}_T$ is taken) contributes to the evaluation of the final outcome of the decision process.

An adequate representation of the underlying random data is needed for any application of stochastic programming. An additional problem related to multistage stochastic programs is the required special structure of the input in a form consistent with (10)–(11). We can think of it as of an oriented graph which starts from a root (the only node at level 0) and branches into nodes at level 1, each corresponding to one of the possible realizations of ω_1, and the branching continues up to nodes at level T assigned to the whole possible data paths $\omega^{T,\bullet}$. A common special arrangement is the *scenario tree* which is based on the additional assumption that there is a one-to-one correspondence between the sections $\omega^{t,\bullet}$ and the nodes of the tree at stage t for $t = 1, \dots, T$. This means that for any node at level t, each of the new observations ω_t must have only one immediate predecessor $\omega^{t-1,\bullet}$, i.e., a node at level $t-1$, and a (finite) number of descendants ω_{t+1} which result in nodes at level $t+1$, $t < T$; compare with (8)–(9). The number of descendants of all nodes at a given level $0 \le t < T$ of the scenario tree can be equal; this is typical for the initial structure of the tree. If this occurs for all stages, the scenario tree is *balanced* and the structure of the tree can be coded as a product of the number of descendants of the root and of nodes at levels $1, \dots, T-1$. For example $3^2 2^2$ describes the structure of a scenario tree for 4 stages with 3 branches from the root and from the first level nodes and 2 branches from nodes at the second and third levels. The total number of scenarios equals the numerical value of this product, i.e., 36. *See Figure 1.*

Two special cases of the scenario tree are worth mentioning:

- For all stages $t = 2, \dots, T$, the conditional probabilities $P(\omega_t | \omega^{t-1,\bullet})$ are independent of $\omega^{t-1,\bullet}$ and are equal to the marginal probabilities $P(\omega_t)$ – the *interstage independence.* In this case, scenario generation methods for two-stage problems, reviewed in [27], apply to each stage separately. Evidently, the scenario tree must be balanced.

- For all stages $t = 2, \dots, T$, the supports $\mathcal{S}_t(\omega^{t-1,\bullet})$ of conditional probability distributions of ω_t conditioned by realizations $\omega^{t-1,\bullet} = \{\omega_1, \dots, \omega_{t-1}\}$ of sections $\omega^{t-1,\bullet}$ are singletons. This means that the scenario tree is nothing else but a "fan" of individual scenarios $\omega^s = \{\omega_1^s, \dots, \omega_T^s\}$ which occur with probabilities $p_s = P(\omega_1^s)\, \forall s$ and, independently of the number of periods, the multiperiod stochastic program reduces to the *two-stage* one. *See Figure 2.*

  Can we have fan of 36 scenarios to relate the figure with the previous one?

An alternative data arrangement relaxes the requirement of unique predecessors in the previous stages and assumes instead special *recombining* properties of the data paths. Recombining trees or lattices help to reduce the number of nodes and, subsequently, the size of the solved problem provided the evaluation of the t-stage coefficients does not depend on the path $\omega^{t-1,\bullet}$ which leads to the respective node at level $t-1$. This is the case for the coefficients in (7).

Except for the two special cases mentioned above, to build a representative scenario tree is a crucial problem in applications. It can be approached from the point of view of a suitable data manipulation ([4], [36], [37], [53]). It includes a decision concerning the number of stages and the branching scheme with an appropriate labeling technique (cf. [50]) to avoid ambiguity in the definition of data and variables. It should reflect both the underlying probability assumptions and the existing data, and be linked with the purpose of the application. It often asks for compromises between a manageable problem size and the desired precision of the results. In this context, authors have looked at design strategies for *aggregating nodes and stages*, see [46] and [70], *trimming or refining of trees*, see [11], [18] and [21] for a reduction technique and importance sampling method based on the concept of the expected value of perfect information (EVPI). The contamination technique can be used to test the influence of including additional scenarios and stages [25]. [34] offers a possibility to design strategies for refining the barycentric scenario trees. We shall describe the importance sampling technique for constructing the scenario tree in Subsection 3.1, and we introduce below the main ideas of the contamination method. We emphasize that in line with the scope of the paper, these two methods are independent of the structure of the model and of the specific application area.

The contamination method applies to general multistage problems rewritten into the form (4), briefly

$$\min F(\boldsymbol{x}, P) := E_\omega\{f_0(\boldsymbol{x}, \omega)\} \tag{12}$$

on a closed nonempty set $\mathcal{X}$ which does not depend on P. This notation is used to underline the dependence of the problem on the chosen probability distribution P of ω on $(\Omega, \mathcal{F})$.

Inclusion of additional scenarios or branches of the scenario tree means to pass from the initial probability distribution P to the probability distribution

$$P_\lambda = (1 - \lambda)P + \lambda Q, \quad 0 \leq \lambda \leq 1. \tag{13}$$

The probability distribution P is contaminated by the probability distribution Q which is carried by the additional scenarios or branches of the scenario tree. For fixed probability distributions P, Q, the expected value in (12) computed for the contaminated distribution P_λ is *linear* in the parameter λ and under mild assumptions, its optimal value

$$\varphi(\lambda) := \min_{\boldsymbol{x} \in \mathcal{X}} F(\boldsymbol{x}, P_\lambda)$$

is a finite concave function on $[0, 1]$ with a derivative $\varphi'(0^+)$ at $\lambda = 0^+$.

Bounds on the optimal value $\varphi(\lambda)$ for an arbitrary $\lambda \in [0, 1]$ follow by properties of concave functions:

$$(1 - \lambda)\varphi(0) + \lambda\varphi(1) \leq \varphi(\lambda) \leq \varphi(0) + \lambda\varphi'(0^+) \quad \forall \lambda \in [0, 1]. \tag{14}$$

An upper bound for the derivative $\varphi'(0^+)$ equals $F(\boldsymbol{x}(0), Q) - \varphi(0)$ where $\boldsymbol{x}(0)$ is an arbitrary optimal solution of the initial problem (12) obtained for the probability distribution P; in case of a unique optimal solution, this upper bound is attained. Hence, the evaluation of bounds in (14) requires the solution of another stochastic program of type (12) for the new distribution Q to get $\varphi(1)$ and evaluation of the expectation $F(\boldsymbol{x}(0), Q)$ at an already known optimal solution of the initial problem (12), but for the contaminating distribution Q.

The contamination method does not depend on any specific assumption about the probability distribution and it provides globally valid bounds. Small values of the contamination parameter λ are typical for various stability studies, the choice of λ may reflect for example the degree of confidence in expert opinions, a problem considered in [47]. By a suitable choice of the contaminating distribution Q, one can study the influence of including additional branches at specified nodes of the initial scenario tree, cf. [25], to test the recommended number of stages and to emphasize the importance of a scenario by increasing its probability.

An illustrative example

Inspired by [47], consider the problem of investment decisions in the debt and equity markets in the US, Germany and Japan. Historical data allows us to construct many scenarios concerning returns of investments in the considered assets categories. Denote these (in principle equiprobable) scenarios by ω^s, $s = 1, \ldots, S$, and let P be the corresponding discrete probability distribution. Assume that for each of these scenarios, an outcome of a feasible investment strategy, say, $\boldsymbol{x} \in \mathcal{X}$ can be evaluated as $f(\boldsymbol{x}, \omega^s)$. Maximization of the expected outcome

$$F(\boldsymbol{x}, P) = \frac{1}{S} \sum_{s=1}^{S} f(\boldsymbol{x}, \omega^s) \text{ with respect to } \boldsymbol{x} \in \mathcal{X}$$

provides the optimal value $\varphi(P)$ and an optimal investment strategy $\boldsymbol{x}(P)$.

The historical data do not cover all possible extremal situations on the market. However, experts in the investment committee may foresee such events. Assume that they are aggregated in an additional scenario ω^*. This is the only atom of the degenerated probability distribution Q, for which the best investment strategy is $\boldsymbol{x}(Q)$—an optimal solution to $\max_{\boldsymbol{x} \in \mathcal{X}} f(\boldsymbol{x}, \omega^*)$. The contamination method is based on the probability distribution P_λ, carried by the scenarios ω^s, $s = 1, \ldots, S$, and on the experts' scenario ω^* with probabilities $\frac{1-\lambda}{S}$ for $s = 1, \ldots, S$, and $p^* = \lambda$. The probability λ assigns a weight to the view of the investment committee and the bounds (14) (multiplied by -1) are valid for all $0 \leq \lambda \leq 1$. They clearly indicate how much the weight λ, interpreted as the degree of confidence to the investor's view, affects the outcome of the portfolio allocation.

The impact of a modification of every single scenario according to the investor's views on the performance of each asset class can be studied in a similar way. We use the initial probability distribution P contaminated by Q, which is now carried by equiprobable scenarios $\hat{\omega}^s = \omega^s + \delta^s$, $s = 1, \ldots, S$. The contamination parameter λ relates again to the degree of confidence in the experts' view.

We now concentrate on the generation of scenario trees for *interstage dependent* components ω_t of ω. We discuss these problems for models that do not assume convexity with respect to the random parameters and for the cases when enough data paths can be generated in accordance with a parametric or nonparametric stochastic model. This specification excludes the generation of scenario trees which exploit extremal properties of families of probability distributions, cf. [33], and scenario trees that are based solely on experts' oppinion, e.g., [3], [30], [31] or [68]. Even within this problem specification, the notion of representative scenarios, or of a representative scenario tree, depends on the real-life background of the underlying optimization problem. The main goal of a scenario generation procedure is to get a representative scenario tree which supports sensible decisions and is acceptable for the user. For further discussions related to requirements on scenario generation for stochastic programming asset and liability management models, see [58] and various other papers included in the collection [71].

3 From data paths to a scenario tree

In many application areas, such as finance, water resource management, and electricity production, there exist advanced continuous and discrete time stochastic models and historical time series that serve to calibrate these models. We refer to [13], [43] and to a survey in [11] for diffusion type models applied in financial problems, and to [15], [16], [38], [42], [67] or [23] for multivariate autoregression models exploited in water resources problems, the modeling of electricity demand or in finance. Using the calibrated model, or its time discretization, one can simulate, in principle, arbitrarily many sample paths of ω. These models employ a specified type of probability distributions, mainly the (multi)normal ones. Successful examples are the global scenario system developed by Towers Perrin [55] or the model FAM [22] for simulation of economic and capital market scenarios.

Nonparametric methods for scenario generation can be applied for very large families of probability distributions which cannot be indexed by a finite dimensional parameter in a natural way; another term used in this connection is *distribution-free* methods. The simplest idea is to use as scenarios the past observations obtained under comparable circumstances and assign them equal probabilities; see for instance [69] for scenarios of future electricity demand in a given period of a year. Similarly, [56] suggests to construct scenarios of joint asset returns for a T-period model as $n - T$ distinct T-tuples of their subsequent observations from n previous periods and to assign them equal probabilities, $(n - T)^{-1}$.

The available data are often in the form of correlated multivariate time series of different lenghts, various time steps, with missing observations, with change points, etc. The dimensionality of the random components ω_t can be reduced by methods of multivariate statistics: *Factor analysis* is mostly applied under the normality assumption, whereas the *principal components* method is distribution-free. Both of these methods aim at explaining the correlation structure of the data by a small number of *independent* factors or components, which can be simulated separately and used to replace the original multidimensional and intercorrelated data process. These ideas appear in various papers, e.g., [14], [41], [43], [56]. A detailed description of the various statistical procedures designed for time series analysis lies beyond the scope of this paper. Let us assume instead that a suitable preprocessing of these time series has resulted in S different equiprobable "observations" of ω and that this input information has been accepted as an approximation of the underlying stochastic process for the considered decision model.

We focus on the generation of scenario trees for the cases when the main random factors have been detected and enough data paths of their realizations can be generated in accordance with the above mentioned parametric or nonparametric methods. The first important step is to delineate the initial structure of the scenario tree, i.e., the number of stages and the branching scheme. The stages are connected with the possibility to take additional decisions based on newly released information. Such information can be obtained at a specific date (expiration of an option), every day, week, month, year, etc. The stages do not necessarily correspond to time periods of an equal length. Typically, the first stage relates to a relatively short time period whereas the last one may cover several years or it may be the end-effect period designed to reduce the effects of approximating an infinite horizon model by the model with finite horizon T, e.g. [9].

For initial screening studies the degree of aggregation of possible future outcomes, which results in a number of branches from the individual nodes, is quite high. It can be verbally described as distinction of "high" or "low", "up" or "down" for branching into two descending nodes, "high", "medium", "low" or "dry", "medium", "wet" for branching into three descending nodes. Another strategy is to use an extensive branching from the root leading to relatively many, say 10, nodes on level 1, use a modest branching from the nodes at the middle of the tree and a relatively poor branching, e.g., into two descendants for the nodes at the last levels of the tree. There are some hints concerning the minimal number of descenadants which come from *problem specific requirements*, such as the necessity to build a model without arbitrage opportunities or to fit some moments of the probability distribution.

Using the tree independent procedures for generating scenarios requires additional steps to build a scenario tree of a prescribed structure. This has been done often by ad hoc crude methods, by cutting and pasting the data paths in a more or less intuitive way. The next possibility mentioned also in [2] and applied in [8] is the *cluster analysis*. It is relatively easy to cluster according to the first component (or subvector) ω_1 of ω and to continue by conditional clustering according to the second components (subvectors) ω_2 of the objects included into the created clusters, etc. To treat properly the interstage dependences, consider instead a *multi-level clustering scheme* which exploits the whole sequences of observed/simulated data $\{\omega_1, \dots, \omega_T\}$:

- For each pair ω^i, ω^j evaluate a suitable dissimilarity measure, e.g.,

$$d_1(\omega^i, \omega^j) = \frac{1}{T} \sum_{t=1}^{T} w_t |\omega_t^i - \omega_t^j|$$

$$d_2(\omega^i, \omega^j) = \frac{1}{T} \sum_{t=1}^{T} w_t (\omega_t^i - \omega_t^j)^2,$$

  where $w_t \geq 0$ are suitably chosen nonincreasing weights. This allows us to put more weight on the differences in the beginning of the sequence.

- Measures of dissimilarity among the compared objects are used in definitions of the standard measures of dissimilarity of clusters and used subsequently in the cluster analysis approaches; see e.g. [39]. The result is K_1 clusters, $C_1^1, \dots, C_1^{K_1}$ represented by $\tilde{\omega}_1^k, k = 1, \dots, K_1$; these can be the mean values or modal values of the first components ω_1 of the observations of ω included into the cluster. Probabilities of $\tilde{\omega}_1^k, k = 1, \dots, K_1$, equal the sum of probabilities of the individual ω^i's belonging to the relevant cluster.

- The clustering procedure continues for each cluster C_1^k separately, starting with the second component ω_2 of the observations included into C_1^k, or equivalently, with the first component ω_1 replaced by $\tilde{\omega}_1^k$ and so on. The desired structure of the scenario tree is taken into account.

For data paths obtained by sampling from a continuous probability distribution, existence of two or more predecessors of a node is small. The arc probabilities are fixed according to the assumption of equiprobable initial observations. A possibility to generate the best approximating scenario tree of a given structure via stochastic approximation technique is suggested in [61].

A *Markov structure of data* can be exploited for conditional generation of scenarios in a way which takes into account the already created structure of the tree; cf. [14], [41]. In this case, ω_t depends only on the preceding component ω_{t-1} and on an additional random variable ε_t which is independent of the history $\omega^{t-1,\bullet}$:

$$\omega_t = \boldsymbol{P}\omega_{t-1} + \varepsilon_t. \tag{15}$$

Here the transition matrix $\boldsymbol{P}$ can depend on t. Interstage independence can be regarded as a special form of the Markov structure (15) with $\boldsymbol{P}$ the zero matrix.

The Markov property (15) allows for a direct sampling from the distribution of ε_t at each node which corresponds to an already obtained realization of ω_{t-1}; the arc probabilities of all descendants of the given node are equal; this appears, e.g., in [19]. Another possibility is to discretize the distribution of ε_t at a given number of points, and add the obtained realizations to the already known past values of $\boldsymbol{P}\omega_{t-1}$. The arc probabilities are fixed according to the used discretization method. This approach was used already in [6], [7].

The recent technique of the *sequential importance sampling* by [10], [11], [12], [18], [21], and others, elaborates further methods based on the Markov structure. It takes into account a given suitably labeled tree structure already in the course of simulation. For details see Subsection 3.1.

Independently repeated procedures for building a scenario tree of a given structure by sampling from the assumed continuous probability distribution P allow for the construction of a stochastic lower bound on the optimal value $\varphi(P)$ of the true underlying stochastic program. Let P^n be one possible discrete probability distribution based on n scenarios sampled from P and $\varphi(P^n)$ the optimal value of the stochastic program based on the corresponding scenario tree. Then a slightly extended demonstration valid for two-stage stochastic programs, cf. [51], implies

$$E\varphi(P^n) \le \varphi(P).$$

The Central Limit Theorem applied to a large number i.i.d. replicas of $\varphi(P^n)$ provides an asymptotic confidence interval for the expectation $E\varphi(P^n)$ whose lower bound gives a lower bound for the true optimal value $\varphi(P)$ at the chosen confidence level. Just as for the contamination technique, this result does not require any special assumptions, such as convexity or saddle property with respect to ω.

3.1 Sequential importance sampling-based scenario generation

Based on (8)–(9), we introduce an importance sampling technique which can be used to construct a set of *objective*-relevant realizations of the vector data process $\omega = \{\omega_t\}_{t=1}^T$

in $(\Omega, \mathcal{F}, P)$. Whenever a sampling technique is adopted for the iterative refinement of the discrete representation of a continuous process, the scenario generator needs to be integrated in a more general framework, as discussed in this section.

The framework is based on the definition of a *scenario tree nodal partition matrix* $\boldsymbol{M} = \{m(i,t)\}$ that uniquely identifies the structure of the associated scenario tree. The matrix, with the number of rows equal to the number of scenarios and the number of columns equal to the number of stages, provides the necessary labelling scheme for the specification of the index sets k_t, $t = 1, \ldots T$, introduced in (8)–(9). In the iterative procedure, the matrix is an input to the *conditional* scenario generator, and an output from the sampling algorithm. A balanced $3^1 2^1 1^1$ tree structure would be described in the matrix as

$$\boldsymbol{M} = \begin{pmatrix} 1 & 2 & 5 & 11 \\ 1 & 2 & 6 & 12 \\ 1 & 3 & 7 & 13 \\ 1 & 3 & 8 & 14 \\ 1 & 4 & 9 & 15 \\ 1 & 4 & 10 & 16 \end{pmatrix}$$

where, row-wise, every scenario is explicitly labelled. The matrix can clearly accommodate any tree configuration in a straight forward way. Parental relations are easily identified in the tree.

Algorithm for sequential sampling

Independently of the chosen importance sampling criterion, the sampling procedure is itself part of a general framework whose driving components are as follows:

- A scenario generator, defined as a general method mapping the current knowledge of the underlying uncertainty into a finite set of future possible realizations, represented by a tree.

- A stochastic program generator, whose inputs are the mathematical formulation of a decision problem and a scenario tree, and whose output is a stochastic programming problem in standard format (SMPS files, see [4] or [37]).

- A solver, possibly interfaced with a sampling algorithm, whose input is the stochastic program in SMPS format and whose outputs are the problem solution and the current estimate or the current value of the sampling criterion.

The sequential sampling algorithm is designed as an iterative procedure. It requires the specification of the number of stages, T, the maximum number of possible iterations, a stopping criterion and the initial scenario tree structure together with the associated nodal partition matrix $\boldsymbol{M}$. At every iteration, the following steps are performed:

- The conditional scenario generator produces selective sample paths of the data process;

- The `core,` `time` and `stoch` files of the corresponding stochastic programming problem are generated in the SMPS format;

- Some version of the stochastic program, such as (8)–(9), is specified and solved;
- Along the tree, the nodal values of the importance sampling criterion are evaluated;
- A new tree structure is defined through an update of the nodal partition matrix.

Various sampling rules can be defined and the procedure can also be improved by modifying the direction of movement along the tree at every iteration. For instance, we either move forward a predefined number of stages or we resample at the current stage. As the algorithm progresses, the scenario generator defines, stage by stage, forward trajectories by resampling the state space of the process. Selective backward and forward resampling can also be imposed, and the scenario generator will always run consistently due to the update of the matrix $\boldsymbol{M}$.

The state space of the random process will in this way be iteratively enriched with a varying number of new trajectories starting in the nodes selected by the importance sampling criterion.

The scenario generation procedure is general, and independent of both the mathematical characterization of the random data process and the adopted sampling criterion. Within the sequential procedure, the expansion of the tree structure can be governed by different rules entering the refinement procedure at different stages.

Scenario generation procedures fitting into our general framework fall into two classes. In one class, the tree is built iteratively based on information from the solution of sample problems. In the other class, the focus is on consistent *a-priori* characterization of the data process. In the first class we find [11], [14] and [18], in the second for example [33] and [55]. The two approaches can be combined.

Data path generation

In previous works (see [10], [11]) we have discussed a possible two-step formulation of scenario generation. In many applications, we need to separate the definition of the random process generated by the coefficients of the problem, i.e., $\omega_{t-1} = \{\boldsymbol{c}_t, \boldsymbol{A}_t, \boldsymbol{B}_t, \boldsymbol{b}_t\}$, $t = 2, \ldots, T$, in (8)–(9), from the random elements that can be regarded as the driving factors of the problem uncertainty. The distinction between a *coefficient process* and a *random data process* is therefore introduced. In general, the former process is defined as a deterministic function of the latter. Unlike the coefficient process, the data process can be modeled according to theoretical and empirical assumptions, and does not need to satisfy the time partition imposed by the stages of the stochastic programming formulation. The coefficient process is by construction a discrete time path-dependent process, consistent with the recourse formulation of the problem, while the characterization of the data process refers to a conceptual underlying continuous time process, say, θ_t, $t \in [1, T]$.

In summary, a scenario generator for a stochastic programming problem (see [19], [53]) can be characterized depending on:

- The planning horizon of the decision problem and the time partition induced by the problem stages and recourse decisions;
- The state transitions of the coefficient process with the associated probability measure;

- The dependency of the coefficient process on the underlying random process and the adopted model for this process.

Industrial project evaluation, as well as asset and liability performance measures, will often include as random element an interest rate process. Long term pension fund and insurance models have in addition to take into account an inflation process. These are examples of random processes—often cross-correlated—that try to capture the core uncertainty of the problem, leading to the definition of the data process. They provide the explanatory variables for the definition of a possibly very large set of coefficients in the mathematical model.

Many recent applications in the fields of risk management and asset allocation have highlighted the need of capturing *extreme events*. The recent crisis in parts of the emerging markets and its impact on financial markets worldwide, have pushed forward the research in this direction. The contamination method described in Section 2 provides a convenient formulation for the inclusion of *catastrophic* events in the scenario set.

At present, we can treat many different types of models for the data process. In particular:

- Random walk models - with or without drift - with fine time discretization and Gaussian noise, adopted in Monte Carlo generators.

- Binomial or trinomial models with a coarser time partition, and characterized by independent increments and theoretical properties for asymptotic convergence towards continuous processes, typical for short term decision problems without continuous control.

- Autoregressive models with random volatility patterns associated with long term planning problems and transitions which in most cases coincide with the original problem stages.

In a multistage setting, the generator of the data process is embedded in the coefficient process generator, and determines the state transition from one node to the next along the scenario tree. From here!

Let us now go back to the sequential framework, and describe in more detail the steps associated with the update of the scenario tree within the algorithm for sequential sampling, see Table 1.

Step 4 uses a predefined discretization of stages into N_t subperiods, $t = 1, \ldots, T$. Step 6 can involve various possibilities, from the call of an extended set of linked subroutines to a simple simulation of a driftless random walk. The actual coefficients for the mathematical model can then be computed from this underlying process. The operator Γ in step 7 typically performs a deterministic nonlinear transformation, mapping a set of realizations of the data process into the current state of the system. More specifically, it describes the set of computations necessary to derive from the generated sample paths of the data process the appropriate set of random coefficients of the model.

Step 8 is needed before generating the numerical solution of the stochastic program in order to allow at every iteration of the sequential procedure a consistent recovery of the history of the process as discussed previously.

Table 1: Scenario tree generation and sequential update

```
read initial nodal partition matrix $\boldsymbol{M}(1)$;
define number of major iterations $J$;
for $j = 1, \ldots, J$   {
    read current sampling algorithm iteration $j$;
    read current number of scenarios $I(j)$;
    set  the stage dependent initial conditions for the simulator;
         if $j = 1$ then $\tau := 1$;
         1. read nodal partition matrix $\boldsymbol{M}(j)$;
         2. read current stage $\tau$;
         3. load $\omega_t, t \leq \tau$;
         4. for $t := \tau + 1, \ldots, T$   {
             5.  for $i = 1, \ldots, I(j)${
                      if $i = 1$ or $m(i,t) \neq m(i-1,t)$ then
                           for $n_t := 1, \ldots, N_t$   {
                               6. run data generator from $m(i,t-1)$ to $m(i,t)$
                                  to obtain $\theta_{n_t}$;
                                  }
                          7. compute stage-dependent coefficients
                              $\omega_t = \Gamma(\theta_{n_t}, n_t = 1, \ldots, N_t)$ according to $\boldsymbol{M}(j)$;
                     }
             }
         8. update process and initial conditions;
         $\tau := \tau + 1$;
}
```

At the beginning of every iteration the current nodal partition matrix defines the index time set for the current iteration, allowing the conditional specification of the coefficient process, and determines a new partition of the state space of the process suggested by the information collected at the end of the previous iteration. To here!

EVPI as importance sampling criterion

Following the *dynamic programming* formulation in (4), which exploits the Markov structure of the decision problem, one can define (cf. [17], [11]) the expected value of perfect information — $\eta_t(\omega^{t,\bullet})$, for $t = 1, \ldots, T$ — as the difference between the optimal value of the stochastic programming problem

$$\pi_t(\omega^{t,\bullet}) = \min_{\boldsymbol{x}_t \in \mathcal{X}_t} E\left\{ f_t(\omega^{t,\bullet}, \boldsymbol{x}^{t-1,\bullet}, \boldsymbol{x}_t) + \varphi_{t+1}(\omega^{t,\bullet}, \boldsymbol{x}^{t,\bullet}) | \mathcal{F}_t \right\} \tag{16}$$

and the associated expected value of wait-and-see problems (assuming *perfect foresight*)

$$\phi_t(\omega^{t,\bullet}) = E[\min_{\boldsymbol{x}_t \in \mathcal{X}_t} \left\{ f_t(\omega^{t,\bullet}, \boldsymbol{x}^{t-1,\bullet}, \boldsymbol{x}_t) + \varphi_{t+1}(\omega^{t,\bullet}, \boldsymbol{x}^{t,\bullet}) \right\} |\mathcal{F}_T]. \tag{17}$$

In (16) and (17), φ_{t+1} expresses the optimal expected value for the remaining stages from $t+1$ to T. The function is evidently nonnegative and at the horizon, it is zero by definition. The dependency of the problem solution on the available information is made explicit in the given formulations by conditioning the expectation operator at stage t on the σ-field $\mathcal{F}_t$ in program (16) and on $\mathcal{F}_T$ in (17). The real-valued processes (16) and (17) are defined at every node of the scenario tree. The set, for $t = 1, \ldots, T$, of nodal EVPI estimates is generated by relaxing *only* the measurability condition with respect to the current σ-field in the original stochastic program.

The EVPI sampling algorithm has been developed as an interface with the MSLiP-OSL solver [21] instantiating the nested Benders decomposition algorithm [1], [35]. It is there used as an importance sampling criterion.

The method has been designed and tested for a dynamic investment problem initially formulated as a stochastic control problem, cf. [12]. It is a 12 stage model which covers a period of 20 years. The scenario generator is based on conditional simulations of a 4-dimensional vector diffusion process which includes dividend rates, stock returns, a short rate and a long rate processes. The associated computational results confirmed, inter alia, the iterative increase of the number of selected sample paths, the increasing root-EVPI value, and the convergence of the first-stage optimal solutions to a stationary state.

In the EVPI-sampling method, the tree structure representing the model uncertainty is updated at every major iteration of the algorithm, and additional sample paths are selected or previous realizations deleted, depending on the nodal EVPI estimates at the current iteration. At termination, particularly in the case of long-term decision problems with many stages, a complex tree structure is typically generated. Ideally, the optimal *here-and-now* solution becomes independent of possible additional scenarios and so does the root-EVPI value, typically adopted as a variable to be monitored for termination.

Associated with the last set of generated scenarios and with the solution of the corresponding stochastic program, a sequence of recourse decisions along every scenario is obtained. This helps to model and detect the extremal events and to study their impact on the obtained first-stage solutions by imposing increasing probablity weights on the scenarios which incorporate these rare events; see the contamination method in Section 2.

The procedure (cf. [11],[18]) results in general in an irregular expansion of the scenario tree. The efficiency of the method depends on the adopted sampling rule, its consistency is enforced at each iteration by updating the nodal partition matrix.

In Subsection 3.3, a method is described which, unlike the one previously considered, does not require an analytical specification of the data model and can be applied to stochastic problems when a poor information structure is assumed.

3.2 Problem oriented requirements

When building the scenario tree one tries to avoid as much as possible any distortion of the available input information. Moreover, the goal of this procedure does not reduce to an approximation of the probability distribution P but rather the goal is to create an input which provides good solutions to the underlying problem. This means, inter alia, that problem oriented requirements should be respected. The motivation comes from various problem areas.

- Scenarios based solely on past observations may ignore possible time trends or exogenous knowledge or expectations of the user; see for instance Mulvey [57] or [47].
- Selection of scenarios should respect the no-arbitrage properties; cf. [23], [45], [46]. This means that scenario-based estimates of future asset prices in a portfolio optimization model should not allow arbitrage opportunities.
- The scenarios coming from historical data need not be directly applicable; [62] suggests a scenario tuning procedure to discover the information hidden in indirect measurement results contained in past records on specific metal melting processes.

On one hand, finding a discrete approximation (a scenario tree) of a continuous distribution (or from sampled data) is statistics. On the other hand, we must remember the purpose of the process, namely to create input to a decision model. The discrete approximation is not in itself a goal. A simple discussion to this effect, but still very useful, can be found in [44]. It is shown how some popular, and from many points very good, three-point approximations of one-dimensional distributions can lead to very bad estimates of entities that depend on the distributions. His example relates to certainty equivalents of games. In the well-known mean-variance model of Markowitz [52], it is known that only the first two moments (including the cross moments) matter for the solution. Therefore, if we choose to solve the Markowitz model with discrete distributions (which is not necessary), all distributions with the same first two moments will produce the same solution. Matching higher order moments will have no effect on the model. In this case, we see that it is the model that determines what properties from the distribution that are important.

Hence, our trees must on one hand represent the underlying distribution, on the other hand be such that the model produces good (first-stage) decisions. We must realize that the scenario trees are *parts of the models, not the data* (the only exception is when the tree represents the full truth of the stochastic process, such as rolling a dice twice).

Explicitly formulated additional requirements concerning properties of the probability distribution can help. The statistical properties can be made concrete through a suitable massaging of the data to obtain the *prescribed moments values*, given a fixed tree structure. This idea has appeared for instance in [9] where at the given stage of a multistage stochastic program, the observed data were grouped and scaled to retain the prescribed values of expectations and variances. One of the reasons was the sought possibility of comparisons with the Markowitz mean-variance model. Another example is in [57] where the expectations are kept fixed at the values declared by the user or [59] where the authors fit their recombining tree so that the first and second moments equal to those used in the continuous (calibrated) model. We shall follow [40] who suggest to build the scenario tree in such a way that some statistical properties of the data process are retained, for instance, there are specified expectations, correlation matrices (also over time) and skewness of the marginal distributions of ω_t. Some of these properties may depend on earlier realizations, representing such as volatility clumping and reversion of the mean. Also the methodology allows for inconsistent data. This will be discussed below.

Why match moments? The question of a possible representation of probability distributions by (infinite) sequences of moments, and approximating them using only a few moments goes back to Tchebycheff and is connected with the moment problem. Moreover, it is possible to prove that given m admissible values of moments, there exists a *discrete* probability distribution with these moments and its support has at most $m+2$ points. We refer to Chapter 5 of [63] for a brief introduction and selected results in this direction. For our purposes it means that given values of certain moments or expectations of continuous functions, say, $\mu_k = \int_\Omega u_k(\omega)dP$, $k = 1,\dots,m$, there exists a modest number of scenarios ω^s, $s = 1,\dots,S$, and their probabilities p_s, $s = 1,\dots,S$, $\sum_s p_s = 1$ so that the moment values are retained, i.e.,

$$\sum_s p_s u_k(\omega^s) = \mu_k,\ 1,\dots,m. \tag{18}$$

To get the scenarios and their probabilities means to find a solution ω^s and p_s, $s = 1,\dots,S$, of the system (18) extended for nonnegativity conditions on probabilities and for the additional constraint $\sum_s p_s = 1$. This is a hihgly nonlinear numerical problem.

Most of the methodology mentioned above assumes that the distribution we are trying to approximate exists, be that because it is given explicitly, or because we are obtaining the data (samples) from one source, for example a simulator, guaranteeing that the data is automatically consistent. But in reality, data often comes from different sources, collected by different people for different reasons, often over different time periods, and we must therefore expect the given specifications to be inconsistent. For example, (18) may not have a solution. We are then faced with three options, to give up, change the data, or try to find a scenario tree which fits as well as possible (but not perfectly) to the data and the given requirements. Giving up is rarely a useful alternative, and changing the data is not sound practice, hence, in our view, the correct approach is to take the data as they are, and continue from there.

3.3 Matching more complex distributions

The system of equations (18) can be further extended for other constraints on the selection of scenarios to represent certain strata, to cover extremal cases, etc. As an example, let us assume that $\omega = (\omega_1, \omega_2)$ is a two-dimensional random vector with the first three moments $\mu_k(1), \mu_k(2)$, $k = 1,2,3$, of the marginal distributions and with the covariance

ρ estimated from the true probability distribution, and we require in addition that in at least one scenario, $\omega_1 \geq l_1, \omega_2 \geq l_2$ to capture a very difficult but unlikely event. Let the discrete two-dimensional distribution which matches the given requirements be carried by S atoms $\omega^s = (\omega_1^s, \omega_2^s), s = 1, \ldots, S$, with probabilities $p_s \geq 0 \, \forall s$, $\sum_s p_s = 1$. Hence, we search values of pairs $(\omega_1^s, \omega_2^s), s = 1, \ldots, S$, and scalars $p_s, s = 1, \ldots, S$, such that

$$\sum_{s=1}^{S} p_s = 1,$$

$$\sum_{s=1}^{S} p_s (\omega_1^s)^k = \mu_k(1) \text{ for } k = 1, 2, 3,$$

$$\sum_{s=1}^{S} p_s (\omega_2^s)^k = \mu_k(2) \text{ for } k = 1, 2, 3,$$

$$\sum_{s=1}^{S} p_s (\omega_1^s - \mu_1(1))(\omega_2^s - \mu_1(2)) = \rho,$$

$$\omega_1^1 \geq l_1, \omega_2^1 \geq l_2,$$

$$p_s \geq 0, \; s = 1, \ldots, S.$$

Similar models can of course be set up for more complex situations. Such a formulation is useful only if we know that the system of equations and inequalities has a solution. If the data is inconsistent or S is too small, we need a more flexible approach, such as the one in [40]. We suggest to use a goal programming version of the same problem, so that an almost feasible solution ω^s, and $p_s, s = 1, \ldots, S$, can be obtained for instance by solving a nonconvex weighted least squares minimization problem, i.e.

$$\min \sum_{k=1}^{3} \alpha_k \left(\sum_{s=1}^{S} p_s (\omega_1^s)^k - \mu_k(1) \right)^2 + \sum_{k=1}^{3} \beta_k \left(\sum_{s=1}^{S} p_s (\omega_2^s)^k - \mu_k(2) \right)^2$$
$$+ \gamma \left(\sum_{s=1}^{S} p_s (\omega_1^s - \mu_1(1))(\omega_2^s - \mu_1(2)) - \rho \right)^2$$

subject to

$$\omega_1^1 \geq l_1, \omega_2^1 \geq l_2,$$

$$p_s \geq 0, \; \text{for } s = 1, \ldots, S.$$

The advantage of this formulation is that the optimal value is zero if the data is consistent (and S is large enough), but that the optimal solution is also a good representation of the data in the case of inconsistency. Hence, also in that case we will get a usefull discrete distribution. The parameters α, β and γ are important only in the case of inconsistency, and can be used to reflect importance and/or quality of data.

Inconsistency can appear if the information about moments comes from different sources, if implicit specifications are inconsistent with explicit ones, etc. Consider for instance a problem which covers two periods. Let us specify the variance of ω_1 and the variance of the sum $\omega_1 + \omega_2$. This is reasonable as many users have difficulties providing conditional statements about second period variances unless these are equal for all periods. By specifying these two variances, we have said something about the correlation over time. If we now explicitly specify correlations over time, we are likely to end up with two inconsistent specifications of the same entity.

From an optimization point of view, the problem is non-convex and has many local optima. A simple heuristic is discussed in [40], and this area is open for more research. There is numerical evidence in favor of performance of stochastic programs based on scenario trees with moment values fitted at each node over those based only on a few randomly sampled realizations, cf. [48] or [49]. We see it as central to be able to fit also higher order moments.

Numerical Example

Let us consider the generation of a tree for a portfolio management model in the energy sector, outlined in [32]. There is uncertainty in spot market prices, inflow going directly into the reservoir, and inflow going directly to the power station. We employ a four period (five stage) model with a total of 256 scenarios. The first period has a length of one week, the second has a length of four weeks, then 16 weeks and finally 64 weeks.

Both prices and inflows are nonstationary time series, for example, inflows are likely to be low in the winter and high in early summer. In order to keep the example simple, we do not consider these effects here.

The basis for generating the scenarios is

- user supplied statistical moments for the first period marginal distributions of all random variables,
- correlations between the variables,
- definition of the state dependent statistical properties and
- bounds on outcomes and probabilities.

A market equilibrium model frequently used for price forecasting in Scandinavia is the Multiarea Power Scheduling (MPS) model developed by the Norwegian Electric Power Research Institute (EFI) (now a part of SINTEF Energy Research), and is described in [29] and [5]. In the MPS, process models describe production, transmission and consumption activities within the Nordic and adjacent areas. The various demand/supply regions are connected through the electrical transmission network. A solution of the model results in a set of equilibrium prices and production quantities, for each week over the time horizon considered (usually 3 years) and for each historical inflow year. The demand side of the model consists of price dependent and price independent loads for each region. Important input for the model are demand, thermal generation costs, and initial reservoir levels. The model is short term in the sense that there are no mechanisms for endogenously increasing production capacity. The MPS recognizes that hydro scheduling decisions are made under the uncertainty of reservoir inflows; to determine the opportunity hydro generation costs production in each region, stochastic dynamic programming is employed on the scheduling problem where production in the region is aggregated into an equivalent reservoir/power station pair.

Table 2: Specifications of market expectations.

Uncertain variable	Distribution property	Period 1	Period 2-4
Spot market price	expected value NOK/MWh	180.0	State dep
	standard deviation	70.0	State dep
	skewness	1.23	1.23
	kurtosis	4.04	4.04
Reservoir inflow	expected value GWh	270.0	State dep
	standard deviation	200.0	State dep
	skewness	1.43	1.43
	kurtosis	1.71	1.71
Station inflow	expected value GWh	90.0	State dep
	standard deviation	70.0	State dep
	skewness	2.76	2.76
	kurtosis	8.17	8.17

Table 3: Specification of correlation.

	Period			
Correlation	1	2	3	4
Price–Reservoir inflow	-0.34	-0.62	-0.78	-0.87
Price–Station inflow	-0.36	-0.63	-0.79	-0.88
Reservoir inflow–Station inflow	0.35	0.62	0.78	0.87

The MPS model generates independent scenarios for price and inflows. However, this structure is not appropriate for multistage programming. What is needed is a *scenario tree* where information is revealed gradually, and not only after the first stage. In the following, we assume that the first four moments and the correlations are the relevant statistical properties. The specifications are given in Tables 2 and 3.

In this example we have modeled state dependent expected values and standard deviations for all uncertain variables. The other statistical properties are assumed state independent, meaning that they are the same in all states of the world at a certain point in time.

The state dependent mean in period $t > 1$ is

$$E(\omega_{it}) = [MR_i E(\omega_{i,t-1}) + (1 - MR_i)\omega_{i,t-1}]\, RPL\_E_{it} \tag{19}$$

where $E(\omega_{it})$ is the expected outcome of random variable i in period t, ω_{it} is the outcome of random variable i in period t, $MR_i \in [0, 1]$ is the mean reversion factor (a high MR_i leads to a large degree of mean reversion), and RPL_E_{it} is a scaling factor depending on the period length relative to the previous period and on the type of random variable i. We let the mean reversion factor, $MR_i = 0.2$ for price and $= 0.3$ for inflow.

Table 4: Adjustment in mean and standard deviation due to nonuniform period length. The numbers for standard deviation are based on analyzing autocorrelation in the time series.

		Period		
Uncertain variable	Parameter	2	3	4
Spot market price	RPL_E	1.0	1.0	1.0
	RPL_SD	0.96	0.74	0.54
Reservoir inflow	RPL_E	4.0	4.0	4.0
	RPL_SD	3.69	2.88	2.16
Station inflow	RPL_E	4.0	4.0	4.0
	RPL_SD	3.77	3.34	2.32

Table 5: Specification of bounds. They are assumed constant for all periods.

Uncertain variable	Relative upper bound	Relative lower bound	Absolute lower bound	
Price	9.00	4.50	25.00	NOK/MWh
Reservoir inflow	5.00	3.25	29.45	GWh
Station inflow	8.00	3.25	2.00	GWh
Probability (cond.)	-	-	0.01	

For standard deviation we assume that the state dependency is given by

$$SD(\omega_{it}) = [VC_i|\omega_{i,t-1} - E(\omega_{i,t-1})| + (1 - VC_i)SD_{BAS}(\omega_{it})]\, RPL\_SD_{it} \tag{20}$$

where $VC_i \in [0, 1]$ is the variance clumping parameter for random variable i (a large VC_i means that variance in the next period is highly influenced by variance in the previous period), and $SD_{BAS}(\omega_{it})$ is the average standard deviation of the outcome of random variable i in period t. We let the volatility clumping parameter, $VC_i = 0.15$ for all variables. The scaling factor RPL_SD_{it} is needed due to the nonuniform period lengths. In Table 3.3 these scaling factors are listed.

There are two types of bounds implemented here; relative and absolute. Relative bounds means that the bounds are expressed in terms of minimum or maximum number of (state dependent) standard deviations from the (state dependent) mean. An absolute bound is expressed in terms of the outcome or the probability. Table 5 lists these bounds.

These requirements are used to produce a four-period tree (which does not have a perfect fit due to inconsistencies). The first three periods are shown in Figure 1.

The data in this example were inconsistent. In such a setting it is very easy to realize that the construction of the scenario tree is part of the modeling, and not part of the data. The scenario tree is the user's view on how the data most well can be represented to give a good model. Hence, when models are to be validated, a subject not discussed