Step decision rules for multistage stochastic programming: a heuristic approach

J. Thénié [*] J.-Ph.Vial[*]

September 27, 2006

Abstract

Stochastic programming with step decision rules, SPSDR, is an attempt to overcome the curse of computational complexity of multistage stochastic programming problems. SPSDR combines several techniques. The first idea is to work with independent experts. Each expert is confronted with a sample of scenarios drawn at random from the original stochastic process. The second idea is to have each expert work with step decision rules. The optimal decision rules of the individual experts are then averaged to form the final decision rule. The final solution is tested on a very large sample of scenarios. SPSDR is then tested against two alternative methods: regular stochastic programming on a problem with 3 stages and 2 recourses; robust optimization with affinely adjustable recourses on a 12-stage model. The performance of the new method turns out to be competitive on those examples, while it permits a tighter control on computational complexity than standard stochastic programming.

Acknowledgements The authors acknowledge the support of Fonds National Suisse de la Recherche Scientifique, grant # 100012-105309

1 Introduction

Many real-life problems are concerned with sequential decision making under uncertainty. Unfortunately, those problems are difficult. The authors of [22] "...argue that multi-stage problems, even linear [...] with complete recourse, generically are computationally intractable already when medium-accuracy solutions are sought." Indeed, the complexity of computing expectations with respect to a multidimensional stochastic process in discrete time grows exponentially with the dimensionality of the process and the number of stages. The same authors add the reassuring statement: "Of course, this does not mean that some specific cases of multi-stage stochastic programming problems cannot be solved efficiently. Note that this claim is rather a belief than a statement which we can

[*]HEC/Logilab/Department of Management Studies, University of Geneva, 40 Bd du pont d'Arve, CH-1211 Geneva 4, Switzerland.

rigorously prove." Other authors [9] argue that multistage stochastic programming is at least as difficult as hard combinatorial problems. In view of the practical importance of the problems to solve, it is not absurd to look for computationally tractable heuristics. Because a heuristic approach does not produce solutions that are provably optimal, one has to resort to empirical analysis to test the performance of the heuristic.

Stochastic programming with step decision rules (SPSDR) is one such heuristic. It aims to find efficient contingent strategies for convex stochastic control problems, with linear dynamic constraints and unconstrained state variables. The SPSDR scheme is as follows. At each stage t, one constructs a partition of the set of scenarios based on past history. The decision rule consists in assigning to a decision variable the same value for all scenarios belonging to the same element of the partition. In general, one associate with a scenario at stage t a multi-dimensional random variable representing past history. In this setting, a decision rule as described above can be viewed as a step function on the space of outcomes of the variable. Hence the name, *step decision rule*, SDR in short. A decision rule based on the sole past history meets the non-anticipativity requirement, even though the partition at stage $t+1$ need not be a refinement of stage t. This property gives more freedom in the design of the partitions. In particular, the number of contingent variables in successive stages is not bound to grow in a multiplicative manner as it does on the standard SP approach. This gives a way to get around the issue of exponential growth of the event tree.

The main feature of SPSDR is that it applies to any given discrete realizations, or scenarios, of the stochastic process. No preprocessing such as aggregation and scenario reduction [15] is required. SPSDR works with the original stochastic information, but replaces the original decision process by a more restrictive one. Thus, instead of approximating the stochastic process by a computationally tractable event tree as in [15], we restrict the original decision process to a sub-optimal but manageable one.

The choice of efficient partitions is a critical issue. The first concern is the number ν_t of elements in the partition at stage t. Since a decision rule stipulates that the same decision is to be applied at all sub-scenarios that hit the same element of the partition, one sees that the partition reflects the knowledge or ignorance at t in the decision process induced by the decision rule. The minimal value $\nu_t = 1$ for all t means total ignorance: the decision to be taken at t is unique, independent of the stochastic process realizations. The problem of finding a good decision rule becomes simple, but the solution is likely to be inefficient. The maximal value $\nu_t = N$ for all t, where N is the total number of scenarios means perfect foresight: the decision to be taken at stage t is fully adapted to the realizations. In that case, the problem of finding a good decision rule has N times the size of the previous one. Intermediary values for ν_t reflect partial knowledge, but we do not require that the partitions be nested. In other words, we allow, in our search of decision rules, forgetfulness of the past. The intermediary case compromises between perfect foresight versus total ignorance and small size versus large size for the decision rule selection problem.

The second concern is to achieve a meaningful partition at each t when the number of elements ν_t is given. To do this, we need a measure of quality of a partition. We proceed as follows. We first associate with each element of the partition a realization (or sub-scenario) of the stochastic process. We call this sub-scenario the reference sub-scenario for the element of the partition and we attach to it the probability that a scenario hits this elemental set. This defines a reduced stochastic process. We then introduce a distance between the reduced process and the process associated with the full set of sub-scenarios (ending at t). Following [19] and [15], we choose the Kantorovitch functional to measure this distance. The goal is to find a partition that minimizes this distance; this problem is equivalent to solving a p-median problem.

We have no theoretical argument to select the sequence ν_t: we resort to empirical testing. The efficiency of a decision rule is measured by its performance on the full set of original scenarios instead of the small sampled subset that was used to compute the rule. At this point, we must give a word of explanation on the way we extend a decision rule computed on a sub-sample of scenarios to the entire set of scenarios. This is done by extending the partition of the sub-sample to a partition of the original set of scenarios. Each scenario in the original set is assigned to the closest reference sub-scenario in the sense of the chosen distance measure.

The concept of *linear decision rules*, in short LDR, appeared in the early '60 in the context of optimal stochastic control. It has been developed to solve production planning problems [16] with quadratic objectives. LDR, have been used long ago in the more general context of stochastic programming and chance-constrained programming [12], but somehow neglected since. They were recently resurrected in [5] in the framework of Robust Optimization under the name of Affinely Adjustable Robust counterpart (AARC). AARC offers a computationally tractable alternative to multi-stage problems with uncertain data. Successful applications of this approach to multistage inventory management problems is reported in [1, 4, 6]. These works stimulated investigation of LDR in stochastic programming [22]. With LDR the true decision variables are the coefficients of the affine functions. They have to be fixed in the initial stage. In that respect, stochastic programming with LDR (SPLDR) transforms a multistage SP in an Act-and-See problem, i.e., into a two-stage problem with no recourse. If the initial problem is linear, the SPLDR restriction is also linear and tractable. Unfortunately, it does not seem possible to assess the loss of optimality. The authors of [22] justify the use of LDR with the following heuristic argument: "The rationale behind restricting affine decision rules is the belief that in actual applications it is better to pose a modest and achievable goal rather than an ambitious goal which we do not know how to achieve". The concept of step decision rules seems to be new. As SPLDR, it leads to computationally tractable problems.

It is possible to improve SPSDR by taking convex combinations of independent SDR's. By convexity of the problem, such combination is feasible and expected value of the outcome is smaller (better) than the same convex combination of the outputs of the individual SDR's. It is convenient to view the optimization over a sample of scenarios as the output of an "expert" facing a problem instance associated with a small size sample of realizations of the stochastic process. The average solution can be interpreted as the proposal of a pool of experts. The idea of building several decision rules from independent samples is directly inspired by [18]. A similar use of experts is described in [10, 11] in the framework of machine learning.

Our experiments apply to stochastic control problems in the area of inventory management in the context of supply chain management. Our primary goal is to compare the SPSDR approach to stochastic programming and robust optimization on two problems on which those methods perform well. The first set of experiments compares SPSDR with the standard stochastic programming (SP) approach on the well-known *newsboy problem* with 3 stages. The main conclusion is that SPSDR can produce as good solutions as SP. The second set of experiments deals with the model of *retailer-supplier flexible commitments contracts*, (RSFC). The model has 12 stages; it is not easily amenable to standard SP but AARC solutions are computed in [4]. On this problem, SPSDR is computationally tractable and can accommodate a variety of objective functions, such as min-max, the expected shortfall (or conditional value at risk), the total expected costs or criteria mixing risk measures and expected costs.

The paper is organized as follows. Section 2 gives the general formulation of the stochastic control problem. Section 3 discusses an implementation of standard stochastic programming on this class of problems. Section 4 is devoted to the presentation of stochastic programming with step decision rules. In Section 5, we report the results of our empirical study on two different problems. The last section is a short conclusion.

2 The stochastic control problem

We consider the deterministic control problem with horizon T

$$\begin{aligned}
\min \quad & \phi(x_1, \ldots, x_T, u_1, \ldots, u_{T-1}) && \text{(1a)}\\
& x_t = A_t x_{t-1} + B_t u_{t-1} + C_t, \ t = 2, \ldots T && \text{(1b)}\\
& g_t(u_1, \ldots, u_t) \leq 0, \ t = 1, \ldots T-1 && \text{(1c)}\\
& x_1 = \bar{x}_1. && \text{(1d)}
\end{aligned}$$

In that problem, u_t and x_t are multidimensional real variables, A_t and B_t are matrices of appropriate dimensions and C_t is a vector. The variable u_t is the control and x_t is the state variable. We assume that g_t is convex in $(u_1, \ldots, u_t)$ and ϕ is jointly convex in $(x_1, \ldots, x_T)$ and $(u_1, \ldots, u_{T-1})$, respectively. For the sake of simplicity, we shall use in the sequel the more restrictive formulation with a separable objective

$$\phi(x_1, \ldots, x_T, u_1, \ldots, u_{T-1}) = \sum_{t=1}^{T-1} f_t(x_t, u_t) + f_T(x_T).$$

The problem becomes stochastic when the components of (1) depend on the realization $\{\xi_t\}_{t=1}^T$ of the sequence of random variables $\{\tilde{\xi}_t\}_{t=1}^T$. We make a fundamental assumption on the stochastic process

Assumption 1 *The stochastic process is independent of the decision process.*

The stochastic process $\{\tilde{\xi}_t\}_{t=1}^T$ is endowed with the probability space $(\Xi, \mathcal{S}, \mu)$, where Ξ is the sample space and $\mathcal{S}$ is the σ-algebra of measurable events with respect to μ. If the sample space has finite support, $\mathcal{S}$ is made of all subsets of the finite set of outcomes. It is also convenient to introduce the probability spaces $(\Xi_t, \mathcal{S}_t, \mu_t)_{t=1}^T$ of partial realizations $\sigma_t = (\xi_1, \ldots, \xi_t)$. The event spaces $\mathcal{S}_t$ form a nested sequence of subsets of $\mathcal{S} : \mathcal{S}_1 \subset \mathcal{S}_2 \subset \ldots, \subset \mathcal{S}_T$.

We follow the convention that the selection of the control at t (decision) follows the unfolding of uncertainty at t. We have the sequence

$$[\text{chance move}]_1 \rightarrow [\text{decision}]_1 \rightarrow \cdots \rightarrow [\text{chance move}]_t \rightarrow [\text{decision}]_t \rightarrow \ldots$$

If Ξ_1 is not a singleton, the decision problem breaks down into independent subproblems. Thus we assume that $\Xi_1 = \{\xi_1\}$ is a singleton. The sequential structure of the problem implies that the decision, i.e., control, at t must be made contingent to the history $\sigma_t = \{\xi_1, \ldots, \xi_t\}$. That is, u_t is a function of σ_t, denoted $u_t(\sigma_t)$. In view of the state equation (1b), the state variable x_t must also be made contingent to σ_t. The stochastic version of the control problem (1) is thus

$$\min \quad E[\sum_{t=1}^{T-1} f_t(x_t(\sigma_t), u_t(\sigma_t), \xi_t) + f_T(x_T(\sigma_T), \xi_T)] \tag{2a}$$

$$x_t(\sigma_t) = A_t(\xi_t)x_{t-1}(\sigma_{t-1}) + B_t(\xi_t)u_{t-1}(\sigma_{t-1}) + C_t(\xi_t) \quad \text{a.e.,} \quad t = 2, \ldots T \tag{2b}$$

$$g_t(u_1(\sigma_1), \ldots, u_t(\sigma_t)) \leq 0, \quad \text{a.e,} \; t = 1, \ldots T-1 \tag{2c}$$

$$x_1 = \bar{x}_1. \tag{2d}$$

3 Stochastic programming

So far we only assumed independence of the stochastic process with respect to the decisions. If we further assume finiteness of the sample space, the stochastic control problem can be formulated as a stochastic programming problem and solved as a standard (large scale) mathematical programming problem.

The main problem in sequential decisions under uncertainty is the way uncertainty unfolds over time. If uncertainty unfolds at once at some stage t, the problem essentially becomes a two-stage one and the flavor of sequential decision-making is lost. We are interested in the case of progressive unfolding: the stochastic programming formulation should reflect this fact.

We distinguish different phases in a stochastic programming approach to a control problem. We review them quickly.

Phase 0: Statistical analysis and sampling

In sequential decision making under uncertainty, it is often the case that the underlying stochastic process is best described in continuous time and space whereas the decision process is in discrete time. Multi-stage stochastic programming starts with a statistical pre-processing phase consisting in identifying the stochastic process and estimating its parameters. The next operation consists in sampling scenarios using the distribution of the estimated process. If the underlying process is continuous the scenarios obtained in that manner are almost surely all different as from stage 2 and look like a fan. This representation does not capture the progressive unfolding of uncertainty over time and is not fit, as such, to model the decision process.

Phase 1: Scenario reduction and aggregation

A multi-period decision process associated with a fan of scenarios is essentially a two-stage problem. The usual remedy consists in building a new discrete time/discrete values stochastic process that "best" approximates the fan of scenarios. This new process is represented by a tree, that unfolds with time. If the tree gradually "opens up" with time, as shown in Figure 1, the decision process based on that tree better captures the adaptive nature of the decisions. It is shown in the literature that under suitable assumptions, one can build a balanced tree with the property that the optimal solution of the problem based on that tree provides an estimate of the optimal value of the original problem.

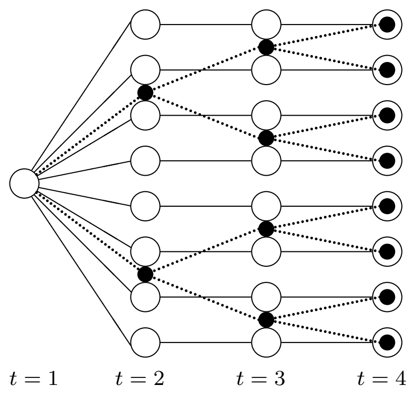

Figure 1: A balanced tree approximating a fan

This phase is described as a scenario reduction and aggregation scheme. We refer to the literature [15] for a detailed description of a scenario reduction and aggregation scheme. To measure the quality of an approximation, some papers use the so-called Kantorovitch functional [19, 15]. We postpone the definition of this distance measure to the next section.

Phase 2: Formulating and solving the deterministic equivalent

In this phase, one adapt the decision process to the balanced tree generated in the previous phase. This operation consists in making the controls, the state variables and the constraints at stage t contingent to each individual stage-t-node in the event tree. This produces a new problem, the so-called *deterministic equivalent*, whose formulation involves many more variables and constraints than the deterministic version. This operation is formal but tedious; it can be made easier by tools like the one described in [23]. The optimal solution of the deterministic equivalent problem provides a prediction of the optimal value of the original problem. It also outputs a solution, consisting of values for the controls at each node of the tree. The solution is thus a stochastic process, with well-defined values on the tree nodes.

Phase 3: Validation

There exist approximation theorems, e.g., [19], that stipulate that under some regularity conditions the optimal expected value of the control problem on the approximation tree tends to the optimal value of the original stochastic control problem. Such results do not say much on how the optimal controls in the deterministic equivalent can be used to determine controls for the original problem and, if this extension can be performed, whether it achieves an objective function value close to the optimum. Those questions are seldom addressed in the literature.

For our study, it is important to implement and test the computed solutions. We proceed as follows. Suppose a scenario is drawn at random according to the probability distribution of the original stochastic process. This scenario is almost surely different from all scenarios of the event tree underlying the deterministic equivalent problem. To find which control which control to apply at stage t, we consider the sub-scenario up to t and look for the closest sub-scenario up to t of the event tree of the deterministic equivalent. The latter determines the control to be applied at t.

It is easy to see that the above operation induces a partition on the set of scenarios at each stage t. This partition can be performed on the full set of scenarios of the original process or on any random sample of such scenarios. An element of the partition is associated with a sub-scenario up to t of the deterministic equivalent tree. We name the sub-scenario on the tree, the reference sub-scenario. The element of the partition includes all scenarios whose sub-scenarios up to t are closer to the reference sub-scenario than to any other sub-scenario on the tree. The extension scheme is pictured on Figure 2.

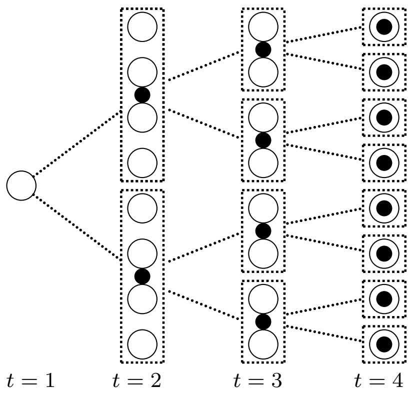

Figure 2: Decision sets to implement the stochastic programming solution

The decision rule consists in applying at any given element of the partition the same control as the one that is prescribed at the reference scenario. Quite naturally, we name each element of the partition a *decision set*.

4 Stochastic programming with a step decision rule

A step decision rule consists in *i*) a sequence of partitions of scenarios —one partition per stage t—, each element of a partition being a decision set, *ii*) a sequence of control values, one per decision set. SPSDR involves several operations. First a finite sample of scenarios is drawn at random from the original process. This sample may form a fan. The second operation consists in the design of decision sets and the computation of an optimal decision rule with respect to those decision sets. The last operation is a validation on a very large sample of scenarios.

4.1 Step decision rule

It is convenient to discuss the two-stage case first. We assume that we are given the discrete set of outcomes of the chance move. A step decision rule consists in a partition into decision sets of this finite set of outcomes and a unique value for the control in each decision set. The partition is built on heuristic grounds. Our approach is inspired by [15]. To this end, we need a distance measure between scenarios, e.g., a l_1, l_2 or l_∞ norm on the sample space Ξ_2. To each element of the partition we associate a representative scenario. We then endow the set of representative scenarios with a probability distribution. Finally, we compute a distance between the two distributions: the original set of scenarios and the set of representative scenarios. This distance is computed as the solution of the so-called the Monge-Kantorovitch transportation problem, which we detail below.

Let $\Xi = \{\xi^1, \ldots, \xi^N\}$ be the sample space with elements in $\mathbb{R}^n$ and let μ and $\bar{\mu}$ be two discrete probability measures on that space. We denote

$$p_i = \mu\{\xi = \xi^i\} \text{ and } \hat{p}_i = \hat{\mu}\{\xi = \xi^i\}.$$

We assume that the support of $\hat{\mu}$ is Ξ_J, where $J \subset \{1, \ldots, N\}$. Thus $\hat{p}_i = 0$ for all $i \notin J$. Given a norm $c : \Xi \times \Xi \to \mathbb{R}$, we estimate the distance between the two probability measures via the Kantorovitch functional κ:

$$\kappa(\mu, \hat{\mu}) = \min\{\sum_{i,j=1}^{N} c(\xi^i, \xi^j)\eta_{ij} \mid \eta_{ij} \geq 0, \sum_{i=1}^{N} \eta_{ij} = \hat{p}_j, \sum_{j=1}^{N} \eta_{ij} = p_i\}. \tag{3}$$

The above problem is known as the Monge-Kantorovitch mass transportation problem. In our experiments, we explore different norms c: the l_1, l_2 and l_∞ norms.

We are interested in finding a set of indices J, with $|J| = s$, and a probability measure $\hat{\mu}$ on Ξ with support Ξ^J such that the distance $\kappa(\mu, \hat{\mu})$ is minimized. This problem is equivalent to a p-median problem. We briefly sketch the proof of the equivalence. Let us assume first that J is given. The problem is then to find $\hat{\mu}$ with support Ξ^J such that the distance is minimized. This amounts to solving the finite dimensional problem in the variables $\hat{p}$ and η

$$\begin{aligned}
\min_{\hat{p},\eta} \quad & \sum_{i,j=1}^{N} c(\xi^i,\xi^j)\eta_{ij} \\
& \sum_{i=1}^{N} \eta_{ij} = \hat{p}_j \ \text{ and } \ \sum_{j=1}^{N} \eta_{ij} = p_i \\
& \eta_{ij} \geq 0, \ \forall i,j \\
& \sum_{j=1}^{N} \hat{p}_j = 1, \ \hat{p}_j \geq 0, j \in J \ \text{ and } \ \hat{p}_j = 0, j \notin J.
\end{aligned}$$

An optimal solution of the mass transportation problem with fixed J is easily obtained as follows. First, we construct a partition of the set $\{1,\dots,N\}$ by assigning each $i \in \{1,\dots,N\}$ to the "closest" element $j \in J$. In case of ties, an element i is assigned to the smallest index j. Assuming that there is no tie, the elements of the partition are defined by

$$I(j) = \{i \in \{1,\dots,N\} \mid j = \arg\min_{k \in J} c(\xi^i,\xi^k)\}, \ \forall j \in J.$$

Since $c(\xi^j,\xi^j) = 0$, then $j \in I(j)$. It is easy to show that the minimum in the mass transportation problem is attained at

$$\hat{p}_j = \begin{cases} \sum\limits_{i \in I(j)} p_i, & j \in J \\ 0 & \text{otherwise} \end{cases}$$

and

$$\eta_{ij} = \begin{cases} 0 & \text{if } i \notin I(j) \\ p_i & \text{otherwise.} \end{cases}$$

In view of the above analysis, we conclude that finding the set J of cardinality s and a probability measure $\hat{\mu}$ on Ξ with support Ξ^J whose distance to μ is minimal amounts to solving a combinatorial optimization problem. Let $c_{ij} = p_i c(\xi^i,\xi^j)$. The optimization problem is

$$\min \quad \sum_{i=1}^{N}\sum_{j=1}^{N} c_{ij}\alpha_{ij} \tag{4a}$$

$$\sum_{j=1}^{N} \alpha_{ij} = 1, \ \forall i \tag{4b}$$

$$0 \leq \alpha_{ij} \leq \beta_j, \forall i,j, \tag{4c}$$

$$\beta_j \in \{0,1\}, \forall j, \ \text{ and } \ \sum_{j=1}^{N} \beta_j = s. \tag{4d}$$

This is precisely the formulation of the p-median problem. It consists in defining s medians among N possible "locations" and assigning each i (for instance, "client") to the closest median. Location j is a median, if and only if $\beta_j = 1$.

Given a solution of the p-median problem with s medians, we construct the decision sets as follows. Let

$$J = \{j_1, \ldots, j_k, \ldots, j_s \mid \beta_{j_k} = 1,\ k = 0, \ldots, s\}.$$

For each j_k, define I^k by

$$I^k = \{i \in \{1, \ldots, N\} \mid j_k = \arg\min_{j \in J} c_{ij}\}. \tag{5}$$

The p-median problem is known to be NP-hard [17]. Nevertheless, there exist efficient algorithms to generate an exact solution (e.g., see the annoted bibliography [20]). There also exist powerful heuristics [14] which generate near-optimal solutions.

Next, we consider the multistage case. We simply extend the concept of decision sets of two-stage case to all stages $t \geq 3$. Like in the traditional approach, we comply with the non-anticipativity requirement by making the decision sets dependent on the past history only. But, contrary to the traditional approach, we do not require that the partition in decision sets at stage $t+1$ be a refinement of the partition at t. Therefore, the partition at any stage t can be built independently of the partitions at any other stage. The only difference with the two-stage case lies in the description of the sample space and the norm c on which the distance between elements of this space is computed. Indeed, at stage t, an element of the space is a scenario $\sigma^i = (\xi_1^i, \ldots, \xi_T^i)$, but the relevant information taken into consideration in the definition of the norm c is only the subscenario $(\xi_1^i, \ldots, \xi_t^i)$. If c is a norm on the space $\mathbb{R}^{N \times t}$ of outcomes, one can define the problem of constructing ν_t decision sets at stage t in the very same way as it was done in the two-stage case.

Let us summarize the decision set design process. We are given an initial set of scenarios as realizations of a T-stage stochastic process. We select a sequence of integer values $\{\nu_2, \ldots, \nu_{T-1}\}$ with $0 < \nu_t \leq N$ for all t. At each stage t, we solve a problem like (4) and construct a partition of the sample space using (5). We thus have the sequence of partitions $\mathcal{P}_2, \ldots, \mathcal{P}_{T-1}$ with $\mathcal{P}_t = \{I_t^1, \ldots, I_t^{\nu_t}\}$.

In our experiments we tried different norms. The one which seem more suitable is a weighted l_1 norm. The weighted norm is defined as follows. Let $\sigma_t = \{\xi_1, \ldots, \xi_t\}$ and $\hat{\sigma}_t = \{\hat{\xi}_1, \ldots, \hat{\xi}_t\}$ be two scenarios up to t. The norm is defined by the recurrence relation

$$c(\sigma_t, \hat{\sigma}_t\} = (1-\alpha)c(\sigma_{t-1}, \hat{\sigma}_{t-1}) + \alpha||\xi_t - \hat{\xi}_t||, \text{ for } t \geq 3, \tag{6a}$$
$$= ||\xi_2 - \hat{\xi}_2||, \text{ for } t = 2. \tag{6b}$$

Here, α is a coefficient between 0 and 1. Values taken by α in our experiments are described in next section.

4.2 SPSDR at work

We consider Phase 0 described in Subsection 3 is common to both SP and SPSDR approaches. If the scenario base set is very large, the problem of finding an optimal decision rule may be too large, even with small cardinality ν_t of the partitions $\mathcal{P}_t$, because the number of constraints is still the same. To cope with this difficulty, we propose to draw a sample Σ' of moderate size N' out of the large base set Σ that was originally given.

Phase 1: Building decisions sets

To build decision sets on Σ' consists in solving $T-2$ p-median problems.

Phase 2: Optimizing the step decision rules

We now assume that we are given decision sets for each stage. Equivalently, we are given a sequence of partitions $\mathcal{P}_2, \ldots, \mathcal{P}_{T-1}$, with $\mathcal{P}_t = \{I_t^1, \ldots, I_t^{\nu_t}\}$. We formulate the optimization problem

$$\min \quad \sum_{i=1}^{N'} p_i [f_T(x_T(\sigma_T^i), \xi_T^i) + \sum_{t=1}^{T-1} f_t(x_t(\sigma_t^i), u_t(\sigma_t^i), \xi_t^i)] \tag{7a}$$

$$x_t(\sigma_t^i) = A_t(\xi_t^i) x_{t-1}(\sigma_{t-1}^i) + B_t(\xi_t^i) u_{t-1}(\sigma_{t-1}^i) + C_t(\xi_t^i) \qquad i = 1, \ldots N', \;\; t = 2, \ldots T \tag{7b}$$

$$g_t(u_t(\sigma_t^i)) \leq 0, \; i = 1, \ldots N', \; t = 1, \ldots T-1 \tag{7c}$$

$$u_t(\sigma_t^i) = v_t^k, \; \forall i \in I_t^k, \;\; t = 2, \ldots T-1, \; k = 1, \ldots \nu_t \tag{7d}$$

$$x_1 = \bar{x}_1. \tag{7e}$$

In this problem, the new constraints (7d) play the role of the non-anticipativity constraints in standard stochastic programming. The true decision variables are the v's.

There is an alternative way of expressing the constraints (7d) of identical decisions on a decision set. Let $\delta_{I_t^k}(\sigma^i)$ be the Dirac function

$$\delta_{I_t^k}(\sigma^i) = \begin{cases} 1 & \text{if } \; i \in I_t^k \\ 0 & \text{otherwise.} \end{cases}$$

Then, we may replace (7d) with the handier expression

$$u_t(\sigma_t^i) = \sum_{k=1}^{\nu_t} \delta_{I_t^k}(\sigma^i) v_t^k, \; i = 1, \ldots N', \; t = 2, \ldots T-1 \tag{8}$$

With this notation we can give a formal definition of a decision rule policy for problem (2) with a given finite set of scenarios.

Definition 1 *A decision rule policy for problem (2) with a given finite set of scenarios Σ is a sequence of partitions $\mathcal{P} = \mathcal{P}_2, \ldots, \mathcal{P}_{T-1}$, with $\mathcal{P}_t = \{I_t^1, \ldots, I_t^{\nu_t}\}$, of respective cardinality ν_t and a set of control values v_t^k, $k = 1, \ldots, \nu_t$, $t = 2, \ldots, T-1$. A decision rule policy is made operational via equation (8).*

Note that the number of control variables at any given stage is proportional to the cardinality of the partition in $\mathcal{P}_t$. An optimal solution of Problem 7 is a prediction for the optimal value of the original problem.

Phase 3: Validation

We extend the decision rule defined on the small sample to the large base set of scenarios. Computing the performance of the decision rule on each scenario becomes a simple decision-free process.

Let us discuss the process of extrapolating a decision rule to the entire base set of scenarios. It consists in assigning to each scenario in the base set and at any stage t a value for the control variables chosen among the values proposed by the decision rule at that stage t. This can be done as follows. Let $\mathcal{P}_t = \cup_{s=1}^{\nu_t} I_t^s$ be the partition of Σ'. The first step consists in assigning to each decision set I_t^s a reference scenario σ^s that has been computed solving the p-median problem. Next, using the distance measure c introduced in (3), we look for the reference scenario that is closest to the current scenario under analysis σ_t. This is given by

$$s = \arg\min_{k \leq \nu_t} c(\sigma_t, \sigma_t^k). \tag{9}$$

The extrapolated decision rule assigns the control v_t^s to the scenario under scrutiny.

Computing the performance of the constructed decision rule on the selected scenario is just routine calculation. By repeating this routine calculation on the full set of scenarios, one obtains a full set of cost realizations. This output can be summarized by statistics like the mean, the median, the α-quantile, and for a better measure of risk, the standard deviation, the absolute mean deviation or the γ-expected shortfall.

4.3 Convex combination of step decision rules

The quality of a decision rule can be improved in different ways. The most obvious one is to increase the size of the sample of scenarios Σ'. However, the complexity of designing the decision sets and computing the decision rules increases notably. Another possibility consists in drawing several samples independently and building for each of them decision sets and decision rules. The computed decision rules can be viewed as advices of *experts*. It is possible to combine those advices, for instance by taking the average of the rules defined by (8). It turns out that the rule built on the pool of experts is more efficient than the individual advices.

Combining expert advice is summarized by the following procedure.

Initialization We are given a (possibly very large) set of scenarios Σ that provide a statistically satisfactory representation of the stochastic process. This set can be constructed.

Experts Repeat for $h = 1$ to H.

1. Draw a sub-sample $\Sigma^h = \{\sigma^{h,1}, \ldots, \sigma^{h,n_h}\}$ with cardinality $|\Sigma^h| = n_h$.
2. Construct decision sets $\cup_k I_t^{h,k}$ according to the heuristics.
3. Compute an optimal decision rule DR^h with respect to Σ^h and the constructed decision sets. The decision rule takes the following form

$$u_t^h(\sigma_t^{h,j}) = \sum_{k=1}^{\nu_t^h} \delta_{I_t^{h,k}}(\sigma_t^{h,j}) v_t^{h,k}.$$

Pooling experts

1. Extend the decision rule DR^h of expert h to all scenarios in Σ. This amounts to constructing, at each t, a partition $\bigcup_{k=1}^{\nu_t^h} J_t^{h,k}$ of Σ. Recall that $\sigma^{h,k}$ is the reference scenario associated with the decision set $I_t^{h,k}$. In view of (9), one assigns $\sigma \in \Sigma$ to the closest reference scenario. By breaking ties if necessary, one obtains the desired partition with the property $I_t^{h,k} \subset J_t^{h,k}$. This expert decision rule is then
$$u_t^h(\sigma_t) = \sum_{k=1}^{\nu_t^h} \delta_{J_t^{h,k}}(\sigma_t) v_t^{h,k}$$
on Σ.
2. Take a convex combination of the single-expert decision rules, usually, the average
$$u_t(\sigma_t) = \frac{1}{H} \sum_{h=1}^{H} u_t^h(\sigma_t). \tag{10}$$

Prediction To compute a prediction of the expected performance of the multiple-expert decision rule we use the $\bigcup_{h=1}^{H} \Sigma^h$ generated scenarios. Rule (10) defines the control and, via (2b), the state variables values on each scenario. The prediction is the average of the associated costs.

Validation

1. Proceed like with the single expert and extend the multiple expert rule (10) to the full set of validation scenarios. Compute the associate state and cost values.
2. Compute statistics, such as the average cost, over the set of all scenarios.

The next theorem justifies the merit of taking a convex combination of decision rules.

Theorem 1 *The expected cost of a convex combination of decision rules is less or equal than the convex combination of the expected costs of the individual decision rules.*

Proof: In view of the convexity of Problem (1), we can formally state that the expected cost of the convex combination of individual decision rules is less than the same convex combination of the expected costs of the individual experts. To prove this statement, it is sufficient to prove it on an individual scenario. Recall that on a scenario, the sequence of controls uniquely defines the sequence of states by equation (7b). First, it is clear that the convex combination of the decision rule satisfies the convex constraint (7c). Next, due to the linearity of the state equation (7b), the state trajectory associated with the combination of decision rules is the convex combination of the individual trajectories. Finally, the convexity of the objective function implies that the function value of the convex combination of decision rules is less or equal to the convex combination of the individual function values. Since the expectation is itself a convex combination, the result follows. A very similar scheme has been used in [18]. The value of combining expert advices has also been exploited in the context of machine learning and statistics [8].

■

4.4 Tractability of SPSDR

The computational effort with SPSDR can be split into three components, the design of the decisions sets, the computation of optimal decision rules for each expert and the pooling of the expert decision rules to predict performance.

Decision sets design Once the norm to evaluate the distance between scenarios and the number of decision sets per stage have been selected, the computational effort consists in solving for each stage a p-median problem. This problem is NP-hard [17], but we need not solve it exactly. A crude heuristic as in [15] may suffice. In our experiments we used the very efficient Variable Neighborhood Decomposition Search metaheuristic of Hansen et al. [14]. The implementation of the VNDS heuristic we use can handle p-median problems with up to 4000 clients, that is 4000 nodes in our case. Note that the computations for each stage and each expert can be performed independently.

Optimal decision rule With the help of equation (8), Problem (7) can be reformulated in the variables v only, which assign values of the control u in each decision set. The number of variables v is equal to the total number of decision sets, a small number in our experiments. In view of the linearity of the state equation, it is easy to compute the value of the objective function on each scenario. If the objective function is non-differentiable, as it is the case in our examples, it is also possible to compute a subgradient of this convex function of v. This information can be exploited to compute an optimal solution with precision ϵ in $O(\frac{\nu}{\epsilon^2})$ iteration, where ν is the dimension of v.

When the number of scenarios per expert is not too large, it is also possible to formulate the problem of finding an optimal decision rule as a deterministic mathematical programming problem. In that framework, each state variable and each state equation must be made contingent to the individual nodes of the scenario tree.

Compute a prediction By pooling the scenario samples submitted to the experts, we obtain a larger sample on which one can evaluate the performance of the decision rule of the pool of experts. This provides a prediction, which appears to be fairly reliable in our experimental study.

5 Empirical study

In this section we propose to assess on empirical grounds the merits of stochastic programming with step decision rules. We face numerous difficulties. Some stem from the fact that we have to decide on the scenario sample size, on the number of decision sets at each stage and on the number of experts, to speak of the most obvious required specifications.

Unfortunately, we are not given clear guidelines to perform this task. The other difficulty is to find benchmarks for our method. We have to choose multistage stochastic problems that can be solved by well-established methods such as standard stochastic programming or robust optimization.

In this paper, we focus on two simple models that have been studied in the literature on supply chain management. The first one is the celebrated newsboy problem in its multistage version. We consider a model with only four periods, an initial decision and two recourse decisions. (There is no recourse in the fourth period. The problem has thus three stages.) This model can be solved by regular stochastic programming with excellent precision. The second model is more complex. It deals with a supply/purchase contract between a producer and a retailer. Monthly target orders are to be set in advance, but adjustments are possible in terms of costly recourses. The model has 13 periods (i.e., 12 stages). A robust solution is proposed in [4].

The solutions obtained by SPSDR in one hand, and by standard stochastic programming and robust optimization in the other hand are tested on very large samples, from one hundred of thousands to one million scenarios. Performance criteria such as expected shortfall, ESf, (at levels 1% or 5%), mean values, and empirical standard deviation are computed. Other criteria can be computed: critical quantiles (at 1% and 5%), median and absolute mean deviation, but we did not use them.

5.1 SPSDR vs. SP on the newsboy problem

The main objective of this example is to show that SPSDR can be applied on a standard multistage stochastic problem and can compete with standard SP approach. Let the deterministic version of the multistage newsboy problem be displayed below.

$$\max \quad \Big(\sum_{t=2}^{T} p_t d_t - p_T[-x_T]^+\Big) - \sum_{t=2}^{T}\big(a_{t-1}u_{t-1} + h_t[x_t]^+ + s_t[-x_t]^+\big) \tag{11a}$$

$$x_t = x_{t-1} + u_{t-1} - d_t,\ t = 2, \dots T, \tag{11b}$$

$$u_t \geq 0,\ t = 1, \dots T-1, \tag{11c}$$

$$x_1 = \bar{x}_1, \tag{11d}$$

with $[x_t]^+ = \max\{0, x_t\}$. In this model, u_t is the order at stage t and d_t is the demand. The inventory level is x_t: a positive value means a physical inventory, and a negative one a shortage. Initial inventory is fixed at $\bar{x}_1 \geq 0$. Unsatisfied demand is backlogged. The parameter h_t is the unit holding cost, and s_t the unit shortage cost. The purchase cost is a_t and the selling prize is p_t. The objective is written as the difference between the revenue and the costs. The total revenue is computed as follows: a demand d_t generates an immediate revenue $p_t d_t$, even though the delivery may be postponed to a later period. At the horizon time, the revenue from the unsatisfied demand $[-x_T]^+$ must be deduced or paid back at the price p_T. (The maximization problem can be modified in a cost minimization if one remarks that the revenue component $\sum_{t=2}^{T} p_t d_t$ is independent of the decision process.)

To make the problem a little more challenging, we choose a model with correlated demands as in [24]. More specifically, the demands form a conditionally heteroskedastic

Gaussian process with unconditional mean and variance μ_t and σ_t. The conditional mean is

$$E(d_{t+1} \mid d_t = d) = \mu_{t+1} + \rho_t \frac{\sigma_{t+1}}{\sigma_t}(d - \mu_t) \tag{12}$$

and the conditional variance

$$\operatorname{Var}(d_{t+1} \mid d_t = d) = \sigma_{t+1}^2(1 - \rho_t^2), \tag{13}$$

where ρ_t is the correlation coefficient between the successive demands d_t and d_{t+1}. The one-step transition is given by

$$d_{t+1}(d_t, \xi_{t+1}) = \mu_{t+1} + \rho_t \frac{\sigma_{t+1}}{\sigma_t}(d_t - \mu_t) + \sigma_{t+1}\sqrt{1 - \rho_t^2}\, \xi_{t+1},$$

where ξ_{t+1} follows a standard normal distribution. We further assume that the $\xi_1, \dots, \xi_T$ are independent and identically distributed.

The data for the numerical example are:

- $T = 4$.
- $\mu_t = 15$, for all $t = 2, \dots T$.
- $a_{t-1} = 1$, $s_t = 0.2$, $h_t = 0.1$ for all $t = 2, \dots T$.
- $p_t = 1.4$ for all $t = 2, \dots T$.
- $\sigma_2 = 2$, and $\sigma_t = \frac{\sigma_2}{\sqrt{1-\rho^2}}$ for all $t = 3, \dots T$.
- $\rho = 0$ or 0.5.

We first discuss a standard implementation of stochastic programming. The main task is to construct a discrete stochastic process that approximates the continuous one. This is done by approximating the standard normal distribution with a discrete one with s elements. The distance between the continuous law and the discrete one is measured according to a suitably adapted version of (3) (to account for the continuous distribution) with the l_1 norm. It is easy to show that the optimal solution in (3) is given by a partition of the real line in s contiguous intervals $[\alpha_i, \beta_i]$ and each interval is represented by its relative median m_i. The total distance is thus $\sum_{i=1}^{s} \int_{\alpha_i}^{\beta_i} |m_i - u| f(u) du$, where $f(u)$ is the density of the standard normal distribution. The bounds of the intervals are determined by the minimization of the total distance. In this way, we obtain a balanced tree with $s \times s \times s = s^3$ scenarios. Note that the number of contingent order decisions u_t at $t = 1, 2, 3$ is $1 + s + s^2$, but the formulation of the deterministic equivalent as a linear programming problem also involves state variables at $t = 4$. This means $(s^{t+1} - 1)/(s - 1)$ nodes. With $s = 20$, the deterministic equivalent is a medium size linear programming problem with 17,261 variables, 8,420 equality constraints and 17,261 non-negativity constraints. With $s = 40$, there are 132,922 variables and non-negativity constraints and 65,640 equality constraints.

To implement SPSDR we chose to work with 10 experts and to submit to each expert a sample of 200 scenarios drawn at random using the continuous distribution of the random process. These samples are all fans. Our second modelization choice is to endow each expert with decision sets of cardinality 10 and 30 in stage $t = 2$ and $t = 3$ respectively.

The decision sets are constructed so as to minimize the distance in (3), that is by solving the p-median problems defined in (4). The norm c is the l_1 norm at $t = 2$ and a convex combination of the l_1 norms on d_2 and d_3 with coefficients 0.35 and 0.65. The expert problem is a linear programming one of moderate size. For instance, with 200 scenarios, and $10 + 30 = 40$ decision sets, the problem has 1,241 variables and non-negativity constraints, and 640 equality constraints. This size is about the same as one would get with the deterministic equivalent for an $8 \times 8 \times 8$ event tree.

5.1.1 Comparison SP / SPSDR

Table 1 presents the results obtained by solving a standard SP with three different discretization factors ($s = 5, 10$ or 20) and by using the SPSDR approach with the previously defined parameters. The results are given for two versions of the stochastic process, one without correlation, the other one with correlation $\rho = 0.5$.

	SP			**SPSDR**
$\rho = 0$	s = 5	s = 10	s = 20	
achieved	15.780	15.868	15.933	15.907
predicted	16.251	16.024	15.974	15.944
prediction error	2.66%	0.92%	0.26%	0.23%
$\rho = 0.5$				
achieved	15.679	15.838	15.927	15.885
predicted	16.251	16.024	15.974	15.919
prediction error	3.52%	1.15%	0.29%	0.21%

Table 1: Newsboy problem: predicted and achieved profit

We observe that the quality of the SP solution and the prediction improve with the level of discretization. Note that the predicted value is always above the achieved value. SPSDR achieves profits that are almost as good as the profit reported by SP with the finer discretization ($s = 20$). The prediction error is small.

The following subsections analyze the variability of the results due to parameter settings. They also study the impact of the random choice of the sample submitted to the experts. Finally they discuss the variability due to the random selection of the validation scenarios.

5.1.2 Variability due to parameter setting

In the experiments of Table 1, the parameters have been fixed to the reported values by a trial and error process. We summarize the main experiments that led to the final choice. In each experiment, we let one parameter vary while the other ones were kept fixed.

Let us first discuss the choice of the norm. Table 2 displays the results with 6 different norms. We recall that a weighted l_1 norm in period 3 is the convex combination of the l_1 norm on d_2 and d_3. The coefficients of the combination are $1-\alpha$ and α, with $\alpha = 0.5, 0.65, 0.8$.

	l_1	l_2	l_∞	weighted l_1		
$\rho = 0$				$\alpha = 0.5$	$\alpha = 0.65$	$\alpha = 0.8$
achieved	15.904	15.895	15.895	15.904	15.907	15.904
predicted	15.978	15.979	15.979	15.978	15.943	15.983
$\rho = 0.5$						
achieved	15.890	15.877	15.876	15.890	15.885	15.896
predicted	15.840	15.828	15.830	15.840	15.919	15.917

Table 2: Newsboy problem : influence of the norm

The first line represents the value found after a validation over 100,000 randomly selected scenarios. (The influence of the random choice of the validation sample is discussed in subsection 5.1.3.) The second line displays predicted values for the objective function. Clearly, the choice of the norm has limited influence on the performance. The l_1 norm seems to perform better than the l_2 and l_∞ norms.

Table 3 displays the influence of the size of the sample submitted to the individual experts. We observe an increase in the size of the sample improves the quality of the solution in the uncorrelated case, but with decreasing gains. In the correlated case, we observe the paradoxical fact that with 400 scenarios the performance is worse than with 200 scenarios only. However, one must recall that we kept the number of decision sets, 10 and 30, fixed. With 400 scenarios and more decision sets, the performance would have improved. The computational cost increases more than proportionally with the number of scenarios. The value 200 seems to be a good compromise.

	$\rho = 0$			$\rho = 0.5$		
sample size	100	200	400	100	200	400
achieved	15.846	15.907	15.919	15.771	15.885	15.843
predicted	15.827	15.944	15.922	16.578	15.919	16.595

Table 3: Influence of the size of the sample submitted to an individual expert

Table 4 shows the influence of an increasing number of experts. If the number of experts in the pool increases, the combined SDR is improved, but the gain from 10 experts to 20 is marginal.

The last parameter under study is the number of decision sets at each period. We tried several combinations: the same number of decision sets per period, or an increasing one. The results can be seen on Table 5.

The table shows that a large number of decision sets (figures on the left-side of the table) has a negative effect on the achieved performance. Indeed, with a large number of

	$\rho = 0$			$\rho = 0.5$		
number of experts	5	10	20	5	10	20
achieved	15.879	15.907	15.923	15.864	15.885	15.896
predicted	15.936	15.944	15.975	16.009	15.919	15.917

Table 4: Influence of the number of experts

decision sets	25, 100	30, 100	50, 50	10, 30
$\rho = 0$				
achieved	15.835	15.838	15.842	15.907
predicted	15.978	15.975	15.955	15.944
pred. error	0.89%	0.86%	0.71%	0.23%
$\rho = 0.5$				
achieved	15.832	15.806	15.828	15.885
predicted	16.599	16.758	16.537	15.919
pred. error	4.62%	5.68%	4.29%	0.21%

Table 5: Influence of decision sets distribution

decision sets, the expert tends to adjust tightly his decision rule to the particular values of the sample. Such decision rule is less efficient on the validation sample.

5.1.3 Variability in the performance

Since the sample of scenarios submitted to an expert is random, the issuing decision rule is also random. To study this effect, we repeated 10 times the construction of a decision rule with 10 experts each time. The other source of randomness in the performance evaluation is the selection of the validation sample itself. To this end, we validated the same decision rule on 10 independent samples of size 100,000. The results of the two experiments are displayed on the same Table 6.

	Different SDR's same validation sample		Different validation samples same SDR	
correlation	0	0.5	0	0.5
achieve mean	15.907	15.885	15.908	15.885
std dev.	0.004	0.006	0.006	0.009
recorded min	15.903	15.876	15.899	15.872
recorded max	15.912	15.894	15.916	15.898

Table 6: Variability of the performance

We observe that the variability of the expected profit is very limited. The relative difference between the extreme values min and max is significantly smaller than the prediction error reported in Table 1.

5.2 SPSDR vs. robust optimization on a retailer-supplier contract model

This section is devoted to an empirical study of SPSDR on a multistage problem with a larger number of stages, the retailer-supplier flexible commitment problem, in short RSFC. In the Newsboy problem we could compare SPSDR to plain stochastic programming, which, due to the small number of stages, provides near optimal solutions. Unfortunately, the RSFC has too many stages to allow for an easy implementation of stochastic programming. We have to look for another benchmark. In the present study, we shall use the approach and the results of [4]. In that paper, the RSFC problem of [2] is solved by a variant of Robust Optimization, named AARC, the affinely adjustable robust counterpart. This variant is designed for dynamic problems. The adaptive nature of the recourses is captured by making them linear functions of the observed stochastic process [5]. The unknown in that formulation are the coefficients of the linear functions.

The deterministic dynamic model is

$$\begin{aligned}
\min_{x,w,q} \quad & -s[x_T] + \sum_{t=1}^{T-1} c_t q_t + \sum_{t=2}^{T} h_t[x_t]^+ + s_t[-x_t]^+ \\
& \qquad + \sum_{t=1}^{T-1} \alpha_t^+[q_t - w_t]^+ + \alpha_t^-[w_t - q_t]^+ && \text{(14a)} \\
& \qquad + \sum_{t=1}^{T-1} \beta^+[w_t - w_{t-1}]^+ + \beta^-[w_{t-1} - w_t]^+ && \text{(14b)} \\
\text{s.t.} \quad & x_t = x_{t-1} + q_{t-1} - d_t, \ t = 2, \dots, T, && \text{(14c)} \\
& L_t \le q_t \le U_t, \ t = 1, \dots, T-1 && \text{(14d)} \\
& x_1 = \bar{x}_1, \ w_1 = \bar{w}_1. && \text{(14e)}
\end{aligned}$$

This model represents a single-product, two-echelon, multi-period supply chain in which inventories are managed periodically over a finite horizon of T periods. At the beginning of the planning horizon the retailer specifies a vector of commitments $w = w_1, ..., w_T$ for the product. These commitments serve as forecasts for the supplier who uses them to determine his production capacity. At the beginning of each period t, the retailer has an inventory of size x_t and he orders a quantity q_t from the supplier at a unit cost c_t. The customers demands d_t are then revealed. The retailer's status at the beginning of the planning horizon is given through the parameters x_1 (initial inventory) and w_0 (a nominal value that might represent the last order prior to the planning horizon or some average of previous orders). Consequently, the following costs are incurred:

- Holding cost $= h_t[x_t]^+$, where h_t are the unit holding costs.
- Shortage cost $= s_t[-x_t]^+$, where s_t are the unit shortage costs.

Moreover, due to the stipulations in the contract, the retailer incurs the following additional costs:

- Penalty due to deviations between committed and actual orders $\alpha_t^+[q_t - w_t, 0]^+ + \alpha_t^-[w_t - q_t]^+$, where α_t^+, α_t^- are the unit penalties for positive and negative deviations, respectively.

- Penalty on deviations between successive commitments $\beta_t^+ \, [w_t - w_{t-1}]^+ + \beta_t^-[w_{t-1} - w_t]^+$, where β_t^+, β_t^- are the associated unit penalties.

Inventory x_{T+1} left at the end of period T has a unit salvage value s. To make sense in our context, the parameter s must be smaller than c_T. To maintain convexity of the objective function in the model, s must satisfy for the terminal period T the inequality

$$h_T - s \geq -s_T.$$

This model is an extension of the multistage newsboy problem (11), but the interpretation of the salvage value s is somehow different. In the above model with commitments we use the definition given in [4] for the sake of consistency.

The original model includes a constraint on cumulative orders, but this constraint turned out to be inactive on the selected instances W12 and A12.

We shall derive SPSDR solutions for different objectives: the total expected cost, the minimization of the maximum cost, a bound on expected shortfall ESf, or conditional value at risk CVaR [3], at the level γ, with $\gamma = 1\%$ or 5%, and a convex combination of the total expected cost and the ESf at 5%. The expected shortfall is a convex functional that can be computed in a stochastic programming framework with discrete events using the following trick [21]. Let C_i the cost associated with the i-th scenario and π_i its probability. The expected shortfall at the level γ is the solution of the problem

$$\min\{z + \frac{1}{\gamma}\sum_{i=1}^{N} \pi_i u_i \mid u_i \geq C_i - z \text{ and } u_i \geq 0,\ i = 1, \ldots, N\}.$$

5.2.1 Results with instance W12

The first set of numerical experiments aims to compare the AARC solution and the solution obtained with SPSDR. To this end, we choose the set of data W12 on which [4] shows that the AARC performs much better than the standard RC solution. The model parameters are displayed on Table 7. The demands are taken to be i.i.d. with a uniform

T	x_1	w_0	c_t	h_t	p_t	s	α_t^+	α_t^-	β_t^+	β_t^-	L_t	U_t
12	0	100	10	2	10	0	10	10	10	10	0	200

Table 7: Data for problem W12 in [4]

distribution on the interval 100 ± 70. The base set of scenarios is obtained by Monte-Carlo sampling on the demand. Its size is 100,000. The AARC solution is the one reported in [4].

It is important to note that robust optimization approach gives us a guaranteed minimum level, but no prediction on the mean cost. This is why the values presented below are difficult to compare. With the best parameter settings given in next paragraph, the SPSDR approach predicts a mean cost of 15985 whereas the RO approach bounds the worst scenario cost by 22722.

Impact of design factors The design factors are: the size of the samples assigned to individual experts, the distance function between scenarios, the number of decision sets at each stage and the number of experts. We tried different combinations, but we report only the results with

- 10 experts.
- 200 scenarios per expert.
- Number of decision sets per stage: $10, 10, 10, 12, 12, 12, 14, 14, 14, 16$ and 16

Note that the first two figures are similar to those used in the Newsboy problem. With this configuration, the total number of decision variables for each expert is $140 + 1 = 141$, that is 5.9% of the total number of nodes in 200 scenarios forming a fan.

In the Newsboy problem, the choice of a distance between scenarios was not critical. For the RSFC problem, this is not so. We tried four different possibilities. The results are given in Table 8.

	l_1	l_2	l_∞	weighted l_1
achieved cost	16044	16030	16077	15980
predicted cost	16013	15978	16000	15985

Table 8: W12: influence of the norm

In this table, the weighted l_1 norm is as defined in Equation (6), with $\alpha = 0.65$. The weighted l_1 norm gives the best results; we shall use it in all subsequent experiments.

Stability As noted in the Newsboy problem, the SPSDR solution is random. We study here the effect of this randomness. We also study the stability of a step decision rule (SDR) with respect to different validation samples.

Table 9 displays the results for two types of stability. The first two columns correspond to the performance of 10 different SPSDR solutions; each SPSDR is obtained by the usual process: drawing 10 samples of size 200, building the individual expert SDR's and averaging them. The last two columns of the table correspond to the same decision rule (obtained by pooling 10 expert decision rules) tested on 10 samples of size 100,000. The column header "StD(a)" means the standard deviation between the 10 different means (achieved and predicted). The row entry "StD(b)" is the mean of the standard deviation of the outcomes *i)* of 10 different SDR and the same validation sample *ii)* of the same SDR on 10 different validation samples.

	10 SDR's same validation sample		10 validation samples same SDR	
	mean cost	StD(a)	mean cost	StD(a)
achieved cost	15992	9	15990	10
StD(b)	2098		2107	
predicted cost	15949	52	15943	44

Table 9: W12: variability of the performance

The table shows that the variability in both cases is limited. Recall that the SDR's are built on samples of fairly small sizes. In that respect, the variability is not surprising, but the table shows that it is very limited.

SPSDR with different objectives The goal of this section is to analyze the SPSDR solutions with different objective functions, some geared towards risk control, other focusing on the expected costs.

The different objective functions are

- The expected cost,
- One of the risk measures: ESf at 5%, ESf at 1% and maximum cost value.
- The combination with equal weights of the expected cost and the ESf at 5%.

Objectives	mean cost	mixture	expected shortfall		max value
Perf. criterion			5%	1%	
predicted mean	15985	16554	17105	17409	17409
achieved mean	15980	16587	17140	17441	17441
standard dev.	2047	1131	857	768	767
ESf 1%	24644	22748	22271	22031	22028
min recorded cost	12744	14748	15612	16007	16011
max recorded cost	31200	31251	30458	29674	29676

Table 10: W12: performance criteria with different objective functions

In Table 10, the columns are ranked by increasing concern for risk in the objective function. One observes that the risk performance deteriorates when moving from right to left, while the expected cost performance improves. This result is in line with the intuition. The row "standard dev." displays the standard deviation on the validation sample.

Clearly, the sample size on which the expert decision rule is designed and optimized is too small to lead to accurate prediction when the objective is a risk measure. For instance, the pool of expert predict a worse case of 17409, but the achieved worse case is

29676. The results are more satisfactory for the first two criteria: the expected cost and the mixture. With the mean cost criterion, the prediction error is less than 0.1%. Notice that the standard deviation decreases steadily when the risk content of the optimized criterion increases.

Robust optimization with different immunization levels In [4], the focus was on minimizing the maximum cost for all demands within a given uncertainty set. This result is an implementable policy to be evaluated on the validation sample. This policy is very conservative. It perfectly controls the worse case, but does poorly on the mean cost. This is not surprising, because the mean cost is not part of the model. An alternative to this conservative approach consists in looking for a worse case policy on a reduced uncertainty set, e.g., shrinking the original uncertainty set by a certain percentage. However, the validation process will still be performed on a sample drawn from the original distribution. This sample is likely to include scenarios lying outside the reduced uncertainty, and on some of these scenarios, the robust optimization policy may behave worse than expected. However, the reduced uncertainty set eases the constraints in the AARC model and makes it possible to achieve better performance on the average cost.

To illustrate that point, we computed the AARC solution with various levels of immunization. In W12, a level of immunization ρ defines an uncertainty set for the individual demands equal to $100 \pm \rho \times 70$. The policies corresponding to the different levels of immunization are simulated on the same validation sample. The results appear on Table 11. In this table, the AARC solution is a prediction on the maximum cost and not on the

immunization	29%	43%	57%	71%	86%	100%
achieved mean cost	18158	18367	18777	19386	20195	21204
std dev	1458	1116	795	516	316	253
AARC solution	15064	16595	18127	19659	21191	22722
max recorded cost	25998	24760	23724	22732	21840	22284

Table 11: W12: AARC with different levels of immunization

mean. The table shows that a lower level of immunization makes it possible to achieve lower mean costs, but the maximum recorded cost increases. Moreover the prediction error on the worse case cost increases significantly. With the W12 data, the constraints on the recourse are always respected, wathever the immunization level.

SPSDR versus AARC It is possible to compare SPSDR and AARC from Tables 10 and 11. To highlight the differences, we have plotted on Figure 3 the empirical densities of the distribution of costs. For SPSDR we used the optimization criteria "mean cost", "mixture" and "worst case"; for AARC, we chose the immunization levels 29%, 71% and 100%.

The picture confirms the figures in the two tables. The shift to the right (higher cost) of the curves when the objective focuses on risk (SPSDR) or the immunization increases (AARC) is striking. In the meantime, the density is increasingly concentrated. As far

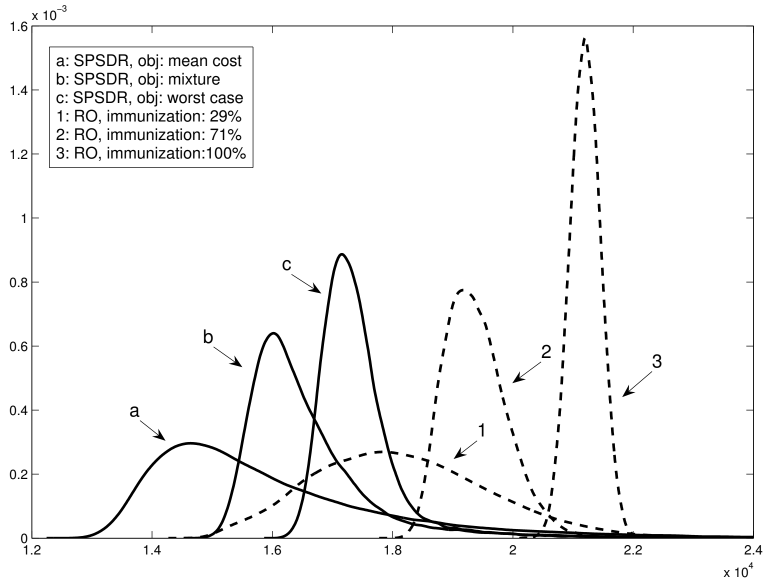

Figure 3: W12: empirical density curves

as average costs are concerned, the SPSDR is certainly more efficient, but one must not forget that the AARC is concerned with the rightmost part of the curve. For costs above 22000, the curves seem to coincide, but this is not so. The AARC dominates SPSDR in this area.

The last picture (Figure 4) shows the commitments selected by SPSDR and AARC. Commitments are first stage decisions. They cannot be modified at later stage, while all other decisions can be re-evaluated in a rolling or folding horizon perspective.

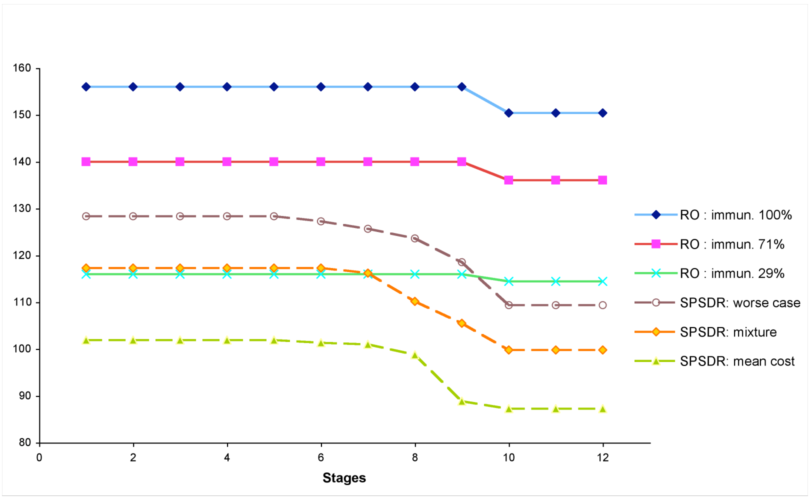

Figure 4: W12: commitments

5.2.2 Results with instance A12

This data set, as described in Table 12, is caracterized by a positive lower bound $L_t = 44, \forall t = 1, \ldots T$, and an upper bound which varies with time. The shortage cost is constant until $t = T - 1$ and increases at T. The demands are taken to be i.i.d. with a uniform distribution on the interval 64 ± 44.8 (70% uncertainty). The validation sample of size 100,000 is obtained by Monte-Carlo sampling on the demand process. The AARC solution is the one reported in [4]: with a 100% immunization, the RO guaranteed level is 5863.32.

T	x_1	w_0	c_t	h_t	s	α_t^+	α_t^-	β_t^+	β_t^-
12	57	12	1.01	0.3	1.13	0.43	0.58	0.37	0.04

t	1	2	3	4	5	6	7	8	9	10	11	12
L_t	44	44	44	44	44	44	44	44	44	44	44	44
U_t	76	54	66	88	68	60	82	53	53	78	72	63
p_t	1	1	1	1	1	1	1	1	1	1	1	6.7

Table 12: Data set A12 for RSFC problem in [4]

The SPSDR design parameters for this experiment are the same as for W12, namely, samples of 200 scenarios for each of the 10 experts and a number of decision sets per stage $10, 10, 10, 12, 12, 12, 14, 14, 14, 16$ and 16. The distances between sub-scenarios are taken with respect to the weighted l_1 norm (6), with $\alpha = 0.65$. We performed the same analyses as with W12, but we only discuss the main results.

Stability Table 13 displays results of two distinct stability computations.

	10 SDR's same validation sample		10 validation samples same SDR	
	mean cost	StD(a)	mean cost	StD(a)
achieved mean	1084.6	0.9	1084.2	1.3
StD(b)	284.3		285.2	
predicted mean	1081.9	7.5	1079.6	5.5

Table 13: A12: variability in the performance

The rows and columns have the same meaning as in Table 9. We observe the same stability properties as before.

SPSDR with different objectives Table 14 displays predicted and achieved mean costs with different objective functions.

We can observe that the prediction error is very small ($< 0.5\%$).

Objectives	mean cost	mixture	expected shortfall		max value
Performance criterion			5%	1%	
predicted mean	1083	1092	1121	1200	1172
achieved mean	1086	1093	1123	1202	1175
standard dev.	287	236	236	356	332
ESf 1%	2602	2433	2453	2764	2674
min recorded cost	801	812	845	820	828
max recorded cost	4158	3962	3975	4143	4044

Table 14: A12: performance criteria with different objective functions

Immunization level Table 15 displays the results of AARC with different levels of immunization.

immunization	29%	43%	57%	71%	86%	100%
achieved mean cost	1159	1153	1194	1216	1238	1267
std dev.	260	255	299	328	354	386
AARC solution	1397	2191	3088	4006	4935	5863
max recorded cost	3961	3961	3961	3987	4117	4242

Table 15: A12: AARC with different levels of immunization

We observe that the AARC solution value is way above the achieved mean cost when the immunization level is high. One must be careful using those results: when the immunization level is low, the majority of scenarios generates recourses which do not respect constraints. if the recourse does not lie between the upper and lower bound, it is simply truncated to its appropriate bound.

We also notice that the SPSDR with the "mean cost" and "mixture" objectives give better results. Figures 5 and 6 illustrate a comparison between SPSDR with the mean objective (respectively, the worse case) and AARC with 29% (resp., 100%) immunization levels.

6 Conclusion

Multistage stochastic programming problems are computationally intractable [22]. The main reasons are the complexity of computing expectations with respect to a multidimensional stochastic process and the exponential growth of the event tree as the number of stages increases. To get around these difficulties we propose a heuristic approach based on step decision rules. The idea is to solve several independent random problems. Each problem corresponds to a sample of scenarios drawn at random according to the distribution of the stochastic process. These samples, which often take the form of a fan of scenarios, need not be large. For each sample, one constructs decision sets and associates to them

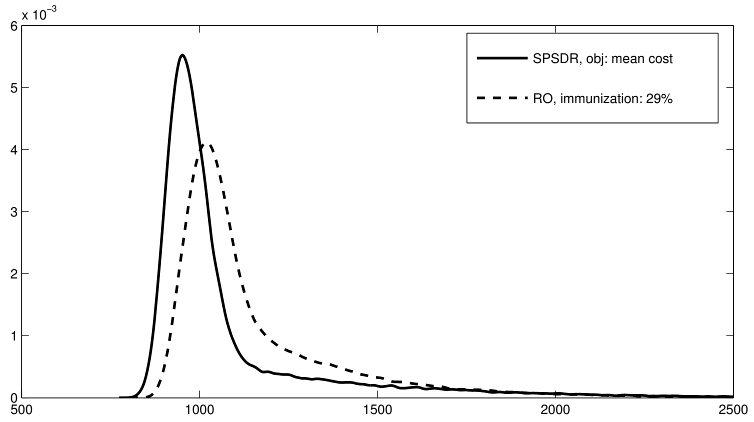

Figure 5: Empirical density of the cost for A12: SPSDR (mean cost objective) vs. AARC (immunization level 29%)

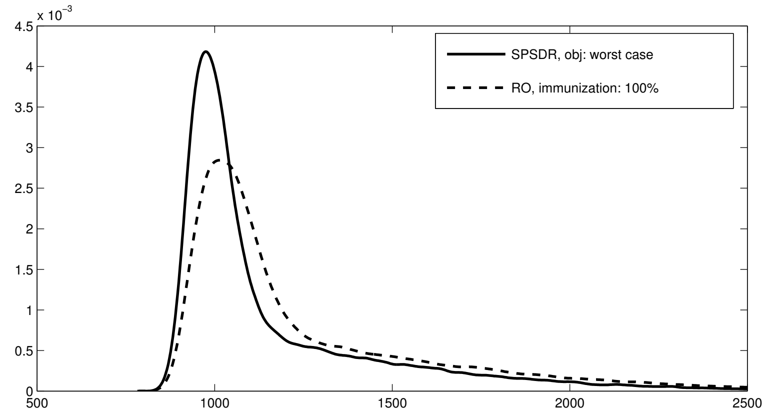

Figure 6: Empirical density of the cost for A12: SPSDR (worse case objective) vs. AARC (immunization level 100%)

a step decision rule. Optimal step decision rules for each sample are then computed as solutions of tractable convex programs. The optimal step decision rules associated with the different samples are combined to generate the final solution. We named this process, pooling the experts advices. Finally the solution is validated on a very large sample of scenarios.

The new approach has been tested against two alternative methods: regular stochastic programming on a problem with only 3 stages and 2 recourses; robust optimization with affinely adjustable recourses on a 12-stage model. Our experimental results show that SPSDR is competitive on these models and probably has a potential on other problems. SPSDR has several nice characteristics. Contrary to regular stochastic programming, it is not necessary to construct event trees that progressively unfold with time. It is possible to start with a sample of scenarios forming a fan and build decision sets whose number grows mildly with stages. An other nice feature is the small dimension and the computational tractability of the individual expert problems. The last feature is the flexibility in choosing the objective function, a definite plus over robust optimization.

References

[1] E. Adida and G. Perakis. A robust optimization approach to dynamic pricing and inventory control with no backorders. *Mathematical Programming*, 107(1-2):97 – 129, 2006.

[2] R. Anupindi and Y. Bassok. *Quantitative Models for Supply Chian Management*, chapter Supply Contracts with Quantity Commitments and Stochastic Demand. Kuwer Academic Publishers, 1998.

[3] P. Artzner, F. Delbaen, J.-M. Eber and D. Heath. Coherent measures of risk. *Mathematical Finance*, 9:203–229, 1999.

[4] A. Ben-Tal, B. Golany, A. Nemirovski and J.-Ph. Vial. Retailer-supplier flexible commitments contracts: a robust optimization approach. *Manufacturing and service operations management*, 7:248–271, 2005.

[5] A. Ben-Tal, A. Goryashko, E. Guslitzer and A. Nemirovski. Adjusting robust solutions of uncertain linear programs. *Mathematical Programming*, 99(2):351–376, 2004.

[6] D. Bertsimas and A. Thiele. A robust optimization approach to inventory theory. *Operations Research*, 54(1):150–168, 2006.

[7] J.R. Birge and F. Louveaux. *Introduction to stochastic programming.* Springer-Verlag New York, Inc., 1997.

[8] N. Cesa-Bianchi, Y. Freund, D. Haussler, D.P. Helmbold, R.E. Schapire and M.K. Warmuth. How to use expert advice. *Journal of the ACM*, 44(3):427–485, 1997.

[9] M. Dyer and L. Stougie. Computational complexity of stochastic programming problems. Technical report, School of Computing, University of Leeds, England, dyer@comp.leeds.ac.uk, leen@win.tue.nl, 2005.

[10] Y. Freund and R. Schapire. Experiments with a New Boosting Algorithm. in Proceedings of the Thirteenth International Conference on Machine Learning, pp. 148156, Morgan Kaufmann, 1996.

[11] Y. Freund and R. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[12] S.J. Garstka and R. J.B. Wets. On decision rules in stochastic programming. *Mathematical Programming*, 7:117–143, 1974.

[13] J.-L. Goffin, A. Haurie and J.-Ph. Vial. Decomposition and nondifferentiable optimization with the projective algorithm. *Management Science*, 38(2):284–302,1992.

[14] P. Hansen, N. Mladenovic and D. Perez-Brito. Variable neighborhood decomposition search. *Journal of Heuristics*, 7:335–350, 2001.

[15] H. Heitsch and W. Römisch. Scenario reduction algorithms in stochastic programming. *Computational optimization and applications*, 24:187–206, 2003.

[16] C.C. Holt, F. Modigiliani, J.F. Muth and H. Simon. *Planning Production, Inventories, and Work Force*. Prentice-Hall, New Jersey, 1960.

[17] O. Kariv and L. Hakimi. An algorithmic approach to network location problems ii: the p-medians. *SIAM Journal of Applied Mathematics*, 37(3):539–560, 1979.

[18] Y. Nesterov and J.-Ph. Vial. Confidence level solutions for stochastic programming. Tech. rep., Department of Management Studies, University of Geneva, 2000, revised 2001.

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

[20] J. Reese. Methods for solving the p-median problem: An annotated bibliography. Technical report, Department of Mathematics, Trinity University, August 11, 2005.

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

[22] A. Shapiro and A. Nemirovski. On complexity of stochastic programming problems. *Continuous Optimization: Current Trends and Applications*, pp. 111-144, V. Jeyakumar and A.M. Rubinov (Eds.), Springer, 2005.

[23] J. Thénié, C. van Delft and J.-Ph. Vial. Automatic formulation and decomposition of stochastic programs via algebraic modeling languages. to appear in *Computational Management Science*, 2006.

[24] C. van Delft and J.-Ph. Vial. A practical implementation of stochastic programming: an application to the evaluation of option contracts in supply chains. *Automatica*, 40:743–756, 2004.