On conditional cuts for Stochastic Dual Dynamic Programming

W. van Ackooij[a,*], X. Warin[a]

[a]*EDF R&D. OSIRIS; 7, Boulevard Gaspard Monge ; F-91120 Palaiseau Cedex France.*

Abstract

Multi stage stochastic programs arise in many applications from engineering whenever a set of inventories or stocks has to be valued. Such is the case in seasonal storage valuation of a set of cascaded reservoir chains in hydro management. A popular method is Stochastic Dual Dynamic Programming (SDDP), especially when the dimensionality of the problem is large and Dynamic programming no longer an option. The usual assumption of SDDP is that uncertainty is stage-wise independent, which is highly restrictive from a practical viewpoint. When possible, the usual remedy is to increase the state-space to account for some degree of dependency. In applications this may not be possible or it may increase the state space by too much. In this paper we present an alternative based on keeping a functional dependency in the SDDP - cuts related to the conditional expectations in the dynamic programming equations. Our method is based on popular methodology in mathematical finance, where it has progressively replaced scenario trees due to superior numerical performance. On a set of numerical examples, we too show the interest of this way of handling dependency in uncertainty, when combined with SDDP. Our method is readily available in the open source software package StOpt.

Keywords: SDDP algorithm, Stochastic optimization, Nonsmooth optimization, Conditional expectations
2010 MSC: 90C15, 65C35

1. Introduction

Dealing with uncertainty is vital in many real-life applications. An interesting way to formalize such a setting is a multistage stochastic program wherein one also accounts for the possibility of acting on past observed uncertainty. These models are quite popular in areas such energy [1–5], transportation [6, 7] and finance [8–10]. The usual underlying assumption is that uncertainty can somehow be presented or approximated using a scenario tree. The resulting mathematical programming problem is generally large-scale and non-trivial or impossible to solve using a monolithic method. The use of specialized algorithms that employ decomposition techniques (and very often sampling) appear crucial

*Corresponding author
Email addresses: wim.van-ackooij@edf.fr (W. van Ackooij), xavier.warin@edf.fr (X. Warin)

for an efficient numerical resolution. In this family two popular approaches are *Nested Decomposition* – ND – of [11] and *Stochastic Dual Dynamic Programming* – SDDP – of [12]. Mathematically speaking, both methods are close cousins, in that they approximate the cost-to-go functions (resulting from dynamic programming) using piecewise linear approximations, defined by cutting planes computed by solving linear programs in a so-called *backward step*.

The difference between the ND and SDDP algorithm resides in how uncertainty is handled when trying to establish an upper bound on the optimal value. The ND algorithm will use the entire scenario tree, whereas the SDDP algorithm will employ sampling procedures in order to achieve this. The ND algorithm therefore does not require any particular assumption on the scenario tree, but as a consequence is typically only applied to multistage stochastic problems of moderate size (with some hundred or a few thousand scenarios). In order to mitigate the effect that only "smaller" trees can be handled, methods for constructing representative but small scenario trees have therefore received significant attention. Let us mention the pioneering work [13] on two-stage programs and some subsequent extensions to the multistage case: e.g., [14–18]. Other ideas to reduce the computational burden rely on combining adaptive partitioning of scenarios with regularization techniques from convex optimization, e.g., [19, 20]. These ideas can also be extended to the multistage setting as done recently in [21].

It is therefore generally acknowledged that SDDP can handle larger scenario trees by combining decomposition and sampling, [12]. This method is highly popular as illustrated by the numerous applied papers, e.g., [1, 3, 7, 14, 22, 23]. The method however requires the assumption that the underlying stochastic process is stagewise independent (see also [24–26] for other methods employing sampling). Combined with a possibility to share cuts among different nodes of the scenario tree, the optimization strategies proposed in these references mitigate the curse of dimensionality further. However, the assumption that uncertainty is stagewise independent is quite a strong assumption when familiar with the actual observation of data. Whenever the underlying stochastic process is Markovian, it is generally suggested to increase the dimension of the state vector in order to revert back to a stagewise independence assumption (c.f., the discussion in [27]). As observed in [28] such an increase may be detrimental to computational efficiency especially if the increase in the dimension of the state vector is significant. The authors [28] suggest a way to partially mitigate this effect. In this paper we suggest another approach for efficiently estimating the conditional expectations appearing in the dynamic programming equations. The suggested method does not require increasing the size of the state vector whenever the underlying stochastic process is not stagewise independent. Our approach of estimating the conditional expectation is based on the observation that such expectations are orthogonal projections onto an appropriate functional space. The method suggested here is very popular in mathematical finance since the first work of Tsilikis and Van Roy [29] and the closely related Longstaff Schwarz method [30]. The latter has become the most used method to deal with optimal stopping problem, i.e., the optimization problem related to computing the optimal stopping time. The optimal stopping time is the best moment to exercise a given option. In the banking system, this method has become the reference method due to its easy implementation, its efficiency in moderate dimensions, and the easiness to calculate the sensibility of the optimal value (the "greeks"). In Energy market, quants use this method to valuate and hedge their

portfolio (see [31] for an example of a gas storage hedging simulation).

In a recent paper, Bouchard and Warin [32] developed a variant of this method based on linear regression on adaptive meshes: they showed that this method was superior to the original Longstaff-Schwarz method based on global polynomial regression by avoiding oscillations of the regressor (the Runge effect). Indeed, the method was compared to different methods to price American options and to estimate the sensitivity of the options to the initial prices (the delta of the option) in dimension 1 to 6. In dimension 3 or above, the proposed regression method appeared to be clearly superior to optimal quantization [33–36] and the Malliavin approach [37, 38]. Finally let us mention that the here suggested regression methods are also used to approximate conditional expectations in numerical schemes [39] used to solve Backward Stochastic Differential Equations (BSDE). The latter BSDEs provide a way to solve quasi-linear equations: the first papers [40, 41] have given birth to many follow up papers on the topic (see for example the references in the recent work of [42]). One can also use these building block to design schemes to solve full nonlinear equations [43].

To conclude the introduction, let us briefly mention that the mathematical properties of the SDDP algorithm have been extensively investigated. A first formal proof of almost-sure convergence of multistage sampling algorithms akin to SDDP is due to Chen and Powell [24]. This proof was extended by Linowsky and Philpott to cover the SDDP algorithm and other sampling-based methods in [44]. The convergence analysis of SDDP was revisited in [45], and more recently in [27]. Recent publications have addressed the issue of incorporating risk-averse measures into the SDDP algorithm: [3, 22, 27, 46–49]. Polyhedral risk measures were studied in [50] and [46]. As mentioned in [3], theoretical foundations for a risk averse approach based on conditional risk mappings were developed in [51]. It is shown is [27] how to incorporate convex combinations of the expectation and *Average Value-at-Risk* into the SDDP algorithm. Numerical experiments on this idea have been reported in many publications; see for instance [22] and [3].

This paper is organized as follows. Section 2 presents the general structure of multistage stochastic linear programs and show how the key difficulty resides in computing conditional expectations. We present the general idea of approximating conditional expectations in Section 3 and our algorithm in section 3.2. Finally Section 4 contains a series of numerical experiments showing numerical consistency of the suggested method. We also compare the method with other variants, wherein it appears that the suggested approach is preferable. The paper ends with concluding remarks and research perspectives.

2. Preliminaries on multistage stochastic linear programs and decomposition

Multistage stochastic programs explicitly model a series of decisions interspaced with the partial observation of uncertainty. If the given set of possible realizations of the underlying stochastic process is discrete, uncertainty can be represented by a scenario tree. Multistage stochastic linear programs (MSLPs) are a special case of this class wherein the underlying modelling structure is linear/affine. Hence, on a scenario tree the resulting model is a very large linear program. In this section we will provide an overview of two popular decomposition approaches for MSLPs, namely nested decomposition and

stochastic dual dynamic programming. From an abstract viewpoint coming from non-linear non-smooth optimization, both methods are variants of Kelley's cutting plane method [52].

Consider the following multistage stochastic linear program:

$$\min_{\substack{x_1 \in X_1 \\ A_1 x_1 = b_1}} c_1^\top x_1 + \mathbb{E}_{|\xi_1} \left[\min_{\substack{x_2 \in X_2 \\ B_2 x_1 + A_2 x_2 = b_2}} c_2^\top x_2 + \mathbb{E}_{|\xi_{[2]}} \left[\cdots + \mathbb{E}_{|\xi_{[T-1]}} \left[\min_{\substack{x_T \in X_T \\ B_T x_{T-1} + A_T x_T = b_T}} c_T^\top x_T \right] \right] \right], \tag{1}$$

where some of (or all) data $\xi = (c_t, B_t, A_t, b_t)$ can be subject to uncertainty for $t = 2, \ldots, T$. The expected value $\mathbb{E}_{|\xi_{[t]}}[\,\cdot\,]$ is taken with respect to the conditional probability measure of the random vector $\xi_t \in \Xi_t \subseteq \mathbb{R}^{m_t}$, defining the stochastic process $\{\xi_t\}_{t=1}^T$. Furthermore, $X_t \neq \emptyset$, $t = 1, \ldots, T$, are polyhedral convex sets that do not depend on the random parameters, which we denote by $X_t := \{x_t \in \mathbb{R}^{n_t}_+ \mid D_t x_t = d_t\}$, for an appropriate matrix D_t and vector d_t.

For numerical tractability, we assume that the number N of realizations (scenarios) of the data process is finite, i.e., support sets Ξ_t ($t = 1, \ldots, T$) have finite cardinality. This is, for instance, the case in which (1) is a SAA approximation of a more general MSLP problem (having continuous probability distribution). For a discussion of the relation of such a problem with the underlying true problem relying on a continuous distribution we refer to [27].

Under these assumptions, the dynamic programming equations for problem (1) take the form

$$Q_t(x_{t-1}, \xi_t) := \begin{cases} \min\limits_{x_t \in \mathbb{R}^{n_t}} & c_t^\top x_t + \mathcal{Q}_{t+1}(x_t)(\xi_t) \\ \text{s.t.} & A_t x_t = b_t - B_t x_{t-1} \\ & x_t \in X_t\,, \end{cases} \tag{2}$$

where

$$\mathcal{Q}_{t+1}(x_t)(\xi_t) := \mathbb{E}_{|\xi_{[t]}}[Q_{t+1}(x_t, \xi_{t+1})]\,, \quad \text{for } \; t = T-1, \ldots 1\,, \tag{3}$$

and $\mathcal{Q}_{T+1}(x) \equiv 0$ by definition. The first-stage problem becomes

$$\begin{cases} \min\limits_{x_1 \in \mathbb{R}^{n_1}} & c_1^\top x_1 + \mathcal{Q}_2(x_1)(\xi_1) \\ \text{s.t.} & A_1 x_1 = b_1 \\ & x_1 \in X_1\,. \end{cases} \tag{4}$$

An implementable policy for (1) is a collection of functions $\bar{x}_t = \bar{x}_t(\xi_{[t]})$, $t = 1, \ldots, T$. Such a policy gives a decision rule at every stage t of the problem based on a realization of the data process up to time t. A policy is feasible for problem (1) if it satisfies all the constraints for every stage t.

As in [27], we assume that the *cost-to-go* functions $\mathcal{Q}_t$ are finite valued, in particular we assume *relatively complete recourse*. Since the number of scenarios is finite, the cost-to-go functions are convex piecewise linear functions [53, Chap. 3].

We note that the stagewise independence assumption (e.g., as in [12]) simplifies (3) to $\mathcal{Q}_{t+1}(x_t)(\xi_t) = \mathcal{Q}_{t+1}(x_t) = \mathbb{E}[Q_{t+1}(x_t, \xi_{t+1})]$ since the random vector ξ_{t+1} is independent of its history $\xi_{[t]} = (\xi_1, \ldots, \xi_t)$. As pointed out in [27], in some cases stagewise dependence

can be dealt with by adding state variables to the model. We will provide an alternative to that general methodology. Prior to doing so, let us briefly mention the main steps of ND and SDDP.

2.1. Decomposition

As already mentioned, two very important decomposition techniques for solving multistage stochastic linear programs are Nested Decomposition (see [54]), and the SDDP algorithm (see [12]). Both methods have two main steps:

- *Forward step*, that goes from stage $t = 1$ up to $t = T$ solving subproblems to define feasible policies $\bar{x}_t(\xi_{[t]})$. In this step an (estimated) upper bound $\overline{z}$ for the optimal value is determined.

- *Backward step*, that comes from stage $t = T$ up to $t = 1$ solving subproblems to compute linearizations that improve the cutting-plane approximations for the cost-to-go functions $\mathcal{Q}_t$. In this step a lower bound $\underline{z}$ is obtained.

Below we discuss these steps, starting with the backward one.

Backward step.. Let $\bar{x}_t = \bar{x}_t(\xi_{[t]})$ be a trial decision at stage $t = 1, \ldots, T-1$, and $\check{\mathcal{Q}}_t$ be a current approximation of the cost-to-go function $\mathcal{Q}_t$, $t = 2, \ldots, T$, given by the maximum of a collection of cutting planes. At stage $t = T$ the following problem is solved

$$\underline{Q}_T(\bar{x}_{T-1}, \xi_T) = \left\{ \begin{array}{ll} \min\limits_{x_T \in \mathbb{R}^{n_T}} & c_T^\top x_T \\ \text{s.t.} & A_T x_T = b_T - B_T \bar{x}_{T-1} \\ & x_T \in X_T \end{array} \right. \tag{5}$$

for all $\xi_T = (c_T, B_T, A_T, b_T) \in \Xi_T$. Let $\bar{\pi}_T = \bar{\pi}_T(\xi_T)$ be an optimal dual solution of problem (5). Then $\alpha_T(\xi_{T-1}) := \mathbb{E}_{|\xi_{T-1}}[b_T^\top \bar{\pi}_T]$ and $\beta_T(\xi_{T-1}) := -\mathbb{E}_{|\xi_{T-1}}[B_T^\top \bar{\pi}_T] \in \partial \mathcal{Q}_T(\bar{x}_{T-1})(\xi_{T-1})$ define the linearization

$$\begin{aligned} q_T(x_{T-1})(\xi_{T-1}) &:= \beta_T^\top(\xi_{T-1}) x_{T-1} + \alpha_T(\xi_{T-1}) \\ &= \quad \mathcal{Q}_T(\bar{x}_{T-1})(\xi_{T-1}) + \langle \beta_T(\xi_{T-1}), x_{T-1} - \bar{x}_{T-1} \rangle \,, \end{aligned}$$

satisfying

$$\mathcal{Q}_T(x_{T-1}) \geq q_T(x_{T-1}) \quad \forall\, x_{T-1} \,,$$

and $\mathcal{Q}_T(\bar{x}_{T-1})(\xi_{T-1}) = q_T(\bar{x}_{T-1})(\xi_{T-1})$, i.e., q_T is a supporting plane for $\mathcal{Q}_T$ (both functions of ξ_{T-1}). This linearization is added to the collection of supporting planes of $\mathcal{Q}_T$: $\check{\mathcal{Q}}_T$ is replaced by $\check{\mathcal{Q}}_T(x_{T-1})(\xi_{T-1}) := \max\{\check{\mathcal{Q}}_T(x_{T-1})(\xi_{T-1}), q_T(x_{T-1})(\xi_{T-1})\}$. In other words, the cutting-plane approximation $\check{\mathcal{Q}}_T$ is constructed from a collection J_T of linearizations:

$$\check{\mathcal{Q}}_T(x_{T-1})(\xi_{T-1}) = \max_{j \in J_T} \{ \beta_T^{j\top}(\xi_{T-1}) x_{T-1} + \alpha_T^j(\xi_{T-1}) \} \,.$$

By starting with a model approximating the cost-to-go function from below (e.g. $\mathcal{Q}_t \equiv -\infty$ for all stages), the cutting-plane updating strategy will ensure that $\check{\mathcal{Q}}_T \leq \mathcal{Q}_T$. The

updated model $\check{\mathcal{Q}}_T$ is then used at stage $T-1$, and the following problem needs to be solved for all $t = T-1, \ldots, 2$:

$$\begin{aligned}\underline{Q}_t(\bar{x}_{t-1}, \xi_t) = &\left\{\begin{array}{ll}\min\limits_{x_t \in \mathbb{R}^{n_t}} & c_t^\top x_t + \check{\mathcal{Q}}_{t+1}(x_t)(\xi_t) \\ \text{s.t.} & A_t x_t = b_t - B_t \bar{x}_{t-1} \\ & x_t \in X_t\end{array}\right. \\ \equiv &\left\{\begin{array}{ll}\min\limits_{x_t, r_{t+1} \in \mathbb{R}^{n_t} \times \mathbb{R}} & c_t^\top x_t + r_{t+1} \\ \text{s.t.} & A_t x_t = b_t - B_t \bar{x}_{t-1} \\ & \beta_{t+1}^{j\top}(\xi_t) x_t + \alpha_{t+1}^j(\xi_t) \le r_{t+1}, \quad j \in J_{t+1} \\ & x_t \in X_t\,.\end{array}\right.\end{aligned} \tag{6}$$

Let $\bar{\pi}_t = \bar{\pi}_t(\xi_t)$ and $\bar{\rho}_t^j = \bar{\rho}_t^j(\xi_t)$ be optimal Lagrange multipliers associated with the constraints $A_t x_t = b_t - B_t \bar{x}_{t-1}$ and $\beta_{t+1}^{j\top}(\xi_{t-1}) x_t + \alpha_{t+1}^j(\xi_{t-1}) \le r_{t+1}$, respectively. Then the linearization

$$q_t(x_{t-1})(\xi_{t-1}) := \beta_t^\top(\xi_{t-1}) x_{t-1} + \alpha_t(\xi_{t-1}) = \mathbb{E}_{|\xi_{t-1}}[\underline{Q}_t(\bar{x}_{t-1}, \xi_t)] + \langle \beta_t(\xi_{t-1}), x_{t-1} - \bar{x}_{t-1} \rangle$$

of $\mathcal{Q}_t$ is constructed with

$$\alpha_t(\xi_{t-1}) := \mathbb{E}_{|\xi_{t-1}}[b_t^\top \bar{\pi}_t + \sum_{j \in J_{t+1}} \alpha_{t+1}^j(\xi_t) \bar{\rho}_t^j] \quad \text{and} \quad \beta_t(\xi_{t-1}) := -\mathbb{E}_{|\xi_{t-1}}[B_t^\top \bar{\pi}_t] \tag{7}$$

and satisfies $\mathcal{Q}_t(x_{t-1})(\xi_{t-1}) \ge q_t(x_{t-1})(\xi_{t-1}) \;\; \forall\, x_{t-1}$.

Once the above linearization is computed, the cutting-plane model at stage t is updated: $\check{\mathcal{Q}}_t(x_{t-1})(\xi_{t-1}) = \max\{\check{\mathcal{Q}}_t(x_{t-1})(\xi_{t-1}), q_t(x_{t-1})(\xi_{t-1})\}$. Since $\check{\mathcal{Q}}_{t+1}$ can be a rough approximation (at early iterations) of $\mathcal{Q}_{t+1}$, the linearization q_t is not necessarily a supporting plane (but a cutting plane) for $\mathcal{Q}_{t+1}$: the inequality $q_t \le \mathcal{Q}_t$ might be strict for all x_{t-1} feasible (at the first iterations).

At the first stage, the following LP is solved

$$\begin{aligned}\underline{z} = &\left\{\begin{array}{ll}\min\limits_{x_1 \in \mathbb{R}^{n_1}} & c_1^\top x_1 + \check{\mathcal{Q}}_2(x_1)(\xi_1) \\ \text{s.t.} & A_1 x_1 = b_1 \\ & x_1 \in X_1\end{array}\right. \\ \equiv &\left\{\begin{array}{ll}\min\limits_{x_1, r_2 \in \mathbb{R}^{n_1} \times \mathbb{R}} & c_1^\top x_1 + r_2 \\ \text{s.t.} & A_1 x_1 = b_1 \\ & \beta_2^{j\top}(\xi_1) x_1 + \alpha_2^j(\xi_1) \le r_2, \quad j \in J_2 \\ & x_1 \in X_1\,.\end{array}\right.\end{aligned} \tag{8}$$

The value $\underline{z}$ is a lower bound for the optimal value of (1). The computed cutting-plane models $\check{\mathcal{Q}}_t$, $t = 2, \ldots, T$, and a solution $\bar{x}_1$ of problem (8) can be used for constructing an implementable policy as follows.

Forward step.. Given a scenario $\xi = (\xi_1, \ldots, \xi_T) \in \Xi_1 \times \ldots \times \Xi_T$ (realization of the stochastic process), decisions $\bar{x}_t = \bar{x}_t(\xi_{[t]})$, $t = 1, \ldots, T$, are computed recursively going

forward with $\bar{x}_1$ being a solution of (8), and $\bar{x}_t$ being an optimal solution of

$$\left\{\begin{array}{ll}\min\limits_{x_t\in\mathbb{R}^{n_t}} & c_t^\top x_t + \check{\mathcal{Q}}_{t+1}(x_t)(\xi_t)\\ \text{s.t.} & A_t x_t = b_t - B_t\bar{x}_{t-1}\\ & x_t \in X_t\end{array}\right. \equiv \left\{\begin{array}{ll}\min\limits_{x_t, r_{t+1}} & c_t^\top x_t + r_{t+1}\\ \text{s.t.} & A_t x_t = b_t - B_t\bar{x}_{t-1}\\ & \beta_{t+1}^{j\top}(\xi_t)x_t + \alpha_{t+1}^j(\xi_t) \le r_{t+1}, \quad j\in J_{t+1}\\ & x_t \in X_t \subseteq \mathbb{R}^{n_t}, r_{t+1}\in\mathbb{R}\,.\end{array}\right. \tag{9}$$

for all $t = 2,\ldots,T$, with $\check{\mathcal{Q}}_{T+1} \equiv 0$. Notice that $\bar{x}_t$ is a function of $\bar{x}_{t-1}$ and $\xi_t = (c_t, A_t, B_t, b_t)$, i.e., $\bar{x}(\xi_{[t]})$ is a feasible and implementable policy for problem (1) (up to stage t). As a result, the value

$$\overline{z} = \mathbb{E}\left[\sum_{t=1}^{T} c_t^\top \bar{x}_t(\xi_{[t]})\right] \tag{10}$$

is an upper bound for the optimal value of (1) as long as all the scenarios $\xi \in \Xi$ are considered for computing the policy. This is the case of nested decomposition. However, the forward step of the SDDP algorithm consists in taking a sample $\mathcal{J}$ with $M < N$ scenarios ξ^j of the data process and computing $\bar{x}_t(\xi_{[t]}^j)$ and the respective values $z(\xi^j) = \sum_{t=1}^T c_t^\top \bar{x}_t(\xi_{[t]}^j)$, $j = 1,\ldots,M$. The sample average $\tilde{z} = \mathbb{E}_{|\mathcal{J}}[z(\xi)]$ and the sample variance $\tilde{\sigma}^2 = \mathbb{E}_{|\mathcal{J}}[(z(\xi^j) - \tilde{z})^2$ are easily computed. The sample average is an unbiased estimator of the expectation (10) (that is an upper bound for the optimal value of (1)). In the case of subsampling, $\tilde{z} + 1.96\tilde{\sigma}/\sqrt{M}$ gives an upper bound for the optimal value of (1) with confidence of about 95%. As a result, a possible stopping test for the SDDP algorithm is $\tilde{z} + 1.96\tilde{\sigma}/\sqrt{M} - \underline{z} \le \epsilon$, for a given tolerance $\epsilon > 0$. We refer to [27, Sec.3] for a discussion on this subject.

3. On Conditional cuts in SDDP

The assumption that the random data process is stagewise independent is useful in order to simplify the expressions given in section 2 such as (7). Essentially it deletes the additional dependency on the stochastic process of the cuts. This has a clear advantage in terms of storage, but the stagewise independence is also unrealistic in many situations. We will present here an alternative method for dealing with the conditional expectations that has the following advantages:

- There is no need to explicitly set up a scenario tree when initial data is available in the form of a set of scenarios $\{(\xi_1^s, ..., \xi_T^s)\}_{s\in\mathcal{S}}$.
- There is no need to increase the dimension of the state-vector in order to account for stagewise dependency.

However our underlying assumption is that the random process is Markovian, i.e., satisfies:

$$\xi_{t+1} = f(\xi_t, \eta_t), \tag{11}$$

where the random data or innovation process $\eta_t \in \mathbb{R}^{p_t}$ is independent of $\xi_{[t]}$ and f is a given map defined on $(\mathbb{R}^{m_t} \times \mathbb{R}^{p_t})$ with values in $\mathbb{R}^{m_{t+1}}$.

3.1. Dealing with conditional expectations - Linear Regression

One of the difficulties in solving problem (1) is closely tied in with evaluating conditional expectations. Indeed, e.g., equation (7) makes it apparent how this introduces an additional dependency with respect to the situation wherein only expectations are evaluated. The easiest way of computing conditional expectations is by setting up a tree or Markov chain to describe the evolution of ξ_t and employ the transition probabilities to simply evaluate the conditional expectation. However, this approach, when only historical data is available, leads to modelization errors by introducing an artificial representation of the underlying process in the form of a tree when only a set of samples of this process was available. Although the statistical quality of such a representation can be studied, we rather suggest to circumvent it altogether by directly employing a Monte Carlo approach both in the *forward* and *backward* steps.

We suppose that in the *backward* step the process is sampled through the Markovian dynamic of equation (11) once and for all with S samples ξ_t^k with $k = 1, ..., S$. These samples could simply be the historical data supposed to be Markovian.

It is well known that for any given t the conditional expectation $\mathbb{E}_{|\xi_t}(.)$ is an orthogonal projection in the space of square integrable function so we will suppose in the sequel that all functions g of $\xi = \{\xi\}_{t>0}$ of which we want estimate the conditional expectation satisfy $\mathbb{E}[g(\xi_t)^2] < \infty$ for $t \leq T$: such functions g are said to be bounded in L_2. As a consequence, for any map $g : \Xi_{t+1} \to \mathbb{R}$, the conditional expectation satisfies $\mathbb{E}_{|\xi_t}(g(\xi_{t+1})) = F(\xi_t)$ for a certain function $F : \Xi_t \to \mathbb{R}$. In order to keep the computational burden manageable as well as to limit storage requirements, we will assume that F can be represented as a linear combination of a given set of base-functions. For instance, one could take Chebyshev or Legendre polynomials known to be an orthonormal basis for L_2 bounded functions as was originally proposed by [30].

Now for a fixed t and given set $\psi_{t,1}, \ldots, \psi_{t,P_t}$ of base functions the map F can be approximated with the help of S drawn samples by (linearly) regressing $\left\{g(\xi_{t+1}^s)\right\}_{s=1}^S$ on $\{(\psi_{t,1}(\xi_t^s), ..., \psi_{t,P_t}(\xi_t^s))\}_{s=1}^S$. This gives the following approximation:

$$\hat{\mathbb{E}}^S(g(\xi_{t+1}) \mid \xi_t) = \sum_{i=1}^{P_t} \alpha_{t,i}^* \psi_{t,i}(\xi_t), \tag{12}$$

where $\alpha_{t,1}^*, ..., \alpha_{t,P_t}^*$ is the optimal solution to the problem

$$\min_{\alpha_1,\ldots,\alpha_{P_t}} \sum_{s=1}^{S} \left(g(\xi_{t+1}^s) - \sum_{i=1}^{P_t} \alpha_i \psi_{t,i}(\xi_t^s) \right)^2.$$

Since the choice of an appropriate well performing basis (uniformly in t) and for a given set of instances of (1) may be tricky, we will consider instead the adaptive method developed in [32]. It relies on setting up a class of local base functions. In order to present the idea, let us denote with $\xi_{t,i}^s$ the ith coordinate of sample s of $\xi_t \in \mathbb{R}^{m_t}$, where $i = 1, ..., m_t$. Now taking coordinate-wise extrema over the set of scenarios we define for each $i = 1, ..., m_t$, $\bar{\xi}_{t,i} = \max_{s=1,\ldots,S} \xi_{t,i}^s$, $\underline{\xi}_{t,i} = \min_{s=1,\ldots,S} \xi_{t,i}^s$. As shown in Figure 1, we now partition the set $\prod_{i=1}^{m_t} [\underline{\xi}_{t,i}, \bar{\xi}_{t,i}]$ in a set of hypercubes $D_{t,\ell}$, $\ell = 1, ..., L_t$. For each

given $\ell = 1, ..., L_t$, i.e., on each hypercube, we pick a family of mappings $\{\psi_q\}_{q\geq 0}$ having support on $D_{t,\ell}$. Typically the family $\{\psi_q\}_{q\geq 0}$ will be that of all monomials in $\mathbb{R}^{m_t}$, implying that $m_t + 1$ coefficients need to be stored for each element $\ell = 1, ..., L_t$ of the partition. We emphasize that the approximation is nonconforming in the sense that we do not assure the continuity of the approximation. However, it has the advantage to be able to fit any, even discontinuous, function. The total number of degrees of freedom is equal to $(1 + m_t)L_t$ which quality-wise should be related to the sample size S. Hence, in order to avoid oscillations, the partition is set up so that each element roughly contains the same number of samples. Using this approximation, the normal equation is always well conditioned leading to the possibility to use the Choleski method, which, as shown in [32], is most efficient for solving the regression problem. We even note that the problem can be decomposed in L_t regressions with $m_t \times m_t$ regression matrices. Using a partial sort algorithm in each direction it is then possible to get roughly the same number of samples in each cell as explained in [32]. The use of this partitioning procedure is illustrated on Figure 1.

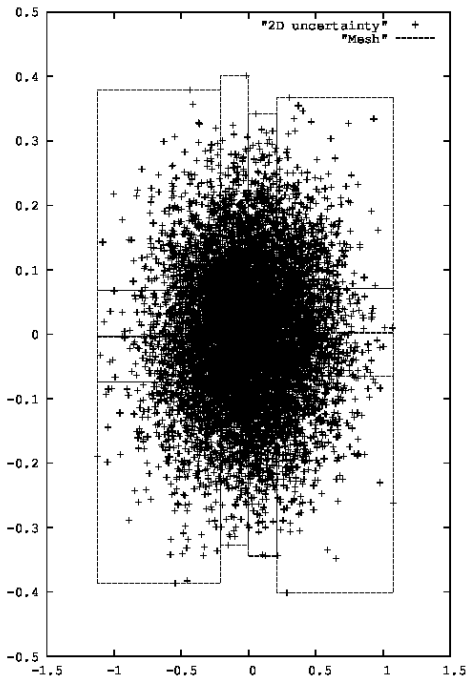


Figure 1: Example of the support of the local basis functions in dimension 2. Here 4 intervals are chosen for each direction giving a total of $L = 16$ hypercubes.

As explained in the introduction, the numerical results presented in [32] demonstrate the superiority of this local method with respect to various competing ways of estimating conditional expectations such as quantization or Malliavin regression. We note that a constant per mesh approximation can be used instead of a linear one, giving a far less efficient method if the functions to approximate are regular or piecewise regular.

3.2. The SDDP algorithm with conditional cuts and local base functions

In this section we provide the description of our variant of the SDDP algorithm relying on conditional cuts and using local base functions. A variant using only global base functions is readily extracted. In the algorithm below, we add the index s to emphasize the dependency of the data on the scenario. Let us also note that for a given current trial decision $\bar{x}_t = \bar{x}_t(\xi_{[t]})$ at stage t, that this can be associated with a given mesh $D_{t,h(t)}$, for $h(t) \in \{1, ..., L_t\}$. The value $h(t)$ is such that $\xi_t \in D_{t,h(t)}$. We will emphasize this by speaking of the trial decision $(\bar{x}_t, h(t))$.

Backward step.. Let $(\bar{x}_t, h(t))$ be a trial decision at stage $t = 1, \ldots, T-1$ and $\check{\mathcal{Q}}_t$ be a current approximation of the cost-to-go function $\mathcal{Q}_t$, $t = 2, \ldots, T$, given by the maximum of a collection of (functional) cutting planes. At stage $t = T$ the following problem is solved for all s such that $\xi_{T-1}^{(s)} \in D_{T-1,h(T-1)}$

$$\underline{Q}_T(\bar{x}_{T-1}, \xi_T^{(s)}) = \left\{ \begin{array}{ll} \min\limits_{x_T \in \mathbb{R}^{n_T}} & (c_T^{(s)})^\top x_T \\ \text{s.t.} & A_T^{(s)} x_T = b_T^{(s)} - B_T^{(s)} \bar{x}_{T-1} \\ & x_T \in X_T \end{array} \right. \tag{13}$$

Let $\bar{\pi}_T^{(s)}$ be an optimal dual solution of problem (13). Now let $\alpha_T(y)$, $\beta_T(y)$ be the numerically approximated conditional expectations of $b_T^\top \bar{\pi}_T$ and $B_T^\top \bar{\pi}_T$ respectively while employing the estimates of (12). We will denote this as $\alpha_T(y) := \hat{\mathbb{E}}^S[b_T^\top \bar{\pi}_T | \xi_{T-1} = y]$ and $\beta_T(y) := -\hat{\mathbb{E}}^S[B_T^\top \bar{\pi}_T | \xi_{T-1} = y]$.

Now define the linearization

$$\begin{aligned} q_T(x_{T-1})(\xi_{T-1}) &:= \beta_T^\top(\xi_{T-1}) x_{T-1} + \alpha_T(\xi_{T-1}) \\ &= \hat{\mathbb{E}}^S \left[\underline{Q}_T(\bar{x}_{T-1}, \xi_T) | \xi_{T-1} \right] + \langle \beta_T(\xi_{T-1}), x_{T-1} - \bar{x}_{T-1} \rangle. \end{aligned}$$

This linearization is used to update the current cutting plane model as follows:

$$\check{\mathcal{Q}}_T(x_{T-1})(\xi_{T-1}) := \max\{\check{\mathcal{Q}}_T(x_{T-1})(\xi_{T-1}), q_T(x_{T-1})(\xi_{T-1})\}.$$

We note that for $\xi \notin D_{T-1,h(T-1)}$, $q_T(x_{T-1})(\xi) = 0$ so that the model is only locally updated. We will emphasize this by letting $J_{t+1}(D_{t,h(t)})$ be the index set of linearization added while exploring mesh $D_{t,h(t)}$.

The updated model $\check{\mathcal{Q}}_T$ is then used at stage $T-1$, and the following problem needs to be solved for all $t = T-1, \ldots, 2$ and every s such that $\xi_{t-1}^{(s)} \in D_{t-1,h(t-1)}$:

$$\underline{Q}_t(\bar{x}_{t-1}, \xi_t^{(s)}) = \left\{ \begin{array}{ll} \min\limits_{x_t \in \mathbb{R}^{n_t}} & (c_t^{(s)})^\top x_t + \check{\mathcal{Q}}_{t+1}(x_t)(\xi_t^{(s)}) \\ \text{s.t.} & A_t^{(s)} x_t = b_t^{(s)} - B_t^{(s)} \bar{x}_{t-1} \\ & x_t \in X_t \end{array} \right. \tag{14}$$

$$\equiv \left\{ \begin{array}{ll} \min\limits_{x_t, r_{t+1} \in \mathbb{R}^{n_t} \times \mathbb{R}} & (c_t^{(s)})^\top x_t + r_{t+1}^{(s)} \\ \text{s.t.} & A_t^{(s)} x_t = b_t^{(s)} - B_t^{(s)} \bar{x}_{t-1} \\ & \beta_{t+1}^{j\top}(\xi_t^{(s)}) x_t + \alpha_{t+1}^j(\xi_t^{(s)}) \le r_{t+1}^{(s)}, \quad j \in J_{t+1}(D_{t,h(t)}) \\ & x_t \in X_t \,. \end{array} \right. \tag{15}$$

Let $\bar{\pi}_t^{(s)}$ and $\left(\bar{\rho}_t^j\right)^{(s)}$ be optimal Lagrange multipliers associated with the constraints $A_t^{(s)}x_t = b_t^{(s)} - B_t^{(s)}\bar{x}_{t-1}$ and $\beta_{t+1}^{j\top}(\xi_t^{(s)})x_t + \alpha_{t+1}^j(\xi_t^{(s)}) \leq r_{t+1}^{(s)}$, respectively. Then the linearization defined for $\xi_{t-1} \in D_{t-1,h(t-1)}$:

$$\begin{aligned} q_t(x_{t-1})(\xi_{t-1}) &:= \beta_t^\top(\xi_{t-1})x_{t-1} + \alpha_t(\xi_{t-1}) \\ &= \hat{\mathbb{E}}^S[\underline{Q}_t(\bar{x}_{t-1}, \xi_t)|\xi_{t-1}] + \langle \beta_t(\xi_{t-1}), x_{t-1} - \bar{x}_{t-1}\rangle \end{aligned}$$

is constructed with

$$\alpha_t(\xi_{t-1}) := \hat{\mathbb{E}}^S[b_t^\top \bar{\pi}_t + \sum_{j\in J_{t+1}} \alpha_{t+1}^j \bar{\rho}_t^j|\xi_{t-1}] \quad \text{and} \quad \beta_t(\xi_{t-1}) := -\hat{\mathbb{E}}^S[B_t^\top \bar{\pi}_t|\xi_{t-1}]. \tag{16}$$

Once the above linearization is computed, the cutting-plane model at stage t is updated At the first stage, the following problem is solved :

$$\begin{aligned} \underline{z} = \begin{cases} \min\limits_{x_1\in\mathbb{R}^{n_1}} & (c_1)^\top x_1 + \check{\mathcal{Q}}_2(x_1)(\xi_1) \\ \text{s.t.} & A_1x_1 = b_1 \\ & x_1 \in X_1 \end{cases} \\ \equiv \begin{cases} \min\limits_{x_1, r_2\in\mathbb{R}^{n_1}\times\mathbb{R}} & (c_1)^\top x_1 + r_2^{(m)} \\ \text{s.t.} & A_1x_1 = b_1 \\ & \beta_2^{j\top}(\xi_1)x_1 + \alpha_2^j(\xi_1) \leq r_2, \quad j \in J_2 \\ & x_1 \in X_1\,. \end{cases} \end{aligned} \tag{17}$$

In particular since ξ_1 is deterministic $D_{1,1}$ is the (only) degenerate mesh consisting of ξ_1.

Forward step.. The forward step is essentially unaltered. For a given scenario $\xi = (\xi_1, \ldots, \xi_T) \in \Xi_1 \times \ldots \times \Xi_T$, first calculate for each t, $h(t) \in \{1, ..., L_T\}$ such that $\xi_t \in D_{t,h(t)}$. The decisions $\bar{x}_t = \bar{x}_t(\xi_{[t]})$, $t = 1, \ldots, T$, are computed recursively going forward with $\bar{x}_1$ being a solution of (17), and $\bar{x}_t$ being an optimal solution of

$$\begin{array}{ll} \min\limits_{x_t} & c_t^\top x_t + \check{\mathcal{Q}}_{t+1}(x_t)(\xi_t) \\ \text{s.t.} & A_tx_t = b_t - B_t\bar{x}_{t-1} \\ & x_t \in X_t \subseteq \mathbb{R}^{n_t} \end{array} \equiv \begin{cases} \min\limits_{x_t, r_{t+1}} & c_t^\top x_t + r_{t+1} \\ \text{s.t.} & A_tx_t = b_t - B_t\bar{x}_{t-1} \\ & \beta_{t+1}^{j\top}(\xi_t)x_t + \alpha_{t+1}^j(\xi_t) \leq r_{t+1},\ j \in J_{t+1}(D_{t,h(t)}), \\ & x_t \in X_t \subseteq \mathbb{R}^{n_t}, r_{t+1} \in \mathbb{R}\,. \end{cases} \tag{18}$$

for all $t = 2, \ldots, T$, with $\check{\mathcal{Q}}_{T+1} \equiv 0$. The trial for the next backward recursion is defined as $(\bar{x}_t, h(t))$ so that in the next backward step cuts at $\bar{x}_t$ will only be generated for the visited meshes $D_{t,h(t)}$. We choose to pick the same stopping criteria as in any usual implementation of SDDP (see [27, Remark 1] for a discussion on this matter). The advantage of only adding cuts for visited meshes is that this mitigates the additional burden of storing additional coefficients. Moreover this cut may be only relevant locally for a given level of ξ_t.

Using the classical SDDP method when the random data process is stagewise independent, the number of samples used in the forward path is generally chosen equal to one. However, due to accumulated errors during the backward resolution the $\check{\mathcal{Q}}_t$ estimation

based on cutting planes can be far from the corresponding true supporting planes. In the case of SDDP with conditional cuts, we want to add one cut for each mesh $D_{t,h(t)}$. Because at each date t, the probability for ξ_t to be in a given mesh is $\frac{1}{L_t}$, it is numerically better to take a number of samples equal to L_t.

Remark 1. *For a given accuracy, in the case where a constant per mesh approximation is used to estimate the conditional expectations, the number of Monte Carlo trajectories S and the number of meshes L_t taken are found to be higher than in the linear approximation. So both the backward resolution and the forward resolution are more costly with the constant per mesh approximation than with the linear approximation. We therefore suggest to use linear per mesh approximation.*

4. Numerical experiments

The SDDP method with or without conditional cuts have been implemented in StOpt [55] an open source library containing tools to solve optimization problems both in continuous or discrete time. In particular, inventory / stock problems (such as cascaded reservoir management problems, e.g., [5, 47, 56] or gas storage, e.g., [57, 58]) can be optimized using either the Dynamic Programming method or the Stochastic Dual Dynamic Programming method. The subsequent experiments were carried out using this library. All computations are achieved on a cluster composed of Intel Xeon CPU E5-2680 v4 processors whereas all linear programs (14), (17) and (18) are solved using COIN CLP version 1.15.10.

4.1. Valuing storage facing the market

We begin with a test case involving gas storage in one dimension and we will extend it artificially in higher dimensions. All experiments are achieved with two processors and 14 cores each.

4.1.1. The initial test case in one dimension.

We suppose that we want to value a storage facility where the owner has the possibility to inject gas from the market or withdraw it to the market in order to maximize his expected earnings. The price model is a classical HJM model used for example in [31]:

$$\frac{dF(T,t)}{F(T,t)} = \sigma e^{-\alpha(T-t)} dW_t \tag{19}$$

where σ is the volatility of the model, α the mean reverting, W a Brownian motion on a probability space $(\Omega, \mathcal{F}, \mathbb{P})$ endowed with the natural (completed and right-continuous) filtration $\mathbb{F} = \{\mathcal{F}_t\}_{t \leq T}$ generated by W up to some fixed time horizon $T > 0$. The spot price is naturally defined as $S_t := F(t,t)$.

We suppose that each date, $t_i = i\Delta T$ with $\Delta T = \frac{T}{N}$, $i = 0, ..., N$, the gas volume C_i has to satisfy the constraint $0 \leq C_i \leq C_{\texttt{max}}$. Noting $a_{\texttt{in}} > 0$ the maximal injection during ΔT, $a_{\texttt{out}} > 0$ the maximal withdrawal from the storage that we suppose to be constant for

simplification purposes, the gain function associated with a command $x_i \in [-a_{\texttt{out}}, a_{\texttt{in}}]$ at date $i\Delta t$ is then

$$\varphi_i(x_i) = -S_{i\Delta T} x_i,$$

where the price $S_{i\Delta T}$ is supposed to be constant between t_i and t_{i+1}. Besides the associated flow equation is given by

$$C_{i+1} = C_i + x.$$

The gain function associated with a strategy $x = \{x_i\}_{i=0,N}$ is a function depending on the initial storage value C_0, the initial spot price S_0:

$$J(C_0, x, S_0) = \sum_{i=0}^{N} \varphi_i(x_i),$$

and the optimization problem can be written as

$$J^*(C_0, S_0) = \sup_{x \in U} J(C_0, x, S_0),$$

where U is the set of non anticipative feasible strategies.

In the test case, we suppose that we want to optimize the assets on a one year ($T = 1$) time horizon with annual values $\sigma = 0.94$, $\alpha = 0.29$. The decision is taken every week so $N = 52$ and we pick the following initial forward curve:

$$F(0, i\Delta T) = 50 + 10 \sin(4\pi \frac{i}{N}). \tag{20}$$

The maximal storage capacity is $C_{\texttt{max}} = 360000$ energy units and the injection and withdrawal rates are $a_{\texttt{in}} = 60000$, $a_{\texttt{out}} = 45000$ energy units respectively. The ratios $\frac{C_{\texttt{max}}}{a_{\texttt{in}}}$, $\frac{C_{\texttt{max}}}{a_{\texttt{out}}}$ indicate a rather slow storage (see [55] for different examples of storages). At date 0 the storage facility is full.

We first value the storage facility with the dynamic programming method (see [32] for details on the methodology). We optimize the storage with 6 meshes, 20000 trajectories in optimization giving a value of $2.774e+07$ under the assumption that the control is bang-bang (i.e., either equal to $a_{\texttt{in}}$ or $a_{\texttt{out}}$). Simulating the optimal control on 100000 sample paths, we obtain an estimation of $2.784e+07$, which shows good convergence.

Next we use the SDDP method with conditional cuts described in section 3.2. To this end, we take 10000 trajectories in the backward pass to estimate the conditional expectations using $I_1 = 10$ meshes for the regressions in equation (12). We use I_1 simulations at each step of the forward pass to discover new storage volumes that will be used in the next backward pass. Every 10 SDDP iterations, the convergence of the scheme is estimated by recalculating a forward path with 20000 new trajectories. The algorithm is stopped when the relative difference between the forward and the backward iteration is lower than 0.1%. On Figure 2, we plot the convergence of the SDDP scheme with respect to the number of iterations. The global algorithm takes 210 seconds to converge using the COIN CLP solver distributing the LPs on the 28 cores. Off course the SDDP approach is not competitive with respect to the DP approach which takes 22 seconds to converge using the parallel DP solver of the StOpt library [55].

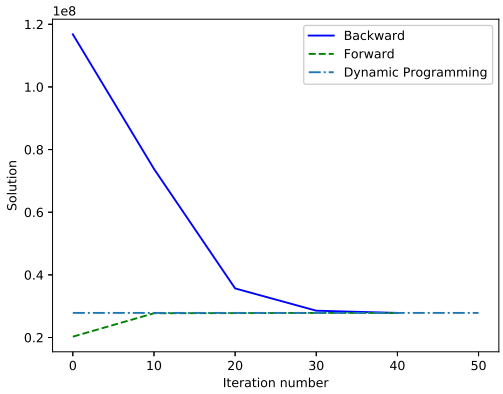
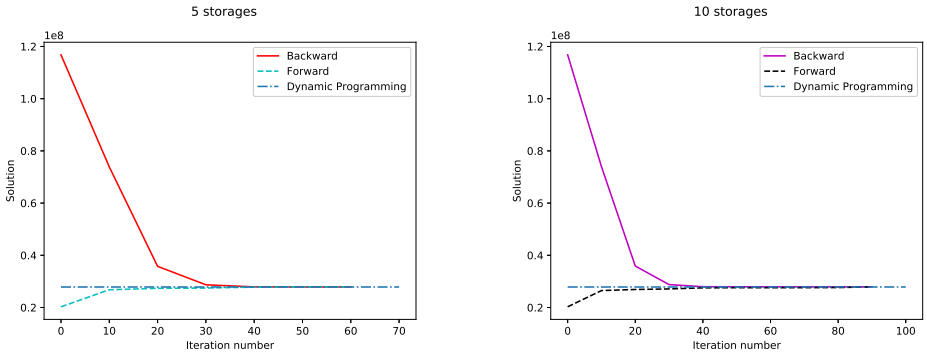


Figure 2: Comparison of optimal values in the backward and forward pass for one gas storage facility.

4.1.2. An artificial test case in higher dimension

In order to test how the method scales as a function of the dimension, we take the same data supposing that we dispose of n similar gas storages that we want to value jointly. Each transition problem is then written as a n dimensional Linear Program solved with COIN CLP. Of course, in this case, the value of n storages is simply n times the value of a single storage.

On figure 3, we plot the convergence of the SDDP scheme for $n = 5$ and $n = 10$ by giving the value of a single storage estimated as the value of n storages with the SDDP algorithm divided by n. The case with $n = 5$ takes 550 seconds to converge, while the case with $n = 10$ takes 1500 seconds. We note that a direct application of dynamic programming in this situation is simply no longer possible, whereas (our variant of) SDDP scales up very well.



(a) 5 Storage facilities

(b) 10 Storage facilities

Figure 3: Comparison of optimal values in the backward and forward pass for several gas storage facilities.

4.2. A simple storage facility problem facing a consumption constraint

In this section we compare our conditional cut approach with the classical SDDP method (i.e., increasing the state-space) for dealing with uncertainty following an AR(1) model. All experiments are once again achieved with two processors with 14 cores each.

4.2.1. A case in one dimension

In this test case we suppose that the price gas is deterministic so is given by the future price (20). The characteristics of the storage facility are the same as in section 4.1 but we suppose that we have an injection cost per unit of injected gas equal to 0.1.

The owner of the storage facility is allowed to buy gas on the market and to inject it into the storage (paying the injection cost) but is not allowed to sell it on the market. The gas withdrawn can only be used to satisfy the consumption D following an AR(1) (e.g., [59]) process between two dates $t_i = i\frac{T}{N}$ and $t_{i+1} = (i+1)\frac{T}{N}$:

$$D_{i+1} - \tilde{D}_{i+1} = \kappa_d(D_i - \tilde{D}_i) + \sigma_d \epsilon_i,$$

where ϵ_i is independent white noise with zero mean and variance 1, κ_d is set equal to 0.9, $\sigma_d = 1000$ and $\tilde{D}$ is an average consumption rate satisfying:

$$\tilde{D}_i = 22000 + 7000 \sin(4\pi \frac{i}{N}).$$

We denote with x_i^b the quantity of gas bought at date t_i, $x_i^{\mathtt{in}} \in [0, a_{\mathtt{in}}]$ the quantity of gas injected and $x_i^{\mathtt{out}} \in [-a_{\mathtt{out}}, 0]$ the quantity of gas withdrawn. We thus have the following constraint to satisfy at each date t_i:

$$x_i^b = D_i + x_i^{in} + x_i^{out}.$$

The following objective function characterizes the cost of a strategy $x^b = \left\{x_i^b\right\}_{i=0,N}$ by the value :

$$J(C_0, x^b, D_0) = \sum_{i=0}^{N} -\varphi_i(x_b^b),$$

where D_0 is the initial consumption value that we suppose to be equal to $\tilde{D}_0$ and the optimization problem can be written as

$$J^*(C_0, D_0) = \inf_{x \in U} J(C_0, x, D_0).$$

Due to the AR(1) structure of the demand, it is possible to add D_i to the state vector x_i in the algorithm 3.2 as proposed by [12]. We can then test two approaches:

-Approach A: SDDP with conditional cuts as suggested in section 3.2. Here x_i is only the stock level and $\xi_i = (D_i)$ is a dimensional Markov process. All cuts coefficients are one dimensional functions and have to be estimated by one dimensional regressions. In the test case we use $N_c^A = 2000$ trajectories in optimization and conditional cuts coefficients are estimated with an $I_1 = 8$ grid. At each step of the SDDP algorithm we use 8 forward simulations to discover the new states used in the next backward step.

-Approach B: SDDP without conditional cuts but with an augmented state vector. Here $x_i = (x_i^b, D_i)$, and at each date t_i all cuts coefficients are constant. This method is related to nested Monte Carlo: a set of N_i simulations is chosen to simulate ϵ_t. So at a given date in the backward pass, we are left with N_i LPs to solve for each visited state in the previous forward simulation. It turns out that in order to be able to be able to reach the 0.1% criterium for difference between the forward and the backward iteration, we have to take $N_i = 2000$. At each step of the SDDP algorithm we use 1 forward simulation to discover the new states used in the next backward step.

On Figure 4, we give the results obtained by the two approaches until convergence of the backward and forward values to within a given (relative) accuracy of 0.1%. Approach

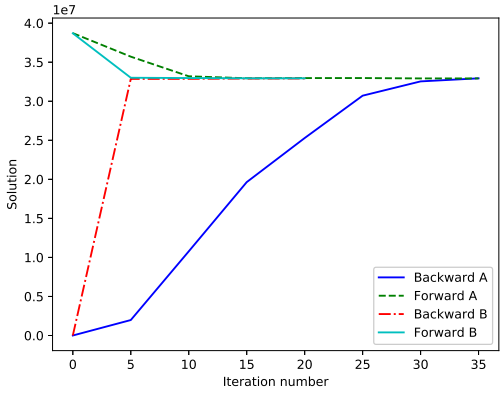


Figure 4: Comparison of optimal values in the backward and forward pass for two variants of SDDP for a load constraint.

A takes 109 seconds to converge while Approach B takes 66 seconds to converge. On this simple case, Approach B is superior to Approach A, especially because of a better convergence rate in the backward step.

4.2.2. Artificial case in dimension 10

In order to test how these methods scale up, we pick the test case of section 4.2.1 but this time with 10 similar storage facilities. We also multiply average load as well as σ_d by 10. Consequently the expected value of the optimal cost is 10 times the value of the case with one storage facility.

On Figure 5, we give the convergence results obtained by the two approaches. In order to be able to get a relative given accuracy of 0.1% it is necessary to increase N_i to 4000. Both approaches allow us to obtain the same value, showing consistency of the suggested method. Although approach B, obtains a better estimation of the optimal value earlier on in the algorithm, approach A turns out to be faster. Indeed, for reaching a 0.1% asserted gap, Approach A takes 660 seconds to converge while Approach B takes 775 (+17%) seconds.

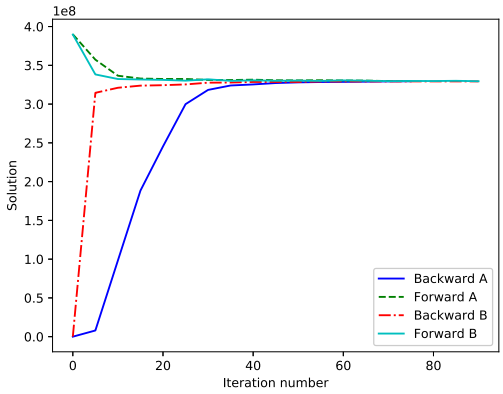


Figure 5: Comparison of optimal values in the backward and forward pass for two variants of SDDP for ten gas storage facilities with a coupling load constraint.

4.3. Valuing a more complex storage facility when facing a consumption constraint

In this section we take the test case 4.2 and allow the prices to vary following the dynamic given in section 4.1. As in the case described in section 4.1, the classical SDDP method cannot deal with this kind of uncertainty. Note that this is so since price uncertainty affects the cost coefficients of the transition problems (e.g., (14)), so that increasing the state space would make these problems bi-linear (i.e., no-longer convex). In the two following subsections we show that our method can deal efficiently with multidimensional uncertainty. All calculation are achieved on a cluster of 8 processors with 14 cores each.

4.3.1. A case in one dimension

Similarly to the sections 4.1 and 4.2 the following objective function characterizes the cost of a strategy $x^b = \{x_i^b\}_{i=0,N}$:

$$J(C_0, x^b, S_0, D_0) = \sum_{i=0}^{N} -\varphi_i(x_b^b).$$

The optimization problem can be written as

$$J^*(C_0, S_0, D_0) = \inf_{x \in U} J(C_0, x, S_0, D_0).$$

We use the SDDP method with conditional cuts as suggested in section 3.2. Here x_i is only the stock level and $\xi_i = (S_{i\Delta T}, D_i)$ is a two dimensional Markov process. All cut coefficients are two dimensional functions and have to be estimated by two dimensional regressions. In the test case we use $N_c^A = 32000$ trajectories in optimization and conditional cuts coefficients are estimated with an $(I_1, I_2) = (8, 4)$ grid (see Figure 1). At each step of the SDDP algorithm we use 32 forward simulations to discover the new states used in the next backward step.

On Figure 6, we give the results until convergence of the backward and forward values with a given (relative) accuracy of 0.1%. The convergence is achieved in 845 seconds.

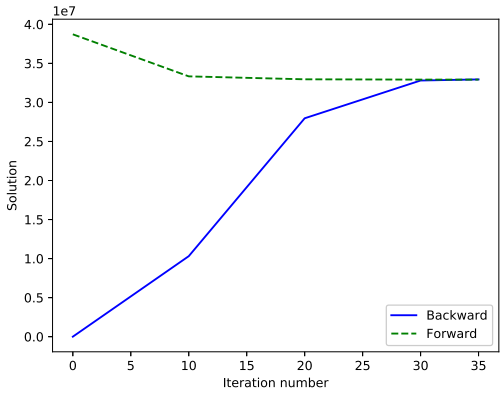


Figure 6: SDDP backward and forward iterations for a gas storage facility with a coupling load constraint.

4.3.2. An artificial test case in dimension 10.

In order to test how the method scales up, we pick the test case of section 4.3.1 but this time with 10 similar storage facilities. We also multiply average load as well as σ_d by 10. Consequently the expected value of the optimal cost is 10 times the value of the case with one storage facility.

On Figure 7, we give the convergence results obtained by the conditional cuts method. Once again our method converges very efficiently even for a high dimension problem with

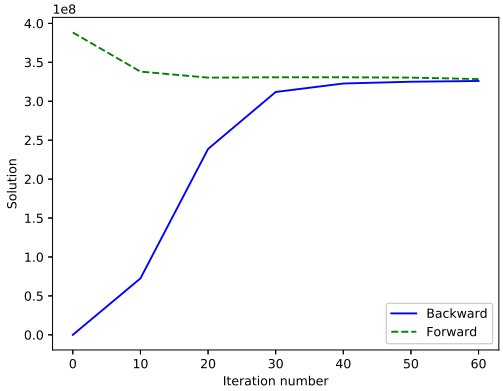


Figure 7: SDDP backward and forward iterations for ten gas storage facilities with a coupling load constraint.

convergence reached in 2400 seconds.

5. Conclusion

Using conditional cuts we have developed an effective methodology to value easily high dimensional storages problems. Getting rid of building trees for the backward part, the method is easy to use even on a set of initial scenarios, perhaps resulting from historical data, or any other set of scenarios assumed to be Markovian. The method can also be employed whatever the elements of the transition problems impacted by uncertainty. This is not the case for the classic approach, consisting of increasing the state space dimension, the use of which is restricted to right hand side uncertainty.

Moreover, when the dimension of the uncertainties is low and in the special case of auto regressive models, we have shown that the use of conditional cuts is competitive with the classical approach developed in [12] consisting of augmenting the state space to account for past dependency.

At last, because our method is a pure Monte Carlo one, risk optimization such as CVaR or VaR optimization should be much more accurate than classical approaches.

References

[1] V. Goel, I. E. Grossmann, A stochastic programming approach to planning of offshore gas field developments under uncertainty in reserves, Computers & Chemical Engineering 28 (8) (2004) 1409 – 1429. doi:http://dx.doi.org/10.1016/j.compchemeng.2003.10.005.

[2] M. Pereira, S. Granville, M. Fampa, R. Dix, L. Barroso, Strategic bidding under uncertainty: A binary expansion approach, IEEE Transactions on Power Systems 11 (1) (February 2005) 180–188.

[3] A. Shapiro, W. Tekaya, J. P. da Costa, M. P. Soares, Risk neutral and risk averse stochastic dual dynamic programming method, European Journal of Operational Research 224 (2) (2013) 375 – 391. doi:http://dx.doi.org/10.1016/j.ejor.2012.08.022.

[4] S. Rebennack, Combining sampling-based and scenario-based nested benders decomposition methods: application to stochastic dual dynamic programming, Mathematical Programming 156 (1) (2016) 343–389. doi:10.1007/s10107-015-0884-3.

[5] V. L. de Matos, D. P. Morton, E. C. Finardi, Assessing policy quality in a multistage stochastic program for long-term hydrothermal scheduling, Annals of Operations Research (2016) 1–19doi:10.1007/s10479-016-2107-6.

[6] B. Fhoula, A. Hajji, M. Rekik, Stochastic dual dynamic programming for transportation planning under demand uncertainty, in: 2013 International Conference on Advanced Logistics and Transport, 2013, pp. 550–555. doi:10.1109/ICAdLT.2013.6568518.

[7] Y. T. Herer, M. Tzur, E. Yücesan, The multilocation transshipment problem, IIE Transactions 38 (3) (2006) 185–200. doi:10.1080/07408170500434539.

[8] G. C. Pflug, W. Römisch, Modeling, Measuring and Managing Risk, World Scientific Publishing Company, 2007.

[9] J. Dupačová, Portfolio Optimization and Risk Management via Stochastic Programming, Osaka University Press, 2009.

[10] J. Dupačová, J. Polívka, Asset-liability management for czech pension funds using stochastic programming, Annals of Operations Research 165 (1) (2009) 5–28. doi:10.1007/s10479-008-0358-6.

[11] J. R. Birge, Decomposition and partitioning methods for multistage stochastic linear programs, Operations Research 33 (5) (1985) 989–1007.

[12] M. Pereira, L. Pinto, Multi-stage stochastic optimization applied to energy planning, Mathematical Programming 52 (2) (1991) 359–375.

[13] J. Dupačová, N. Gröwe-Kuska, W. Römisch, Scenario reduction in stochastic programming : An approach using probability metrics, Mathematical Programming 95 (3) (2003) 493–511.

[14] W. de Oliveira, C. Sagastizábal, D. D. J. Penna, M. E. P. Maceira, J. M. Damázio, Optimal scenario tree reduction for stochastic streamflows in power generation planning problems, Optimization Methods and Software 25 (6) (2010) 917–936.

[15] G. C. Pflug, Version-independence and nested distributions in multistage stochastic optimization, SIAM Journal on Optimization 20 (3) (2010) 1406–1420. doi:10.1137/080718401.

[16] H. Heitsch, W. Römisch, Scenario tree generation for multi-stage stochastic programs, in: M. Bertocchi, G. Consigli, M. Dempster (Eds.), Stochastic Optimization Methods in Finance and Energy: New Financial Products and Energy Market Strategies, Vol. 163 of International Series in Operations Research & Management Science, Springer-Verlag, 2011, pp. 313–341.
[17] G. C. Pflug, A. Pichler, A distance for multistage stochastic optimization models, SIAM Journal on Optimization 22 (1) (2012) 1–23. doi:10.1137/110825054.
[18] R. M. Kovacevic, A. Pichler, Tree approximation for discrete time stochastic processes: a process distance approach, Annals of Operations Research 235 (1) (2015) 395–421. doi:10.1007/s10479-015-1994-2.
[19] Y. Song, J. Luedtke, An adaptive partition-based approach for solving two-stage stochastic programs with fixed recourse, SIAM Journal on Optimization 25 (3) (2015) 1344–1367. doi:10.1137/140967337.
[20] W. van Ackooij, W. de Oliveira, Y. Song, An adaptive partition-based level decomposition for solving two-stage stochastic programs with fixed recourse, To Appear in Informs Journal on Computing (2016) 1–18.
[21] W. van Ackooij, W. de Oliveira, Y. Song, On regularization with normal solutions in decomposition methods for multistage stochastic programs, Submitted preprint, available http://www.optimization-online.org/DB_HTML/2017/01/5806.html (2017) 1–28.
[22] A. Philpott, V. L. de Matos, Dynamic sampling algorithms for multi-stage stochastic programs with risk aversion, European Journal of Operational Research 218 (2) (2012) 470 – 483. doi:http://dx.doi.org/10.1016/j.ejor.2011.10.056.
[23] F. B. Rodríguez, W. de Oliveria, E. Finardi, Application of scenario tree reduction via quadratic process to medium-term hydrothermal scheduling problem, To appear in IEEE Transactions on Power Systems.
[24] Z. L. Chen, W. B. Powell, Convergent cutting-plane and partial-sampling algorithm for multistage stochastic linear programs with recourse, Journal of Optimization Theory and Applications 102 (3) (1999) 497–524. doi:10.1023/A:1022641805263.
[25] M. Hindsberger, A. Philpott, Resa: A method for solving multi-stage stochastic linear programs, in: SPIX Stochastic Programming Symposium, Berlin, 2001.
[26] C. Donohue, J. R. Birge, The abridged nested decomposition method for multistage stochastic linear programs with relatively complete recourse, Algorithmic Operations Research 1 (1) (2006) 20–30.
[27] A. Shapiro, Analysis of stochastic dual dynamic programming method, European Journal of Operational Research 209 (2011) 63–72.
[28] A. R. de Queiroz, D. P. Morton, Sharing cuts under aggregated forecasts when decomposing multistage stochastic programs, Operations Research Letters 41 (2013) 311–316.
[29] J. Tsilikis, B. Van Roy, Optimal stopping of Markov processes: Hilbert space theory, approximation algorithms, and application to pricing high dimensional financial derivatives, IEEE Transaction on Automatic Control 44 (1999) 1840–1851.
[30] F. A. Longstaff, E. S. Schwartz, Valuing american options by simulation: a simple least-squares approach, Review of Financial studies 14 (1) (2001) 113–147.
[31] X. Warin, Gas storage hedging, in: R. A. Carmona, P. del Moral, P. Hu, N. Oudjane (Eds.), Numerical Methods in Finance, Vol. 12 of Springer Proceedings in Mathematics, Springer, 2012, pp. 421–445.
[32] B. Bouchard, X. Warin, Monte-Carlo valuation of American options: Facts and new algorithms to improve existing methods, in: R. A. Carmona, P. del Moral, P. Hu, N. Oudjane (Eds.), Numerical Methods in Finance, Vol. 12 of Springer Proceedings in Mathematics, Springer, 2012, pp. 215–255.
[33] G. Pagès, J. Printems, Optimal quadratic quantization for numerics: the gaussian case, Monte Carlo Methods and Applications 9 (2) (2003) 135–166.
[34] G. Pagès, H. Pham, J. Printems, Optimal quantization methods and applications to numerical problems in finance, in: Handbook of computational and numerical methods in finance, Springer, 2004, pp. 253–297.
[35] G. Pagès, J. Printems, Optimal quantization for finance: from random vectors to stochastic processes, Handbook of Numerical Analysis 15 (2008) 595–648.
[36] G. Pagès, J. Printems, Functional quantization for numerics with an application to option pricing, Monte Carlo Methods and Applications 11 (4) (2005) 407–446.
[37] E. Fournié, J.-M. Lasry, J. Lebuchoux, P.-L. Lions, N. Touzi, Applications of malliavin calculus to monte carlo methods in finance, Finance and Stochastics 3 (4) (1999) 391–412.
[38] E. Fournié, J.-M. Lasry, J. Lebuchoux, P.-L. Lions, Applications of malliavin calculus to monte-carlo

methods in finance. ii, Finance and Stochastics 5 (2) (2001) 201–236.
[39] B. Bouchard, N. Touzi, Discrete-time approximation and monte-carlo simulation of backward stochastic differential equations, Stochastic Processes and their applications 111 (2) (2004) 175–206.
[40] E. Gobet, J.-P. Lemor, X. Warin, A regression-based monte carlo method to solve backward stochastic differential equations, The Annals of Applied Probability 15 (3) (2005) 2172–2202.
[41] J.-P. Lemor, E. Gobet, X. Warin, Rate of convergence of an empirical regression method for solving generalized backward stochastic differential equations, Bernoulli 12 (5) (2006) 889–916.
[42] E. Gobet, P. Turkedjiev, Approximation of backward stochastic differential equations using malliavin weights and least-squares regression, Bernoulli 22 (1) (2016) 530–562.
[43] A. Fahim, N. Touzi, X. Warin, A probabilistic numerical method for fully nonlinear parabolic pdes, The Annals of Applied Probability 21 (4) (2011) 1322–1364.
[44] K. Linowsky, A. Philpott, On the convergence of sampling-based decomposition algorithms for multistage stochastic programs, Journal of Optimization Theory and Applications 125 (2) (2005) 349–366. doi:10.1007/s10957-004-1842-z.
[45] A. Philpott, Z. Guan, On the convergence of stochastic dual dynamic programming and related methods, Operations Research Letters 36 (4) (2008) 450–455.
[46] V. Guigues, W. Römisch, Sampling-based decomposition methods for multistage stochastic programs based on extended polyhedral risk measures, SIAM Journal on Optimization 22 (2) (2012) 286–312. doi:10.1137/100811696.
[47] V. Guigues, C. Sagastizábal, The value of rolling horizon policies for risk-averse hydro-thermal planning, European Journal of Operations Research 217 (2012) 219–240. doi:10.1016/j.ejor.2011.08.017.
[48] J. Dupačová, V. Kozmík, Structure of risk-averse multistage stochastic programs, OR Spectrum 37 (3) (2015) 559–582. doi:10.1007/s00291-014-0379-2.
[49] T. Homem de Mello, B. Pagnoncelli, Risk aversion in multistage stochastic programming: A modeling and algorithmic perspective, European Journal of Operations Research 249 (2016) 188–199.
[50] A. Eichhorn, W. Römisch, Polyhedral risk measures in stochastic programming, SIAM Journal on Optimization 16 (1) (2005) 69–95. doi:10.1137/040605217.
[51] A. Ruszczyński, A. Shapiro, Conditional risk mappings, Mathematics of Operations Research 31 (3) (2006) 544–561. doi:10.1287/moor.1060.0204.
[52] J. Kelley, The cutting-plane method for solving convex programs, Journal of the Society for Industrial and Applied Mathematics 8 (4) (1960) 703–712.
[53] A. Shapiro, D. Dentcheva, A. Ruszczyński, Lectures on Stochastic Programming. Modeling and Theory, Vol. 9 of MPS-SIAM series on optimization, SIAM and MPS, Philadelphia, 2009.
[54] A. Ruszczyński, A. Swietanowski, Accelerating the regularized decomposition method for two stage stochastic linear problems, European Journal of Operational Research 101 (2) (1997) 328–342. doi:10.1016/S0377-2217(96)00401-8.
[55] H. Gevret, J. Lelong, X. Warin, Stochastic optimization library in c++ (2016).
URL https://hal.archives-ouvertes.fr/hal-01361291
[56] W. van Ackooij, R. Henrion, A. Möller, R. Zorgati, Joint chance constrained programming for hydro reservoir management, Optimization and Engineering 15 (2014) 509–531.
[57] P. Henaff, I. Laachir, F. Russo, Gas storage valuation and hedging. a quantification of the model risk, Arxiv.
[58] R. Carmona, M. Ludkovski, Valuation of energy storage: an optimal switching approach, Quantitative Finance 10 (4) (2010) 359–374.
[59] R. Shumway, D. Stoffer, Time Series Analysis and Its Applications, 1st Edition, Springer, 2000.