Energy contracts management by stochastic programming techniques

Zhihao Cen, J. Frederic Bonnans, Thibault Christel

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Energy contracts management by stochastic programming techniques

Zhihao Cen — J. Frédéric Bonnans — Thibault Christel

N° 7289

May 2010

Thème NUM

ISSN 0249-6399 ISRN INRIA/RR--7289--FR+ENG

INSTITUT NATIONAL
DE RECHERCHE
EN INFORMATIQUE
ET EN AUTOMATIQUE

INRIA

centre de recherche
SACLAY - ÎLE-DE-FRANCE

Energy contracts management by stochastic programming techniques

Zhihao Cen* , J. Frédéric Bonnans† , Thibault Christel‡

Thème NUM — Systèmes numériques
Équipes-Projets Commands

Rapport de recherche n° 7289 — May 2010 — 22 pages

Abstract: We consider the problem of optimal management of energy contracts, with bounds on the local (time step) amounts and global (whole period) amounts to be traded, integer constraint on the decision variables and uncertainty on prices only. After building a finite state Markov chain by using vectorial quantization tree method, we rely on the stochastic dual dynamic programming (SDDP) method to solve the continuous relaxation of this stochastic optimization problem. An heuristic for computing sub optimal solutions to the integer optimization problem, based on the Bellman values of the continuous relaxation, is provided. Combining the previous techniques, we are able to deal with high-dimension state variables problems. Numerical tests applied to realistic energy markets problems have been performed.

Key-words: stochastic programming, multi-stage, dual dynamic programming, quantization tree.

* Total, Paris La Defense, COMMANDS team, INRIA-Saclay and CMAP, Ecole Polytechnique, Palaiseau, France (zhihao.cen@polytechnique.edu).

† COMMANDS team, INRIA-Saclay, CMAP, Ecole Polytechnique, Palaiseau, and Laboratoire de Finance des Marchés de l'Energie, Paris, France (Frederic.Bonnans@inria.fr).

‡ Total, Paris La Defense, France (thibault.christel@total.com).

Centre de recherche INRIA Saclay – Île-de-France
Parc Orsay Université
4, rue Jacques Monod, 91893 ORSAY Cedex
Téléphone : +33 1 72 92 59 00

Gestion de contrats d'énergie par des techniques de programmation stochastique

Résumé : Nous considérons le problème de la gestion optimale de contrats d'énergie, avec bornes sur les quantités locales et globales, des contraintes d'intégrité sur les variables de décision et une incertitude ne portant que sur les prix. Après avoir construit une chaîne de Markov en état finie par la méthode d'arbre de quantisation, nous nous appuyons sur la méthode de programmation stochastique dynamique duale (SDDP) pour résoudre la relaxation continue de ce problème d'optimisation stochastique. Une heuristique de calcul de solutions sous optimales du problème en nombres entiers, basée sur les valeurs de Bellman du problème relaxé, est proposée. Combinant les deux techniques précédentes, nous sommes en mesure de traiter des problèmes de grande dimension. Des tests numériques appliqués à des problèmes réalistes de marchés de l'énergie ont été réalisés.

Mots-clés : programmation stochastique, multi étapes, programmation dynamique duale, arbre de quantisation.

1 Introduction

The physical energy markets are articulated around long term supply and demand contracts. Their purpose is to accommodate both producers and consumers according to their needs. On one side, a producer will be keen on selling energy on a long term basis for cash flow certainties allowing him to schedule other investments. On the other side, it ensures the consumer of a certainty on volumes and acts as a hedge against short term uncertainties.

Long term energy contracts typically have the following features. First of all, they are associated with a granularity defining the periods at which the energy is exchanged. Second of all, the amounts of traded energy are contractually upper and lower bounded during each period as well as on the overall contract. The motivation behind these bounds is generally to allow the consumer to modulate his energy purchase. Finally, the price at which the energy is bought or sold at a given period is function of the index spot price at that time which is a stochastic variable. Therefore, the optimal management of long term contracts implies for the option's holder to successively choose the best possible decision over the quantity to be bought or sold according to the price information available at that moment. Each decision taken will impact the remainder of the contract. In addition, one might consider the situation where the decisions to be made are discrete. Indeed, unlike pipeline natural gas where it is generally possible to assume that any quantity of gas can be exchanged as long as it fulfills the contractual boundaries, this is no longer true for sea shipped commodities such as Liquefied Natural Gas (LNG). In fact, the quantity of energy to be exchanged consists in the number of LNG vessels to allocate to a destination. The combination of stochastic and discrete decision makes the problem very difficult, and it is natural to consider a continuous relaxation of the problem, at least as a first step.

For surveys on stochastic programming, we refer to [5, 13, 22]. Unless the underlying random space has finite support, our model is an optimization problem over infinite-dimensional function spaces, which is difficult to solve. Analytical solutions are not available except for very simple models. Numerical approximation methods have been largely studied in the literature.

Most approximation schemes are based on discretization of the underlying random space by a scenario tree. A survey and evaluation of popular scenario generation techniques is provided in [14]. Among them, let us mention sampling-based methods (quasi Monte-Carlo based method) [22, Chapter 3], moment matching methods [11], tree reduction methods based on probability metric [6, 8], recombining tree [15]. Once the underlying probability space becomes discrete, every finite dimensional optimization algorithms can be applied without difficulties. However, the problem size grows polynomially with the number of discretization points and exponentially with the number of stages. So, getting a high precision for the value problem and the computation of a solution is still a big challenge.

Another main approximation method is to project the control function or cost-to-go function into a finite dimension functional space [4, 23, 24]. One has hence to solve a finite dimensional problem, whose dimension is the cardinality of the functional basis. The advantage of this method is that it does not need discretization of the underlying probability space. Numerical simulation can be done with Monte-Carlo based methods, and thus the convergence properties

will be given by classical Monte-Carlo methods as well as Hilbert space theory. This method has been successfully applied to many problems [20]. However, in practice, the result depends strongly on the basis functions chosen, and when the state variable (and random variable) dimension increases, the dimension of basis function space often increases exponentially.

Apart from these two main approximation methods, some special methods have been proposed for certain specific problems. For example, Pereira and Pinto [17] study the problem where randomness only appears on right-hand-side of constraint and is independent from one stage to another. Philpott and Guan [18] have studied the convergence of this method. Shapiro recently analyses the statistical properties and rate of convergence of this method [21]. This method can also be applied to ARMA process [12].

The main contribution of this article is to introduce a new discretization method of the underlying probability space, namely the vectorial quantization tree method. We apply the traditional dual decomposition method to the continuous relaxation of this problem. The numerical test results present good performance when the two methods are combined. Like other dual decomposition methods applied to multi-stage stochastic programs, this algorithm provides upper and lower bounds of the optimal value for the initial problem, both in a statistical sense. This article is organised as follows: in section 2, we present the model and some properties. In section 3, we introduce the quantization tree, and then the algorithm for the continuous relaxation in section 4. Finally, numerical tests are performed over realistic energy management problems in section 5. A heuristic for computing integer suboptimal solutions based on the Bellman values obtained by continuous relaxation is also provided.

2 Problem formulations

2.1 Framework

We may formulate the class of problems under consideration in the following setting. First of all, we have a discrete time Markov process $(\boldsymbol{\xi}_t)_{t\in[1,T]}$ in the probability space $L^2(\Omega, \mathcal{F}, \mathbf{P}; \mathbb{R}^d)$. We may write its dynamics as

$$\boldsymbol{\xi}_{t+1} = f(W_t, \boldsymbol{\xi}_t, \alpha_t) \quad t = 1, \dots, T-1, \tag{1}$$

where $\boldsymbol{\xi}_1$ is deterministic, $(\alpha_t)_{t\in[1,T-1]}$ is a sequence of deterministic parameters, and $(W_t)_{t\in[1,T-1]}$ is a sequence of independent and identically-distributed (i.i.d.) random variables on $L^2(\Omega, \mathcal{F}, \mathbf{P}; \mathbb{R}^d)$, independent of $(\boldsymbol{\xi}_t)$; finally f is a measurable mapping. The canonic filtration associated with $(\boldsymbol{\xi}_t)$ is denoted by $\mathcal{F}_t = \sigma(\boldsymbol{\xi}_s, 1 \le s \le t)$. The decision problem to be solved has the following expression:

$$\begin{aligned} \inf_{(u_t)_{t\in[1,T]}} \quad & \mathbb{E}\left[\sum_{t=1}^{T} c_t(\boldsymbol{\xi}_t)\cdot u_t + g(\boldsymbol{\xi}_T, x_{T+1})\right] \\ \text{s.t.} \quad & u_t \text{ is } \mathcal{F}_t - \text{measurable}, \quad u_t \in \mathfrak{U}_t \quad \text{a.s.} \\ & x_{t+1} = x_t + A_t u_t \\ & x_1 = 0,\ x_{T+1} \in \mathfrak{X}_{T+1}. \end{aligned} \tag{2}$$

where $u_t \in \mathbb{R}^n$ is the control variable, $\mathfrak{U}_t$ is a compact nonempty polyhedral set in $\mathbb{R}^n$, $x_t \in \mathbb{R}^m$ is the state variable representing the storage, $\mathfrak{X}_{T+1}$ is a nonempty polyhedral set in $\mathbb{R}^m$, $A_t \in \mathbb{R}^{m\times n}$ tells how the decision modifies the storage, $c_t(\cdot) : \mathbb{R}^d \to \mathbb{R}^n$ is a running cost function by unit on control, and $g(\boldsymbol{\xi}, x) : \mathbb{R}^d \times \mathbb{R}^m \to \mathbb{R}$ is convex lower semi-continuous (l.s.c.) function of x and can often be interpreted as a penalty function for final state x_{T+1}.

Lemma 2.1. *If $\sum_{t=1}^{T} A_t\mathfrak{U}_t \cap \mathfrak{X}_{T+1} \neq \emptyset$, problem (2) has a nonempty set of solutions in $L^\infty(\Omega, \mathcal{F}, \mathbf{P}; \mathbb{R}^n)$.*

Proof. Since the sets $(\mathfrak{U}_t), t = 1, \ldots, T$ are compact, it suffices to prove the existence of a solution in $H := L^2(\Omega, \mathcal{F}, \mathbf{P}; \mathbb{R}^n)$. The set of feasible policies is closed and convex, and hence, weakly closed. The final cost $\mathbb{E}[g(\boldsymbol{\xi}_T, x_{T+1})]$ is a l.s.c. convex function over $L^2(\Omega, \mathcal{F}, \mathbf{P}; \mathbb{R}^m)$, and therefore is weakly l.s.c. in this space; so is the all cost function $(c_t(\boldsymbol{\xi}_t)u_t)$. Extracting if necessary a subsequence, we may assume that a minimizing sequence say u^k, necessarily bounded in H, is weakly convergent in this space. Passing to the limit in the state equation and the constraint on the decision and using the weak l.s.c. of the cost function, we obtain that the weak limit of u^k is solution of (2). □

This condition $\sum_{t=1}^{T} A_t u_t \cap \mathfrak{X}_{T+1} \neq \emptyset$ is always satisfied in this article to make sure of the existence of a solution.

Remark 2.2. The function to be minimized is not strictly convex. Therefore the minimum is not necessarily unique.

2.2 Dynamic programming principle

The problem under study has the desired structure for the dynamic programming principle to hold. Let us define the realization up to time t of the signal as $\boldsymbol{\xi}_{[t]} = (\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_t)$. The Bellman value at stage t, denoted by $Q(t, x_t, \boldsymbol{\xi}_{[t]})$, is the value of the variant of the original problem obtained when starting at time t with state variable(storage) x_t:

$$\begin{array}{rlll} Q(t, x_t, \boldsymbol{\xi}_{[t]}) := & \inf\limits_{(u_s)_{s\in[t,T]}} & \mathbb{E}\left[\sum\limits_{s=t}^{T} c_s(\boldsymbol{\xi}_s)\cdot u_s + g(\boldsymbol{\xi}_T, x_{T+1})\right] & (3) \\ & \text{s.t.} & u_s \text{ is } \mathcal{F}_s - \text{measurable}, \quad u_s \in \mathfrak{U}_s \quad \text{a.s.} & \\ & & x_{s+1} = x_s + A_s u_s,\ s = 1, \ldots, T, & \\ & & x_{T+1} \in \mathfrak{X}_{T+1} & \end{array}$$

The optimal value of original problem is $Q(1, 0, \xi_1)$. In view of the Markov property of $(\boldsymbol{\xi}_t)_{t\in[1,T]}$, $Q(t, x_t, \boldsymbol{\xi}_{[t]})$ is actually a function of $(t, x_t, \boldsymbol{\xi}_t)$. The following *dynamic programming principle* is easily established:

$$\begin{array}{rlll} Q(t, x_t, \boldsymbol{\xi}_t) := & \inf\limits_{u_t\in\mathfrak{U}_t} & c_t(\boldsymbol{\xi}_t)\cdot u_t + \mathcal{Q}(t+1, x_{t+1}, \boldsymbol{\xi}_t) & (4) \\ & \text{s.t.} & A_t u_t + x_t = x_{t+1} & \end{array}$$

where $\mathcal{Q}(\cdot,\cdot,\cdot)$ is the conditional expectation of the value at time $t+1$ obtained in state x_{t+1}:

$$\mathcal{Q}(t+1, x_{t+1}, \boldsymbol{\xi}_t) := \mathbb{E}\left[Q(t+1, x_{t+1}, \boldsymbol{\xi}_{t+1})|\mathcal{F}_t\right], \quad t = 1, \ldots, T-1, \qquad (5)$$

and

$$\mathcal{Q}(T+1, x_{T+1}, \boldsymbol{\xi}_T) = \begin{cases} g(\boldsymbol{\xi}_T, x_{T+1}) & \text{if } x_{T+1} \in \mathfrak{X}_{T+1}, \\ +\infty & \text{otherwise.} \end{cases} \tag{6}$$

2.3 Feasibility

The property of *relatively complete recourse* does not hold generally in this setting. That is, it may happen that for given time t and x_t, such that the associated Bellman value is finite (implying the existence of a sequence of feasible decisions with the corresponding final time belongs to $\mathfrak{X}_{T+1}$), some $u \in \mathfrak{U}_s$ may result in a non feasible state variable $x_{t+1} := x_t + A_t u_t$, whose associated Bellman value will therefore be infinite. That is, the final state constraint induces implicit constraints on the decision at time t.

The SDDP approach includes a mechanism for dealing with this situation, by generating feasibility cuts (similar to Bender's feasibility cuts for a two stage problem but combined with backward recursion). However, generating feasibility cuts may be a long process and we prefer to deal with them by a direct approach, taking advantage of the specific following property: no uncertainty occurs in the state dynamics. In order to guarantee that the final constraints may be fulfilled, let us define the sets of feasible control:

$$\mathfrak{U}^{ad} := \left\{ (u_t)_{t\in[1,T]} \,\middle|\, \sum_{t=1}^{T} A_t u_t \in \mathfrak{X}_{T+1},\ u_t \in \mathfrak{U}_t \right\}. \tag{7}$$

With time $t \in [1, T]$ and some state x_t is associated the (possibly empty) local set of feasible control

$$\mathfrak{U}_t^{ad}(x_t) := \left\{ u_t \in \mathfrak{U}_t \,\middle|\, x_t + \sum_{s=t}^{T} A_s u_s \in \mathfrak{X}_{T+1},\ u_s \in \mathfrak{U}_s, s = t, \dots, T \right\}. \tag{8}$$

Hence, we can also define $\mathfrak{X}_t = \left\{ x \,|\, \mathfrak{U}_t^{ad}(x) \neq \emptyset \right\}$, such that only for $x_t \in \mathfrak{X}_t$, there exists feasible control leading to the final state set $\mathfrak{X}_{T+1}$:

$$\mathfrak{X}_t := \left\{ x_t : \exists u_s \in \mathfrak{U}_s\ s \geq t,\ s.t.\ x_t + \sum_{s=t}^{T} A_s u_s \in \mathfrak{X}_{T+1} \right\} = \mathfrak{X}_{T+1} - \sum_{s=t}^{T} A_s \mathfrak{U}_s. \tag{9}$$

Since $\mathfrak{X}_{T+1}$ and $\mathfrak{U}_t$ are polyhedral sets, so is $\mathfrak{X}_t$. If we replace the decision u_t by a sequence $(u_t, \dots, u_T)$ such that $x_t \in \mathfrak{X}_t$, we get the property of relatively complete recourse. The counterpart is an increase of dimension of the decision variable, which in our LNG application (see second example in section 5) is acceptable.

2.4 Stochastic dual dynamic programming (SDDP) formulation

We next take advantage of a partial convexity property of the Bellman functions.

Proposition 2.3. *For any $t = 1$ to T, the functions $Q(t, x_t, \boldsymbol{\xi}_t)$ and $\mathcal{Q}(t, x_t, \boldsymbol{\xi}_{t-1})$ are convex and l.s.c. functions of x_t.*

Proof. We prove the convexity and l.s.c. property by backward induction. In view of (6), they hold when $t = T + 1$. The conditional expectation (5) keeps the convexity and l.s.c. property on x_t. For $t \leq T$, the convexity follows by induction over t, from T to 1, using (4). To prove the l.s.c., in view of theorem [19, Theorem 1.17] on parametric minimization, the property holds as the expression on right hand side of (4) is proper, l.s.c. and level-bounded in u_t locally uniformly in x_t [19, Definition 1.16]. By induction, both properties follow for all $t = 1, \ldots, T$. □

Remark 2.4. $Q(t, x_t, \boldsymbol{\xi}_t)$ and $\mathcal{Q}(t+1, x_{t+1}, \boldsymbol{\xi}_t)$ are in general not convex functions of $\boldsymbol{\xi}_t$.

We denote by f^* the Fenchel conjugate of an extended real valued function f on an Euclidean space X, identified with its dual, defined for each $x^* \in X$ by

$$f^*(x^*) := \sup_{x \in X} \left(x^* \cdot x - f(x) \right). \tag{10}$$

Let f^{**} denote the biconjugate of f, defined by $f^{**}(x) = \sup_{x^* \in X^*} \left(x \cdot x^* - f^*(x^*) \right)$. By the Moreau Fenchel theorem [19, Thm 12.1], $f = f^{**}$ whenever f is convex, l.s.c. and proper (the latter means that f is always greater than $-\infty$, and not always equal to $+\infty$). Applying this theorem to $Q(t, x_t, \boldsymbol{\xi}_t)$ allows to write this function as the supremum of a family, depending on t and $\boldsymbol{\xi}_t$, of affine functions of x:

$$Q(t, x_t, \boldsymbol{\xi}_t) = Q^{**}(t, x_t, \boldsymbol{\xi}_t) \quad = \quad \sup_{x_t^* \in \mathbb{R}^m} x_t^* \cdot x_t - Q^*(t, x_t^*, \boldsymbol{\xi}_t),$$

where Q^* denotes the conjugate of Q w.r.t. x only.

Thus, we can use the stochastic dual decomposition approach on the state variable x_t [5, 22], but not on the random variable $\boldsymbol{\xi}_t$. The corresponding formulation of the dynamic programming principle is

$$\begin{aligned}
Q(t, x_t, \boldsymbol{\xi}_t) &= \inf_{u_t} \quad c_t(\boldsymbol{\xi}_t) \cdot u_t + \mathcal{Q}(t+1, x_{t+1}, \boldsymbol{\xi}_t) && (11)\\
&= \inf_{u_t} \quad c_t(\boldsymbol{\xi}_t) \cdot u_t + \mathcal{Q}^{**}(t+1, x_{t+1}, \boldsymbol{\xi}_t) && (12)\\
&\quad s.t. \quad A_t u_t + x_t = x_{t+1}\\
\text{(feasibility)} &\quad u_t \in \mathfrak{U}_t^{ad}(x_t)\\
\text{(optimality cut)} &\quad \begin{cases} \mathcal{Q}^{**}(t+1, x_{t+1}, \boldsymbol{\xi}_t) \geq \mathbb{E}\left[Q^{**}(t+1, x_{t+1}, \boldsymbol{\xi}_{t+1}) | \mathcal{F}_t \right] \\ Q^{**}(t+1, x_{t+1}, \boldsymbol{\xi}_{t+1}) \geq x^* \cdot x_{t+1} - Q^*(t+1, x^*, \boldsymbol{\xi}_{t+1}), \quad \forall x^*. \end{cases}
\end{aligned}$$

This is the formulation we will apply to our algorithm.

However, in this formulation, one still needs to calculate conditional expectations. Many numerical methods have been proposed, such as PDE or Monte-Carlo. Here, we choose to discretize the random space. Considering the size of the problem that we are interested in solving (see second example in section 5), standard scenario trees will cause an increase of the dimension of the problem such that it will blow up numerically. To avoid this point, we rely on the vectorial quantization tree which builds a Markov chain with finitely many states.

2.5 Heuristic method for integer solution

We return to the integer solution for problem:

$$\begin{array}{ll} \inf\limits_{(u_t)_{t\in[1,T]}} & \mathbb{E}\left[\sum\limits_{t=1}^{T} c_t(\boldsymbol{\xi}_t)\cdot u_t + g(\boldsymbol{\xi}_T, x_{T+1})\right] \\ \text{s.t.} & u_t \text{ is } \mathcal{F}_t - \text{measurable}, \quad u_t \in \mathbb{Z}^n \cap \mathfrak{U}_t \quad \text{a.s.} \\ & x_{t+1} = x_t + A_t u_t \\ & x_1 = 0,\ x_{T+1} \in \mathfrak{X}_{T+1}. \end{array} \tag{13}$$

and the assumption that $\mathfrak{U}^{ad} \cap \mathbb{Z}^{n\times T} \neq \emptyset$.

Our heuristic method is to look for the integer control using the Bellman value obtained from continuous relaxation:

$$\begin{array}{lll} Q(t, x_t, \boldsymbol{\xi}_t) & = \inf\limits_{u_t} & c_t(\boldsymbol{\xi}_t)\cdot u_t + \mathcal{Q}^{**}(t+1, x_{t+1}, \boldsymbol{\xi}_t) \\ & s.t. & A_t u_t + x_t = x_{t+1} \\ \text{(feasibility)} & & u_t \in \mathbb{Z}^n \cap \mathfrak{U}_t^{ad}(x_t) \\ \text{(optimality cut)} & & \begin{cases} \mathcal{Q}^{**}(t+1, x_{t+1}, \boldsymbol{\xi}_t) \geq \mathbb{E}\left[Q^{**}(t+1, x_{t+1}, \boldsymbol{\xi}_{t+1})|\mathcal{F}_t\right] \\ Q^{**}(t+1, x_{t+1}, \boldsymbol{\xi}_{t+1}) \geq x^*\cdot x_{t+1} - Q^*(t+1, x^*, \boldsymbol{\xi}_{t+1}), \quad \forall x^*. \end{cases} \end{array} \tag{14}$$

3 Vectorial Quantization Tree

Quantization method started in the early 50's and has been applied in information science and signal processing. Recently, V. Bally and G. Pagès used this technique in order to discretize the probability space and later applied it to financial engineering [1, 2]. In this section, the main ideas and some important results of vectorial quantization are presented.

3.1 Optimal quantization

Optimal quantization of random vectors consists in finding the best possible approximation (in L^p norm sense) of a $\mathbb{R}^d-$valued random vector $\boldsymbol{\xi} \in L^p(\Omega, \mathcal{F}, \mathbf{P})$ by a measurable function $\phi(\boldsymbol{\xi})$ taking values in a finite set Γ of cardinal of at most N values. That is, one wishes to solve the minimization problem:

$$\min\left\{\|\boldsymbol{\xi} - \phi(\boldsymbol{\xi})\|_p \,|\phi : \Omega \to \mathbb{R}^d, \text{measurable}, |\phi(\Omega)| \leq N\right\} \tag{15}$$

and $\Gamma = \phi(\Omega)$. Let us note $\hat{\boldsymbol{\xi}} = \phi(\boldsymbol{\xi})$. This error is defined as minimal *distortion* $D_N^{\mu,p} = \|\boldsymbol{\xi} - \hat{\boldsymbol{\xi}}\|_p^p$ of quantization, where μ is the probability measure of $\boldsymbol{\xi}$. One gets easily that $\phi = \mathbf{Proj}_\Gamma$, where $\mathbf{Proj}_\Gamma$ denotes the projection on the grid $\Gamma = \left\{\xi^1, \ldots, \xi^n\right\}$ following the nearest neighbor rule following L^p distance.

We say that $C(\xi^i)$ is a *Voronoi tessellation* of N-tuple Γ if for all $\xi^i \in \Gamma$, $C(\xi^i)$ is a Borel set satisfying:

$$C(\xi^i) = \mathbf{Proj}_\Gamma^{-1}(\xi^i) \subset \left\{\xi \mid |\xi - \xi^i|_p = \min_{\xi^j\in\Gamma} |\xi - \xi^j|_p\right\}. \tag{16}$$

Following the definition, a N-tuple does have infinitely many Voronoi tessellations. However, each tessellation always has the same closure and the same

boundary. If the distribution of $\boldsymbol{\xi}$ is strongly continuous, i.e., the measure μ of hyperplane is 0, we will not bother with boundaries and one may think as if there is only one tessellation. This hypothesis is satisfied in our application.

Therefore, the problem of finding optimal quantization becomes a global optimization problem, which is in general difficult to solve. However, one can easily derive the following properties (e.g. [2, property 1-3]):

- The minimal distortion is decreasing w.r.t. N.
- $\lim_{N\to\infty} \min_{|\Gamma|\leq N} \|\boldsymbol{\xi} - \hat{\boldsymbol{\xi}}\|_p = 0.$

One important property is the rate of the convergence to zero, which is referred as Zador's theorem.

Theorem 3.1. *[7, Theorem 6.2] If $\mathbb{E}[|\boldsymbol{\xi}|^{p+\eta}] < \infty$ for some $\eta > 0$, then,*

$$\lim_N \left(N^{p/d} \min_{|\Gamma|\leq N} \|\boldsymbol{\xi} - \hat{\boldsymbol{\xi}}\|_p^p \right) = J_{p,d} \left(\int |g|^{d/(d+p)}(u) du \right)^{1+p/d}$$

where $\mathbf{P}(du) = g(u)\lambda_d(du) + \nu, \nu \perp \lambda_d$ (λ_d Lebesgue measure on $\mathbb{R}^d$). The constant $J_{p,d}$ corresponds to the case of the uniform distribution on $[0,1]^d$.

Remark 3.2. This theorem implies that $\min_{|\Gamma|\leq N} \|\boldsymbol{\xi} - \hat{\boldsymbol{\xi}}\|_p = O(N^{1/d})$.

3.2 Quantization (recombining) tree

The quantization tree $(\hat{\boldsymbol{\xi}}_t)_{t\in[1,T]}$ is a finite state Markov chain, such that at each stage t, $\hat{\boldsymbol{\xi}}_t$ is an optimal quantization of the original probability space $\boldsymbol{\xi}_t$ in the discrete time process.

Let us denote by N_t be the number of quantization points at stage t, and set $N := \sum_{t=1}^T N_t$ with $N_1 := 1$. We first set the number of total quantization points N, which depends on the calculation capacity of the machine and on the required precision of the result.

The next step is to determine the number of quantization points taken at each stage N_t. If $\boldsymbol{\xi}_t \in L^2(\Omega, \mathcal{F}_t, \mathbf{P}; \mathbb{R}^d)$, following [2, proposition 8] in order to minimize the L^2 distortion of the whole tree, one gets that

$$N_t = \left\lceil \frac{a_t}{\sum_{\tau=1}^T a_\tau} N \right\rceil \quad a_t := \|\boldsymbol{\xi}_t\|_{d/(d+2)}^{d/2(1+d)} \qquad t = 2, \ldots, T. \tag{17}$$

We still have to evaluate probability transitions for the Markov chain. Let p_t^{ij} be the transition probability from ξ_t^i to ξ_{t+1}^j, defined as

$$p_t^{ij} = \mathbf{P}\left[\mathbf{Proj}_{\Gamma_{t+1}}(\boldsymbol{\xi}_{t+1}) = \xi_{t+1}^j | \mathbf{Proj}_{\Gamma_t}(\boldsymbol{\xi}_t) = \xi_t^i \right] \tag{18}$$

In practice, an analytical computation of the probability transitions is out of reach for $d \geq 2$. One efficient way is to compute the transition by a Monte-Carlo method, i.e. p_t^{ij} is to be estimated by

$$\hat{p}_t^{ij} = \frac{\sum_{m=1}^M \mathbf{1}_{\{\mathbf{Proj}_{\Gamma_t}(\boldsymbol{\xi}_t^m) = \xi_t^i\}} \mathbf{1}_{\{\mathbf{Proj}_{\Gamma_{t+1}}(\boldsymbol{\xi}_{t+1}^m) = \xi_{t+1}^j\}}}{\sum_{m=1}^M \mathbf{1}_{\{\mathbf{Proj}_{\Gamma_t}(\boldsymbol{\xi}_t^m) = \xi_t^i\}}} \tag{19}$$

where M is the number of sample in Monte-Carlo simulation.

For the detail of quantization tree building, particularly the extended competitive learning vector quantization (CLVQ) algorithm, we refer to [2].

3.3 Dual dynamic programming formulation on quantization tree

In the sequel we denote the quantized process by $(\hat{\boldsymbol{\xi}}_t)$, the associated filtration by $(\hat{\mathcal{F}}_t)$, and (for the problem similar to (2) in which random variables $(\boldsymbol{\xi}_t)$ are replaced by $(\hat{\boldsymbol{\xi}}_t)$) the associated Bellman value by $\hat{Q}(t+1, x_{t+1}, \hat{\boldsymbol{\xi}}_t)$. The corresponding dynamic programming principle reads

$$\begin{aligned} \hat{Q}(t, x_t, \hat{\boldsymbol{\xi}}_t) := \quad & \inf_{u_t \in \mathfrak{U}_t^{ad}(x_t)} \quad c_t(\hat{\boldsymbol{\xi}}_t) \cdot u_t + \hat{\mathcal{Q}}(t+1, x_{t+1}, \hat{\boldsymbol{\xi}}_t) \\ & \text{s.t.} \quad A_t u_t + x_t = x_{t+1} \end{aligned} \tag{20}$$

where $\hat{\mathcal{Q}}(\cdot,\cdot,\cdot)$ is the conditional expectation of the value at time $t+1$ obtained in state x_{t+1}:

$$\hat{\mathcal{Q}}(t+1, x_{t+1}, \hat{\boldsymbol{\xi}}_t) := \mathbb{E}\left[\hat{Q}(t+1, x_{t+1}, \hat{\boldsymbol{\xi}}_{t+1}) | \hat{\mathcal{F}}_t\right], \quad t = 1, \ldots, T-1, \tag{21}$$

and

$$\hat{\mathcal{Q}}(T+1, x_{T+1}, \hat{\boldsymbol{\xi}}_T) = \begin{cases} g(\hat{\boldsymbol{\xi}}_T, x_{T+1}) & \text{if } x_{T+1} \in \mathfrak{X}_{T+1}, \\ +\infty & \text{otherwise.} \end{cases} \tag{22}$$

In the algorithm to be studied next, only a finite set of affine lower bounds of the Bellman values $\hat{Q}(t, \cdot, \hat{\boldsymbol{\xi}}_t)$ are available. Denote by M_t^i the corresponding set for the realization of ξ_t^i. We see that the maximum of these affine lower bounds is itself a convex lower bound of the Bellman values, denoted by :

$$\theta(t, x, \xi_t^i, M_t^i) := \max\left\{x^* \cdot x - e; \ \ (x^*, e) \in M_t^i\right\} \tag{23}$$

and the associated conditional expectation appearing in the dynamic programming equation, when the Bellman values are replaced by the lower bounds $\theta(t, x, \xi_t^i, M_t^i)$:

$$\vartheta(t+1, \cdot, \xi_t^i, M_{t+1}) := \mathbb{E}[\theta(t+1, \cdot, \xi_{t+1}, M_{t+1}^j) \tilde{v} \hat{\mathcal{F}}_t] = \sum_{\xi_{t+1}^j \in \Gamma_{t+1}} \hat{p}_t^{ij} \theta(t+1, \cdot, \xi_{t+1}^j, M_{t+1}^j) \tag{24}$$

where $M_{t+1} := \left(M_{t+1}^j, \xi_{t+1}^j \in \Gamma_{t+1}\right)$.

We may then write the resulting *approximate dynamic programming equation*, in which the Bellman value is replaced by its lower estimate, in the following form:

$$\begin{aligned} \inf_{u_t \in \mathfrak{U}_t^{ad}(x_t)} \quad & c_t(\xi_t^j) \cdot u_t + \vartheta(t+1, x_{t+1}, \xi_t^i, M_{t+1}) \\ \text{s.t.} \quad & x_{t+1} = x_t + A_t u_t, \\ & \begin{cases} \vartheta(t+1, x_{t+1}, \xi_t^i, M_{t+1}) = \sum_{\xi_{t+1}^j \in \Gamma_{t+1}} \hat{p}_t^{ij} \theta(t+1, x_{t+1}, \xi_{t+1}^j, M_{t+1}^j) \\ \theta(t+1, x_{t+1}, \xi_{t+1}^j, M_{t+1}^j) \geq x^* \cdot x_{t+1} - e; \ \ \forall (x^*, e) \in M_{t+1}^j. \end{cases} \end{aligned} \tag{25}$$

Starting from the initial state $x = 0$, generating trajectories for $(\boldsymbol{\xi}_t)$, and for each of these trajectories, solving the above linear program at each time, we obtain a suboptimal policy (in the spirit of approximate dynamic programming). We will call this calculation a forward operation. Simultaneously, each time we solve (25), from the duality theory for linear programs we obtain an affine lower bound of $\theta(t, \cdot, \hat{\boldsymbol{\xi}}_t)$ (and hence, also of $\mathcal{Q}(t, \cdot, \hat{\boldsymbol{\xi}}_t)$), exact at the point x_t. This is what we will call a backward elementary step. We perform a backward induction by propagating this information over a particular trajectory, from the final time to the initial one.

In the above discussion we make no use anymore of the original (non quantized) random variable. However, it is sound, in order to obtain an upper bound of the cost, to use the original probability law for computing trajectories. At the same time, for computing the decision variable at each step, we solve the approximate dynamic programming problem, where for $\hat{\boldsymbol{\xi}}_t$ we take the projection of $\boldsymbol{\xi}_t$ over the quantized grid. The problem to be solved then is

$$\begin{array}{ll} \inf\limits_{u_t \in \mathfrak{U}_t^{ad}(x_t)} & c_t(\boldsymbol{\xi}_t) \cdot u_t + \vartheta(t+1, x_{t+1}, \mathbf{Proj}_{\Gamma_t}(\boldsymbol{\xi}_t) = \xi_t^i, M_{t+1}) \\ \text{s.t.} & x_{t+1} = x_t + A_t u_t, \\ & \begin{cases} \vartheta(t+1, x_{t+1}, \xi_t^i, M_{t+1}) = \sum_{\xi_{t+1}^j \in \Gamma_{t+1}} \hat{p}_t^{ij} \theta(t+1, x_{t+1}, \xi_{t+1}^j, M_{t+1}^j) \\ \theta(t+1, x_{t+1}, \xi_{t+1}^j, M_{t+1}^j) \geq x^* \cdot x_{t+1} - e; \ \ \forall (x^*, e) \in M_{t+1}^j. \end{cases} \end{array} \tag{26}$$

However, during the numerical tests, we find that this formulation is still too computationally intensive. We therefore provide an aggregated version of the above problem:

$$\begin{array}{ll} \inf\limits_{u_t \in \mathfrak{U}_t^{ad}(x_t)} & c_t(\boldsymbol{\xi}_t) \cdot u_t + \vartheta(t+1, x_{t+1}, \mathbf{Proj}_{\Gamma_t}(\boldsymbol{\xi}_t) = \xi_t^i, \tilde{M}_{t+1}^i) \\ \text{s.t.} & x_{t+1} = x_t + A_t u_t, \\ & \vartheta(t+1, x_{t+1}, \xi_t^i, \mathcal{M}_{t+1}^i) \geq x^* \cdot x_{t+1} - e; \ \ \forall (x^*, e) \in \mathcal{M}_{t+1}^i, \end{array} \tag{27}$$

where the set $\mathcal{M}_{t+1}^i$ of affine lower bounds is obtained by aggregating the new lower bound for each $\xi_{t+1}^j \in \Gamma_{t+1}$ (each time they are generated) into a single one taking into account the probability transitions:

$$\mathcal{M}_{t+1}^i := \left\{ (x^*, e) = \sum_{\xi_{t+1}^j \in \Gamma_{t+1}} \hat{p}_t^{ij} ((x^*)^j, e^j), ((x^*)^j, \, e^j) \in M_{t+1}^j \right\}. \tag{28}$$

Therefore, this expression still provides lower bounds. It looses some information provided by optimality cuts. Hence, it will require more iterations to converge (supported in our numerical tests). But it simplifies subproblem (26), and numerical tests show that it can reduce the computing time a lot.

The version of heuristic method (14) for integer solution follows the same idea:

$$\begin{array}{ll} \inf\limits_{u_t \in \mathbb{Z}^n \cap \mathfrak{U}_t^{ad}(x_t)} & c_t(\boldsymbol{\xi}_t) \cdot u_t + \vartheta(t+1, x_{t+1}, \mathbf{Proj}_{\Gamma_t}(\boldsymbol{\xi}_t) = \xi_t^i, \tilde{M}_{t+1}^i) \\ \text{s.t.} & x_{t+1} = x_t + A_t u_t, \\ & \vartheta(t+1, x_{t+1}, \xi_t^i, \mathcal{M}_{t+1}^i) \geq x^* \cdot x_{t+1} - e; \ \ \forall (x^*, e) \in \mathcal{M}_{t+1}^i, \end{array} \tag{29}$$

However, it only works for forward pass, as we cannot obtain the dual information from such mixed integer programming problem.

4 Algorithm

After having discretized the probability space, it is natural to use stochastic dual dynamic programming method on the resulting finite state Markov chain. Since the Bellman value (and its approximation by some optimality cuts) is non-convex w.r.t. the underlying random variable, one needs to calculate the cuts on every vertex of quantization tree. The algorithm iterates over a forward pass and a backward pass. The forward pass generates a set of random trajectories following the dynamic of Markov chain (1) and runs the optimization on the fly on these trajectories. It provides an upper bound of the optimal value. The backward pass updates the optimality cuts on each vertex and computes a lower bound at first stage.

4.1 Forward pass

The forward pass generates trajectories $\boldsymbol{\xi}^m, m = 1, \ldots, M_f$ following the dynamic of Markov chain (1), i.e. out of sample for quantization tree. For each trajectory, starting from the first stage ($x_1 = 0$), the current decision u_t and the state variable x_{t+1} for next stage are computed by solving (27).

After having computed all the trajectories, an objective value v^m for each trajectory $m = 1, \ldots, M_f$ is available. Thus, the forward statistic can be obtained as follows:

$$\overline{v} = \frac{1}{M_f} \sum_{m=1}^{M_f} v^m \quad \text{and} \quad s = \frac{1}{M_f} \sqrt{\sum_{m=1}^{M_f} (v^m - \overline{v})^2}.$$

Remark 4.1. Also notice that the control $(u_t)_{t\in[1,T]}$ computed in this algorithm is sub-optimal since one approximates $\mathcal{Q}(t+1, x_{t+1}, \boldsymbol{\xi}_t)$ by $\hat{\mathcal{Q}}(t+1, x_{t+1}, \hat{\boldsymbol{\xi}}_t)$. However, $(u_t)_{t\in[1,T]}$ belongs to the domain $\mathfrak{U}^{ad}$ and is $\mathcal{F}_t-$measurable, hence is feasible for the original problem. As it is a minimization problem, each v^m is sub-optimal, so $\overline{v}$ is an upper bound in statistic sense for the original problem: the 95% asymptotic confidence interval is given by $[\overline{v} - 1.96s, \overline{v} + 1.96s]$.

Remark 4.2. Like any Monte-Carlo type methods, the convergence rate is $1/\sqrt{M_f}$, which is slow. In addition, the number of subproblems to solve in the forward pass is proportional to the number of forward pass trajectories M_f. Therefore, some variance reduction techniques are very useful here. In practice, we use a quite efficient control variate technique, see section 5.

Remark 4.3. Since the calculation of each trajectory is independent, it is possible to use parallel computing in order to speed up the calculation.

4.2 Backward pass

The goal of backward pass is to update the optimality cuts used in (27). It follows the quantization tree, i.e. in sample calculation. One still uses the formulation of sub-problem (27), where $\hat{\boldsymbol{\xi}}_t \in \Gamma_t$, and x_t have been stored during the previous forward pass. Once the sub-problem (27) has been solved, the optimality cut is computed from the dual value associated to the dynamic equation $A_t u_t + x_t = x_{t+1}$. We then add this cut into the optimality cuts collection of vertex M_t^i.

Remark 4.4. Optimality cuts have to be updated for each vertex of the quantization tree due to the non-convexity of $Q(t, x_t, \boldsymbol{\xi}_t)$ (and $\hat{Q}(t, x_t, \hat{\boldsymbol{\xi}}_t)$) w.r.t. $\boldsymbol{\xi}_t$ (and $\hat{\boldsymbol{\xi}}_t$). The number of sub-problems to solve in the backward pass is proportional to the quantization tree size N times the number of trajectories denoted M_b. Since the state variable values are stored during the forward pass, M_b is always inferior to M_f. But if we are using all M_f trajectories for the backward pass (M_f is relatively large owing to remark 4.2), the number of sub-problems would be $N \times M_f$ which is computationally unmanageable. So, one idea is to select only a small subsample M_b samples to be computed in the backward pass in order to allow the program to finish in a reasonable time. Another reason is that in practise, some state values x_t are very close one another, so the sub-problem will return almost the same dual information which will not help us much well approximate $\hat{Q}(t, x_t, \hat{\boldsymbol{\xi}}_t)$.

However, the process of selecting M_b trajectories among the M_f ones depends on the nature of the problem on which the algorithm is applied. In the numerical tests in section 5, the selection follows the rejection method which consists in checking whether the distance between the new x_t and the existing ones is below a predefined threshold or not. This is a random way with the requirement that they are not too close each other.

Remark 4.5. Similarly to the forward pass, the backward pass algorithm may also use parallel computation, because at stage t, each sub-problems (26) using different (x_t, ξ_t^i) are independent from each other.

After solving the sub-problem at the first stage, an optimal value $\underline{v}$ is obtained, which is a lower bound for the original problem, since this value is calculated by Bender's decomposition. However, $\underline{v}$ is calculated according to the quantization tree, and would differ if calculated on another tree. The error caused by this approximation can be bounded, as discussed in the next subsection.

Remark 4.6. The stopping condition is to check whether $\underline{v} \in [\overline{v} - 1.96s, \overline{v} + 1.96s]$. However, the confidence interval is sometimes too large making this condition too easy to satisfy. To avoid this problem, one adds another stopping condition stating that the backward pass value $\underline{v}$ should be stable: $|\underline{v}^{it} - \underline{v}^{it-1}| \leq \epsilon |\underline{v}^{it}|$, where $it \in \mathbb{Z}_{++}$ is the iteration number.

4.3 Error analysis on quantization

In order to estimate the error caused by quantization, we refer to the result in H. Heitsch and W. Römisch [10, 9].

Theorem 4.7. *[10, Theorem 2.1] Assume that the solution set $\mathfrak{U}^{ad}$ is nonempty, and bounded. Then, there exists positive constants L and δ such that the estimate*

$$|v(\boldsymbol{\xi}) - v(\hat{\boldsymbol{\xi}})| \leq L(\|\boldsymbol{\xi} - \hat{\boldsymbol{\xi}}\|_2 + D_f(\boldsymbol{\xi}, \hat{\boldsymbol{\xi}})) \tag{30}$$

holds for all random element $\hat{\boldsymbol{\xi}} \in L^2(\Omega, \mathcal{F}, \mathbf{P}; \mathbb{R}^d)$ with $\|\boldsymbol{\xi} - \hat{\boldsymbol{\xi}}\|_2 \leq \delta$, where

$$D_f(\boldsymbol{\xi}_t, \hat{\boldsymbol{\xi}}_t) = \inf_{u \in S(\boldsymbol{\xi}), \hat{u} \in S(\hat{\boldsymbol{\xi}})} \sum_{t=2}^{T-1} \max \left\{ \|u_t - \mathbb{E}[u_t \tilde{v} \mathcal{F}_t(\hat{\boldsymbol{\xi}})]\|_2, \|\hat{u}_t - \mathbb{E}[\hat{u}_t \tilde{v} \mathcal{F}_t(\boldsymbol{\xi})]\|_2 \right\}, \tag{31}$$

where $S(\boldsymbol{\xi})$ (resp. $S(\hat{\boldsymbol{\xi}})$) is the optimal solution of with random variable $\boldsymbol{\xi}$(resp. $\hat{\boldsymbol{\xi}}$), and $v(\boldsymbol{\xi})$ (resp. $v(\hat{\boldsymbol{\xi}})$) are the related optimal values.

Since $(\hat{\boldsymbol{\xi}}_t)$ is the quantization of $(\boldsymbol{\xi}_t)$, then $\mathcal{F}_t(\hat{\boldsymbol{\xi}}) \subset \mathcal{F}_t(\boldsymbol{\xi})$. The optimal control $\hat{u}_t$ is $\mathcal{F}_t(\hat{\boldsymbol{\xi}})$ measurable, hence it is $\mathcal{F}_t(\boldsymbol{\xi})$ measurable. Then, we have for (31):

$$D_f(\boldsymbol{\xi}_t, \hat{\boldsymbol{\xi}}_t) = \inf_{u\in S(\boldsymbol{\xi}), \hat{u}\in S(\hat{\boldsymbol{\xi}})} \sum_{t=2}^{T-1} \|u_t - \mathbb{E}[u_t \tilde{v} \mathcal{F}_t(\hat{\boldsymbol{\xi}})]\|_2 \tag{32}$$

Our framework is a little different from the problem studied in [10, 9], since our problems have the final cost $g(\boldsymbol{\xi}_T, x_{T+1})$. But however, their result is suitable here if we have the Lipschitz property on this function. The proof is similar with the one in their paper.

This error bound consists in 2 terms, the first one of right hand side (30) is the l^2-distance between the original random process and the approximated one, the second one is the distance between the 2 filtrations over the optimal control. The first term can be measured by theorem 3.1, which provides the convergence rate to 0. However, it is very difficult to provide a convergence rate to (32).

4.4 Algorithm

Here is the whole algorithm (fig. 1):

Step 0	**Construction quantization tree (possible off-line)**
	build one quantization tree and compute transition probability.
Step 1	**Initialisation**
	let $u_1 \in \mathfrak{U}_1^{ad}$ the current optimal solution and compute state variable value x_2;
	let $\underline{v}^0 = -\infty$.
Step 2	**Forward pass**
	create M_f diffusion process following dynamic (1);
	for $t = 2, \ldots, T$ do:
	for $m = 1, \ldots, M_f$ do:
	calculate $\hat{\boldsymbol{\xi}}_t^m = \mathbf{Proj}_{\Gamma_t}(\boldsymbol{\xi}_t^m)$,
	solve the subproblem (27);
	compute the forward statistic $\overline{v}$ and s.
Step 3	**Backward pass**
	for $t = T, \ldots, 2$ do:
	for $m = 1, \ldots, M_b$ do:
	solve the subproblem (27) for all vertices in Γ_t;
	calculate new optimality cut and add it into M_t;
	$t = 1$
	solve the first stage problem and calculate the backward value $\underline{v}^{it}$.
Step 4	**Check stopping condition**
	if $\underline{v} \in [\overline{v} - 1.96s, \overline{v} + 1.96s]$ and $\lvert\underline{v}^{it} - \underline{v}^{it-1}\rvert \leq \epsilon \lvert\underline{v}^{it-1}\rvert$
	STOP;
	else
	go to Step 2.

Figure 1: Algorithm

5 Numerical tests

In this section, two numerical tests are performed. The first one is a swing option, which is an example where both state variable and random variable are scalar. The second one concerns a LNG portfolio optimization, which is far more complex. The algorithm has been successfully applied on both examples.

5.1 Price model

First, we introduce the price modeled as a Markov chain $(\boldsymbol{\xi}_t)$ in the following numerical tests. It takes as inputs the market volatilities and the forward curve. The forward price at time s with maturity t, $F(s,t)$ follows under the equivalent martingale measure (EMM) $\mathbf{Q}$ the following dynamic:

$$\ln F^i(s+1,t) - \ln F^i(s,t) = \sum_{j=1}^{d} \frac{\sigma_{ij}}{\sqrt{T-1}} W_s^j - \frac{1}{2}\sum_{j=1}^{d} \frac{\sigma_{ij}^2}{T-1} \quad i = 1,\dots,d \quad (33)$$

where $\sigma \in \mathbb{R}^{d\times d}$ is the volatility of the whole period, W_s is a $d-$dimension i.i.d. random variable following a normal distribution $N(0,\mathbf{1})$.

Hence, each of the forward price is a martingale under $\mathbf{Q}$ and the spot prices at time t are given by $\boldsymbol{\xi}_t = F(t,t)$:

$$\boldsymbol{\xi}_t^i = F^i(t,t) = F^i(1,t)\exp\left(\frac{\sigma_i}{\sqrt{T-1}}\cdot(\sum_{s=1}^{t-1} W_s) - \frac{1}{2}\frac{|\sigma_i|^2}{T-1}(t-1)\right) \quad (34)$$

5.2 Swing option

Swing options are fairly common in the energy market. It allows its holder to purchase a total amount of commodity during a fixed time period. A basic swing option contract can be formulated as:

$$\begin{aligned} \sup_{(u_t)} \quad & \mathbb{E}\left[\sum_{t=1}^{T}(\boldsymbol{\xi}_t - K_t)\cdot u_t\right] \\ s.t. \quad & u_t \in [0,1] \quad \text{and} \quad \mathcal{F}_t - \text{measurable} \quad \forall t \\ & \sum_{t=1}^{T} u_t \in [L,U] \end{aligned} \quad (35)$$

where K_t is the strike price, assumed to be deterministic, L and U are lower bound and upper bound of the final condition.

Let us first flip the sign of the objective function to $(K_t - \boldsymbol{\xi}_t)$ in order to switch it to a minimization problem. In this test, we consider that $K_t = F(1,t)$ without loss of generality.

In the case when $(L,U) \in \mathbb{Z}^2$, Bardou et al [3] show that the optimal control is bang-bang, i.e. $u_t \in \{0,1\}$. Thus, dynamic programming can be directly applied in this framework by discretizing the state space x_t. Bardou et al. have applied this idea for the state variable and use a quantization tree to discretize the underlying random space. However, in the presence of more complex conditions on the control variable such as $u_t \in [l(x_t), u(x_t)]$ (like in most of storage capacities technical features), the bang-bang strategy is still available but the discretization of the state space x_t may become intractable.

5.2.1 Control variate

Since the forward pass produces a large confidence interval with a small M_f, the following control variate technique has been introduced to reduce it. Let us consider the following variable:

$$v_{cv}(\boldsymbol{\xi}) = \sum_{t=1}^{T}(K_t - \boldsymbol{\xi}_t). \tag{36}$$

Observe that $E[v_{cv}(\boldsymbol{\xi})] = 0$ since $\boldsymbol{\xi}_t$ is a martingale with expectation K_t.

5.2.2 Numerical result

In the following test, without losing any generality, we take $F(1,t) = 1, \forall t = 1,\dots,T$ in the price model (33). The results are compared to another method using bang-bang strategy and the Longstaff-Schwartz method [16]. During the simulation, the quantization tree has 3000 points in total. We set parameters $M_f = 1000$ and $M_b = 10$. We use 1000 scenarios in the Longstaff-Schwartz method.

$T = 50$		$[L,U] = [0,25]$			$[L,U] = [20,30]$		
		$\sigma = 0.5$	$\sigma = 0.75$	$\sigma = 1$	$\sigma = 0.5$	$\sigma = 0.75$	$\sigma = 1$
SDDP.	$\underline{v}$	4.31	6.43	8.49	1.96	2.93	3.88
	c.i.	[4.04, 4.40]	[5.91, 6.51]	[7.86, 8.62]	[1.77, 1.97]	[2.66, 3.00]	[3.68, 4.16]
BB+LSM.	c.i.	[4.19, 4.47]	[6.23, 6.71]	[8.22, 8.94]	[1.81, 2.17]	[2.69, 3.31]	[3.57, 4.42]

Figure 2: Quantization tree + dual dynamic programming versus bang-bang strategy + Longstaff-Schwartz method in swing option pricing with control variate

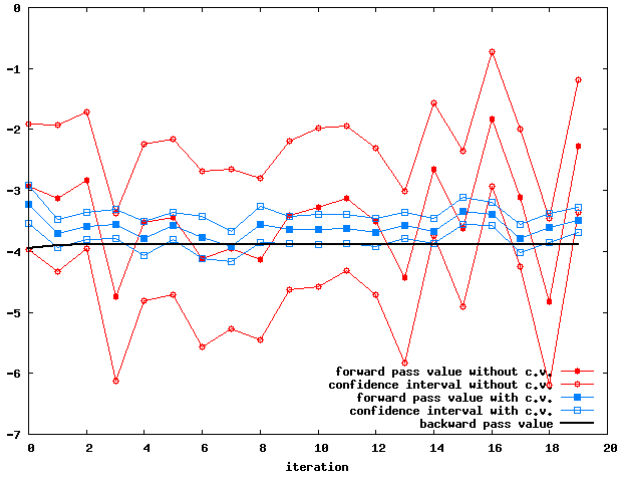

Figure 3: Result of swing option pricing using quantization method and dual dynamic programming. $[L,U] = [20,30]$, $\sigma = 1.0$

We show in fig. 3 how the program behaves during 20 iterations. We observe that the backward value is stable from 3rd iteration and that most of the following backward values belong to the confidence interval obtained without control variable. Also, some of them belong to the confidence interval obtained with control variable.

5.3 Dynamic portfolio optimization

We now consider a gas trading portfolio. A trading company purchases liquefied natural gas(LNG.) from a set of producing countries indexed at a price formula and sells it to consuming countries at another other price formula (see fig. 4 for the main market and tab. 1 for the price formulae). Annual quantity and price formulae have been agreed contractually, the latter are functions of the future prices of major energy markets $\boldsymbol{\xi}_t$, such as crude oil price (OIL), north American natural gas price (NA NG), etc. The goal is to find a strategy that optimizes the portfolio value. The decision is made over the quantity of LNG transported on each route. *At that point, we relax the discrete nature of the decision variables.* The first difficulty lies in the number of indices we have to consider when we model the portfolio: $\boldsymbol{\xi}_t \in \mathbb{R}^2$. The second difficulty is linked to the dimension of the state variable $x_t \in \mathbb{R}^m$, corresponding to the total number of production and delivery countries, which is also in high dimension, $m = 6$ in the simulation.

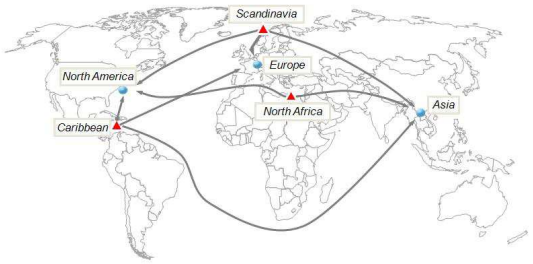

Figure 4: A fictive supply and demand portfolio, as well as the possible route. ▲: producing country; •: consuming country.

Port	Cargo Size	Annual QC. $(TBtu^1)$	Monthly QC. $(TBtu^1)$	Price formula $(\$/MMBtu^1)$
Caribbean	3.0	[46.0, 54.0]	[0.0, 6.0]	NA NG $-$ 0.1
Scandinavia	3.0	[21.0, 29.0]	[0.0, 3.0]	$\begin{cases} 0.05\text{OIL} + 2.5 & \text{if OIL} \leq 75 \\ 0.07\text{OIL} + 1.0 & \text{elsewise} \end{cases}$
North Africa	4.0	[96.0, 100.0]	[0.0, 12.0]	$\begin{cases} 0.9\text{NA NG} + 0.4 & \text{if NA NG} \leq 5 \\ 0.8\text{NA NG} + 0.9 & \text{otherwise} \end{cases}$
North American	3.0, 4.0	[81.0, 89.0]	[0.0, 8.0]	NA NG
Europe	3.0, 4.0	[66.0, 74.0]	[0.0, 7.0]	$0.055\text{OIL} + 0.8$
Asia	3.0, 4.0	[16.0, 24.0]	[0.0, 4.0]	$0.08\text{OIL} - 0.8$

Table 1: Constraints and price formulae

So, the problem is

$$\begin{aligned} \sup_{(u_t)_{t\in[1,T]}} \quad & \mathbb{E}\left[\sum_{t=1}^{T} \tilde{c}_t(\boldsymbol{\xi}_t)\cdot u_t\right] \\ \text{s.t.} \quad & \sum_{t=1}^{T} A_t u_t \in \mathfrak{X}_{T+1} \\ & u_t \in \mathfrak{U}_t \quad \text{and} \quad \mathcal{F}_t - \text{measurable} \end{aligned} \tag{37}$$

Let us flip the sign of the objective function: $c_t = -\tilde{c}_t$ to switch to a minimization problem.

The forward price $F(1,t)$ in (33) is read from the energy market. Here is the value used in test:

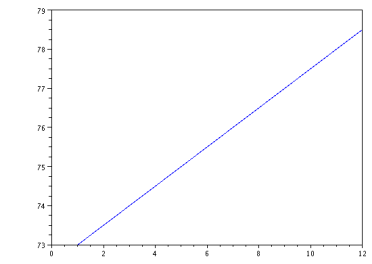

Figure 5: OIL price (US$ /barrel)

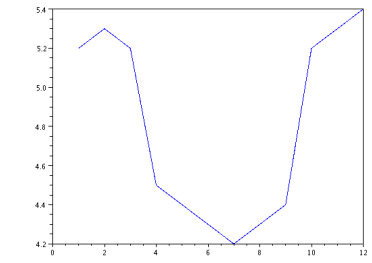

Figure 6: NA NG price (US$ / MMBtu)

Please notice that it is sufficient to build a tree for $\log(\boldsymbol{\xi}_t)$, which follows a Gaussian distribution. The quantization tree was processed with $N = 8000$ $\mathbb{R}^2$-valued elementary quantizers dispatched on $T = 12$ stages. The estimation of transition probability was carried out by Monte-Carlo method using 10^9 samples.

5.3.1 Control variate

Similarly to the previous example, we keep using a control variate to reduce the large confidence interval. Let us consider the following variable:

$$v_{cv}(\boldsymbol{\xi}) = \sum_{t=1}^{T} c_t(\boldsymbol{\xi}_t) u_t^* \tag{38}$$

[1] MMBtu stands for a million british thermal unit, a TBtu is a tera british thermal unit thus equivalent to a Million MMBtu.

where u_t^* is one of the optimal solutions for problem

$$\begin{aligned} \inf_{(u_t)_{t\in[1,T]}} \quad & \sum_{t=1}^{T} \mathbb{E}\left[c_t(\boldsymbol{\xi}_t)\right] u_t \\ \text{s.t.} \quad & \sum_{t=1}^{T} A_t u_t \in \mathfrak{X}_{T+1} \\ & u_t \in \mathfrak{U}_t \qquad t = 1, \ldots, T \end{aligned} \tag{39}$$

Let v_{cv}^* be the optimal value.

This is a deterministic linear program, which is straightforward to solve. The only technical point is the way to evaluate $\mathbb{E}[c_t(\boldsymbol{\xi}_t)]$. One possible method is using quantization:

$$\mathbb{E}[c_t(\boldsymbol{\xi}_t)] \approx \sum_{\xi_t^i \in \Gamma_t} p_t^i c_t(\xi_t^i)$$

Clearly one has $\mathbb{E}[v_{cv}(\boldsymbol{\xi})] = v_{cv}^*$.

5.3.2 Results

The process parameters in the test are volatility $\sigma_1 = \sigma_2 = 66\%$, and correlation $\rho_{12} = 0.3$. We apply algorithm fig.1, with the following parameters: forward pass samples $M_f = 3000$ to 6000, backward pass sample $M_b = 8$ to 12. The program is done in C++ with Cplex 10.1, and the tests were performed on a PC with a 2.2 GHz Dual Core AMD CPU and 16GByte main memory.

The numerical result is (fig. 7):

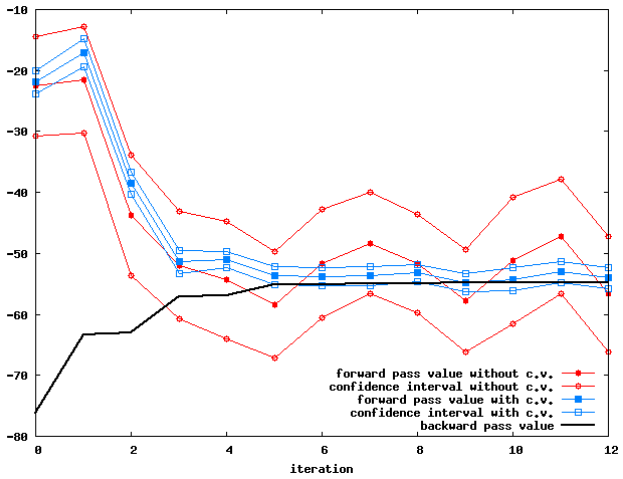

Figure 7: result on iterations

Notice that the value of the backward pass is almost stable from the 6th iteration and most of the following values are within the confidence interval of the forward value generated without control variate, which represents circa 20% of the forward pass value. This is extremely important. However, the confidence interval generated with control variate represents circa 2% of the

	upper bound	confidence interval of u.b.	lower bound	control variate value
Continuous	-54.0464	1.79238	-54.7472	-31.5404
Integer	-31.8133	4.29456		-28.0352

Table 2: Comparison of optimal value of continuous problem and integer problem

forward pass value. This shows that the control variate proves to be adequate in this example. Moreover, the algorithm converges in 13 iterations, which is very efficient. However, if we test this algorithm with larger volatilities, or with more producing/consuming countries, it will require more iterations to converge.

Below are the graphs representing the CPU time of each iterations (fig. 8 and fig. 9):

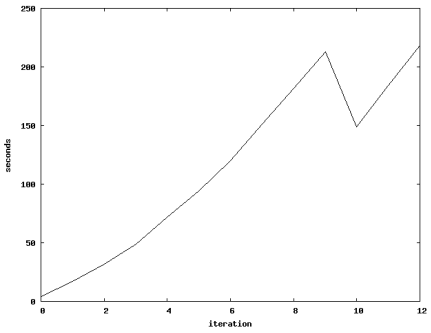

Figure 8: forward CPU time

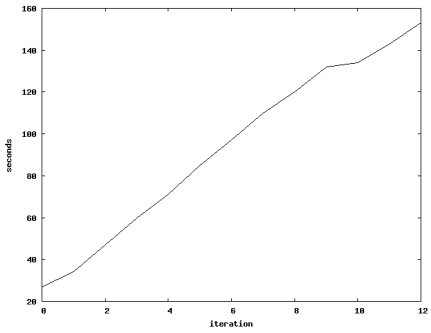

Figure 9: backward CPU time

This program does not consume much time in each iteration. However the calculation time increases sharply when the iteration goes on. We know that the forward computation time is proportional to the number of forward samples M_f, and that the backward computing time is proportional to the number of vertices in the quantization tree N and to the number of backward samples M_b. Hence, we can modify these parameters to reduce the calculation time when it becomes too long. But if we do so, we will loose the precision by enlarging the confidence interval.

Integer solution As explained in section 2.5, we can compute feasible and sub optimal integer solutions by combining the Bellman values of the continuous relaxation and integer constraints on the decision variables during the forward step. The results are represented in table 2. The control variate value of -28.0352 is obtained when solving (39) taking account of integer constraints on control variable. The heuristic gives an expected value of -31.8133, with a 95% asymptotic confidence interval of 4.29456, with 5000 forward samplings.

References

[1] V. Bally and P. Pagès. Error analysis of the quantization algorithm for obstacle problems. *Stochastic Processes & Their Applications*, 106(1):1–40,

2003.

[2] V. Bally and P. Pagès. A quantization algorithm for solving discrete time multi-dimensional optimal stopping problems. *Bernoulli*, 9(6):1003–1049, 2003.

[3] O. Bardou, S. Bouthemy, and G. Pagès. When are swing option bang-bang and how to use it?, 2007.

[4] D.P. Bertsekas and J.N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996.

[5] J.R. Birge and F. Louveaux. *Introduction to Stochastic Programming*. Springer, New York, 2000.

[6] J. Dupačová, N. Gröwe-Kuska, and W. Römisch. Scenario reduction in stochastic programming–an approach using probability metrics. *Math. Program. Ser. A*, 95:493–511, 2003.

[7] S. Graf and H. Luschgy. *Foundations of quantization for probability distributions*. Springer-Verlag, Berlin, 2000.

[8] H. Heitsch and W. Römisch. Scenario reduction algorithms in stochastic programming. *Computational Opti. and Appli.*, 24:187–206, 2003.

[9] H. Heitsch and W. Römisch. Stability and scenario trees for multistage stochastic programs, 2006. preprint 324, DFG Research Center Matheon "Mathematics for key technologies".

[10] H. Heitsch, W. Römisch, and C. Strugarek. Stability of multistage stochastic programs. *SIAM J. on Optimization*, 17:511–525, 2006.

[11] K. Hoyland, M. Kaut, and S.W. Wallace. A heuristic for moment-matching scenario generation. *Computational Optimization and Applications*, 24:169–185, 2003.

[12] G. Infanger and D.P. Morton. Cut sharing for multistage stochastic linear programs with interstage dependency. *Mathematical Programming*, 75:241–256, 1996.

[13] P. Kall and S.W. Wallace. *Stochastic Programming*. John Wiley & Sons, Chichester, 1994.

[14] M. Kaut and S.W. Wallace. Evaluation of scenario-generation methods for stochastic programming, 2003. [Online: Stand 2008-09-04T17:04:18Z].

[15] C. Küchler and S. Vigerske. Decomposition of multistage stochastic programs with recombining scenario trees.

[16] F. Longstaff and E. Schwartz. Valuing american options by simulation : a simple least squares approach. *Review of Financial Studies*, 14:113–148, 2001.

[17] M.V.F. Pereira and L.M.V.G. Pinto. Multi-stage stochastic optimization applied to energy planning. *Mathematical Programming*, 52:359–375, 1991.

[18] A.B. Philpott and Z. Guan. On the convergence of stochastic dual dynamic programming and related methods. *Operation Research Letters*, 2008.

[19] R.T. Rockafellar and R. J-B. Wets. *Variational analysis*. Grundlehren der mathematischen Wissenschaften. Springer-Verlag, first edition, 1997.

[20] B. Van Roy and J.N. Tsitsiklis. Regression methods for pricing complex american-style options. *IEEE transactions on Neural Networks*, 12(4):694–703, 2001.

[21] A. Shapiro. Analysis of stochastic dual dynamic programming method, 2009.

[22] A. Shapiro, D. Dentcheva, and A. Ruszczyński. *Lectures on Stochastic Programming : Modelling and Theory*. SIAM, Philadelphia, 2009.

[23] J.N. Tsitsiklis. An analysis of temporal-difference learning with function approximation. *IEEE transactions on automatic control*, 42(5):674–690, 1997.

[24] J.N. Tsitsiklis and B. Van Roy. Optimal stopping of Markov processes : Hilbert space theory, approximation algorithms, and an application to pricing high-dimensional financial derivatives. *IEEE transactions on automatic control*, 44(10):1840–1851, 1999.

Centre de recherche INRIA Saclay – Île-de-France
Parc Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 Orsay Cedex (France)

Centre de recherche INRIA Bordeaux – Sud Ouest : Domaine Universitaire - 351, cours de la Libération - 33405 Talence Cedex
Centre de recherche INRIA Grenoble – Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier
Centre de recherche INRIA Lille – Nord Europe : Parc Scientifique de la Haute Borne - 40, avenue Halley - 59650 Villeneuve d'Ascq
Centre de recherche INRIA Nancy – Grand Est : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex
Centre de recherche INRIA Paris – Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex
Centre de recherche INRIA Rennes – Bretagne Atlantique : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex
Centre de recherche INRIA Sophia Antipolis – Méditerranée : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399