A Multi-Quantile Approach for Open-Loop Stochastic Dynamic Programming Problems ⋆

Rodrigo T. N. Cardoso *,**** **Ricardo H. C. Takahashi** **
Frederico R. B. Cruz *** **Carlos M. Fonseca** ****,†

* *Department of Physics and Mathematics, Centro Federal de Educação Tecnológica de Minas Gerais, MG Brazil (e-mail: rodrigoc@des.cefetmg.br)*
** *Department of Mathematics, Universidade Federal de Minas Gerais, (e-mail: taka@mat.ufmg.br)*
*** *Department of Statistics, Universidade Federal de Minas Gerais, (e-mail: fcruz@est.ufmg.br)*
**** *Faculty of Science and Technology, Universidade do Algarve, Portugal, (e-mail: cmfonsec@ualg.pt)*
† *CEG-IST, Instituto Superior Técnico, Universidade Técnica de Lisboa, Portugal*

Abstract: This paper proposes a multi-quantile approach for solving open-loop continuous-variable discrete-time stochastic dynamic programming problems in systems with non-standard probability distribution functions. Instead of using the expected value of the objective function for building the optimization criterion, the decision maker performs a choice on the decision variables over the objective function value quantiles. The proposed procedure relies on a Monte Carlo simulation of the unknown process input outcomes, associated with an open-loop multiobjective optimization. The optimal control comes from a trade-off analysis that considers, for instance, the risk associated with each policy versus its yield.

Keywords: Stochastic Dynamic Programming, Multiobjective Optimization, Model Predictive Control.

1. INTRODUCTION

Stochastic dynamic programming problems are usually formulated as the multi-period optimization of the expected value of an objective function that is additive over time, constrained by a stochastic dynamic system [Bertsekas (1995)]. This procedure makes sense when the stochastic variables that drive the dynamic system are endowed with properties that allow some kind of “certainty-equivalence” [Bertsekas (2005)].

The present paper considers stochastic dynamic systems driven by variables with arbitrary probability distribution. Inspired in the stochastic dominance concept [Levy (1992)], and using Monte Carlo simulation of the stochastic variables, a multiobjective optimization is performed taking into account the quantiles of the objective function. This allows trade-offs between the risk and the expected yield to be considered in the decision-making. The dynamic system input variables are chosen in an open-loop scheme. Both the dynamic constraint and the end-point constraint are described by an inequality considering the pre-image of a closed set around the pre-established goal [Bertsekas and Rhodes (1971)]. This procedure is similar to the model predictive control technique – which has been shown to present suboptimal dynamic programming behavior [Camacho and Bordons (2004); Bertsekas (2005)].

This article is structured as follow: Section 2 states the problem. The proposed methodology is explained in Section 3. The inventory control, a classical case study, is considered in Section 4. Finally, Section 5 presents the final discussions, the conclusions and topics for future research.

⋆ This work was supported by Brazilian Agencies CNPq, CAPES and FAPEMIG, and the Portuguese Agency FCT.

2. PROBLEM STATEMENT

Define the cost function J as:

$$J = \sum_{k=0}^{N-1} g_k\big(x[k], u[k]\big) + g_N\big(x[N]\big). \tag{1}$$

The dynamic optimization problem is stated here as:

$$\min_{u[0],\ldots,u[N-1]} Q_\alpha\left(J\right) \tag{2}$$

subject to:

$$\begin{cases} x[k+1] = f\big(x[k], u[k], w[k]\big), \\ k = 0, 1, \ldots, N-1; \\ x[0] = x_0 \text{ and } ||x[N] - x^*||_\infty \le \epsilon \end{cases} \tag{3}$$

in which: $k = 0, \ldots, N$ are the time stages; N is the horizon; $x[k]$ is the state variable at stage k with dimension n; $u[k]$ is the decision variable at stage k with dimension p; $w[k]$ is the random disturbance variable at stage k with dimension q. The probability distribution function of $w[k]$ is denoted by $\psi(k)$, and is supposed to be known. The function $Q_\alpha(\cdot)$ represents the α-quantile of its stochastic argument. The constraint $x[N] = x^*$

is called the *end-point constraint*, with x^* a given goal that the state vector should reach. As in Bertsekas and Rhodes (1971), it is supposed that the end-point constraint is relaxed through a square-ball of radius ϵ around the given goal.

Other constraints may be considered:

$$h_i(x[0], \ldots, x[N], u[0], \ldots, u[N-1]) \leq 0, \tag{4}$$

with each $h_i : \mathbb{R}^{(N+1)\times n+N\times p} \mapsto \mathbb{R}$, for $i = 1, \ldots, m$.

Denote by $\mathcal{F}$ the feasible set of decision variables $\mathcal{U} = [u[0], \cdots, u[N-1]]$ that obey (3) and (4).

The problem (2) becomes a *multiobjective optimization problem* (MOP) when the simultaneous minimization of several quantiles $Q_\alpha(J)$ is considered, for $\alpha = \{\alpha_1, \ldots, \alpha_r\}$:

$$\min_{u[0],\ldots,u[N-1]} Q_\alpha(J) = (Q_{\alpha_1}(J), \ldots, Q_{\alpha_r}(J)) \tag{5}$$

Consider the vectorial function $F(\mathcal{U}) = Q_\alpha(J(\mathcal{U}))$, $F : \mathcal{F} \mapsto \mathbb{R}^r$. For defining the solution of MOPs, define the *dominance* relation, $\mathcal{U}_a \prec \mathcal{U}_b$ ($\mathcal{U}_a$ dominates $\mathcal{U}_b$), which means that given two different decision variable vectors, $\mathcal{U}_a$ and $\mathcal{U}_b$ (which lead respectively to the cost vectors $F(\mathcal{U}_a)$ and $F(\mathcal{U}_b)$), then we have $F_i(\mathcal{U}_a) \leq F_i(\mathcal{U}_b),\ \forall\ i = 1, 2, \ldots, r$ and $F_i(\mathcal{U}_a) < F_i(\mathcal{U}_b)$, for some $i = 1, 2, \ldots, r$, in which r is the total number of objectives of the problem. MOPs are stated as problems of finding a solution inside a set $\mathcal{P}$, defined as the set of all decision vectors that are not dominated by any other one:

$$\mathcal{P} \triangleq \{\mathcal{U} \in \mathcal{F} \mid \nexists\, \mathcal{U}_a \in \mathcal{F} \text{ such that } \mathcal{U}_a \prec \mathcal{U}\}. \tag{6}$$

The set $\mathcal{P}$ is called the *Pareto-optimal set*. The image-set of the Pareto-optimal set $\mathcal{P}$ by the objective function F, or $F(\mathcal{P})$, is the *Pareto-front*.

It should be mentioned that a relation of dominance between the probability distribution functions of two stochastic variables has been defined as the *stochastic dominance* relation [Levy (1992)]. Consider $Q_{\alpha_i}(F)$ e $Q_{\alpha_i}(G)$ representing the α_i-quantile of the accumulated distributions F and G, respectively. It is said that function F has stochastic dominance over function G if $Q_{\alpha_i}(F) \geq Q_{\alpha_i}(G)$, for all i, with the strict inequality holding for at least one α_i. The comparison considering a finite number of quantiles, as in (5), can be considered an approximation of the comparison by the stochastic dominance.

In multi-objective dynamic programming problems, each non-dominated solution corresponds to a non-dominated policy. For instance, Li and Haimes (1989) presented a survey on multi-objective dynamic programming and Trzaskalik and Sitarz (2007) proposed a procedure that considers a partially ordered criteria structure in dynamic programming. However, the approach proposed here is out of the traditional multi-objective discrete-time dynamic programming methods.

2.1 The linear case

An important case occurs when the dynamics is linear:

$$x[k+1] = Ax[k] + Bu[k] + Cw[k], \tag{7}$$

in which A, B and C are matrices with appropriate dimensions. Because the dynamic optimization is performed here in open-loop, each state $x[k]$ in (1), (3) and (4) can be re-written as:

$$x[k] = x[0] + \sum_{j=0}^{k-1} A^j \left(Bu[k-j-1] + Cw[k-j-1]\right). \tag{8}$$

For a disturbance sequence fixed in the expected values $\{\hat{w}[0], \ldots, \hat{w}[N-1]\}$, the constraint (3) is:

$$\begin{cases} e_i^T \big(A^N x_0 + \sum_{k=0}^{N-1} A^k Bu[N-1] + \\ + \sum_{k=0}^{N-1} A^k C\hat{w}[N-i-1] - (x^* + e_i\epsilon)\big) \leq 0, \\ \\ -e_i^T \big(A^N x_0 + \sum_{k=0}^{N-1} A^k Bu[N-k-1] + \\ + \sum_{k=0}^{N-1} A^k C\hat{w}[N-i-1] - (x^* - e_i\epsilon)\big) \leq 0, \\ \\ i = 1, \ldots, n; \\ \\ x_0 \text{ and } x^* \text{ are given;} \end{cases} \tag{9}$$

with e_i denoting the i-th canonical basis vector.

3. THE PROPOSED APPROACH

This paper proposes a computationally tractable scheme for stochastic dynamic programming problems formulated as in (5). This procedure fits in the *model predictive control* [Camacho and Bordons (2004); Bertsekas (2005)], a technique which has been developed in the field of automatic control in the 1980's [Garcia et al. (1989)], and whose applications come mainly from the field of chemical engineering. The main distinctive features of the proposed procedure are:

- Arbitrary probability distribution functions are allowed for the stochastic disturbances, instead of the usual assumption of specific pdf's (in most of the cases, Gaussian pdf's);
- The trade-off analysis between risk and yield is explicitly delivered as the outcome of the proposed algorithm, instead of the traditional outcome that delivers the control sequence that maximizes the "expected value" of the yield.

Using this scheme, the proposed strategy has to be run once by time stage, in order to account for the new information that becomes available each time a stochastic variable is instantiated. An implicit *feedback effect*, in this case, comes from: for each stage $k = 0, \ldots, N-1$, as soon as the actual state $x[k]$ becomes known, run the optimization procedure, obtaining the sequence of control actions $\{u[k], \ldots, u[N-1]\}$, and apply just the actual decision variable $u[k]$. This means that, although generating open-loop control actions, the proposed strategy becomes virtually closed-loop with the step-by-step updating of the empirically observed problem variables feeding a new run at each step.

It is known that open-loop procedures can deal with rather general shapes of objective function and constraints on the states or decision variables [Bertsekas (1995)], since mathematical programming techniques or evolutionary algorithms are available for different problems with different features. This article proposes considering the stochastic disturbance via *Monte Carlo simulations*, obtainable within a quite reasonable amount of computational effort [Ross (2002)]. Monte Carlo simulation is a well-known and useful method to determine probabilities by using highly intensive computational experiments. Solving a dynamic programming problem by means of simulation is not a novelty though. Indeed, neuro-dynamic programming is a well-known dynamic programming approach that employs Monte Carlo sampling in stochastic settings [Bertsekas and Tsitsiklis (1996)], among other ones.

The proposed scheme for the calculus of the control action of each stage may be summarized as:

Proposed Scheme:

For each stage $k = 0, \dots, N-1$:

1. Generate randomly several sequences of disturbances $\{w[k], \dots, w[N-1]\}$ with the given probability distribution function;
2. As soon as the empirical state $x[k]$ becomes known, perform the multiobjective optimization (5) in open-loop. Different quantiles of objective function are taken as different objective functions, and non-dominated control sequences $\{u[k], \dots, u[N-1]\}$ are selected. The objective function is evaluated using all disturbances generated in step 1.
3. A trade-off analysis is performed, considering a decision criterion, for instance, a risk measure of the non-dominated decision variables, evaluated from the different quantiles. Choose the next step which optimizes the decision criteria. Apply just the next decision variable $u[k+1]$.
4. Make $k \leftarrow k+1$, and return to step 1.

It is remarkable that: **(i)** the convergence of the proposed scheme depends on the convergence of the Monte Carlo simulations [Ross (2002)], and on the chosen open-loop multiobjective optimization procedure [Chankong and Peng (1983)]; **(ii)** the optimal open-loop cost is an upper bound for the optimal closed-loop cost [Bertsekas (2005)].

3.1 The optimization engine

In this work, a multiobjective genetic algorithm [Fonseca and Fleming (1995)] is employed for the open-loop optimization. An algorithm of this class has been chosen due to some nice characteristics that those algorithms present:

- They do not require “strong properties” of the objective functions, like convexity, smoothness or unimodality. Instead, the only requirement is usually stated as that the objective functions present “weak locality” (which roughly means that the function values should present autocorrelation that decreases with the distance in the decision variable space).
- They deliver an entire set of estimates of the Pareto-optimal set in a single run.

Multiobjective genetic algorithms work according to the following general scheme delineated in Algorithm 1.

The specific version of multiobjective genetic algorithm employed in this work was the *Nondominated Sorting Genetic Algorithm* (NSGA-II), presented in [Deb et al. (2002)]. As distinctive features, the NSGA-II has introduced a fast non-dominated sorting procedure, an elitism-preserving selection, and a parameterless niching operator for diversity preservation (crowding distance comparison operator), leading to an enhanced computational complexity. NSGA-II also incorporates simple and efficient penalty-parameterless approach for solving constraints.

Algorithm 1 Pseudocode for Multiobjective Genetic Algorithm

• A set of initial tentative solutions is generated randomly. The tentative solutions are the *individuals* and the whole set is the *population*.

while (not stop criterion) **do**

  • The objective functions are evaluated, and the dominance relations among all individuals are computed. “Fitness values” are assigned to all individuals in the population, with the greater values assigned to the non-dominated individuals.

  • Individuals of the population are chosen randomly for composing a new population, with the relative “chances” depending on their fitness values. In this way, the individuals that are dominated by a smaller number of other individuals have greater probabilities to be chosen. This *selection* mechanism finishes when the new population becomes of the same size of the former one (at the end, some individuals will have been chosen several times, and others will have not been chosen at all).

  • The individuals of the population receive disturbances that are called *mutations* in the case of one individual being perturbed for generating another one, and are called *crossovers* when two individuals are combined in order to generate other ones.

end while

---

4. SIMULATION RESULTS

4.1 Problem Statement

This case study is intended to show a simple application of the proposed methodology in a classical example of stochastic dynamic programming: *the inventory control* [Bertsekas (1995)]. The problem consists in placing orders over a vector of items at discrete-time stages so as to meet a stochastic demand. For an inventory problem, the variables are inherently discrete since items are counted, but the range of levels for an item is too large to be practical for a discrete-variable dynamic programming solution. Then, the variables, or their relaxation, will be considered as real numbers.

This paper studies a multiproduct case with a warehousing constraint, as in Beyer et al. (2001). That paper has shown that if the cost functions are stationary and separable, the demands are independent, and the feasible set is described by linear constraints, a myopic ordering policy [Ignall and Veinott (1969)] is optimal for this problem, when optimizing according to the expected value of the objective function.

The variables considered here are: the integer k is the index corresponding to the time interval stage; N is the horizon; the problem vector size n corresponds to the number of commodities to be considered; each component of the state vector $x[k]$ is either the inventory level (the stock available) or the backlog level (the postponed quantity) of the corresponding commodity at the beginning of stage k; each control action vector $u[k]$ is the amount to be ordered at the beginning of stage k; and each disturbance vector $w[k]$ is the stochastic customer demand during stage k. It is supposed that the probability distribution function of each commodity is known. The initial inventory position $x[0] = x_0$ is given, as well as the goal x^*, which is the requested inventory level at the final stage.

The surplus balance equation is defined by the difference equation:

$$x[k+1] = x[k] + u[k] - w[k]. \tag{10}$$

This is a linear system as in (7), in which A and B are the identity matrix and C is its opposite. Note that $x[k] > 0$ represents an inventory level and $x[k] < 0$ represents a backlog level. The system must reach an ϵ-radius sphere around the target x^* at the final stage. Thus, the inventory level might evolve throughout the stages according to the open-loop equations presented in (9).

Since disposals are not allowed and the warehouse space is limited, the constraints of this problem are:

$$\begin{cases} u_i[k] \geq 0; \\ \sum_i (x_i[k] + u_i[k]) \leq M; \\ k = 0, \ldots, N-1, \\ i = 1, \ldots, n, \end{cases} \tag{11}$$

in which M is the warehouse space capacity.

Consider an additive cost-function as in (1). A V-shaped function per stage: $g_k\big(x[k], u[k]\big)$ is used, composed by: a purchasing cost, represented by a row vector d_k, per unit that was ordered, added to a fixed cost D_k when $u[k] \neq 0$; and a penalty for a positive stocks (interpreted as a holding cost), represented by a row vector c_k^+, per unit that was held, added to a penalty for a backlogs (interpreted as a backorder cost), represented by a row vector c_k^-, per unit that was backordered. Each decision variable may be less than a known constant B_k.

Due to the use of an open-loop deterministic approach, the inventory level in each stage $x[k]$ on the expressions of the objective function (1) and in the constraints (11) must be rewritten as a function just of the initial state and of the sequence of control variables, as in (8). Thus, the dynamic programming problem can be formulated as in (5). Five quantiles have been chosen to compose the optimization criteria vector: $Q_{0,10}$, $Q_{0,25}$, $Q_{0,50}$, $Q_{0,75}$ and $Q_{0,90}$.

In order to illustrate the ability of the proposed methodology for dealing with arbitrary pdf's, this study considers $n = 10$ products, each of them with a customer demand following a bi-modal probability distribution function. The bi-modal distribution arises as a result of a process that picks the stochastic variable from two distinct Gaussian distributions, with the specific distribution being chosen as a result of a binomial process. A distribution with mean 100 and standard deviation 10 is chosen with probability 0.3, and a distribution with mean 200 and standard deviation 10 is chosen with probability 0.7.

The optimization horizon is $N = 12$, and each stage k corresponds to a month. The initial stock is assumed to be null ($x_0 = 0$), and the inventory level at the final stage $x[N]$ must be close to zero ($x^* = 0$). This assumption corresponds to the case of products that are subject to design cycles of one year: each year a new model is launched, and the last year model becomes "obsolete".

The fixed purchasing cost D_k is considered to be 200, for all k. Each coordinate of vector d_k is $5 \times (0.99)^k$. Each coordinate of the unitary holding cost c_k^+ is considered as $0.5 \times (0.99)^k$ and each coordinate of the unitary backorder cost c_k^- is supposed to be $5.5 \times (0.99)^k$. The radius of the relaxation on the end-point constraint is supposed to be $\epsilon = 10$. Each commodity can be ordered between 0 and $B_k = 500$, for all k. The warehouse capacity is $M = 1000$.

In all simulations, the program has generated $20,000$ sequences of disturbance vectors $w[k]$. However, sometimes, the convergence was reached with only $1,000$ simulations (convergence considering the median of the objective function). The open-loop optimization problem has been solved using NSGA-II, with real encoding, selection by binary tournament and polynomial mutation. The algorithm parameter values are listed below: number of generations: 200, population size: 150, crossover rate: 0.70, index of distribution for crossover: 10, mutation rate: 0.05, index of distribution for mutation: 10. All algorithms have been coded in MATLAB [1] and are available from the authors upon request. Its functions `randn` and `quantile` have been used to generate normal disturbance and to estimate quantiles, respectively. Each complete run of NSGA-II for solving the problem in this setting has spent about $45,000$ seconds in a Intel(R) Core(TM)2 Duo 2.5 GHz. The time scale is of the order of 10 hours, which is *much smaller* than the time scale of a stage between two control actions, which is of the order of one month.

4.2 Analysis of Results

For instance, take commodity 1. Figure 1 shows the objective function values that have been found (in the vertical axis, from bottom to up) for each one of the final non-dominated solution (in the horizontal axis). For a better analysis, all solutions have been sorted by quantile $Q_{0,50}$. The lines corresponding to the quantiles $Q_{0,10}$ and $Q_{0,25}$ have been overlapped, and appear as a single line (the lowest one) in the graphic. This means that all cases between these quantiles have lead to the same objective function values.

The lines corresponding to quantiles $Q_{0,75}$ and $Q_{0,90}$ follow a tendency that is similar to the quantile $Q_{0,50}$. Note that the first solution (that will be called as *Policy A*) corresponds to a policy that optimizes both quantiles 0.5, 0.75 and 0.9, and the last solution (that will be called as *Policy B*) corresponds to a policy that optimizes both quantiles 0.1 and 0.25.

Policy A presents a range of objective function value between 2.1 and 3.5×10^5 between quantiles $Q_{0,10}$ and $Q_{0,90}$, and Policy B presents objective function value between 1.6 and 4.1×10^5 between the same quantiles. Policy A seems to be more robust than Policy B (with lower variability and lower quantile $Q_{0,90}$), which fits better to a more conservative decision maker. On the other hand, Policy B leads to the lowest $Q_{0,10}$ and $Q_{0,25}$ cost function values, although associated with the greatest $Q_{0,50}$, $Q_{0,75}$ and $Q_{0,90}$ cost values. Because of this, it might fit better to an optimistic decision maker.

Illustrating another kind of analysis that the proposed methodology allows: if the decision maker is able to spend a cost value of no more than 4.0×10^5, only the solutions up to 170 should be considered, since these solutions have the quantile $Q_{0,90}$ under the allowed level. As a by-product, the risk of application of the chosen policy can be properly evaluated – which can be used as a raw information for performing a hedge operation.

Another decision criterion can be built on the basis of the weighted probability of having inventory or backlog during the

[1] MATLAB is a trademark of The MathWorks, Inc.

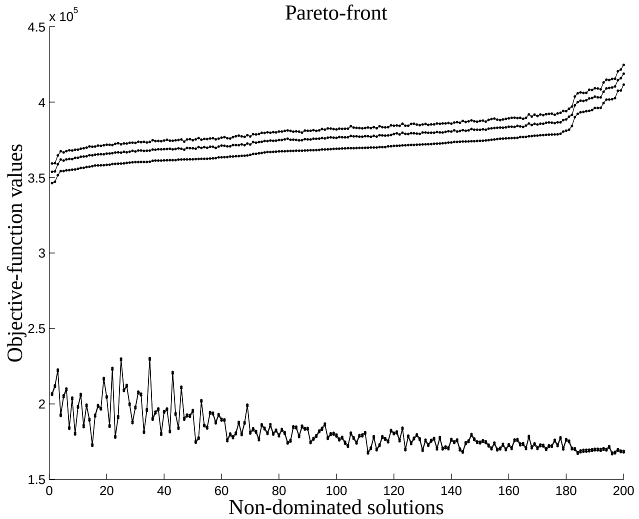

Fig. 1. Values of the functions (from bottom to up), considering quantiles: $Q_{0,10}$, $Q_{0,25}$, $Q_{0,50}$, $Q_{0,75}$ and $Q_{0,90}$ (in the vertical axis), for each one of the final non-dominated solutions (in the horizontal axis). The quantiles $Q_{0,10}$ and $Q_{0,25}$ are represented by a single line (the lowest one).

time horizon. Figure 2 shows this trade-off. Each final non-dominated solution corresponds in this figure to a point that represents the probability of having a positive inventory multiplied by the sum of the corresponding total amount (considering all commodities) and the probability of having backlog multiplied by the sum of the corresponding total amount. It is noticeable that Policy A is at the left upper side of this graphic, and Policy B is in the right lower side.

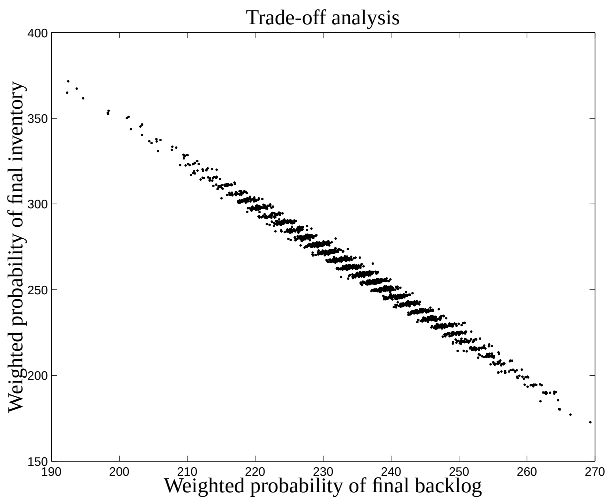

Fig. 2. Trade-off between the sum of the total amount multiplied by the probability of having inventory (horizontal axis) or backlog (vertical axis), for each final non-dominated solution.

A validation simulation has been performed: each non-dominated open-loop policy came from Figure 1 has been applied in another $20,000$ sequences of disturbance vectors. Figure 3 considers the boxplot of the simulated objective function for 20 non-dominated policies linearly equally picked between Policies A and B. To perform a comparison, this figure also shows a line marking the mean of simulated objective function — this is the standard optimization criteria. It must be pointed that Policy A also minimizes the mean of the objective function.

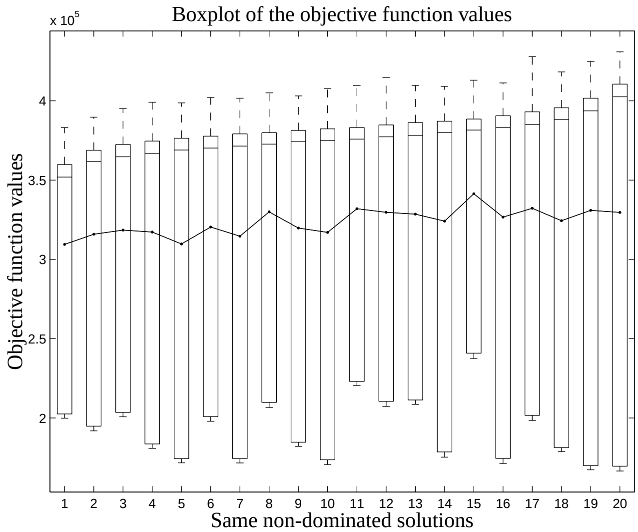

Fig. 3. Boxplot of the simulated objective function (in the vertical axis) for 20 final non-dominated solution linearly equally picked between policies A and B (in the horizontal axis). The line marks the mean of its simulated objective function.

For now, consider a *second example*, considering no warehouse capacity constraint. The fixed purchasing cost D_k is 5, for all k. Each coordinate of vector d_k is 1. Each coordinate of the both unitary holding and backorder cost c_k^+ is considered as 0.1. All other parameters have been the same.

Also take commodity 1. Like in Figure 1, Figure 4 shows the value of the found objective-functions (in the vertical axis, from bottom to up) for each of final non-dominated solution (in the horizontal axis). All individuals have also been sorted by its quantile $Q_{0,50}$. The three lines more below, came from quantiles $Q_{0,10}$, $Q_{0,25}$, and $Q_{0,50}$ follow the same tendency, as well the lines from $Q_{0,75}$ and $Q_{0,90}$. The central line corresponds to the mean (the ordinary criterion). Note the most amplified trade-off: the policy which minimizes the mean (as well the three first quantiles), also maximizes the variance, and the policy which minimizes the variance (came from the minimum of the two last quantiles), maximizes the mean.

For these data, consider a simulation with the *open-loop feedback scheme*: for each month k, for same 50 sequences of demand previously generated, as soon as the inventory level becomes available, run the open-loop optimization procedure, obtaining one ordering sequence for each generated demand sequence, but applying just the present order, for each demand.

For instance, consider two cases: ordering according to quantiles $Q_{0.1}$ (called *Policy 1*) and $Q_{0.9}$ (called *Policy 2*). Figure 5 shows the boxplot of the objective function value considering this two open-loop feedback policies. Comparatively, Figure 6 shows the boxplot of the objective function value considering the pure open-loop Policies 1 and 2, for same sequences of demand which have been already generated. Note that the open-loop feedback cost is lower than the pure open-loop cost, but follows the same tendency: Policy 1 posses lower mean and higher variance.

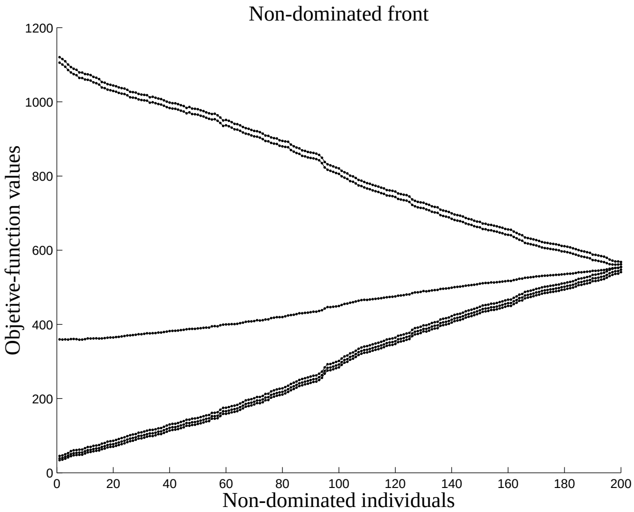

Fig. 4. Values of the functions (from bottom to top), considering quantiles: $Q_{0,10}$, $Q_{0,25}$, $Q_{0,50}$, the mean, and quantiles $Q_{0,75}$ and $Q_{0,90}$ (in the vertical axis), for each of final non-dominated individual (in the horizontal axis). Lines from $Q_{0,10}$, $Q_{0,25}$, and $Q_{0,50}$ follow the same tendency, as well lines from $Q_{0,75}$ and $Q_{0,90}$. The central line corresponds to the mean.

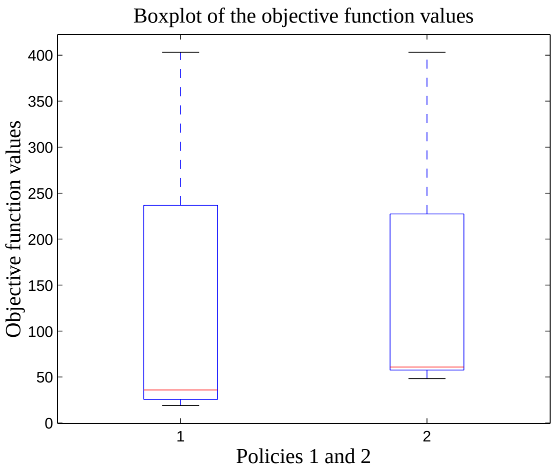

Fig. 5. Boxplot of the simulated objective function (in the vertical axis) for Policies 1 and 2 (in the horizontal axis) considering the open-loop feedback process.

5. DISCUSSION AND CONCLUSIONS

This paper has presented a preliminary study of a multiobjective approach that deals with stochastic discrete-time real-variable dynamic programming problems via a multiquantile analysis – considering some quantiles of the cost function as the objectives to be minimized. The scheme is based on Monte Carlo simulations coupled with a model predictive control algorithm. It is motivated by the need of a systematic way for dealing with problems driven by stochastic variables with arbitrary probability distributions, taking into account also the variability and risk as the optimization criteria, instead of just considering expectations. Doing that, the decision maker can choose a more risky alternative with better expected objective function value, a more conservative one with smaller expected yield, or some other policy between those ones.

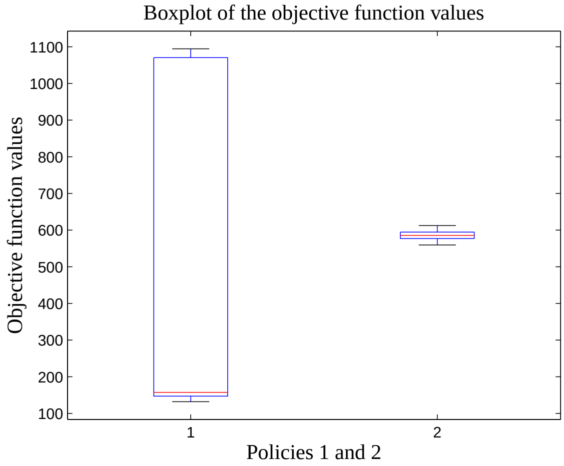

Fig. 6. Boxplot of the simulated objective function (in the vertical axis) for Policies 1 and 2 (in the horizontal axis) considering the pure open-loop process.

REFERENCES

Bertsekas, D.P. (1995). *Dynamic Programming and Optimal Control*. Athena Scientific.

Bertsekas, D.P. (2005). Dynamic programming and suboptimal control: a surveu from adp to mpc. *European Journal of Control*, 11(4-5).

Bertsekas, D.P. and Rhodes, I.B. (1971). On the minimax reachability of target sets and target tubes. *Automatica*, 7, 233–247.

Bertsekas, D.P. and Tsitsiklis, J.N. (1996). *Neuro-Dynamic Programming*. Athena Scientific.

Beyer, D., Sethi, S.P., and Sridhar, R. (2001). Stochastic multiproduct inventory models with limited storage. *Journal of Optimization Theory and Applications*, 111(3), 553–588.

Camacho, E.F. and Bordons, C. (2004). *Model Predictive Control*. Springer-Verlag.

Chankong, V. and Peng, T.K.C. (1983). *Multiobjective Decision Making - Theory and Methodology*. Elsevier.

Deb, K., Pratab, A., Agrawal, S., and Meyarivan, T. (2002). A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transaction on Evolutionary Computation*, 2(6).

Fonseca, C.M. and Fleming, P.J. (1995). An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, 3, 1–16.

Garcia, C.E., Prett, D.M., and Morari, M. (1989). Model predictive control - theory and practice - a survey. *Automatica*, 25(3), 335–348.

Ignall, E.J. and Veinott, A.F. (1969). Optimality of myopic inventory policies for several substitute products. *Management Science*, 18, 284–304.

Levy, H. (1992). Stochastic dominance and expected utility: Survey and analysis. *Management Science*, 38(4), 555–593.

Li, D. and Haimes, Y.Y. (1989). Multiobjective dynamic programming: The state of the art. *Control Theory and Advanced Technology*, 5(4), 471–482.

Ross, S.M. (2002). *Simulation*. Academic Press, San Diego, CA.

Trzaskalik, T. and Sitarz, S. (2007). Discrete dynamic programming with outcomes in random variables structures. *European Journal of Operational Research*, 177, 1535–1548.