Journal of Pure and Applied Mathematics: Advances and Applications
Volume 6, Number 2, 2011, Pages 105-133

ABOUT PROJECTIONS IN THE IMPLEMENTATION OF STOCHASTIC QUASIGRADIENT METHODS TO SOME INVENTORY CONTROL PROBLEMS

STEFAN M. STEFANOV

Department of Informatics
Neofit Rilski South-Western University
Blagoevgrad
Bulgaria
e-mail: stefm@swu.bg

Abstract

In this paper, we consider inventory control problems in a stochastic environment. After a brief description of stochastic quasigradient methods (SQM) for solving stochastic programming problems, algorithms of polynomial complexity are proposed for projecting the current approximation, generated by the SQM, onto feasible regions of two important inventory control problems. Convergence of suggested algorithms is proved, and some examples and results of computational experiments are presented.

1. Introduction

Consider the following inventory control model in a stochastic environment ([3]). Determine the amounts x_j of inventory units (facilities) at locations (points) j, $j = 1, \ldots, n$, in order to meet the demand

---

2010 Mathematics Subject Classification: 90C15, 90C20, 90C25, 90C30.

Keywords and phrases: stochastic programming, stochastic quasigradient methods, inventory control, projection, algorithms.

Received October 29, 2011

ω_j, $j = 1, \ldots, n$, respectively. Since the demand $\boldsymbol{\omega} = (\omega_1, \ldots, \omega_n)$ is random, we know only its distribution function $H(\overline{\boldsymbol{\omega}}) = \mathbf{P}\{\omega_1 \leq \overline{\omega_1}, \ldots, \omega_n \leq \overline{\omega_n}\}$. At the moment of decision making concerning $\mathbf{x} = (x_1, \ldots, x_n)$, the actual value of the demand $\boldsymbol{\omega} = (\omega_1, \ldots, \omega_n)$ is not known.

Suppose that we have made a decision $\mathbf{x}$ about the quantities of materials, facilities, etc., and that the actual demand turned out to be $\boldsymbol{\omega}$. We have to pay for both oversupply and shortfalls. The penalty charged at the j-th location is $\psi_1^j(\omega_j - x_j)$, if $\omega_j \geq x_j$, and $\psi_2^j(x_j - \omega_j)$, if $\omega_j < x_j$, where the functions ψ_1^j and ψ_2^j are nondecreasing. In the simplest case, these functions are linear and the total penalty is

$$\sum_{j=1}^{n} \max\{p_j(\omega_j - x_j),\, q_j(x_j - \omega_j)\}, \tag{1}$$

where $p_j \geq 0$ and $q_j \geq 0$ are the expenses for storage and losses because of deficit for unit of the j-th facility, $j = 1, \ldots, n$, respectively.

In most cases, $\mathbf{x}$ should be determined such that the average penalty is minimal, that is, to minimize the following function:

$$F(\mathbf{x}) = \mathbf{E}_{\boldsymbol{\omega}} f(\mathbf{x}, \boldsymbol{\omega}) = \mathbf{E}_{\boldsymbol{\omega}}\left\{\sum_{j=1}^{n}\left(p_j \int_0^{x_j}(x_j - \omega_j)\mathbf{P}(d\omega_j) + q_j \int_{x_j}^{\infty}(\omega_j - x_j)\mathbf{P}(d\omega_j)\right)\right\},$$

where $\mathbf{E}_{\boldsymbol{\omega}}$ denotes the mathematical expectation with respect to $\boldsymbol{\omega}$.

Often there are some constraints on $\mathbf{x}$.

If the capacity of warehouse (volume of the store), we have to use to keep the inventory facilities is α and we have to order quantity of the j-th product, which is at least a_j and at most b_j, $j = 1, \ldots, n$, we obtain the following minimization problem:

Find $\mathbf{x} = (x_1, \ldots, x_n)$ such that

$$\min_{\mathbf{x}}\{F(\mathbf{x}) = \mathbf{E}_{\boldsymbol{\omega}} f(\mathbf{x}, \boldsymbol{\omega})\} \tag{2}$$

subject to

$$\mathbf{x} \in X, \tag{3}$$

where X is defined through

$$\sum_{j=1}^{n} d_j x_j \le \alpha, \quad d_j > 0, \quad j = 1, \dots, n, \tag{4}$$

$$a_j \le x_j \le b_j, \quad j = 1, \dots, n, \tag{5}$$

or

$$\sum_{j=1}^{n} d_j x_j = \alpha, \quad d_j > 0, \quad j = 1, \dots, n, \tag{6}$$

$$a_j \le x_j \le b_j, \quad j = 1, \dots, n. \tag{7}$$

Relations "$\le$" and "$=$" in (4) and (6) mean "warehouse may not be completely filled" and "capacity of warehouse must be completely used", respectively.

Here ω_j are random variables in closed segments $[R1_j, R2_j]$, $j = 1, \dots, n,$ respectively.

The function $F(\mathbf{x})$ in (2) can be written in the following form:

$$F(\mathbf{x}) = \mathbf{E}_{\boldsymbol{\omega}} f(\mathbf{x}, \boldsymbol{\omega}) = \mathbf{E}_{\boldsymbol{\omega}} \sum_{j=1}^{n} f_j(x_j, \omega_j), \tag{2'}$$

where

$$f_j(x_j, \omega_j) = \begin{cases} p_j(x_j - \omega_j), & \text{if } x_j \ge \omega_j, \\ q_j(\omega_j - x_j), & \text{if } x_j < \omega_j. \end{cases}$$

Since $f(\mathbf{x}, \boldsymbol{\omega})$ is a nondifferentiable function at $\mathbf{x} = \boldsymbol{\omega}$, then $F(\mathbf{x})$ is also a nondifferentiable function.

Problem (2)-(3) is known as the *multi-commodity facility location problem* or as the *inventory control problem* and it is a special case of a (perspective) stochastic programming problem, that is, a problem of the form

$$\text{Find } \mathbf{x} = (x_1, \dots, x_n) \text{ such that}$$

$$\min_{\mathbf{x}} \left\{ F^0(\mathbf{x}) \equiv \mathbf{E}_{\boldsymbol{\omega}} f^0(\mathbf{x}, \boldsymbol{\omega}) = \int f^0(\mathbf{x}, \boldsymbol{\omega}) \mathrm{P}(d\boldsymbol{\omega}) \right\}$$

subject to

$$F^i(\mathbf{x}) \equiv \mathbf{E}_{\boldsymbol{\omega}} f^i(\mathbf{x}, \boldsymbol{\omega}) = \int f^i(\mathbf{x}, \boldsymbol{\omega}) \mathrm{P}(d\boldsymbol{\omega}) \leq 0, \quad i = 1, \dots, m,$$

$$\mathbf{x} \in X \subset \mathbb{R}^n.$$

The functions $F^i(\mathbf{x}),\ i = 0, 1, \dots, m,$ are called *regression functions*, and $\mathbf{E}_{\boldsymbol{\omega}}$ denotes the mathematical expectation with respect to $\boldsymbol{\omega}$.

Stochastic quasigradient methods (SQM) for solving stochastic optimization problems were suggested by Ermoliev ([1-3]).

Given the problem

$$\min F(\mathbf{x})$$

subject to

$$\mathbf{x} \in X,$$

where X is a “deterministic” set.

SQM are defined through

$$\mathbf{x}^{k+1} = \prod\nolimits_X \left(\mathbf{x}^k - \rho_k \xi^k \right), \quad k = 0, 1, \dots, \tag{8}$$

where

- $\mathbf{x}^0$ is an arbitrary initial guess (initial approximation);
- $\prod_X(\mathbf{y})$ is a projection operation of $\mathbf{y}$ onto the feasible region X;

- ρ_k is a step-size;

- $\xi^k = \xi^k(\boldsymbol{\omega})$ is a step direction and $\xi^k(\boldsymbol{\omega})$ is a random vector such that

$$\mathbf{E}_{\boldsymbol{\omega}}(\xi^k / \mathbf{x}^0, \mathbf{x}^1, \dots, \mathbf{x}^k) = a_k \hat{\mathbf{F}}_{\mathbf{x}}(\mathbf{x}^k) + \mathbf{b}^k, \quad k = 0, 1, \dots, \tag{9}$$

where $a_k > 0$ is a random variable; $\mathbf{b}^k = (b_1^k, \dots, b_n^k)$ is a random vector, measurable with respect to the σ-algebra $\mathcal{B}^k$ induced by the family of random variables $(\mathbf{x}^0, \dots, \mathbf{x}^k)$; $\hat{\mathbf{F}}_{\mathbf{x}}(\mathbf{x}^k)$ is a generalized gradient of $F(\mathbf{x})$ at $\mathbf{x}^k$; $\mathbf{E}_{\boldsymbol{\omega}}(\xi^k / \mathbf{x}^0, \dots, \mathbf{x}^k)$ is the conditional mathematical expectation of ξ^k subject to $\mathbf{x}^0, \dots, \mathbf{x}^k$; and ρ_k's are also measurable with respect to $\mathcal{B}^k$.

When $a_k \equiv 1$ and $\mathbf{b}^k \equiv \mathbf{0}$, then ξ^k is said to be *a stochastic generalized gradient* (or *a stochastic quasigradient*) of $F(\mathbf{x})$. Method (8)-(9) is called the *stochastic quasigradient method* (SQM).

SQM are direct methods. Convergence theorems for the SQM have been proved under certain requirements for ξ^k, ρ_k (see, e.g., [1]).

SQM are slow methods. That is why one of the main problems concerning their implementation is the choice of the step-size sequence $\{\rho_k\}$. The simplest way is to do this before the iterative process. Convergence theory states that any sequence with the properties

$$\rho_k \geq 0; \quad \rho_k \to 0,\ k \to \infty; \quad \sum_{k=0}^{\infty} \rho_k = \infty; \quad \sum_{k=0}^{\infty} \rho_k^2 < \infty,$$

may be used as a step-size sequence. However, this approach does not use the valuable information obtained during the iterative process.

A modern method for choosing ρ_k's is the so-called *adaptive step-size regulation* ([17]).

As Gaivoronski ([3]) has pointed out, due to the specificity of stochastic programming problems and stochastic quasigradient methods (slow convergence, nonmonotonicity, and sometimes oscillatory behaviour), it is advisible to average the values of variables and the objective function during a certain number of last iterations and take these average values as the final approximation to the solution.

The second basic problem regarding implementation of SQM is finding the projection of a current point $\mathbf{y}^k \equiv \mathbf{x}^k - \rho_k \xi^k$ onto the feasible region X. As it is known, this is equivalent to solving the quadratic optimization problem, called the *projection problem*

$$\min \frac{1}{2} \|\mathbf{y}^k - \mathbf{x}\|^2,$$

$$\mathbf{x} \in X.$$

This problem has to be solved at *each* iteration of algorithm performance. That is why projection is the most onerous and time-consuming part of the SQM (and of any projection method of gradient type for constrained optimization) and we need efficient algorithms for solving this problem.

The third important question concerning implementation of SQM is calculation of stochastic quasigradient of the function to be minimized.

For example, the components of the stochastic quasigradient of $F(\mathbf{x})$ $(2')$ at iteration k are

$$\xi_j^k = \widehat{f}_j (x_j^k, \omega_j^k) = \begin{cases} p_j, & \text{if } x_j^k \geq \omega_j^k \\ -q_j, & \text{if } x_j^k < \omega_j^k, \end{cases} \quad j = 1, \dots, n, \tag{10}$$

where x_j^k is the j-th component of $\mathbf{x}$ at iteration number k and ω_j^k is j-th component of the observation of $\boldsymbol{\omega}$ at iteration number k.

Algorithms for finding a projection onto a set defined by an inequality/equality constraint and bounds on the variables are suggested in [5, 6], stochastic programming is discussed, e.g., in [1, 4, 7]. Related problems and methods for solving them are considered in [8-16].

This paper is devoted to the development of efficient polynomial algorithms for finding a projection onto the set X defined by (4)-(5) and (6)-(7), respectively. The rest of the paper is organized as follows: In Section 2, characterization theorems (necessary and sufficient conditions) for the projection problem are stated and proved. In Section 3, algorithms of polynomial complexity for solving the projection problems are proposed and their convergence is proved. In Section 4, some extensions concerning both statement of problems and characterization theorems are considered. In Section 5, results of some computational experiments are presented.

2. On the Projection in the Implementation of SQM to Inventory Control Problems

As it was pointed out in the Introduction, we need efficient algorithms for finding a projection of a point onto certain feasible regions.

Consider the problem of finding projection of an arbitrary point $\hat{\mathbf{x}} = (\hat{x}_1, \dots, \hat{x}_n) \in \mathbb{R}^n$ onto the set X defined by (4)-(5) and (6)-(7). This problem is equivalent to the following quadratic programming problem:

$$\min \left\{ c(\mathbf{x}) \equiv \sum_{j=1}^{n} c_j(x_j) \equiv \frac{1}{2} \sum_{j=1}^{n} (x_j - \hat{x}_j)^2 \right\},$$

$$\mathbf{x} \in X.$$

Denote this problem by $(P^{\le})$ in the first case (feasible region X defined by (4)-(5)) and by $(P^{=})$ in the second case (feasible region X defined by (6)-(7)).

Since $c(\mathbf{x})$ is a strictly convex function and X is a convex closed set, then this problem always has a *unique* solution when $X \neq \emptyset$.

First consider the case when X is defined by (4)-(5).

Assumptions:

(1.a) $a_j \le b_j$ for all $j = 1, \dots, n$. If $a_k = b_k$ for some $k, 1 \le k \le n$, then the value $x_k := a_k = b_k$ is determined in advance.

(1.b) $\sum_{j=1}^n d_j a_j \le \alpha$. Otherwise, the constraints (4) and (5) are inconsistent and $X = \emptyset$. In addition to this assumption, we suppose that $\alpha \le \sum_{j=1}^n d_j b_j$ in some cases, which are specified below.

The Lagrangian for problem $(P^{\le})$ is

$$L(\mathbf{x}, \mathbf{u}, \mathbf{v}, \lambda) = \frac{1}{2}\sum_{j=1}^{n}(x_j - \hat{x}_j)^2 + \lambda\left(\sum_{j=1}^{n} d_j x_j - \alpha\right)$$

$$+ \sum_{j=1}^{n} u_j(a_j - x_j) + \sum_{j=1}^{n} v_j(x_j - b_j),$$

where $\lambda \in \mathbb{R}_+^1$; $\mathbf{u}, \mathbf{v} \in \mathbb{R}_+^n$, and $\mathbb{R}_+^n$ consists of all vectors with n real nonnegative components.

Theorem 1. *A feasible solution* $\mathbf{x}^* = (x_1^*, \dots, x_n^*) \in X$ *defined by* (4)-(5) *is an optimal solution to problem* $(P^{\le})$, *if and only if there exists some* $\lambda \in \mathbb{R}_+^1$ *such that*

$$x_j^* = a_j, \quad j \in J_a^\lambda \overset{\text{def}}{=} \left\{ j : \lambda \ge \frac{\hat{x}_j - a_j}{d_j} \right\}, \tag{11}$$

$$x_j^* = b_j, \quad j \in J_b^\lambda \overset{\text{def}}{=} \left\{ j : \lambda \le \frac{\hat{x}_j - b_j}{d_j} \right\}, \tag{12}$$

and

$$x_j^* = \hat{x}_j - \lambda d_j, \quad j \in J^\lambda \overset{\text{def}}{=} \left\{ j : \frac{\hat{x}_j - b_j}{d_j} < \lambda < \frac{\hat{x}_j - a_j}{d_j} \right\}. \tag{13}$$

Proof. The Karush-Kuhn-Tucker (KKT) necessary and sufficient optimality conditions for the minimum solution $\mathbf{x}^*$ are

$$x_j^* - \hat{x}_j + \lambda d_j - u_j + v_j = 0, \quad j = 1, \dots, n, \tag{14}$$

$$u_j(a_j - x_j^*) = 0, \quad j = 1, \dots, n, \tag{15}$$

$$v_j(x_j^* - b_j) = 0, \quad j = 1, \dots, n, \tag{16}$$

$$\lambda \left(\sum_{j=1}^{n} d_j x_j^* - \alpha \right) = 0, \quad \lambda \in \mathbb{R}_+^1, \tag{17}$$

$$\sum_{j=1}^{n} d_j x_j^* \le \alpha, \tag*{(18) ≡ (4)}$$

$$a_j \le x_j^* \le b_j, \quad j = 1, \dots, n, \tag*{(19) ≡ (5)}$$

$$u_j \in \mathbb{R}_+^1, \quad v_j \in \mathbb{R}_+^1, \quad j = 1, \dots, n. \tag{20}$$

Here λ, u_j, and v_j, $j = 1, \dots, n$, are the Lagrange multipliers associated with the constraints (4), $a_j \le x_j$, $x_j \le b_j$, $j = 1, \dots, n$, respectively. If $a_j = -\infty$ or $b_j = +\infty$ for some j, we do not consider the corresponding condition (15) [(16), respectively] and Lagrange multiplier u_j [v_j, respectively].

(i) Let $\mathbf{x}^* = (x_1^*, \dots, x_n^*)$ be an optimal solution to $(P^{\le})$. Then there exist constants λ, u_j, and v_j, $j = 1, \dots, n$, such that KKT conditions (14)-(20) are satisfied. Consider both possible cases for λ.

(1) Let $\lambda > 0$. Then system (14)-(20) becomes (14), (15), (16), (19), (20), and

$$\sum_{j=1}^{n} d_j x_j^* = \alpha,$$

that is, the inequality constraint (4) is satisfied with an equality for x_j^*, $j = 1, \dots, n$, in this case.

(a) If $x_j^* = a_j$, then $u_j \geq 0, v_j = 0$ according to (16). Therefore $x_j^* - \hat{x}_j = u_j - \lambda d_j \geq -\lambda d_j$. Since $d_j > 0$, then

$$\lambda \geq \frac{\hat{x}_j - x_j^*}{d_j} \equiv \frac{\hat{x}_j - a_j}{d_j}.$$

(b) If $x_j^* = b_j$, then $u_j = 0$ according to (15), $v_j \geq 0$. Therefore $x_j^* - \hat{x}_j = -v_j - \lambda d_j \leq -\lambda d_j$. Hence,

$$\lambda \leq \frac{\hat{x}_j - x_j^*}{d_j} \equiv \frac{\hat{x}_j - b_j}{d_j}.$$

(c) If $a_j < x_j^* < b_j$, then $u_j = v_j = 0$ according to (15) and (16). Therefore $x_j^* = \hat{x}_j - \lambda d_j$ according to (14).

Since $d_j > 0$, $j = 1, \dots, n$, $\lambda > 0$ by the assumptions, then $\hat{x}_j - x_j^* \geq 0$. From $b_j > x_j^*$, $x_j^* > a_j$, it follows that

$$b_j - \hat{x}_j \geq x_j^* - \hat{x}_j, \quad x_j^* - \hat{x}_j \geq a_j - \hat{x}_j.$$

Using $\hat{x}_j - x_j^* \geq 0$ and $d_j > 0$, we obtain $\lambda = \dfrac{\hat{x}_j - x_j^*}{d_j} \leq \dfrac{\hat{x}_j - a_j}{d_j}$ and $\lambda = \dfrac{\hat{x}_j - x_j^*}{d_j} \geq \dfrac{\hat{x}_j - b_j}{d_j}$, that is,

$$\frac{\hat{x}_j - b_j}{d_j} \leq \lambda \leq \frac{\hat{x}_j - a_j}{d_j}.$$

Since we are not in subcases (a), (b), these inequalities are strict. Hence,

$$\frac{\hat{x}_j - b_j}{d_j} < \lambda < \frac{\hat{x}_j - a_j}{d_j}.$$

(2) Let $\lambda = 0$. Then system (14)-(20) becomes (15), (16), (18), (19), (20), and

$$x_j^* - \hat{x}_j - u_j + v_j = 0, \quad j = 1, \dots, n.$$

(a) If $x_j^* = a_j$, then $u_j \geq 0, v_j = 0$ according to (16). Therefore $a_j - \hat{x}_j \equiv x_j^* - \hat{x}_j = u_j \geq 0$. Multiplying both sides of this inequality by $-\frac{1}{d_j}$ (< 0 by the assumption), we obtain

$$\frac{\hat{x}_j - a_j}{d_j} \leq 0 \equiv \lambda.$$

(b) If $x_j^* = b_j$, then $u_j = 0$ according to (15), $v_j \geq 0$. Therefore $b_j - \hat{x}_j \equiv x_j^* - \hat{x}_j = -v_j \leq 0$. Multiplying this inequality by $-\frac{1}{d_j} < 0$, we get

$$\frac{\hat{x}_j - b_j}{d_j} \geq 0 \equiv \lambda.$$

(c) If $a_j < x_j^* < b_j$, then $u_j = v_j = 0$ in accordance with (15) and (16). Therefore $x_j^* - \hat{x}_j = 0$, that is, $x_j^* = \hat{x}_j$. Since $b_j > x_j^*, x_j^* > a_j$, $j = 1, \dots, n$, by the assumption, then

$$b_j - \hat{x}_j \geq x_j^* - \hat{x}_j = 0, \quad 0 = x_j^* - \hat{x}_j \geq a_j - \hat{x}_j.$$

Multiplying both inequalities by $-\frac{1}{d_j} < 0$, we obtain

$$\frac{\hat{x}_j - b_j}{d_j} \leq 0 \equiv \lambda, \quad \lambda \equiv 0 \leq \frac{\hat{x}_j - a_j}{d_j}.$$

Since we are not in subcases (a), (b), these inequalities are strict, that is, in subcase (c), we have

$$\frac{\hat{x}_j - b_j}{d_j} < \lambda < \frac{\hat{x}_j - a_j}{d_j}.$$

In order to describe subcases (a), (b), (c) for both cases (1) and (2) of part (i), it is convenient to introduce the following index sets: J_a^λ, J_b^λ, J^λ defined by (11), (12), and (13), respectively. Obviously, $J_a^\lambda \cup J_b^\lambda \cup J^\lambda = \{1, \ldots, n\}$. Thus, the "necessity" part is proved.

(ii) Conversely, let $\mathbf{x}^* \in X$ and components of $\mathbf{x}^*$ satisfy (11), (12), and (13), where $\lambda \in \mathbb{R}^1_+$.

(1) If $\lambda > 0$, then $x_j^* - \hat{x}_j < 0$, $j \in J^\lambda$, according to (13) and $d_j > 0$. Set

$$\lambda = \frac{\hat{x}_j - x_j^*}{d_j} \; (> 0) \text{ from (13)}: \sum_{j \in J_a^\lambda} d_j a_j + \sum_{j \in J_b^\lambda} d_j b_j + \sum_{j \in J^\lambda} d_j\,(\hat{x}_j - \lambda d_j) = \alpha,$$

$$u_j = v_j = 0, \qquad \text{for } j \in J^\lambda;$$

$$u_j = a_j - \hat{x}_j + \lambda d_j \; (\geq 0 \text{ according to the definition of } J_a^\lambda),$$

$$v_j = 0, \text{ for } j \in J_a^\lambda;$$

$$u_j = 0,\; v_j = \hat{x}_j - b_j - \lambda d_j \; (\geq 0 \text{ according to the definition of } J_b^\lambda),$$

$$\text{for } j \in J_b^\lambda.$$

By using these expressions, it is easy to check that conditions (14), (15), (16), (17), and (20) are satisfied; conditions (18) and (19) are satisfied according to the assumption $\mathbf{x}^* \in X$.

(2) If $\lambda = 0$, then $x_j^* - \hat{x}_j = 0$, $j \in J^\lambda$, according to (13) and

$$J^{\lambda=0} = \left\{ j : \frac{\hat{x}_j - b_j}{d_j} < 0 < \frac{\hat{x}_j - a_j}{d_j} \right\}.$$

Since $d_j > 0$, then $\hat{x}_j - b_j < 0,\ \hat{x}_j - a_j > 0,\ j \in J^0$. Therefore $x_j^* = \hat{x}_j \in (a_j, b_j)$.

Set

$$\lambda = \frac{\hat{x}_j - x_j^*}{d_j} (= 0); \quad u_j = v_j = 0, \qquad \text{for } j \in J^{\lambda=0};$$

$$u_j = a_j - \hat{x}_j + \lambda d_j = a_j - \hat{x}_j\ (\geq 0),\ v_j = 0, \quad \text{for } j \in J_a^{\lambda=0};$$

$$u_j = 0,\ v_j = \hat{x}_j - b_j - \lambda d_j = \hat{x}_j - b_j\ (\geq 0), \quad \text{for } j \in J_b^{\lambda=0}.$$

Obviously conditions (14), (15), (16), and (20) are satisfied; conditions (18) and (19) are also satisfied according to the assumption $\mathbf{x}^* \in X$; and condition (17) obviously is satisfied for $\lambda = 0$.

In both cases (1) and (2) of part (ii), $x_j^*, \lambda, u_j, v_j,\ j = 1, \ldots, n,$ satisfy KKT conditions (14)-(20), which are necessary and sufficient conditions for a feasible solution to be an optimal solution to a convex minimization problem. Therefore, $\mathbf{x}^*$ is an optimal solution to problem $(P^{\leq})$. ■

Theorem 1 is important because it describes components of the optimal solution to problem $(P^{\leq})$ only through the Lagrange multiplier λ associated with the inequality constraint (4).

Since we do not know the optimal value of λ from Theorem 1, we define an iterative process with respect to the Lagrange multiplier λ and we prove convergence of this process in section The Algorithms.

From $d_j > 0$ and $a_j \leq b_j,\ j = 1, \ldots, n,$ it follows that

$$ub_j \stackrel{\text{def}}{=} \frac{\hat{x}_j - b_j}{d_j} \leq \frac{\hat{x}_j - a_j}{d_j} \stackrel{\text{def}}{=} la_j, \quad j = 1, \ldots, n,$$

for the expressions by means of which we define the sets J_a^λ, J_b^λ, and J^λ.

The problem how to ensure a feasible solution to problem $(P^{\leq})$, which is an assumption of Theorem 1, is discussed after the statement of the corresponding algorithm.

Consider problem $(P^{=})$ of finding a projection onto a set X of the form (6)-(7)

$(P^{=})$

$$\min \left\{ c(\mathrm{x}) = \sum_{j=1}^{n} c_j(x_j) = \frac{1}{2}\sum_{j=1}^{n}(x_j - \hat{x}_j)^2 \right\},$$

$$\mathbf{x} \in X,$$

where X is defined by (6)-(7).

Assumptions:

(2.a) $a_j \leq b_j$, for all $j = 1, \ldots, n$.

(2.b) $\sum_{j=1}^{n} d_j a_j \leq \alpha \leq \sum_{j=1}^{n} d_j b_j$. Otherwise, the constraints (6) and (7) are inconsistent and the feasible region (6)-(7) is empty.

In this case, the following Theorem 2, which is analogous to Theorem 1, holds true.

For the sake of simplicity, throughout Theorem 2, we will use the same notation $\lambda, u_j, v_j, j = 1, \ldots, n,$ for the Lagrange multipliers associated with (6), $a_j \leq x_j$, $x_j \leq b_j$, respectively, as we have used them in Theorem 1. We hope that this would not lead to confusion.

Theorem 2. *A feasible solution* $\mathbf{x}^* = (x_1^*, \ldots, x_n^*) \in X$ *defined by* (6)-(7) *is an optimal solution to problem* $(P^{=})$, *if and only if there exists some* $\lambda \in \mathbb{R}^1$ *such that*

$$x_j^* = a_j, \quad j \in J_a^\lambda \stackrel{\text{def}}{=} \left\{ j : \lambda \geq \frac{\hat{x}_j - a_j}{d_j} \right\}, \tag{21}$$

$$x_j^* = b_j, \quad j \in J_b^\lambda \stackrel{\text{def}}{=} \left\{ j : \lambda \leq \frac{\hat{x}_j - b_j}{d_j} \right\}, \tag{22}$$

and

$$x_j^* = \hat{x}_j - \lambda d_j, \quad j \in J^\lambda \stackrel{\text{def}}{=} \left\{ j : \frac{\hat{x}_j - b_j}{d_j} < \lambda < \frac{\hat{x}_j - a_j}{d_j} \right\}. \tag{23}$$

The proof of Theorem 2 is omitted because it is similar to the proof of Theorem 1.

3. The Algorithms

3.1. Analysis of the optimal solution to problem $(P^{\leq})$

Before the formal statement of the algorithm for problem $(P^{\leq})$, we will discuss some properties of the optimal solution to this problem, which turn out to be useful.

By using (11), (12), and (13), the condition (17) can be written as follows:

$$\lambda \left(\sum_{j \in J_a^\lambda} d_j a_j + \sum_{j \in J_b^\lambda} d_j b_j + \sum_{j \in J^\lambda} d_j \left(\hat{x}_j - \lambda d_j \right) - \alpha \right) = 0, \quad \lambda \geq 0. \tag{17$'$}$$

Since the optimal solution $\mathbf{x}^*$ to problem $(P^{\leq})$ obviously depends on λ, we consider the components of $\mathbf{x}^*$ as functions of λ for different $\lambda \in \mathbb{R}_+^1$

$$x_j^* = x_j(\lambda) = \begin{cases} a_j, & j \in J_a^\lambda, \\ b_j, & j \in J_b^\lambda, \\ \hat{x}_j - \lambda d_j, & j \in J^\lambda. \end{cases} \tag{24}$$

Obviously $x_j(\lambda)$, $j = 1, \ldots, n$, are piecewise linear, monotonically nondecreasing, piecewise differentiable functions of λ with two breakpoints at $\lambda = \dfrac{\hat{x}_j - a_j}{d_j}$ and $\lambda = \dfrac{\hat{x}_j - b_j}{d_j}$.

Let

$$\delta(\lambda) \stackrel{\text{def}}{=} \sum_{j \in J_a^\lambda} d_j a_j + \sum_{j \in J_b^\lambda} d_j b_j + \sum_{j \in J^\lambda} d_j (\hat{x}_j - \lambda d_j) - \alpha. \tag{25}$$

If we differentiate (25) with respect to λ, we get

$$\delta'(\lambda) \equiv -\sum_{j \in J^\lambda} d_j^2 < 0, \tag{26}$$

when $J^\lambda \neq \emptyset$, and $\delta'(\lambda) = 0$ when $J^\lambda = \emptyset$. Hence $\delta(\lambda)$ is a *monotone nonincreasing function* of λ, $\lambda \in \mathbb{R}^1_+$, and $\max_{\lambda \geq 0} \delta(\lambda)$ is attained at the minimum feasible value of λ, that is, at $\lambda = 0$.

Case 1. If $\delta(0) > 0$, in order that (17′) and (18) ≡ (4) be satisfied, there exists some $\lambda^* > 0$ such that $\delta(\lambda^*) = 0$, that is,

$$\sum_{j=1}^n d_j x_j^* = \alpha, \tag{27}$$

which means that the inequality constraint (4) is satisfied with an equality for λ^* in this case.

Case 2. If $\delta(0) < 0$, then $\delta(\lambda) < 0$ for all $\lambda \geq 0$, and the maximum of $\delta(\lambda)$ with $\lambda \geq 0$ is $\delta(0) = \max_{\lambda \geq 0} \delta(\lambda)$ and it is attained at $\lambda = 0$ in this case. In order that (17′) be satisfied, λ must be equal to 0. Therefore $x_j^* = \hat{x}_j$, $j \in J^{\lambda=0}$, according to (13).

Case 3. In the special case when $\delta(0) = 0$, the maximum $\delta(0) = \max_{\lambda \geq 0} \delta(\lambda)$ of $\delta(\lambda)$ is also attained at the minimum admissible value of λ, that is, for $\lambda = 0$, because $\delta(\lambda)$ is a monotone nonincreasing function in accordance with the above discussion.

As we have seen, for the optimal value of λ, we have $\lambda \geq 0$ in all possible cases, as the KKT condition (17) requires. We have shown that in Case 1, we need an algorithm for finding λ^*, which satisfies the KKT conditions (14)-(20) but such that λ^* satisfies (18) with an equality. In order that this be fulfilled, the set

$$X^= \stackrel{\text{def}}{=} \left\{ \mathbf{x} \in \mathbb{R}^n : \sum_{j=1}^{n} d_j x_j = \alpha, \quad a_j \leq x_j \leq b_j, \quad j = 1, \ldots, n \right\}$$

must be nonempty. That is why we have required $\alpha \leq \sum_{j=1}^{n} d_j b_j$ in some cases in addition to the assumption $\sum_{j=1}^{n} d_j a_j \leq \alpha$ (see Assumption (1.b)). We have also used this in the proof of Theorem 1, part (ii), when $\lambda > 0$.

From the equation $\delta(\lambda) = 0$, where $\delta(\lambda)$ is defined by (25), we are able to obtain a closed form expression for λ

$$\lambda = \left(\sum_{j \in J^\lambda} d_j^2 \right)^{-1} \left(\sum_{j \in J_a^\lambda} d_j a_j + \sum_{j \in J_b^\lambda} d_j b_j + \sum_{j \in J^\lambda} d_j \hat{x}_j - \alpha \right), \tag{28}$$

because $\delta'(\lambda) < 0$ according to (26) when $J^\lambda \neq \emptyset$ (it is important that $\delta'(\lambda) \neq 0$). This expression of λ is used in the algorithm suggested for problem $(P^{\leq})$. It turns out that for our purpose, without loss of generality, we can assume that $\delta'(\lambda) \neq 0$, that is, $\delta(\lambda)$ depends on λ, which means that $J^\lambda \neq \emptyset$.

At iteration k of the implementation of algorithms, denote by $\lambda^{(k)}$ the value of the Lagrange multiplier associated with constraint (4) [(6), respectively], by $\alpha^{(k)}$ the right-hand side of (4) [of (6), respectively]; by $J^{(k)}, J_a^{\lambda(k)}, J_b^{\lambda(k)}, J^{\lambda(k)}$ the current sets $J = \{1, \ldots, n\}, J_a^{\lambda}, J_b^{\lambda}, J^{\lambda}$, respectively.

3.2. Algorithm 1

The following algorithm for solving problem $(P^{\leq})$ is based on Theorem 1:

Algorithm 1 (for problem $(P^{\leq})$)

(1) (Initialization) $J := \{1, \ldots, n\}$, $k := 0$, $\alpha^{(0)} := \alpha$, $n^{(0)} := n$, $J^{(0)} := J$, $J_a^{\lambda} := \emptyset$, $J_b^{\lambda} := \emptyset$, initialize $\hat{x}_j$, $j \in J$. If $\sum_{j=1}^{n} d_j a_j \leq \alpha$, go to (2) else go to (10).

(2) Construct the sets J_a^0, J_b^0, J^0 (for $\lambda = 0$). Calculate

$$\delta(0) := \sum_{j \in J_a^0} d_j a_j + \sum_{j \in J_b^0} d_j b_j + \sum_{j \in J^0} d_j \hat{x}_j - \alpha.$$

If $\delta(0) \leq 0$, then $\lambda := 0$, go to (9),

else if $\delta(0) > 0$, then

if $\alpha \leq \sum_{j=1}^{n} d_j b_j$ go to (3)

else if $\alpha > \sum_{j=1}^{n} d_j b_j$ go to (10) (there does not exist $\lambda^* > 0$ such that $\delta(\lambda^*) = 0$).

(3) $J^{\lambda(k)} := J^{(k)}$. Calculate $\lambda^{(k)}$ by using the explicit expression of λ (28). Go to (4).

(4) Construct the sets $J_a^{\lambda(k)}$, $J_b^{\lambda(k)}$, $J^{\lambda(k)}$ through (11), (12), (13) (with $j \in J^{(k)}$ instead of $j \in J$) and find their cardinalities $|J_a^{\lambda(k)}|$, $|J_b^{\lambda(k)}|$, and $|J^{\lambda(k)}|$, respectively. Go to (5).

(5) Calculate

$$\delta(\lambda^{(k)}) := \sum_{j \in J_a^{\lambda(k)}} d_j a_j + \sum_{j \in J_b^{\lambda(k)}} d_j b_j + \sum_{j \in J^{\lambda(k)}} d_j (\hat{x}_j - \lambda d_j) - \alpha^{(k)}.$$

Go to (6).

(6) If $\delta(\lambda^{(k)}) = 0$ or $J^{\lambda(k)} = \emptyset$, then $\lambda := \lambda^{(k)}$, $J_a^{\lambda} := J_a^{\lambda} \cup J_a^{\lambda(k)}$, $J_b^{\lambda} := J_b^{\lambda} \cup J_b^{\lambda(k)}$, $J^{\lambda} := J^{\lambda(k)}$, go to (9),

else if $\delta(\lambda^{(k)}) > 0$ go to (7),

else if $\delta(\lambda^{(k)}) < 0$ go to (8).

(7) $x_j^* := a_j$ for $j \in J_a^{\lambda(k)}$, $\alpha^{(k+1)} := \alpha^{(k)} - \sum_{j \in J_a^{\lambda(k)}} d_j a_j$, $J^{(k+1)} := J^{(k)} \backslash J_a^{\lambda(k)}$, $n^{(k+1)} := n^{(k)} - |J_a^{\lambda(k)}|$, $J_a^{\lambda} := J_a^{\lambda} \cup J_a^{\lambda(k)}$, $k := k + 1$. Go to (3).

(8) $x_j^* := b_j$ for $j \in J_b^{\lambda(k)}$, $\alpha^{(k+1)} := \alpha^{(k)} - \sum_{j \in J_b^{\lambda(k)}} d_j b_j$, $J^{(k+1)} := J^{(k)} \backslash J_b^{\lambda(k)}$, $n^{(k+1)} := n^{(k)} - |J_b^{\lambda(k)}|$, $J_b^{\lambda} := J_b^{\lambda} \cup J_b^{\lambda(k)}$, $k := k + 1$. Go to (3).

(9) $x_j^* := a_j$ for $j \in J_a^{\lambda}$; $x_j^* := b_j$ for $j \in J_b^{\lambda}$; $x_j^* := \hat{x}_j - \lambda d_j$ for $j \in J^{\lambda}$. Go to (11).

(10) Problem $(P^{\leq})$ has no optimal solution because $X = \emptyset$ or there does not exist a $\lambda^* > 0$ satisfying Theorem 1.

(11) End.

Remark 1. In order to avoid a possible "endless loop" in computer programming of the algorithm, the criterion of Step (6) to go to Step (9) at iteration k usually is $\delta(\lambda^{(k)}) \in [-\varepsilon, \varepsilon]$ instead of $\delta(\lambda^{(k)}) = 0$, where $\varepsilon > 0$ is some given tolerance value up to which the equality $\delta(\lambda^*) = 0$ may (for problem $(P^{\leq})$) or must (for problem $(P^{=})$) be satisfied.

3.3. Convergence and complexity of Algorithm 1

Theorem 3. *Let* $\{\lambda^{(k)}\}$ *be the sequence generated by Algorithm* 1. *Then*

(i) *if* $\delta(\lambda^{(k)}) > 0$, *then* $\lambda^{(k)} \leq \lambda^{(k+1)}$;

(ii) *if* $\delta(\lambda^{(k)}) < 0$, *then* $\lambda^{(k)} \geq \lambda^{(k+1)}$.

Proof. Denote by $x_j^{(k)}$ the components of $\mathbf{x}^{(k)} = (x_j)_{j \in J^{(k)}}$ at iteration k of the implementation of Algorithm 1.

Taking into consideration (26), Cases 1-3, Step (2) (the sign of $\delta(0)$) and Step (3) of Algorithm 1, it follows that $\lambda^{(k)} \geq 0$ for each k. Since $x_j^{(k)}$ are determined from (13): $x_j^{(k)} = \hat{x}_j - \lambda^{(k)} d_j,\ j \in J^{\lambda(k)}$, substituted in $\sum_{j \in J^{\lambda(k)}} d_j x_j^{(k)} = \alpha^{(k)}$ at Step (3) of Algorithm 1 and since $\lambda^{(k)} \geq 0,\ d_j > 0$, then $x_j^{(k)} - \hat{x}_j \leq 0$, that is, $\hat{x}_j \geq x_j^{(k)}$.

(i) Let $\delta(\lambda^{(k)}) > 0$. Using Step (7) of Algorithm 1 (which is performed when $\delta(\lambda^{(k)}) > 0$), we get

$$\sum_{j \in J^{\lambda(k+1)}} d_j x_j^{(k)} \equiv \sum_{j \in J^{(k+1)}} d_j x_j^{(k)}$$

$$= \sum_{j \in J^{(k)} \setminus J_a^{\lambda(k)}} d_j x_j^{(k)} = \alpha^{(k)} - \sum_{j \in J_a^{\lambda(k)}} d_j x_j^{(k)}. \tag{29}$$

Let $j \in J_a^{\lambda(k)}$. According to definition (11) of $J_a^{\lambda(k)}$, we have

$$\frac{\hat{x}_j - a_j}{d_j} \leq \lambda^{(k)} = \frac{\hat{x}_j - x_j^{(k)}}{d_j}.$$

Multiplying this inequality by $-d_j < 0$, we obtain $a_j - \hat{x}_j \geq x_j^{(k)} - \hat{x}_j$. Therefore $a_j \geq x_j^{(k)}$, $j \in J_a^{\lambda(k)}$.

From (29), using that $d_j > 0$ and Step (7), we get

$$\sum_{j \in J^{\lambda(k+1)}} d_j x_j^{(k)} = \alpha^{(k)} - \sum_{j \in J_a^{\lambda(k)}} d_j x_j^{(k)} \geq \alpha^{(k)} - \sum_{j \in J_a^{\lambda(k)}} d_j a_j$$

$$= \alpha^{(k+1)} = \sum_{j \in J^{\lambda(k+1)}} d_j x_j^{(k+1)}.$$

Since $d_j > 0$, $j = 1, \ldots, n$, then there exists at least one $j_0 \in J^{\lambda(k+1)}$ such that $x_{j_0}^{(k)} \geq x_{j_0}^{(k+1)}$. Then

$$\lambda^{(k)} = \frac{\hat{x}_j - x_{j_0}^{(k)}}{d_{j_0}} \leq \frac{\hat{x}_j - x_{j_0}^{(k+1)}}{d_{j_0}} = \lambda^{(k+1)},$$

which was to be proved in part (i). We have used that the relationship between $\lambda^{(k)}$ and $x_j^{(k)}$ is given by (13) for $j \in J^{\lambda(k)}$ according to Step (3) of Algorithm 1 and that $\hat{x}_j \geq x_j^{(k)}$, $j \in J^{\lambda(k)}$, according to (13) with $\lambda^{(k)} \geq 0$ and $d_j > 0$.

The proof of part (ii) is omitted because it is similar to the proof of part (i). ∎

Consider the feasibility of $\mathbf{x}^* = (x_j^*)_{j \in J}$, generated by Algorithm 1.

Components $x_j^* = a_j$, $j \in J_a^{\lambda}$, and $x_j^* = b_j$, $j \in J_b^{\lambda}$, obviously satisfy (5). From

$$\frac{\hat{x}_j - b_j}{d_j} < \lambda \equiv \frac{\hat{x}_j - x_j^*}{d_j} < \frac{\hat{x}_j - a_j}{d_j}, \quad j \in J^{\lambda},$$

and $d_j > 0$, it follows that $a_j - \hat{x}_j < x_j^* - \hat{x}_j < b_j - \hat{x}_j$, $j \in J^{\lambda}$. Therefore $a_j \leq x_j^* \leq b_j$ for $j \in J^{\lambda}$. Hence all x_j^*, $j = 1, \ldots, n$, satisfy (5).

We have proved that if $\delta(\lambda)|_{\lambda=0} \geq 0$ and $X^{=} \neq \emptyset$, then there exists a $\lambda^* \geq 0$ such that $\delta(\lambda^*) = 0$. Since at Step (3) we determine $\lambda^{(k)}$ from the equality $\sum_{j \in J^{\lambda(k)}} d_j x_j^{(k)} = \alpha^{(k)}$ for each k, then (4) is satisfied with an equality in this case. Otherwise, if $\delta(0) < 0$, then we set $\lambda = 0$ (Step (3)) and we have $\sum_{j \in J} d_j x_j(0) - \alpha \equiv \delta(0) < 0$, that is, (4) is satisfied as a strict inequality in this case.

Therefore Algorithm 1 generates $\mathbf{x}^*$, which is feasible for problem $(P^{\leq})$, which is an assumption of Theorem 1.

Remark 2. Theorem 3, definitions of J_a^{λ}, J_b^{λ}, $J^{(\lambda)}$ in (11), (12), and (13), respectively, and Steps (7)-(9) of Algorithm 1 allow us to state that the calculation of λ, operations $x_j^* := a_j$, $j \in J_a^{\lambda(k)}$ (Step (7)), $x_j^* := b_j$, $j \in J_b^{\lambda(k)}$ (Step (8)), and the construction of J_a^{λ}, J_b^{λ}, and J^{λ} are in accordance with Theorem 1.

At each iteration, Algorithm 1 determines the value of at least one variable (Steps (7)-(9)) and at each iteration, we solve a problem of the form $(P^{\leq})$ but of less dimension (Steps (3)-(8)). Therefore, Algorithm 1 is finite and it converges with at most $n = |J|$ iterations, that is, the iteration complexity of Algorithm 1 is $\mathcal{O}(n)$.

Step (1) takes time $\mathcal{O}(n)$. Step (2) (construction of sets J_a^0, J_b^0, J^0, calculation of $\delta(0)$ and checking whether X is empty) also takes time $\mathcal{O}(n)$.

The calculation of $x_j^{(k)}$, $j = 1, \dots, n$, and $\lambda^{(k)}$ requires $\mathcal{O}(n)$ time (Step (3)). Step (4) takes $\mathcal{O}(n)$ time because of the construction of $J_a^{\lambda(k)}$, $J_b^{\lambda(k)}$, and $J^{\lambda(k)}$. Step (5) also requires $\mathcal{O}(n)$ time, and Step (6) requires constant time. Each of Steps (7)-(9) takes time, which is bounded by $\mathcal{O}(n)$ because at these steps we assign some of x_j's the final value, and since the number of all x_j's is n, then Steps (7)-(9) take time $\mathcal{O}(n)$. Hence, the algorithm has $\mathcal{O}(n^2)$ running time and it belongs to the class of strongly polynomially bounded algorithms.

As the computational experiments show, the number of iterations of the algorithm performance is not only at most n, but also it is much, much less than n for great n. In fact, this number does not depend on n but only on the three index sets defined by (11), (12), and (13). In practice, Algorithm 1 has $\mathcal{O}(n)$ running time.

3.4. Algorithm 2 (for problem $(P^=)$) and its convergence

After analysis of the optimal solution to problem $(P^=)$, similar to that to problem $(P^\leq)$, we suggest the following algorithm for solving problem $(P^=)$.

Algorithm 2 (for problem $(P^=)$)

(1) (Initialization) $J := \{1, \dots, n\}$, $k := 0$, $\alpha^{(0)} := \alpha$, $n^{(0)} := n$, $J^{(0)} := J$, $J_a^\lambda := \emptyset$, and $J_b^\lambda := \emptyset$, initialize $\hat{x}_j$, $j \in J$. If $\sum_{j=1}^n d_j a_j \leq \alpha \leq \sum_{j=1}^n d_j b_j$, go to (2) else go to (9).

(2) $J^{\lambda(k)} := J^{(k)}$. Calculate $\lambda^{(k)}$ by using the explicit expression of λ. Go to (3).

(3) Construct the sets $J_a^{\lambda(k)}$, $J_b^{\lambda(k)}$, and $J^{\lambda(k)}$ through (21), (22), (23) (with $j \in J^{(k)}$ instead of $j \in J$) and find their cardinalities $|J_a^{\lambda(k)}|$, $|J_b^{\lambda(k)}|$, and $|J^{\lambda(k)}|$. Go to (4).

Steps (4)-(8) are the same as Steps (5)-(9) of Algorithm 1, respectively.

(9) Problem $(P^{=})$ has no optimal solution because the corresponding feasible set X defind by (6)-(7) is empty.

(10) End.

A theorem, analogous to Theorem 3, holds for Algorithm 2, which guarantees the "convergence" of $\lambda^{(k)}$, $J^{\lambda(k)}$, $J_a^{\lambda(k)}$, and $J_b^{\lambda(k)}$ to the optimal λ, J^{λ}, J_a^{λ}, and J_b^{λ}, respectively.

Theorem 4. *Let* $\{\lambda^{(k)}\}$ *be the sequence generated by Algorithm* 2. *Then*

(i) *if* $\delta(\lambda^{(k)}) > 0$, *then* $\lambda^{(k)} \leq \lambda^{(k+1)}$;

(ii) *if* $\delta(\lambda^{(k)}) < 0$, *then* $\lambda^{(k)} \geq \lambda^{(k+1)}$.

The proof of Theorem 4 is similar to the proof of Theorem 3.

It can be shown that Algorithm 2 also has $\mathcal{O}(n)$ running time, and point $\mathbf{x}^* = (x_1^*, \ldots, x_n^*)$ generated by this algorithm is feasible for problem $(P^{=})$, which is an assumption of Theorem 2.

4. Extensions

If it is allowed that $d_j = 0$ for some j in problems $(P^{\leq})$ and $(P^{=})$, then for such indices j, we cannot construct the expressions $\dfrac{\hat{x}_j - a_j}{d_j}$ and $\dfrac{\hat{x}_j - b_j}{d_j}$ by means of which we define sets J_a^{λ}, J_b^{λ}, and J^{λ} for the corresponding problem. In such cases, x_j's are not involved in (4) [in (6), respectively] for such indices j.

Denote

$$J = \{1, \ldots, n\}, \qquad Z0 = \{j \in J : d_j = 0\}.$$

Here "0" denotes the "computer zero". In particular, when $J = Z0$ and $\alpha = 0,$ then X is defined only by (5) [by (7), respectively].

Theorem 5. *Problem* $(P^{\leq})$ *can be decomposed into two subproblems*: $(P1^{\leq})$ *for* $j \in Z0$ *and* $(P2^{\leq})$ *for* $j \in J \setminus Z0.$

The optimal solution to $(P1^{\leq})$ *is*

$$x_j^* = \begin{cases} a_j, & j \in Z0 \text{ and } \hat{x}_j \leq a_j, \\ b_j, & j \in Z0 \text{ and } \hat{x}_j \geq b_j, \\ \hat{x}_j, & j \in Z0 \text{ and } a_j < \hat{x}_j < b_j, \end{cases} \tag{30}$$

that is, subproblem $(P1^{\leq})$ *itself is decomposed into* $n_0 \equiv |Z0|$ *independent problems.*

The optimal solution to $(P2^{\leq})$ *is given by* (11), (12), (13) *with* $J := J \setminus Z0.$

The proof of Theorem 5 is omitted because it repeats in part the proof of Theorem 1. Analogous result holds for problem $(P^{=})$.

Theorem 6. *Problem* $(P^{=})$ *can be decomposed into two subproblems*: $(P1^{=})$ *for* $j \in Z0$ *and* $(P2^{=})$ *for* $j \in J \setminus Z0.$

The optimal solution to $(P1^{=})$ *is also given by* (30). *The optimal solution to* $(P2^{=})$ *is given by* (21), (22), (23) *with* $J := J \setminus Z0.$

Thus, with the use of Theorems 5 and 6, we can express components of the optimal solutions to problems $(P^{\leq})$ and $(P^{=})$ *without* the necessity of calculating the expressions $\dfrac{\hat{x}_j - a_j}{d_j}$ and $\dfrac{\hat{x}_j - b_j}{d_j}$ with $d_j = 0.$

5. Numerical Experiments

In this section, we present results of some numerical experiments obtained by applying a SQM with adaptive step-size regulation to multi-commodity inventory control problems of the form $(P^{\leq})$ and $(P^{=})$. The projection of the current approximation onto the feasible region was found by using the polynomial algorithms suggested in this paper.

Example 1.

$$\min\left\{F(\mathbf{x}) = E_{\boldsymbol{\omega}}\sum_{j=1}^{5}\max\{p_j(x_j - \omega_j),\ q_j(\omega_j - x_j)\}\right\}$$

subject to

$$x_1 + x_2 + 2x_3 + 3x_4 + x_5 = 200,$$

$$0 \leq x_1 \leq 50,$$

$$0 \leq x_2 \leq 7,$$

$$0 \leq x_3 \leq 7,$$

$$0 \leq x_4 \leq 80,$$

$$0 \leq x_5 \leq 25.$$

Here

$$\mathbf{P} = (p_1, p_2, p_3, p_4, p_5) = (1, 0, 3, 1, 2),$$

$$\mathbf{q} = (q_1, q_2, q_3, q_4, q_5) = (3, 4, 1, 2, 3),$$

and ω_j, $j = 1, 2, 3, 4, 5,$ are random variables uniformly distributed on the closed segments

$$[0, 60], [0, 15], [0, 17], [0, 90], [0, 40],$$

respectively.

Obviously, this is a problem of the form (2)-(3) with $n = 5$ and feasible set X of the form (6)-(7).

Optimal solution (by using SQM) (values of the last 10 iterations have been averaged):

$$\mathbf{x}^* = (42.08259, 6.98305, 3.76966, 41.86273, 17.80680).$$

Optimal value of $F(\mathbf{x})$: $F(\mathbf{x}^*) = 96.63854$.

The equality constraint is satisfied with tolerance: 0.00000000419.

This test example can also be solved analytically by using nonlinear programming approach. The analytical expression of the objective function is

$$F(\mathbf{x}) = \frac{1}{30}x_1^2 + \frac{2}{15}x_2^2 + \frac{2}{17}x_3^2 + \frac{1}{60}x_4^2 + \frac{1}{16}x_5^2 - 3x_1 - 4x_2 - x_3 - 2x_4 - 3x_5 + 278.5.$$

Analytic solution (by using a nonlinear programming approach, [17]):

$$\mathbf{x}^* = (41.88057, 7.00000, 2.48092, 41.27456, 22.33456).$$

Optimal value of $F(\mathbf{x})$: $F(\mathbf{x}^*) = 98.10089$.

Example 2.

$$\min\left\{F(\mathbf{x}) = E_{\boldsymbol{\omega}}\sum_{j=1}^{6}\max\{p_j(x_j - \omega_j),\ q_j(\omega_j - x_j)\}\right\}$$

subject to

$$3x_1 + 4x_2 + 7x_3 + 5x_5 + x_6 \le 872,$$

$$0 \le x_1 \le 45,$$

$$0 \le x_2 \le 56,$$

$$0 \le x_3 \le 32,$$

$$0 \le x_4 \le 27,$$

$$0 \le x_5 \le 456,$$

$$0 \le x_6 \le 45.$$

Here

$$\mathbf{p} = (p_1, p_2, p_3, p_4, p_5, p_6) = (5, 7, 8, 3, 5, 1),$$

$$\mathbf{q} = (q_1, q_2, q_3, q_4, q_5, q_6) = (4, 56, 7, 8, 4, 4),$$

and ω_j, $j = 1, 2, 3, 4, 5, 6$, are uniformly distributed random variables on closed segments

$$[0, 56], [0, 57], [0, 36], [0, 34], [0, 468], [0, 65],$$

respectively.

Obviously, this is a problem of the form (2)-(3) with $n = 6$ and feasible set X of the form (4)-(5).

Optimal solution (by using SQM) (values of the last 10 iterations have been averaged):

$$\mathbf{x}^* = (25.13417, 48.01999, 15.62868, 25.32763, 49.51728, 33.14936).$$

Optimal value of $F(\mathbf{x})$: $F(\mathbf{x}^*) = 972.59309$.

The effectiveness of algorithms for problems $(P^{\le})$ and $(P^{=})$ has been tested by many other examples.

References

[1] Yu. Ermoliev, Methods for Stochastic Programming, Nauka, Moscow, 1976 (in Russian).

[2] Yu. Ermoliev and A. Gaivoronski, Stochastic Quasigradient Methods and Their Implementation, Working Paper A-2361, IIASA, Laxenburg, Austria, 1984.

[3] A. Gaivoronski, Stochastic quasigradient methods and their implementation, (Yu. Ermoliev and R. J. -B. Wets (Eds.)), Numerical Techniques for Stochastic Optimization, Springer Verlag, Berlin, (1988), 313-351.

[4] P. Kall and S. W. Wallace, Stochastic Programming, Wiley, New York, 1994.

[5] R. T. Rockafellar and R. J.-B. Wets, A note about projections in the implementation of stochastic quasigradient methods, (Yu. Ermoliev and R. J.-B. Wets (Eds.)), Numer. Tech. Stoch. Optim., Springer Verlag, Berlin, (1988), 385-392.

[6] S. M. Stefanov, Facility Location Problems and Methods for Solving Them, M. S. Thesis, Sofia University, 1989, pp. 1-84 (in Bulgarian).

[7] S. M. Stefanov, Nondifferentiable and Stochastic Programming, PhD Dissertation, 1991, pp. 1-86.

[8] S. M. Stefanov, Convex separable minimization problems with linear constraint and bounds on the variables, Appl. Math. Engg. Eco., Heron Press, Sofia, (2002), 392-402.

[9] S. M. Stefanov, Method for solving a convex integer programming problem, Int. J. Math. Math. Sci. 2003(44) (2003), 2829-2834.

[10] S. M. Stefanov, Convex quadratic minimization subject to a linear constraint and box constraints, Appl. Math. Res. Express 2004(1) (2004), 17-42.

[11] S. M. Stefanov, Convex separable minimization problems with a linear constraint and bounded variables, Int. J. Math. Math. Sci. 2005(9) (2005), 1339-1363.

[12] S. M. Stefanov, Minimizing a convex separable exponential function subject to linear inequality or equality constraint and bounds on the variables, Journal of Interdisciplinary Mathematics 9(1) (2006), 207-226.

[13] S. M. Stefanov, An efficient method for minimizing a convex separable logarithmic function subject to a convex inequality constraint or linear equality constraint, J. Appl. Math. Deci. Sci. 2006 (2006), 1-19 (Article ID 89307).

[14] S. M. Stefanov, Minimization of a convex linear-fractional separable function subject to a convex inequality constraint or linear inequality constraint and bounds on the variables, Appl. Math. Res. Express 2006 (2006), 1-24 (Article ID 36581).

[15] S. M. Stefanov, Applications of convex separable unconstrained nonsmooth optimization to numerical approximation with respect to l_1- and l_∞-norms, Journal of Interdisciplinary Mathematics 10(6) (2007), 767-788.

[16] S. M. Stefanov, Solution of some convex separable resource allocation and production planning problems with bounds on the variables, Journal of Interdisciplinary Mathematics 13(5) (2010), 541-569.

[17] S. Urasiev, Adaptive Stochastic Quasigradient Procedures, (Yu. Ermoliev and R. J.-B. Wets (Eds.)), Numer. Tech. Stoch. Optim., Springer Verlag, Berlin, (1988), 373-384.

■