An Augmented Lagrangian Decomposition Method for Chance-Constrained Optimization Problems

Xiaodi Bai
Department of Applied Mathematics, College of Science,
Zhejiang University of Technology, Hangzhou, Zhejiang 310023, P.R. China, xdbai@zjut.edu.cn

Jie Sun
School of Business, National University of Singapore, Singapore, jsun@nus.edu.sg

Xiaojin Zheng
School of Economics and Management, Tongji University, Shanghai 200092, P.R. China, xjzheng@tongji.edu.cn

Abstract: Joint chance-constrained optimization problems under discrete distributions arise frequently in financial management and business operations. These problems can be reformulated as mixed-integer programs. The size of reformulated integer programs is usually very large even though the original problem is of medium size. This paper studies an augmented Lagrangian decomposition method for finding high-quality feasible solutions of complex optimization problems, including non-convex chance-constrained problems. Different from the current augmented Lagrangian approaches, the proposed method allows randomness to appear in both the left-hand-side matrix and the right-hand-side vector of the chance constraint. In addition, the proposed method only requires to solve a convex subproblem and a 0-1 knapsack subproblem at each iteration. Based on the special structure of the chance constraint, the 0-1 knapsack problem can be computed in quasi-linear time that keeps the computation for discrete optimization subproblems at relatively low level. The convergence of the method to a first-order stationary point is established under certain mild conditions. Numerical results are presented in comparison with a set of existing methods in the literature for various real-world models. It is observed that the proposed method compares favorably in terms of the quality of the best feasible solution obtained within a certain time for large size problems, particularly when the objective function of the problem is non-convex or the left-hand-side matrix of the constraints is random.

Key words: chance-constrained optimization problem; finite discrete distribution; alternating direction method of multipliers; augmented Lagrangian algorithm; first-order stationary point

History:

1. Introduction

Consider the following joint chance-constrained optimization problem

$$\begin{aligned} &\min f(x) \\ &\text{s.t. } \mathbb{P}\{g(x,\xi)\geq 0\}\geq 1-\alpha, \\ &\qquad x\in X, \end{aligned} \tag{P}$$

where $f:\ \Re^n\to\Re$ is a continuously differentiable function, $g=(g_1,\dots,g_m):\Re^n\times\Re^s\to\Re^m$, $g_j(x,\xi)$ $(j=1,\dots,m)$ are continuously differentiable concave functions of x, ξ is a random vector taking values in $\Re^s$, $\alpha\in(0,0.5)$ is a given small risk level, $\mathbb{P}$ denotes the probability, and X is a nonempty compact convex set in $\Re^n$. Throughout the paper, we assume that problem (P) is feasible. Problem (P) can be viewed as a stochastic version of the conventional constrained optimization problem. The probabilistic constraint in (P) is also called the *joint chance constraint*, which allows violation of the constraints $g(x,\xi)\geq 0$ with a small probability α. It is a reasonable relaxation of requiring $g(x,\xi)\geq 0$ to be held for all possible realizations of ξ, which could be very expensive or impossible in many situations.

Chance-constrained (or probabilistically constrained) optimization problems are introduced by Charnes et al. (1958), Charnes and Cooper (1959), Miller and Wagner (1965), and Prékopa (1970) and have been studied extensively in the stochastic optimization literature. Readers are referred to Prékopa (2003) and Shapiro et al. (2009) for comprehensive reviews on the theory and applications of those problems.

One of the major difficulties in solving problem (P) is the nonconvexity caused by the probabilistic constraint (Prékopa 2003, Lagoa et al. 2005, Henrion and Strugarek 2008). Several approximation approaches have been proposed to build tractable approximations to problem (P), which can be optimized efficiently using standard methods, such as convex programming (Rockafellar and Uryasev 2000, Nemirovski and Shapiro 2006a), CVaR based robust optimization (Chen et al. 2010), sequential convex approximation using various DC (difference of two convex functions) representations of the non-convex probability function (Hong et al. 2011, Hu et al. 2013, Shan et al. 2014), trust-region-based sequential quadratic programming method (Curtis et al. 2018), scenario approximations (Calafiore and Campi 2005, 2006, Nemirovski and Shapiro 2006b), the progressive hedging method (Rockafellar and Sun 2019), and many others.

In the literature, there is also extensive research under the assumption of finite discrete distribution on ξ. As shown by Ruszczyński (2002), under the assumption of finite discrete distribution, problem (P) can be reformulated as a mixed-integer problem and thus can be solved in the framework of branch-and-bound methodology. Recently, Luedtke (2014) proposes a branch-and-cut decomposition algorithm for chance-constrained mathematical programs with finite support. Adam and Branda (2016) derive necessary optimality conditions based on a relaxation reformulation with complementarity constraints. Motivated by these studies, we focus on the special case of problem (P) that satisfies the following two assumptions:

(A1) There are only finitely many realizations (scenarios) of ξ, i.e.,

$$\mathbb{P}(\xi = \xi^i) = p_i, \quad i = 1, \ldots, N, \quad \sum_{i=1}^{N} p_i = 1.$$

(A2) The constraint functions $g_j(x, \xi)$ $(j = 1, \ldots, m)$ are affine in x.

Assumption (A1) arises frequently in applications, either by using a set of past data, or by using sample approximations of the underlying distribution. Related results can be found, say, in Luedtke and Ahmed (2008) and Shapiro et al. (2009).

Assumption (A2) is also extensively used in the literature. Let us assume that the chance constraint takes the following form:

$$\mathbb{P}(Tx \geq \eta) \geq 1 - \alpha, \tag{1}$$

where T is an $m \times n$ random matrix and η is a random vector in $\Re^m$. We refer problem (P) with constraint (1) as the linear (joint) chance-constrained programming (LCCP). Two notable examples of LCCP are the Value-at-Risk (VaR) constrained portfolio selection problem and the chance-constrained transportation problem. In the literature, there are two research streams. One assumes that only the left-hand-side technology matrix T is random and the other assumes that only the right-hand-side vector η is random. Van Ackooij et al. (2011) and Cheng and Lisser (2012) propose efficient algorithms for LCCP when η is deterministic and T is a random matrix with certain continuous distributions. For instance, each row of T is normally distributed. Extensive research has been carried out for LCCP when the matrix T is deterministic and the right-hand-side vector η is random. Several authors considered mixed-integer linear programming (MILP) reformulations for

LCCP when η has finite discrete distribution. By identifying p-efficient points, Prékopa (1990) constructs improved MILP reformulations for LCCP. Lejeune and Noyan (2010) discuss how to generate p-efficient points using mathematical programming approaches. On the other hand, Dentcheva and Martinez (2012, 2013) propose a progressive augmented Lagrangian method to solve the chance-constrained optimization problem with random right-hand-side vector. Other methods for this case include the lower and upper bounds of LCCP approach of Dentcheva et al. (2000), the branch-and-bound methods for LCCP by Beraldi and Ruszczyński (2002a,b), the method of strengthened MILP reformulations of LCCP by Luedtke et al. (2010), the methods of disjunctive programming proposed by Sen (1992), the method based on extended disjunctive formulations by Vielma et al. (2012), the branch-reduce-cut algorithm based on domain reduction and linear programming relaxation for LCCP with a general distribution by Cheon et al. (2006), and the pattern-based modeling and solution method of Lejeune (2012). For more general non-convex chance-constrained problems, the systematic reformulation and algorithm is proposed by Lejeune and Margot (2016) for solving a class of non-convex stochastic programming problems with random technology matrix and stochastic quadratic inequality.

As problem (P) is in general non-convex, exact global solution methods are only suitable for problems of small to medium size. For large-size problems, it is natural to consider local or approximation methods. Those methods can not guarantee the global optimality but are capable of finding a high-quality feasible solution of (P) in reasonable computing time. The purpose of this paper is to present an efficient local method of this type for problem (P) under assumptions (A1) and (A2), where both T and η are random.

The contributions of this paper are as follows.

- We present an augmented Lagrangian decomposition formulation of problem (P). When certain variables are fixed, the formulation can lead to a nonlinear (possibly non-convex) programming subproblem over the set X or a 0-1 linear knapsack subproblem. We introduce an additional proximal term to guarantee the resulted subproblem to be convex and show that the 0-1 linear knapsack subproblem can be solved in quasi-linear time for the case of equal probabilities.
- Under certain mild conditions, we prove that the method converges to a first-order stationary point of problem (P).

- To evaluate the performance of the proposed method, we conduct computational experiments on real-world application problems such as the VaR-constrained portfolio selection model, the chance-constrained transportation model with random supply and the chance-constrained supply chain model with random demand and a non-convex objective function. The numerical results show that the proposed method is capable of finding suboptimal solutions of high quality for the tested problems.
- We compare the proposed method with a set of existing methods in the literature, some of which are aimed at finding an exact solution rather than a stationary point. The comparison results indicate that the proposed method is favorable particularly for the large-scale tested problems with a non-convex objective function or with a non-deterministic technical matrix.

After the first draft of this paper has been submitted, a referee kindly introduces us to three recent papers by Dentcheva et al. (2000) and Dentcheva and Martinez (2012, 2013). These papers use the idea of augmented Lagrangian methods to solve chance-constrained programming problems with discrete distributions, therefore bear some similarity to our proposed method. However, there are some major differences between the proposed method and the aforementioned methods (the existing methods for short). First, the existing methods aim at solving exactly the problem while our study primarily aims at finding high-quality feasible solution for complex problems, including non-convex chance-constrained problems. Second, the existing methods are applicable to the problems with random right-hand-side vector and can not be applied to the problems with random technology matrix. Third, the existing methods penalize inequality constraints, while our method penalizes equality constraints. Thus, our augmented Lagrangian function is different from the existing methods. Lastly, the existing methods compute p-efficient points repeatedly by mixed-integer programming techniques, while the proposed method reduces the work to the level of solving a 0-1 knapsack subproblem per iteration. Meanwhile, another referee points us to a paper by Ahmed et al. (2017) that proposes two new Lagrangian dual problems for chance constrained stochastic programs with discrete distribution. Both approaches in Ahmed et al. (2017) and the proposed method are applied to the integer reformulation of chance-constrained problems. The major difference between approaches of Ahmed et al. (2017) and the proposed method is that the methods in Ahmed et al. (2017) focus on the strength of the dual bounds, while the proposed method focus on finding the high quality feasible solution of the primal problem.

The remainder of the paper is organized as follows. In Section 2, we derive the augmented Lagrangian decomposition formulation of (P) and study the two subproblems obtained by fixing some variables. In Section 3, we describe the augmented Lagrangian decomposition method and prove its convergence to a first-order stationary point of (P). We conduct computational experiments in Section 4 on test problems from the aforementioned real-world application models. Finally, we make a few concluding remarks in Section 5.

2. Augmented Lagrangian Decomposition Formulation and Subproblems

We start this section by introducing an augmented Lagrangian reformulation for the mixed-integer program (MIP for short) formulation of problem (P). We then explore the special structure of this reformulation that eventually leads to an efficient numerical scheme for solving problem (P).

Let $d_i \in \Re^m$ be a lower bound vector of $g(x,\xi^i)$ over X for $i=1,\dots,N$. Since X is compact and $g(x,\xi^i)$ is affine in x, d_i is finite for $i=1,\dots,N$. Without loss of generality, we assume that $d_i \le 0$. By introducing a binary variable $z_i \in \{0,1\}$ for each scenario ξ^i, $i=1,\dots,N$, the probabilistic constraint $\mathbb{P}\{g(x,\xi)\ge 0\} \ge 1-\alpha$ can be expressed as

$$g(x,\xi^i) \ge z_i d_i,\ i=1,\dots,N,\ \sum_{i=1}^{N} p_i z_i \le \alpha,\ z\in\{0,1\}^N.$$

Hence, problem (P) can be reformulated as the following mixed-integer problem (Ruszczyński 2002):

$$\begin{aligned} \min\ & f(x) \\ \text{s.t.}\ & g(x,\xi^i) \ge z_i d_i,\ i=1,\dots,N, \\ & \sum_{i=1}^{N} p_i z_i \le \alpha,\ z\in\{0,1\}^N, \\ & x \in X. \end{aligned} \tag{MIP$_0$}$$

By introducing $y_i = g(x, \xi^i)$ $(i = 1, \ldots, N)$, we can rewrite the problem (MIP$_0$) as

$$\begin{aligned} \min\ & f(x) \\ \text{s.t. } & y_i = g(x, \xi^i),\ i = 1, \ldots, N, \\ & y_i \geq z_i d_i,\ i = 1, \ldots, N, \\ & \sum_{i=1}^{N} p_i z_i \leq \alpha,\ z \in \{0,1\}^N, \\ & x \in X. \end{aligned} \tag{MIP$_1$}$$

The constraints $y_i = g(x, \xi^i)$ $(i = 1, \ldots, N)$ can be viewed as *linkage* constraints, which can be enforced approximately by an augmented Lagrangian approach. Consider the following augmented Lagrangian function for problem (MIP$_1$):

$$\mathcal{L}(x, y, \lambda) := f(x) + \sum_{i=1}^{N} \lambda_i^T [y_i - g(x, \xi^i)] + \frac{\rho}{2} \sum_{i=1}^{N} \|y_i - g(x, \xi^i)\|^2, \tag{2}$$

where each $\lambda_i \in \Re^m$ is a multiplier vector and $\rho > 0$ is a penalty parameter. The resulting augmented Lagrangian decomposition formulation is

$$\begin{aligned} \min\ & \mathcal{L}(x, y, \lambda) \\ \text{s.t. } & (y, z) \in \Omega, \\ & x \in X, \end{aligned} \tag{ALP}$$

where

$$\Omega = \left\{ (y, z) \in \Re^{m \times N} \times \{0,1\}^N \mid y_i \geq z_i d_i,\ \sum_{i=1}^{N} p_i z_i \leq \alpha,\ i = 1, \ldots, N \right\}. \tag{3}$$

The structure of (ALP) suggests to apply the alternating direction method of multipliers for solving it. When applying the alternating direction method, however, an issue arises. We notice that for given $\lambda = \bar{\lambda}$, when the variables $(y, z) = (\bar{y}, \bar{z}) \in \Omega$ are fixed, problem (ALP) becomes

$$\min \mathcal{L}(x, \bar{y}, \bar{\lambda}) \quad \text{s.t. } x \in X. \tag{4}$$

If $f(x)$ is non-convex, the function $\mathcal{L}(x, \bar{y}, \bar{\lambda})$ may be non-convex, and then we can not guarantee that problem (4) is convex, which will bring complications in the analysis of the algorithm. As a remedy, we introduce a proximal term to the objective function of problem (4) to make the resulted subproblem be convex.

For a continuously differentiable and strictly convex function $\phi(x)$ on $\Re^n$, the Bregman distance D_ϕ associated with $\phi(x)$ for points $x^1, x^2 \in \Re^n$ is defined as

$$D_\phi(x^1, x^2) = \phi(x^1) - \phi(x^2) - \nabla\phi(x^2)^T(x^1 - x^2). \tag{5}$$

According to the property of the strictly convex function, we have $\phi(x^1) > \phi(x^2) + \nabla\phi(x^2)^T(x^1 - x^2)$ if $x^1 \neq x^2$. Thus, the function D_ϕ is nonnegative when $\phi(x)$ is strictly convex. By adding the function $D_\phi(x, \bar{x})$, named as *the Bregman proximal term*, to the objective function of problem (4), we obtain the following problem

$$\min \mathcal{L}(x, \bar{y}, \bar{\lambda}) + D_\phi(x, \bar{x}) \ \text{ s.t. } x \in X. \tag{6}$$

If $f(x)$ is convex, we can choose $\phi(x) = \frac{1}{2}\|x\|_H^2$ with $H \succ 0$, where $\|x\|_H^2 := x^T H x$, then $D_\phi(x, \bar{x}) = \frac{1}{2}\|x - \bar{x}\|_H^2$. On the other hand, if $f(x)$ is non-convex, we can choose $\phi(x)$ properly to make the problem (6) be convex. For example, since the Hessian of $f(x)$ is bounded on X, we can choose $\phi(x) = \frac{1}{2}\|x\|_H^2 - f(x)$ with $H \in \mathcal{H}$, where

$$\mathcal{H} = \{H \succ 0 \mid H - \nabla^2 f(x) \succ 0, \ \forall x \in X\}.$$

It it obvious that $\phi(x)$ is strictly convex. Then the objective function of problem (6) becomes

$$\begin{aligned}\mathcal{L}(x, \bar{y}, \bar{\lambda}) + D_\phi(x, \bar{x}) = & \sum_{i=1}^N \bar{\lambda}_i^T[\bar{y}_i - g(x, \xi^i)] + \frac{\rho}{2}\sum_{i=1}^N \|\bar{y}_i - g(x, \xi^i)\|^2 \\ & + f(\bar{x}) + \nabla f(\bar{x})^T(x - \bar{x}) + \frac{1}{2}\|x - \bar{x}\|_H^2.\end{aligned}$$

Since $g_j(x, \xi^i)$ $(j = 1, \ldots, m)$ are affine functions of x for each i, the objective function of problem (6) is a convex function of x. Thus, problem (6) is a convex programming problem of variables x over the compact convex set X.

Next, we consider the subproblem of problem (ALP) when $x = \bar{x} \in X$ is fixed. For given $\lambda = \bar{\lambda}$, problem (ALP) becomes

$$\min \mathcal{L}(\bar{x}, y, \bar{\lambda}) \ \text{ s.t. } (y, z) \in \Omega, \tag{7}$$

where Ω is defined in (3). Let $\bar{\delta}$ denote the optimal value of problem (7). Then

$$\bar{\delta} = f(\bar{x}) - \tau + \frac{\rho}{2}\min\left\{\sum_{i=1}^N \|y_i - w_i\|^2 \mid (y, z) \in \Omega\right\}, \tag{8}$$

where $w_i = g(\bar{x}, \xi^i) - \bar{\lambda}_i/\rho \in \Re^m$ and $\tau = \frac{1}{2\rho}\sum_{i=1}^N \|\bar{\lambda}_i\|^2$ is a constant. Let

$$r_i = \min_{y_i \geq 0} \|y_i - w_i\|^2, \ q_i = \min_{y_i \geq d_i} \|y_i - w_i\|^2, \qquad i = 1, \ldots, N. \tag{9}$$

It follows from (3) and (8) that

$$\begin{aligned}\bar{\delta} &= f(\bar{x}) - \tau + \frac{\rho}{2}\min\left\{\sum_{i=1}^N q_i z_i + r_i(1-z_i) \mid \sum_{i=1}^N p_i z_i \leq \alpha, \ z \in \{0,1\}^N\right\} \\ &= \bar{f} + \frac{\rho}{2}\min\left\{\sum_{i=1}^N (q_i - r_i) z_i \mid \sum_{i=1}^N p_i z_i \leq \alpha, \ z \in \{0,1\}^N\right\},\end{aligned} \tag{10}$$

where $\bar{f} = f(\bar{x}) - \tau + \frac{\rho}{2}\sum_{i=1}^N r_i$. Since $d_i \leq 0$, we have $q_i \leq r_i$ for each i. Thus, the minimization problem in (10) is equivalent to the following 0-1 linear knapsack problem.

$$\max\left\{\sum_{i=1}^N (r_i - q_i) z_i \mid \sum_{i=1}^N p_i z_i \leq \alpha, \ z \in \{0,1\}^N\right\}. \tag{11}$$

Although problem (11) is still NP-hard in theory for general finite discrete distribution, it is more tractable in computation than a general MIP approach as it can be solved by dynamic programming approaches very efficiently when p_i $(i = 1, \ldots, N)$ are rational numbers.

We now consider an important case of the 0-1 linear knapsack problem (11) when $p_i = 1/N$ for $i = 1, \ldots, N$. Let $K = \lfloor N\alpha \rfloor$, where $\lfloor N\alpha \rfloor$ is the maximum integer number less than or equal to $N\alpha$. The constraint in the 0-1 linear knapsack problem (11) reduces to $\sum_{i=1}^N z_i \leq K$. Thus, problem (11) is equivalent to the sum of the K largest entries of $\{r_i - q_i\}_{i=1}^N$, which can be computed in $O(N\log(N))$ by ranking the sequence $\{r_i - q_i\}_{i=1}^N$.

Finally, we see that r_i and q_i defined in (9) can be calculated in the following way:

$$r_i = \sum_{j=1}^m r_{ij}, \ q_i = \sum_{j=1}^m q_{ij}, \qquad i = 1, \ldots, N,$$

where

$$r_{ij} = \min_{y_{ij} \geq 0} (y_{ij} - w_{ij})^2 = \begin{cases} w_{ij}^2, \text{ if } w_{ij} < 0, \\ 0, \quad \text{if } w_{ij} \geq 0, \end{cases} \tag{12}$$

$$q_{ij} = \min_{y_{ij} \geq d_{ij}} (y_{ij} - w_{ij})^2 = \begin{cases} (d_{ij} - w_{ij})^2, \text{ if } w_{ij} < d_{ij}, \\ 0, \qquad\qquad\quad \text{if } w_{ij} \geq d_{ij}. \end{cases} \tag{13}$$

Therefore, the optimal y_i in the subproblem (7) can be easily determined by using (9), (12) and (13) via the optimal solution of problem (11).

In summary, we have shown that the two subproblems obtained by respectively fixing x or (y, z) in the augmented Lagrangian decomposition formulation can be solved efficiently. In the next section, we present the detailed algorithm and prove a convergence result.

3. An Augmented Lagrangian Decomposition Method

In this section, we propose an *augmented Lagrangian decomposition method* for problem (P) and prove its convergence to a first-order stationary point of problem (P).

The alternating direction method, also known as the block coordinate descent methods, has been successfully applied to structured convex programming problems arising from image processing and matrix optimization (see for examples, Yin et al. 2008, Goldstein and Osher 2009, Sun and Zhang 2010, He et al. 2012, Zhang et al. 2013, Shen et al. 2014). Here, we present an augmented Lagrangian decomposition method by applying the alternating direction method of multipliers with a proximal term, which appears to be new.

The basic idea of the proposed method is to solve the two subproblems (6) and (7) alternatively while updating the Lagrangian multipliers so as to enforce the least square penalty term to diminish during the iterations. By doing so, we are able to "isolate" the probabilistic constraint from other deterministic constraints, thus alleviating the difficulty of handing the probabilistic constraint. A detailed description of the method is as follows.

Algorithm 1 (The Augmented Lagrangian Decomposition Method).

Step 0. Choose tolerance parameter $\epsilon > 0$, multiplier vector λ^0, penalty parameter $\rho > 0$ and step-size $\kappa > 0$. Choose $x^0 \in X$ and $\phi(x)$. Set the iteration counter $k = 0$.

Step 1. Solve the subproblem problem (7) with $\bar{x} = x^k$ and $\bar{\lambda} = \lambda^k$ to obtain an optimal solution (y^{k+1}, z^{k+1}).

Step 2. Solve the convex programming subproblem (6) with $(\bar{x}, \bar{y}, \bar{\lambda}) = (x^k, y^{k+1}, \lambda^k)$ to obtain an optimal solution x^{k+1}.

Step 3. If $\sum_{i=1}^N \|y_i^{k+1} - g(x^{k+1}, \xi^i)\|^2 \le \epsilon$, stop, x^{k+1} is an approximate optimal solution. Otherwise, go to Step 4.

Step 4. Update the multipliers by

$$\lambda_i^{k+1} = \lambda_i^k + \kappa\rho\big[y_i^{k+1} - g(x^{k+1}, \xi^i)\big], \ i = 1, \dots, N. \tag{14}$$

Set $k := k + 1$, and go to Step 1.

Before establishing the convergence properties of Algorithm 1, we need to present the local optimality conditions for (P). To simplify the proof of the lemmas and the convergence theorem, we make the following assumption in the sequel.

(A3) The probabilities of each scenario $\xi = \xi^i$ $(i = 1, \dots, N)$ are equal, i.e.,

$$p_i = \mathbb{P}(\xi = \xi^i) = 1/N, \quad i = 1, \dots, N.$$

Assumption (A3) has been used in the literatures, see Luedtke et al. (2010), Luedtke (2014), and some references therein. The scenarios can be obtained from the observation of the historical data, or Monte Carlo based samples from some given probability distribution. In such cases, the resulted scenarios often have the same probability.

For $y = (y_1, \dots, y_N) \in \Re^{m \times N}$, where $y_i = (y_{i1}, \dots, y_{im})^T$, define $s(y) = (s(y)_1, \dots, s(y)_N)^T$, where

$$s(y)_i = \min_{j=1,\dots,m} y_{ij}, \qquad i = 1, \dots, N.$$

Let $G(x) = [g(x, \xi^1), \dots, g(x, \xi^N)] \in \Re^{m \times N}$. Then, $g(x, \xi^i) \geq 0$ if and only if $s(G(x))_i \geq 0$. Since each g_j is a linear function of x, $s(G(x))_i$ $(i = 1, \dots, N)$ are concave functions of x. Let $a_{[k]}$ denote the kth smallest entry of a vector $a \in \Re^N$. Then, by Assumption (A3), we have

$$\mathbb{P}\{g(x, \xi) \geq 0\} \geq 1 - \alpha \iff s(G(x))_{[K]} \geq 0, \tag{15}$$

where $K = \lfloor N\alpha \rfloor$. For convenience, we denote in the sequel that $\chi(x) := s(G(x))_{[K]}$.

Therefore, problem (P) can be rewritten as

$$\begin{aligned} &\min f(x) \\ &\text{s.t. } \chi(x) \geq 0, \quad x \in X. \end{aligned} \tag{16}$$

Lemma 1. The function $\chi(x)$ is locally Lipschitz on $\Re^n$.

Proof. Note that each $\chi(x) = s(G(x))_{[K]}$ is a composite function of $u = s(G(x))$ and $u_{[K]}$. Since $s(G(x))_i = \min_{j=1,\dots,m} g_j(x, \xi^i)$ is concave and hence locally Lipschitz on $\Re^n$, it suffices to show that $u_{[K]}$ is a locally Lipschitz function of $u \in \Re^N$ (Clarke 1983). Note that $u_{[K]}$ can be expressed as

$$u_{[K]} = \sum_{j=K}^{N} u_{[j]} - \sum_{j=K+1}^{N} u_{[j]}. \tag{17}$$

Let $\psi_t(u)=\sum_{j=t}^{N} u_{[j]}$. Then, $u_{[K]}=\psi_K(u)-\psi_{K+1}(u)$. Since $\psi_t(u)$ is the sum of the $N-t+1$ largest components of u, it is a convex function of u (see, e.g., Example 3.6 in Boyd and Vandenberghe 2004). Thus, $u_{[K]}$ is the difference of two convex functions and hence is locally Lipschitz function since convex functions and concave functions are locally Lipschitz and the sum of two locally Lipschitz functions is a locally Lipschitz function. □

By Lemma 1, the Clarke generalized gradient of $\chi(x)$ exists for any $x\in\Re^n$. Moreover, $\chi(x)$ can be viewed as a composite of $y=G(x)$ and $s(y)_{[K]}$. By the chain rule (Clarke 1983), we have

$$\partial\chi(x)=\text{conv}\left\{\sum_{i=1}^{N}\nabla g(x,\xi^i)\theta_i:\ \theta_i\in\partial_{y_i}(s(y)_{[K]})\,|_{y=G(x)}\right\}, \tag{18}$$

where $\nabla g(x,\xi^i)\in\Re^{n\times m}$ is the Jacobian matrix of $g(x,\xi^i)$ and $\partial_{y_i}(s(y)_{[K]})\,|_{y=G(x)}$ denotes the partial generalized Clarke gradient of $s(y)_{[K]}$ at $y=G(x)$ with respect to $y_i\in\Re^m$, $i=1,\dots,N$.

We have the following first-order stationary condition for (16) (Clarke 1983): If x^* is a local minimizer of problem (16), then there exists $\pi^*\geq 0$ such that

$$0\in\nabla f(x^*)-\pi^*\partial\chi(x^*)+\mathcal{N}_X(x^*), \tag{19}$$

$$\pi^*\chi(x^*)=0, \tag{20}$$

where $\mathcal{N}_X(x^*):=\{y\in\Re^n\mid\forall x\in X,\ \langle y,x-x^*\rangle\leq 0\}$, which is the normal cone of X at x^*.

We are now ready to state the convergence theorem of Algorithm 1.

Theorem 1. Let (x^*,y^*,λ^*) be any accumulation point of $\{(x^k,y^k,\lambda^k)\}$ generated by Algorithm 1. Assume that $\{\lambda^k\}$ is bounded and

$$\sum_{k=1}^{\infty}\|\lambda^{k+1}-\lambda^k\|^2<\infty. \tag{21}$$

Then, x^* satisfies the first-order stationary conditions (19)-(20).

Before proceeding to the proof of Algorithm 1, a few remarks are in order. The assumption on the boundedness of the multiplier vectors in Theorem 1 is a standard condition in the convergence analysis of augmented Lagrangian methods for non-convex optimization problems (Bertsekas 1982, Luo et al. 2007). Roughly speaking, the condition (21) is to ensure that the difference of multipliers between two iterations shrinks faster than $1/\sqrt{k}$

as the iteration proceeds. Similar conditions have been used in the convergence analysis of alternating direction methods for non-convex optimization problems (Shen et al. 2014).

We need the following lemma.

Lemma 2. Under the assumptions of Theorem 1, it holds that

$$\|x^k - x^{k+1}\| \to 0,\ \|y^k - y^{k+1}\| \to 0,\ k \to \infty. \tag{22}$$

Proof. Since x^{k+1} solves (6) at the kth iteration and $x^k - x^{k+1}$ is a feasible direction with respect to X, we have

$$[\nabla_x \mathcal{L}(x^{k+1}, y^{k+1}, \lambda^k) + \nabla\phi(x^{k+1}) - \nabla\phi(x^k)]^T (x^k - x^{k+1}) \geq 0. \tag{23}$$

From (2) and (5), we see that $\mathcal{L}(x, y, \lambda) + D_\phi(x, x^k)$ is a strongly convex function (with certain parameter $\bar{\rho}/2$) of variable x, we have

$$\begin{aligned} &\mathcal{L}(x^k, y^{k+1}, \lambda^k) - \mathcal{L}(x^{k+1}, y^{k+1}, \lambda^k) - D_\phi(x^{k+1}, x^k) \\ \geq\ &[\nabla_x \mathcal{L}(x^{k+1}, y^{k+1}, \lambda^k) + \nabla\phi(x^{k+1}) - \nabla\phi(x^k)]^T (x^k - x^{k+1}) + \frac{\bar{\rho}}{2}\|x^k - x^{k+1}\|^2. \end{aligned}$$

This together with (23) gives rise to

$$\mathcal{L}(x^k, y^{k+1}, \lambda^k) - \mathcal{L}(x^{k+1}, y^{k+1}, \lambda^k) - D_\phi(x^{k+1}, x^k) \geq \frac{\bar{\rho}}{2}\|x^k - x^{k+1}\|^2. \tag{24}$$

On the other hand, since y^{k+1} solves (7) at the k-iteration and y^k is feasible to (7), we have

$$\mathcal{L}(x^k, y^k, \lambda^k) - \mathcal{L}(x^k, y^{k+1}, \lambda^k) \geq 0. \tag{25}$$

Moreover, by (2) and (14), we have

$$\begin{aligned} &\mathcal{L}(x^{k+1}, y^{k+1}, \lambda^k) - \mathcal{L}(x^{k+1}, y^{k+1}, \lambda^{k+1}) \\ =\ &\sum_{i=1}^N (\lambda_i^k - \lambda_i^{k+1})^T (y_i^{k+1} - g(x^{k+1}, \xi^i)) \\ =\ &-\frac{1}{\kappa\rho}\|\lambda^k - \lambda^{k+1}\|^2. \end{aligned} \tag{26}$$

Combining with (24)-(26), we obtain

$$\begin{aligned} &\mathcal{L}(x^k, y^k, \lambda^k) - \mathcal{L}(x^{k+1}, y^{k+1}, \lambda^{k+1}) \\ =\ &\mathcal{L}(x^k, y^k, \lambda^k) - \mathcal{L}(x^k, y^{k+1}, \lambda^k) + \mathcal{L}(x^k, y^{k+1}, \lambda^k) \\ &-\mathcal{L}(x^{k+1}, y^{k+1}, \lambda^k) + \mathcal{L}(x^{k+1}, y^{k+1}, \lambda^k) - \mathcal{L}(x^{k+1}, y^{k+1}, \lambda^{k+1}) \\ \geq\ &\frac{\bar{\rho}}{2}\|x^k - x^{k+1}\|^2 - \frac{1}{\kappa\rho}\|\lambda^k - \lambda^{k+1}\|^2 + D_\phi(x^{k+1}, x^k) \\ \geq\ &\frac{\bar{\rho}}{2}\|x^k - x^{k+1}\|^2 - \frac{1}{\kappa\rho}\|\lambda^k - \lambda^{k+1}\|^2. \end{aligned} \tag{27}$$

Since $\{x^k\} \subset X$ and by assumption X and $\{\lambda^k\}$ are bounded, we deduce from (14) that $\{y^k\}$ is also bounded. Thus, $\{\mathcal{L}(x^k, y^k, \lambda^k)\}$ is bounded. Summing up both sides of (27) over all k yields

$$\frac{\bar{\rho}}{2}\sum_{k=1}^{\infty}\|x^k - x^{k+1}\|^2 - \frac{1}{\kappa\rho}\sum_{k=1}^{\infty}\|\lambda^k - \lambda^{k+1}\|^2 < \infty. \tag{28}$$

By assumption, $\sum_{k=1}^{\infty}\|\lambda^k - \lambda^{k+1}\|^2 < \infty$, thus (28) implies that

$$\|x^k - x^{k+1}\| \to 0, \qquad k \to \infty. \tag{29}$$

Since $\lambda^k - \lambda^{k+1} \to 0$ $(k \to \infty)$, it follows from (14) that

$$\|y_i^{k+1} - g(x^{k+1}, \xi^i)\| = \frac{1}{\kappa\rho}\|\lambda_i^k - \lambda_i^{k+1}\| \to 0, \qquad k \to \infty. \tag{30}$$

This together with (29) and the continuity of $g(x, \xi^i)$ with respect to x implies that

$$\begin{aligned} y_i^k - y_i^{k+1} &= [y_i^k - g(x^k, \xi^i)] + [g(x^k, \xi^i) - g(x^{k+1}, \xi^i)] \\ &\quad + [g(x^{k+1}, \xi^i) - y_i^{k+1}] \to 0, \qquad k \to \infty. \end{aligned}$$

Therefore, (22) holds. □

Proof of Theorem 1. Let (x^*, y^*, λ^*) be an accumulation point of $\{(x^k, y^k, \lambda^k)\}$. Then, there exists a subsequence $\{(x^k, y^k, \lambda^k)\}_{k\in\mathcal{I}}$ that converges to (x^*, y^*, λ^*). We first note that under Assumption (A3), $(y, z) \in \Omega$ is equivalent to $s(y)_{[K]} \geq 0$. Hence, the subproblem (7) with $\bar{x} = x^k \in X$ can be rewritten as

$$\begin{aligned} &\min\ \mathcal{L}(x^k, y, \lambda^k) \\ &\text{s.t. } \ s(y)_{[K]} \geq 0. \end{aligned} \tag{31}$$

Since y^{k+1} solves (31), there exists $\pi^k \geq 0$ such that

$$\nabla_{y_i}\mathcal{L}(x^k, y^{k+1}, \lambda^k) - \pi^k \zeta_i^k = 0, \qquad i = 1, \ldots, N, \tag{32}$$

$$\pi^k s(y^{k+1})_{[K]} = 0, \tag{33}$$

where $\zeta_i^k \in \partial_{y_i}(s(y^{k+1})_{[K]})$ and

$$\nabla_{y_i}\mathcal{L}(x^k, y^{k+1}, \lambda^k) = \lambda_i^k + \bar{\rho}(y_i^{k+1} - g(x^k, \xi^i)). \tag{34}$$

It is easy to verify that $\partial_{y_i} s(y) = \mathrm{conv}\{e_j \mid y_j = s(y),\ j = 1,\dots,m\}$, where e_j is the jth unit vector in $\Re^m$, and $\partial_i(u_{[K]}) = \mathrm{conv}\{e_i:\ u_i = u_{[K]}\}$, where e_i is the ith unit vector in $\Re^N$. Hence, the set $\partial_{y_i}(s(y^{k+1})_{[K]})$ is bounded away from zero. Consequently, there exists some constant $\delta_0 > 0$ such that $\|\zeta_i^k\| \geq \delta_0$ for all k and $i = 1,\dots,N$. It then follows from (32) that $\{\pi^k\}_{k\in\mathcal{I}}$ is bounded. Also, since the Clarke generalized gradient is a lower semi-continuous set-valued mapping (see Proposition 2.4.4 in Clarke 1983), $\{\zeta^k\}_{k\in\mathcal{I}}$ is bounded. Without loss of generality, we assume that $\pi^k \to \pi^*$ and $\zeta^k \to \zeta^*$ $(k\to\infty,\ k\in\mathcal{I})$. Taking limits in (32)-(34) for $k\in\mathcal{I}$ and using Lemma 2, we get

$$\lambda_i^* - \pi^*\zeta_i^* = 0, \qquad i = 1,\dots,N, \tag{35}$$

$$\pi^* s(y^*)_{[K]} = 0, \tag{36}$$

where $\zeta_i^* \in \partial_{y_i}(s(y^*)_{[K]})$, $\pi^* \geq 0$, $y_i^* = g(x^*,\xi^i)$, $i = 1,\dots,N$.

On the other hand, since x^{k+1} solves the subproblem (6) when $(\bar{x},\bar{y},\bar{z}) = (x^k, y^{k+1}, z^{k+1})$, the first-order optimality condition of (6) is

$$0 \in \nabla_x \mathcal{L}(x^{k+1}, y^{k+1}, \lambda^k) + \nabla\phi(x^{k+1}) - \nabla\phi(x^k) + \mathcal{N}_X(x^{k+1}), \tag{37}$$

where

$$\nabla_x \mathcal{L}(x^{k+1}, y^{k+1}, \lambda^k) = \nabla f(x^{k+1}) - \sum_{i=1}^N \nabla g(x^{k+1},\xi^i)[\lambda_i^k + \bar{\rho}(y_i^{k+1} - g(x^{k+1},\xi^i)].$$

Since the normal cone of a compact set is a lower semi-continuous set-valued mapping (see Proposition 2.4.4 in Clarke 1983), taking limits for $k\in\mathcal{I}$ on (37) and using Lemma 2 and (30), we obtain

$$0 \in \nabla f(x^*) - \sum_{i=1}^N \nabla g(x^*,\xi^i)\lambda_i^* + \mathcal{N}_X(x^*). \tag{38}$$

Note that $\chi(x^*) = s(G(x^*))_{[K]} = s(y^*)_{[K]}$. Combining (35), (36) and (38), and noting the chain rule (18), we conclude that x^* is a first-order stationary point satisfying conditions (19) and (20).

□

4. Numerical Results

We conduct computational experiments to evaluate the performance of Algorithm 1. The main purpose of the computational experiments is to test the capability of the proposed method for finding a high quality solution within a given time limit. We first consider the class of problems with random technology matrices. The test problems are chosen from VaR-constrained portfolio selection problem and the chance-constrained transportation problem. For this class of problems, we compare our method with the greedy method (Reich 2013), the bisection-based CVaR approximation (suggested by a referee), the DC method (Hong et al. 2011), the trust-region-based sequential quadratic approximation method (Curtis et al. 2018), and the CPLEX package applied to the mixed-integer program reformulation (MIP$_0$). We also consider the class of problems with random right-hand-side vector and non-convex objective function. This class of problems arises from the chance-constrained supply chain problem. We compare our method with the progressive augmented Lagrangian method in Dentcheva and Martinez (2013) and the trust-region-based sequential quadratic approximation method in Curtis et al. (2018). All the data files of the test problems are available at `https://sem.tongji.edu.cn/semch_data/faculty_cv/xjz/ccop.html`.

The numerical tests are run on a personal computer equipped with Intel Core i5 CPU(3.3GHz) and 16GB RAM under Windows 7. All the algorithms are implemented in MATLAB R2013b. All linear and convex quadratic subproblems in Algorithm 1 and the mixed-integer linear and quadratic reformulations are solved by the quadratic programming solver and mixed integer programming solver in CPLEX 12.6.3 with MATLAB interface. We use CPLEX default settings and set the time limit to 1800 seconds[1]. The non-convex quadratic programming subproblems in progressive augmented Lagrangian method are solved by SNOPT (Gill et al. 2005, 2018), which is called from a driver program in MATLAB.

The parameters in Algorithm 1 are set as follows. The tolerance parameter is set as $\epsilon = 10^{-6}$; the step length $\kappa = 0.1$; the initial Lagrangian multiplier $\lambda^0 = 0$, and $x^0 = \arg\min\{f(x) \mid x \in X\}$. It is possible to strengthen the values of the lower bound d_i ($i = 1, \dots, N$) by using the techniques proposed in Qiu et al. (2014) and Song et al. (2014).

[1] Since we only check the time limit at the starting point of each iteration, the actual finishing time of an algorithm may be longer than this limit.

However, such strengthening procedure is very time consuming and have no obvious effect in our test. Therefore we simply set $d_i = \min\{g(x,\xi^i) \mid x \in X\}$, $i = 1,\ldots,N$. For all the test problems, we set $p_i = 1/N$ for $i = 1,\ldots,N$. In order to get the accumulation point of the sequence generated by our algorithm, the number of iterations of our method should be as large as possible. In our numerical test, we set the stop criteria as $\sum_{i=1}^N \|y^k - g(x^k,\xi^i)\|^2 \le \epsilon$. Since the test problems are non-convex, none of the tested instance obtained provable local optimal solutions within the time limit. Instead, we use the achieved objective values ("fval" in the tables) as a quality indicator of the solutions.

The notations used in Tables 2-5 are listed in Table 1.

Table 1 Notations used in Tables 2 - 5

ALDM:	the augmented Lagrangian decomposition method (Algorithm 1);
Greedy:	the greedy method proposed in Reich (2013)
BiCVaR:	the bisection-based CVaR approximation described in subsection 4.1
DC:	the DC method proposed by Hong et al. (2011)
CPLEX:	the CPLEX mixed-integer programming solver applied to (MIP$_0$)
SQP-TR:	the trust-region-based sequential quadratic programming proposed in Curtis et al. (2018)
PAL:	the progressive augmented Lagrangian method proposed in Dentcheva and Martinez (2013)
fval:	the average objective value (upper bound) for the 5 test problems
time:	the average computing time in seconds for the 5 test problems
iter:	the average number of iterations of ALDM for the 5 test problems

4.1. Bisection-based CVaR approximation

The idea of using CVaR as a convex conservative approximation of the chance constraint is due to Rockafellar and Uryasev (2000). Under assumption (A1)-(A2), the CVaR approximation of problem (P) can be expressed as

$$\begin{aligned}
&\min\ f(x)\\
&\text{s.t. } t + \tfrac{1}{\alpha N}\sum_{i=1}^N y_i \le 0,\\
&\qquad y_i \ge 0,\ y_i \ge -g_j(x,\xi^i) - t,\ i = 1,\ldots,N,\ j = 1,\ldots,m,\\
&\qquad x \in X,\ t \in \mathbb{R}.
\end{aligned} \tag{P$_{\text{CVaR}}$}$$

The CVaR approximation is the "best" convex approximation of the chance constraint, but it is sometimes too conservative for the problem (P). As suggested by one of the referees, we use a bisection-based CVaR approximation to compute a approximation solution. The algorithm is described as follows.

Algorithm 2 (Bisection-based CVaR approximation for (P)).

Step 0. Given tolerance $\epsilon_1 > 0$. Let $\alpha_l = \alpha$, $\alpha_u = 1$, and $\alpha_0 = (\alpha_l + \alpha_u)/2$. Set the iteration counter $k = 0$.

Step 1. Solve the problem ($\mathrm{P_{CVaR}}$) with $\alpha = \alpha_k$ to obtain an optimal solution x^k.

Step 2. If $\mathbb{P}\{g(x^k,\xi) \geq 0\} \geq 1-\alpha$, let $\alpha_l = \alpha_k$. Otherwise, let $\alpha_u = \alpha_k$.

Step 3. If $\alpha_u - \alpha_l \leq \epsilon_1$, goto Step 4. Otherwise, let $\alpha_k = (\alpha_l + \alpha_u)/2$ and goto Step 1

Step 4. Solve the problem ($\mathrm{P_{CVaR}}$) with $\alpha = \alpha_l$ to obtain an optimal solution x^{CVaR} as the approximation solution of problem (P).

4.2. The VaR-constrained portfolio selection problem

In this subsection, we consider the VaR-constrained mean-variance portfolio selection problem of following form.

$$\begin{aligned} &\min\ \beta x^T \Sigma x - \mu^T x \\ &\text{s.t. } \mathbb{P}(\xi^T x \geq R) \geq 1-\alpha, \\ &\qquad e^T x = 1,\ 0 \leq x \leq u, \end{aligned} \tag{VaR-MV}$$

where ξ is the random vector of returns of n risky assets, Σ and μ are the covariance matrix and expectation of these returns, β is the risk aversion factor, $x = (x_1, x_2, \ldots, x_n)^T$ is the vector of portfolio weights, R is the prescribed minimal level of return, e is the all-one column vector and $0 < u \leq e$ is an upper bound vector for x. This model has been used and studied widely in the field of risk management. For more details, readers are referred to Steinbach (2001) and the reference therein.

In our test, we set $\beta = 2$, $u = 0.5e$ and $\alpha = 0.05$ or 0.10, respectively. To build the test problems for (VaR-MV), we use both real market data and simulated data to construct the asset return samples ξ, the covariance matrix Σ and the expectation μ.

- **Real data set**. We use 2523 daily return data of 435 stocks included in Standard & Poor's 500 index between March 2006 and March 2016. The covariance matrix Σ is calculated by the sample covariance using the 2523 historical data. We generate the test problems with n ranged from 50 to 400. For each n, we generate 5 instances by randomly selecting n stocks from the 435 stocks and $N = 3n$ samples ξ^i $(i = 1, \ldots, N)$ from the 2523

Table 2 **Comparison results for (VaR-MV) with different number of stocks using real data set**

		ALDM			Greedy		BiCVaR		DC		SQP-TR		CPLEX	
α	N	fval	time	iter	fval	time	fval	time	fval	time	fval	time	fval	time
0.05	50	1.9065	2.02	690	1.9192	0.71	2.1611	0.09	2.0039	0.30	1.9545	0.16	1.9049	0.49
0.05	100	1.8562	2.74	730	1.8586	5.93	2.1796	0.28	1.9681	1.32	1.8928	2.49	1.8421	376.29
0.05	150	1.7230	3.67	736	1.7345	20.61	2.0037	0.67	1.8166	5.49	1.7546	16.68	1.7094	1799.92
0.05	200	1.6754	5.36	727	1.6925	57.71	1.9758	1.23	1.7842	16.57	1.7139	271.65	1.6673	1799.94
0.05	250	1.7126	7.50	715	1.7202	141.83	1.9932	2.25	1.8121	38.20	1.9010	1631.03	1.7172	1799.97
0.05	300	1.6878	9.29	722	1.7032	275.14	1.9838	3.38	1.7875	407.40	1.9146	1801.93	1.6897	1799.97
0.05	350	1.6599	12.48	730	1.6883	479.06	1.9507	5.29	1.7763	103.54	1.8921	1803.10	1.6775	1800.01
0.05	400	1.6621	16.05	720	1.6830	848.17	1.9684	7.25	1.7915	182.56	1.9399	1804.72	1.6818	1800.07
0.10	50	1.7845	1.86	655	1.7866	1.07	1.9046	0.08	1.8661	0.42	1.8026	0.14	1.7813	0.44
0.10	100	1.6341	2.62	721	1.6548	9.93	1.8864	0.27	1.6957	2.43	1.6834	1.23	1.6227	557.79
0.10	150	1.5241	3.45	708	1.5389	35.67	1.7176	0.64	1.5989	9.86	1.5340	17.38	1.5140	1586.34
0.10	200	1.4623	5.02	690	1.4888	94.40	1.6539	1.14	1.5430	28.73	1.4863	21.42	1.4559	1799.91
0.10	250	1.4996	7.10	713	1.5208	242.18	1.7098	2.13	1.6098	68.69	1.5865	1282.94	1.4961	1799.96
0.10	300	1.4607	8.69	705	1.4954	454.80	1.6656	3.16	1.5508	118.31	1.4810	1040.13	1.4631	1799.96
0.10	350	1.4374	11.92	727	1.4663	809.06	1.6308	4.51	1.5347	207.04	1.5769	1818.16	1.4398	1800.00
0.10	400	1.4435	15.14	716	1.4662	1443.84	1.6489	6.03	1.5303	371.85	1.6071	1804.09	1.4624	1800.02

daily return data. For each instance, the prescribed annualized return level is set equal to 5%, so the daily return level is $R = 5\%/250 = 0.02\%$.

- **Simulated data set**. To evaluate the performance of the method for large-scale problems, we also construct test problems of (VaR-MV) with n up to 1000 and $N = 3000$, using simulation data generated in a similar fashion as in Nemirovski and Shapiro (2006a). The first asset has deterministic return $r_1 \equiv 1$ and the returns of the remaining $n-1$ assets are random variables with expectations $\mathbb{E}[r_i] = 1 + \bar{r}_i$, where the nominal profits $\bar{r}_i$ vary in [0, 0.1]. The random returns of assets are generated by the following factor model:

$$r_i = \sum_{l=1}^{8} \gamma_{il}\zeta_l + \eta_i, \qquad i = 2, \ldots, n, \tag{39}$$

where $\log\zeta_l \sim \mathcal{N}(0, 0.1^2)$, $\log\eta_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$ with $\sigma_i = \mu_i$, and $\gamma_{il} \geq 0$ are deterministic influence coefficients. The coefficients γ_{il} and the parameters μ_i are chosen in such a way that $\mathbb{E}[\sum_{l=1}^{8} \gamma_{il}\zeta_l] = \bar{r}_i/2$ and $E[\eta_i] = 1 + \bar{r}_i/2$ for all i. For each n, we construct 5 instances by randomly generating N samples of $\xi^i = (r_1, \ldots, r_n)^T$ by the factor model (39). The covariance matrix Σ is calculated using (39). The return level is set as $R = 1.06$.

Table 2 summarizes the comparison results for test problems (VaR-MV), where n ranges from 50 to 400 and $N = 3n$. A real data set is used. From Table 2, we see that the greedy method, ALDM and CPLEX can find solutions with significantly better objective values than those found by the BiCVaR approximation, the DC method and SQP-TR. ALDM can achieve better objective values than the DC method, while it spends less computing time

Table 3 **Comparison results for (VaR-MV) with different number of stocks using simulated data set**

		ALDM			Greedy		BiCVaR		DC		SQP-TR		CPLEX	
α	N	fval	time	iter	fval	time	fval	time	fval	time	fval	time	fval	time
0.05	500	-0.8521	31.38	436	-0.8318	1816.40	-0.8485	99.32	-0.8453	66.15	-0.8512	1605.25	-0.8490	1804.18
0.05	600	-0.8768	48.61	416	-0.8603	1811.49	-0.8739	130.87	-0.8712	79.06	-0.8729	1856.39	-0.8742	1803.79
0.05	700	-0.8953	70.22	412	-0.8806	1819.00	-0.8929	169.30	-0.8909	107.59	-0.8947	1571.27	-0.8925	1802.82
0.05	800	-0.9108	102.22	403	-0.9001	1830.58	-0.9088	217.84	-0.9073	125.32	-0.9102	1889.66	-0.9087	1803.54
0.05	900	-0.9220	146.50	385	-0.9130	1831.99	-0.9202	279.36	-0.9188	118.50	-0.9211	1912.72	-0.9193	1804.72
0.05	1000	-0.9294	187.05	365	-0.9214	1822.50	-0.9279	347.32	-0.9268	132.22	-0.9287	1940.99	-0.9265	1807.54
0.10	500	-0.8571	31.66	427	-0.8298	1819.85	-0.8544	118.89	-0.8492	71.57	-0.8559	1178.94	-0.8527	1801.06
0.10	600	-0.8815	48.03	414	-0.8604	1816.46	-0.8793	154.61	-0.8752	85.62	-0.8806	1550.00	-0.8786	1802.30
0.10	700	-0.8992	69.06	411	-0.8814	1824.01	-0.8974	200.36	-0.8940	86.41	-0.8986	864.67	-0.8966	1801.91
0.10	800	-0.9137	101.77	395	-0.8991	1832.58	-0.9120	264.45	-0.9094	148.16	-0.9132	1254.97	-0.9115	1804.11
0.10	900	-0.9245	135.30	376	-0.9123	1816.85	-0.9231	332.62	-0.9208	111.33	-0.9241	943.95	-0.9220	1807.60
0.10	1000	-0.9313	177.29	366	-0.9209	1833.09	-0.9300	372.87	-0.9283	135.77	-0.9310	1152.32	-0.9274	1807.55

than the DC method when $n \geq 150$. We observe that when $n = 50$ and $n = 100$, CPLEX can find the global optimal solution within 1800 seconds, while ALDM and other methods can only find a feasible solution, but the relative gap between the value found by ALDM and global optimal value is less than 1%. We also observe that ALDM can find slightly better solutions than CPLEX within 1800 seconds for larger size problem. As expected, the computing time of the BiCVaR approximation is much less than the other five methods because it only solves a few convex quadratic programming problem. Note also from Table 2 that the solution time for the greedy method grows sharply when the sample size N increases. This is mainly because the number of quadratic subproblems to be solved in the greedy method is in the order of $O(N^2\alpha)$.

To see the performance of different methods for large-scale problems, we report in Table 3 the comparison results for (VaR-MV) with n up to 1000 and $N = 3000$ using simulation data set. For these large test problems, ALDM appears to perform better than the other methods tested in this paper in terms of the objective values, and the performance of ALDM and DC is robust. We observed that BiCVaR approximation can find a slightly better solution than DC method at the price of using significantly more time, while ALDM can find slightly better solution than BiCVaR in less time. We also observed that for all large-scale instances, due to the large input sample size N, Greedy and CPLEX was terminated because of reaching the CPU time limit (1800 seconds).

In summary, the above comparison results suggest that the proposed augmented Lagrangian decomposition method is effective in finding solutions of good quality for large-scale problems of (VaR-MV) in a reasonable time limit and is favorable compared to the other methods for the tested problems in terms of the trade-off between the solution quality and the computing time.

4.3. The chance-constrained transportation problem

We next consider a probabilistic version of the conventional transportation problem. Suppose there are n suppliers and m customers. The transportation cost for shipping one unit product from supplier k to customer j is c_{kj}. Suppose that the kth supplier's capacity ξ_k is random, while the jth customer's demands d_j is deterministic. The problem is how to make decision on the shipment strategy such that the expected production transportation cost is minimized without the exact knowledge about the supplier capacity.

The chance-constrained transportation problem in our test has the following form:

$$\begin{aligned} \min\ & \sum_{k=1}^{n} \mathbb{E}[\xi_k] \sum_{j=1}^{m} c_{kj} x_{kj} \\ \text{s.t.}\ & \mathbb{P}\left\{ \sum_{k=1}^{n} \xi_k x_{kj} \geq d_j,\ j=1,\ldots,m \right\} \geq 1-\alpha, \\ & \sum_{j=1}^{m} x_{kj} \leq 1,\ x_{kj} \geq 0,\ k=1,\ldots,n,\ j=1,\ldots,m, \end{aligned} \tag{STP}$$

where x_{kj} is the percentage of production shipped from supplier k to customer j. When ξ have finite discrete distributions, problem (STP) is a special case of problem (P).

We randomly generate instances of (STP) with $(n,m) = (40,100)$ and $N = 500$, 750, 1000, 1250, 1500, 1750, 2000, respectively. The cost coefficients c_{kj} are from the uniform distribution $\mathcal{U}[1,5]$ for $k = 1,\ldots,n$ and $j = 1,\ldots,m$. To generate the samples or scenarios of ξ, we first generate μ_k from $\mathcal{U}[100,120]$ and σ_k from $\mathcal{U}[0,10]$ $(k = 1,\ldots,n)$. The samples ξ^i $(i = 1,\ldots,N)$ are then generated from $\mathcal{N}(\mu,\Sigma)$, where $\Sigma = \text{diag}(\sigma_1^2,\ldots,\sigma_n^2)$. Similarly, we generate d_j from $\mathcal{N}(\mu,\Sigma)$ with $\mu_j \sim \mathcal{U}[30,40]$ and $\sigma_j \sim \mathcal{U}[0,5]$ $(j = 1,\ldots,m)$. For each sample size N, we generate 5 random instances and the average computational results are recorded. The risk level is set as $\alpha = 0.05$ and 0.10, respectively.

The comparison results for problem (STP) are summarized in Table 4. The number in brackets in the column "fval" denotes the number of problems which have been solved by SQP-TR. The hyphen "–" in Table 4 indicates that the method can't find any feasible solution for all test problems in the time limit. Comparing the results in Table 4, we see that the average objective values achieved by ALDM, CPLEX and SQP-TR are much better than those achieved by the other methods for the cases with smaller scenarios. The proposed ALDM performs better than the other methods for test problems with a large

Table 4 Comparison results for (STP) with different number of samples

		ALDM			Greedy		BiCVaR		DC		SQP-TR		CPLEX	
α	N	fval	time	iter	fval	time	fval	time	fval	time	fval	time	fval	time
0.05	500	4.3097	15.41	635	4.3230	1801.55	4.3339	14.04	4.3189	23.93	4.3088	1802.12	4.3067	1801.55
0.05	750	4.3122	20.09	664	4.3360	1801.31	4.3380	22.57	4.3259	16.90	4.3124	1803.45	4.3106	1802.56
0.05	1000	4.3172	26.28	688	4.3537	1804.73	4.3445	42.00	4.3370	41.30	4.3217	1804.69	4.3168	1803.91
0.05	1250	4.3614	32.04	699	4.4001	1816.56	4.3860	57.45	4.3781	60.50	4.3713	1806.04	4.3603	1805.02
0.05	1500	4.3191	39.12	709	4.3621	1808.25	4.3408	81.51	4.3367	60.36	4.3512	1807.72	4.3355	1806.43
0.05	1750	4.3354	42.72	701	4.3845	1807.43	4.3548	104.55	4.3478	109.70	4.3911(4)	1808.95	4.3658	1811.94
0.05	2000	4.3523	51.68	737	4.4005	1823.18	4.3683	142.77	4.3699	160.75	4.4287(2)	1810.51	4.3915	1810.13
0.10	500	4.3049	15.94	654	4.3380	1801.10	4.3369	14.05	4.3279	19.78	4.3103	1802.09	4.3082	1801.60
0.10	750	4.3124	21.09	686	4.3577	1801.39	4.3418	24.38	4.3279	28.63	4.3177	1803.40	4.3151	1802.55
0.10	1000	4.3587	25.62	700	4.4217	1803.79	4.3848	39.94	4.3796	53.15	4.3794	1804.70	4.3598	1803.79
0.10	1250	4.3217	30.77	727	4.3844	1804.18	4.3431	56.92	4.3380	118.82	4.3719	1805.96	4.3250	1805.04
0.10	1500	4.3060	39.54	734	4.3772	1807.96	4.3267	83.89	4.3229	174.32	4.3469(4)	1807.57	4.3323	1806.54
0.10	1750	4.3097	43.70	748	4.3769	1808.98	4.3259	100.72	4.3300	227.40	4.3181(1)	1808.73	4.3658	1808.20
0.10	2000	4.3242	52.75	776	4.3937	1811.80	4.3332	128.66	4.3336	371.86	–	1812.99	4.3894	1810.02

Note: The magnitude of ‘fval’ is 10^3

number of scenarios. Compared with CPLEX, the ALDM can obtain feasible solutions in much less time. When the number of scenarios is less than 1250 and $\alpha = 0.05$, ALDM is slightly worse than CPLEX. When the number of scenarios grows larger, the ALDM can find better solution than CPLEX. For all instances, greedy method, CPLEX and SQP-TR are terminated when reaching the CPU time limit (1800 seconds). We also observe that the computing time of the ALDM method is slightly less than that of BiCVaR and DC method. Overall, it appears that the ALDM is efficient for finding good suboptimal solutions of problem (STP) and compare favorably with the greedy method, BiCVaR approximation, the DC method, CPLEX and SQP-TR in terms of the objective value and the computing time.

4.4. The chance-constrained supply chain problem

In this subsection, we consider the class of chance-constrained problems with non-convex objective function and random right-hand-side vector. This class of problems arises from the chance-constrained supply chain problem, which requires the decision maker to build a product supply chain before the customer demands are known exactly. The model in our test is same as the one used in Dentcheva and Martinez (2012, 2013), which has the following form:

$$\begin{aligned} &\min \sum_{k=1}^{n}\sum_{j=1}^{m} c_{kj}x_{kj} + a_{kj}x_{kj}^2 \\ &\text{s.t. } \mathbb{P}\left\{\sum_{k=1}^{n} x_{kj} \geq \xi_j,\ j=1,\ldots,m\right\} \geq 1-\alpha, \\ &\qquad \sum_{j=1}^{m} x_{kj} \leq M_k,\ x_{kj} \geq 0,\ k=1,\ldots,n,\ j=1,\ldots,m, \end{aligned} \tag{SCP}$$

where x_{kj} is the supply quantities delivered from the kth supplier to the jth customer, and $c_{kj} > 0$ is associated with transporting usually a unit of product from supplier k to customer j. In the objective function, the linear term $c_{kj}x_{kj}$ is the normal transportation cost, and the quadratic term $a_{kj}x_{kj}^2$ $(a_{kj} < 0)$ can be viewed as the cost discount. The kth supplier's total supply quantity is M_k, and the jth customer's demand ξ_j is random. When the random vector $\xi = (\xi_1, \ldots, \xi_m)^T$ has finite discrete distributions, problem (SCP) is a special case of problem (P) with a random right-hand-side vector.

Different from the previous examples, since the objective function is non-convex, we compare only the proposed method with the progressive augmented Lagrangian method proposed in Dentcheva and Martinez (2013), and the trust-region-based sequential quadratic programming proposed in Curtis et al. (2018). In our test, we set $(n, m) = (40, 100)$, $a_{kj} = -\frac{c_{kj}}{2M_k}$, and use the data (c_{kj}, ξ_j, M_k) proposed in Luedtke et al. (2010), which is downloaded from the author's home page `http://homepages.cae.wisc.edu/~luedtkej/`.

Table 5 Comparison results for (SCP) with different number of samples

		ALDM			PAL		SQP-TR	
α	N	fval	time	iter	fval	time	fval	time
0.05	500	3.4017	141.58	1571	3.5991	1335.72	3.3884	1193.63
0.05	1000	3.4781	323.98	3340	3.9048	1617.24	3.5160	1865.56
0.05	1500	3.6721	407.85	3783	3.8773	1631.15	3.8532	1905.80
0.05	2000	3.6891	560.98	4988	3.9153	1711.99	4.1479	1926.68
0.10	500	3.3609	141.13	1574	3.5660	1574.22	3.3445	959.87
0.10	1000	3.4172	347.56	3582	3.8630	1611.93	3.4469	1836.30
0.10	1500	3.6054	398.77	3702	3.8113	1567.82	3.7397	1879.26
0.10	2000	3.6167	667.27	5965	3.9524	1722.89	3.9680	1914.82

Note: The magnitude of 'fval' is 10^7

The comparison results for problem (SCP) are summarized in Table 5. From Table 5, we observe that the proposed ALDM performs better than the other two methods in terms of the quality of solution and the computing time. According to the columns "fval", the objective value of our ALDM is the smallest except the case $N = 500$. That implies that the quality of solution obtained by the ALDM is better than that by the other two methods. According to the columns "time", SQP-TR are often stopped due to the maximal computing time (1800 seconds). Note that the subproblems in PAL and SQP-TR are concave programming and mixed-integer programming, respectively, and thus are time consuming. That might be one of the reason why ALDM perform better than PAL and SQP-TR in our test. The most computing time for the ALDM is less than 700 seconds.

Particularly, when the number of scenarios grows from 500 to 2000, the computing time increases linearly. It is mainly because that the proposed method only requires to solve a convex subproblem and a 0-1 knapsack subproblem at each iteration. The dimension of the convex subproblem is independent of the number of scenarios and the dimension of the 0-1 knapsack subproblem is exactly the number of scenarios. Since we can solve the 0-1 knapsack subproblem very efficiently by the ranking method as shown in Section 2, the ALDM is advantageous in dealing with chance-constrained problems with larger number of scenarios.

5. Conclusions

We have presented an augmented Lagrangian decomposition method for solving the joint linear chance-constrained optimization problem under finite discrete distributions. The objective function of the studied problem can be non-convex and the left-hand side technology matrix and the right-hand side vector in the chance constraints can be both random. Different from other approximation methods in the literature, the proposed method is an iterative algorithm applied to the augmented Lagrangian decomposition reformulation of the mixed-integer program formulation of problem (P). The special structure of the probabilistic constraint motivated us to derive tractable subproblems by fixing certain variables and alternatively solving them at each iteration of the algorithm. We established the convergence of the method to the first-order stationary point of the original (possibly non-convex) problem. Preliminary computational results on several real-world chance-constrained models show that the proposed method is promising for finding approximate solutions of good quality. In particular, the proposed method compares favorably with some popular existing methods in terms of the quality of the best feasible solution obtained within a certain time for large size problems, especially when the objective function of the test problem is non-convex or the left-hand-side matrix of the constraints is random.

Acknowledgments

The research of Xiaodi Bai is supported by the National Natural Science Foundation of China Grants 11701511 and 71601169. The research of Xiaojin Zheng, who is the corresponding author of the paper, is supported by the National Natural Science Foundation of China Grant 11671300. The authors wish to thank the anonymous reviewers and the associate editor for the crucial comments and suggestions that helped us to improve the quality of the paper.

References

Adam L, Branda M (2016) Nonlinear chance constrained problems: Optimality conditions, regularization and solvers. *Journal of Optimization Theory and Applications* 170:419–436.

Ahmed S, Luedtke J, Song Y, Xie W (2017) Nonanticipative duality, relaxations, and formulations for chance-constrained stochastic programs. *Mathematical Programming* 162:51–81.

Beraldi P, Ruszczyński A (2002a) A branch and bound method for stochastic integer problems under probabilistic constraints. *Optimization Methods and Software* 17:359–382.

Beraldi P, Ruszczyński A (2002b) The probabilistic set-covering problem. *Operations Research* 50:956–967.

Bertsekas D (1982) *Constrained Optimization and Lagrange Multiplier Methods* (Boston: Academic Press).

Boyd S, Vandenberghe L (2004) *Convex Optimization* (Cambridge University Press).

Calafiore G, Campi M (2005) Uncertain convex programs: Randomized solutions and confidence levels. *Mathematical Programming* 102:25–46.

Calafiore G, Campi M (2006) The scenario approach to robust control design. *IEEE Transactions on Automatic Control* 51:742–753.

Charnes A, Cooper W (1959) Chance-constrained programming. *Management Science* 6:73–79.

Charnes A, Cooper W, Symonds G (1958) Cost horizons and certainty equivalents: an approach to stochastic programming of heating oil. *Management Science* 4:235–263.

Chen W, Sim M, Sun J, Teo C (2010) From CVaR to uncertainty set: Implications in joint chance-constrained optimization. *Operations Research* 58:470–485.

Cheng J, Lisser A (2012) A second-order cone programming approach for linear programs with joint probabilistic constraints. *Operations Research Letters* 40:325–328.

Cheon M, Ahmed S, Al-Khayyal F (2006) A branch-reduce-cut algorithm for the global optimization of probabilistically constrained linear programs. *Mathematical Programming* 108:617–634.

Clarke F (1983) *Optimization and Nonsmooth Analysis* (New York: Wiley).

Curtis F, Wächter A, Zavala V (2018) A Sequential Algorithm for Solving Nonlinear Optimization Problems with Chance Constraints. *SIAM Journal on Optimization* 28:930–958.

Dentcheva D, Martinez G (2012) Augmented lagrangian method for probabilistic optimization. *Annals of Operations Research* 200:109–130.

Dentcheva D, Martinez G (2013) Regularization methods for optimization problems with probabilistic constraints. *Mathematical Programming* 138:223–251.

Dentcheva D, Prékopa A, Ruszczynski A (2000) Concavity and efficient points of discrete distributions in probabilistic programming. *Mathematical Programming* 89:55–77.

Gill P, Murray W, Saunders M (2005) SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Review* 47:99–131.

Gill P, Murray W, Saunders M, Wong E (2018) User's guide for SNOPT 7.7: Software for large-scale nonlinear programming. Center for Computational Mathematics Report CCoM 18-1, Department of Mathematics, University of California, San Diego, La Jolla, CA.

Goldstein T, Osher S (2009) The split Bregman method for ℓ_1-regularized problems. *SIAM Journal on Imaging Sciences* 2:323–343.

He B, Tao M, Yuan X (2012) Alternating direction method with Gaussian back substitution for separable convex programming. *SIAM Journal on Optimization* 22:313–340.

Henrion R, Strugarek C (2008) Convexity of chance constraints with independent random variables. *Computational Optimization and Applications* 41:263–276.

Hong L, Yang Y, Zhang L (2011) Sequential convex approximations to joint chance constrained programs: A Monte Carlo approach. *Operations Research* 59:617–630.

Hu Z, Hong L, Zhang L (2013) A smooth monte carlo approach to joint chance-constrained programs. *IIE Transactions* 45:716–735.

Lagoa C, Li X, Sznaier M (2005) Probabilistically constrained linear programs and risk-adjusted controller design. *SIAM Journal on Optimization* 15:938–951.

Lejeune M (2012) Pattern-based modeling and solution of probabilistically constrained optimization problems. *Operations Research* 60:1356–1372.

Lejeune M, Margot F (2016) Solving chance-constrained optimization problems with stochastic quadratic inequalities. *Operations Research* 64:939–957.

Lejeune M, Noyan N (2010) Mathematical programming approaches for generating p-efficient points. *European Journal of Operational Research* 207:590–600.

Luedtke J (2014) A branch-and-cut decomposition algorithm for solving chance-constrained mathematical programs with finite support. *Mathematical Programming* 146:219–244.

Luedtke J, Ahmed S (2008) A sample approximation approach for optimization with probabilistic constraints. *SIAM Journal on Optimization* 19:674–699.

Luedtke J, Ahmed S, Nemhauser G (2010) An integer programming approach for linear programs with probabilistic constraints. *Mathematical Programming* 122:247–272.

Luo H, Sun X, Li D (2007) On the convergence of augmented Lagrangian methods for constrained global optimization. *SIAM Journal on Optimization* 18:1209–1230.

Miller L, Wagner H (1965) Chance-constrained programming with joint constraints. *Operations Research* 13:930–945.

Nemirovski A, Shapiro A (2006a) Convex approximations of chance constrained programs. *SIAM Journal on Optimization* 17:969–996.

Nemirovski A, Shapiro A (2006b) Scenario approximations of chance constraints. Calafiore G, Dabbene F, eds., *Probabilistic and Randomized Methods For Design Under Uncertainty*, 3–47 (London: Springer-Verlag).

Prékopa A (1970) On probabilistic constrained programming. *Proceedings of the Princeton Symposium on Mathematical Programming*, 113–138.

Prékopa A (1990) Sharp bounds on probabilities using linear programming. *Operations Research* 38:227–239.

Prékopa A (2003) Probabilistic programming. Ruszczyński A, Shapiro A, eds., *Handbooks in Operations Research and Management Science*, volume 10, 267–351 (Amsterdam: Elsevier).

Qiu F, Ahmed S, Dey S, Wolsey L (2014) Covering linear programming with violations. *INFORMS Journal on Computing* 26:531–546.

Reich D (2013) A linear programming approach for linear programs with probabilistic constraints. *European Journal of Operational Research* 230:487–494.

Rockafellar R, Sun J (2019) Solving monotone stochastic variational inequalities and complementarity problems by progressive hedging. *Mathematical Programming* 174:453–471.

Rockafellar R, Uryasev S (2000) Optimization of conditional value-at-risk. *Journal of Risk* 2:21–42.

Ruszczyński A (2002) Probabilistic programming with discrete distributions and precedence constrained knapsack polyhedra. *Mathematical Programming* 93:195–215.

Sen S (1992) Relaxations for probabilistically constrained programs with discrete random variables. *Operations Research Letters* 11:81–86.

Shan F, Zhang L, Xiao X (2014) A smoothing function approach to joint chance-constrained programs. *Journal of Optimization Theory and Applications* 163:181–199.

Shapiro A, Dentcheva D, Ruszczyński A (2009) *Lectures on Stochastic Programming: Modeling and Theory.* MPS-SIAM Series on Optimization (Philadelphia: SIAM).

Shen Y, Wen Z, Zhang Y (2014) Augmented Lagrangian alternating direction method for matrix separation based on low-rank factorization. *Optimization Methods and Software* 29:239–263.

Song Y, Luedtke J, Küçükyavuz S (2014) Chance-constrained binary packing problems. *INFORMS Journal on Computing* 26:735–747.

Steinbach M (2001) Markowitz revisited: Mean-variance models in financial portfolio analysis. *SIAM review* 43(1):31–85.

Sun J, Zhang S (2010) A modified alternating direction method for convex quadratically constrained quadratic semidefinite programs. *European Journal of Operational Research* 207:1210–1220.

Van Ackooij W, Henrion R, Möller A, Zorgati R (2011) On joint probabilistic constraints with Gaussian coefficient matrix. *Operations Research Letters* 39:99–102.

Vielma J, Ahmed S, Nemhauser G (2012) Mixed integer linear programming formulations for probabilistic constraints. *Operations Research Letters* 40:153–158.

Yin W, Osher S, Goldfarb D, Darbon J (2008) Bregman iterative algorithms for ℓ_1-minimization with applications to compressed sensing. *SIAM Journal on Imaging Sciences* 1:143–168.

Zhang S, Ang J, Sun J (2013) An alternating direction method for solving convex nonlinear semidefinite programming problems. *Optimization* 62:527–543.