Journal of Global Optimization **25:** 141–155, 2003.

Cutting Plane Algorithms for Nonlinear Semi-Definite Programming Problems with Applications

HIROSHI KONNO[1], NAOYA KAWADAI[2] and HOANG TUY[3]
[1]*Department of Industrial and Systems Engineering, Chuo-University, Japan;* [2]*Department of Industrial Engineering and Management, Tokyo Institute of Technology, Japan;* [3]*Institute of Mathematics, Hanoi, Vietnam*

Abstract. We will propose an outer-approximation (cutting plane) method for minimizing a function $f(\boldsymbol{X})$ subject to semi-definite constraints on the variables $\boldsymbol{X} \in \boldsymbol{R}^{n\times n}$. A number of efficient algorithms have been proposed when the objective function is linear. However, there are very few practical algorithms when the objective function is nonlinear. An algorithm to be proposed here is a kind of outer-approximation(cutting plane) method, which has been successfully applied to several low rank global optimization problems including generalized convex multiplicative programming problems and generalized linear fractional programming problems, etc. We will show that this algorithm works well when f is convex and n is relatively small. Also, we will provide the proof of its convergence under various technical assumptions.

Key words: Semi-definite programming, Low rank nonconvex problem, Cutting plane algorithm, Outer-approximation, Semi-infinite programming, Ellipsoidal separation, Quadratic regression, Semi-definite logit model

1. Introduction

Semi-definite programming problems (SDP), namely minimization of a linear objective function subject to semi-definite constraints have been under intensive study in recent years. SDP's have surprisingly diverse practical applications in control engineering, conbinatorial optimization, nonconvex quadratic programming, structural design and moment problems to name only a few. For a recent survey of these applications, readers are referred to Vandenberghe-Boyd [19] and Wolkowicz-Saigal-Vandenberghe [21].

Semi-definite programming problems can be viewed as a natural extension of linear programming problems. In fact, we can define a dual problem for a given SDP and the duality relation holds between these pairs of problems. Based upon this observation, primal-dual interior point algorithms and path following algorithms developed for linear programming problems have been extended to SDP's, some of which are even polynomial order algorithms [7]. Further, a number of efficient softwares have been developed and used for solving the problems referred to above.

Recently, Konno et al. [11] proposed an alternative outer approximation (cutting plane) algorithm and demonstrated that it can solve a special class of SDP's very fast, much faster than SDPA 5.0 [5], a widely used software. This algorithm is based upon the observation that an SDP is a linear programming problem with infinitely many linear constraints as observed by Vandenberghe and Boyd [18] and Konno and Kobayashi [13]. This class of problems is called semi-infinite programming problems. To solve an SDP, we first solve a relaxed linear programming problem with finite number of constraints. If the solution satisfies the semi-definite constraint, then we are done. If not, we generate linear constraints which are violated at the optimal solution of the relaxed problem and generate a tighter relaxation of the original problem by adding these constraints. This algorithm can be viewed as an extension of classical cutting plane algorithms proposed by Kelley [10] and Veinott [20] for convex minimization problems. Also, it is similar to outer approximation algorithms successfully applied to low rank concave and d.c. minimization problems [8, 14].

We applied this algorithm to failure discriminant analysis (Konno et al[12, 13]), where two classes of multidimensional financial data are separated by a convex quadratic surface, i.e., by an ellipsoid or paraboloid. This class of problems satisfies the low rank property in the sense that the dimension of semi-definite constraint is small. We showed in Konno et al. [11] that SDP whose rank of semi-definiteness is less than 10 can be solved very fast by the cutting plane algorithm. The success depends upon the fact that we can generate the most violated constraint very cheaply and that an optimal solution of tightly relaxed linear program can be recovered by applying a few dual simplex pivots to the optimal dictionary.

The purpose of this paper is to show that this algorithm can be extended to general nonlinear semi-definite programming problems, i.e., minimization of a not necessarily convex function $f(\boldsymbol{X})$ subject to semi-definite constraints on $\boldsymbol{X} \in R^{n\times n}$. Convergence of the algorithm is guaranteed if the feasible region is bounded. Also, it can be established under coerciveness condition and under a weaker condition when f is concave.

In the next section, we will reformulate a nonlinear semi-definite programming problem as a linearly constrained semi-infinite programming problem. Then we present cutting plane algorithms using two different types of cuts. One is the extension of Kelley's cut which is applicable to any class of SDPs. The second is the extension of Veinott's supporting hyperplane method, which is applicable to the case where the feasible region has an interior. Convergence property of these algorithms will be discussed in detail. Section 3 will be devoted to three successful applications. First is the failure discriminant analysis, where multi-dimensional data are separated into two classes by a convex quadratic surface. The second example is semi-definite regression and the third example is the estimation of failure probability by using a semi-definite logit model.

2. Cutting Plane Algorithm.

Let us consider the following minimization problem:

$$\left|\begin{array}{ll} \text{minimize} & f(X) \\ \text{subject to} & A^i \bullet X = b_i, \quad i = 1, \cdots, m, \\ & X \succeq \mathbf{0}, \end{array}\right. \tag{1}$$

where $A^i = (a^i_{jk}) \in R^{m\times n}$, $b_i \in R^1$ are constants, $f : R^{n\times n} \to R^1$ is a (not necessarily convex) function and $X = (x_{jk}) \in R^{n\times n}$ are variables. Here the matrix product $A^i \bullet X$ denotes the dot product of matrices, i.e.,

$$\sum_{j=1}^{n}\sum_{k=1}^{n} a^i_{jk} x_{jk}.$$

Also, $X \succeq \mathbf{0}$ means that X is a symmetric positive semi-definite matrix. We will assume that the problem (1) contains no redundant constraints.

First, let us note that the equality constraints $A^i \bullet X = b_i, \quad i = 1, \cdots, m$ can be eliminated from (1) by using the standard procedure in linear programming. Let $\boldsymbol{x}_B \in R^m$, $\boldsymbol{x}_N \in R^{n\times n-m}$ (with $\boldsymbol{x}_B = (x_{jk}, (j,k) \in B)$, $\boldsymbol{x}_N = (x_{jk}, (j,k) \in N)$ and $B, N \subset \{1, \cdots, n\} \times \{1, \cdots, n\}$, $|B| = m$, $|N| = n \times n - m$), be the sets of basic and nonbasic variables of x_{jk}'s, respectively, so that

$$\boldsymbol{x}_B = (A_B)^{-1}\boldsymbol{b} - (A_B)^{-1}(A_N)\boldsymbol{x}_N,$$

where A_B, A_N are submatrices of the matrix A of the system of linear constraints in (1). Let $Q(\boldsymbol{x}_N) \in R^{n\times n}$ be the matrix X in which the components $\boldsymbol{x}_{jk}$ with $(j,k) \in B$ are replaced by the corresponding components of $\boldsymbol{x}_B$ defined by (2). Then $Q(\boldsymbol{x}_N)$ is a matrix whose components are linear functions of $\boldsymbol{x}_N$, i.e. a matrix of the form $Q(\boldsymbol{x}_N) = Q_0 + \sum_{(j,k)\in N} x_{jk} Q_{jk}$ with $Q_0, Q_{jk} \in R^{n\times n}$, so the problem (1) can be rewritten as

$$\text{minimize} \quad F(\boldsymbol{x}_N) \qquad \text{subject to} \quad Q(\boldsymbol{x}_N) \succeq \mathbf{0} \tag{2}$$

where $F(\boldsymbol{x}_N) = f(Q_0 + \sum_{(j,k)\in N} x_{jk} Q_{jk})$ is a function of $\boldsymbol{x}_N$. Let us note that F is convex(concave) when f is convex(concave).
The condition $Q(\boldsymbol{x}_N) \succeq \mathbf{0}$ which is a *linear matrix inequality* is equivalent to

$$\boldsymbol{d}^T Q(\boldsymbol{x}_N)\boldsymbol{d} \geqslant 0, \quad {}^\forall\boldsymbol{d} \in B_n \tag{3}$$

where B_n is the n-dimensional unit ball. Therefore, setting $\boldsymbol{y} = \boldsymbol{x}_N \in R^{n\times n-m}$, the problem can be written as

$$(P) \quad \left|\begin{array}{ll} \text{minimize} & F(\boldsymbol{y}) \\ \text{subject to} & \boldsymbol{d}^T Q(\boldsymbol{y})\boldsymbol{d} \geqslant 0, \quad {}^\forall\boldsymbol{d} \in B_n. \end{array}\right. \tag{4}$$

Let us assume that the function $F(\boldsymbol{y})$ is *coercive*, in the sense that

$$|F(\boldsymbol{y})| \to +\infty \quad \text{whenever} \quad \|\boldsymbol{y}\| \to +\infty$$

and consider a polyhedron $\boldsymbol{Y}_0$ in $R^{n\times n-m}$ containing the constraint set of (P), for instance

$$\boldsymbol{Y}_0 = \{\boldsymbol{y} \in R^{n\times n-m} \mid \boldsymbol{d}_j^T Q(\boldsymbol{y})\boldsymbol{d}_j \geqslant 0,\ j = 1, \cdots, J\}.$$

Then the relaxed problem

$$(P_0) \quad \left| \begin{array}{l} \text{minimize} \quad F(\boldsymbol{y}) \\ \text{subject to} \quad \boldsymbol{y} \in \boldsymbol{Y}_0 \end{array} \right. \tag{5}$$

has a finite optimal solution $\boldsymbol{y}^0$, because otherwise there would exist a sequence $\boldsymbol{y}^k$ such that $F(\boldsymbol{y}^k) \to -\infty$ while $\|\boldsymbol{y}^k\| \to +\infty$. Obviously, if $Q(\boldsymbol{y}^0) \succeq \boldsymbol{0}$ then $\boldsymbol{y}^0$ is an optimal solution of the original problem (P).

If $Q(\boldsymbol{y}^0) \not\succeq \boldsymbol{0}$, i.e., $\min\{\boldsymbol{g}^T Q(\boldsymbol{y}^0)\boldsymbol{g} \mid \boldsymbol{g} \in B_n\} < 0$, we will calculate a vector $\boldsymbol{g}$ such that $\boldsymbol{g}^T Q(\boldsymbol{y}^0)\boldsymbol{g} < 0$ and add a new constraint

$$\boldsymbol{g}^T Q(\boldsymbol{y}^0)\boldsymbol{g} \geqslant 0 \tag{6}$$

to the problem (5) above to obtain a tighter relaxation of (P).

We are now ready to describe a basic procedure of the cutting plane algorithm.

Cutting Plane Algorithm (Prototype)

1. $k = 0$.
2. Solve a linearly constrained minimization problem

$$min\ \{F(\boldsymbol{y}) \mid \boldsymbol{y} \in \boldsymbol{Y}_k\}. \tag{7}$$

   and let $\boldsymbol{y}^k$ be its optimal solution.
3. If $Q(\boldsymbol{y}^k) \succeq \boldsymbol{0}$, then $\boldsymbol{y}^k$ is an optimal solution of (P). Otherwise choose a vector $\boldsymbol{g}_k$ such that $\boldsymbol{g}_k^T Q(\boldsymbol{y}^k)\boldsymbol{g}_k < 0$ and

$$\boldsymbol{Y}_{k+1} = \boldsymbol{Y}_k \cap \{\boldsymbol{y} \mid \boldsymbol{g}_k^T Q(\boldsymbol{y})\boldsymbol{g}_k \geqslant 0\}$$

   $k = k + 1$ and go to 2. □

REMARK 1. We implicitly assumed here that the subproblem (7) can be solved in an efficient way. If, for example f is linear or convex, (7) can be solved by standard methods. Also, there exists a number of efficient algorithms when f is concave, or it has so called low rank nonconvex structure, e.g., convex multiplicative functions or generalized linear fractional functions, etc [14].

Convergence property and efficiency of the cutting plane algorithm depends upon the choice of the cut in Step 3. One possible choice is the most violated constraint at $\boldsymbol{y}^k$ which can be obtained by solving the following problem:

$$\min\{\boldsymbol{g}^T Q(\boldsymbol{y}^k)\boldsymbol{g} \mid \boldsymbol{g} \in B_n\} \tag{8}$$

An optimal solution $\boldsymbol{g}_k$ of this problem is given by the eigenvector of $Q(\boldsymbol{y}^k)$ corresponding to the minimal(negative) eigenvalue of $Q(\boldsymbol{y}^k)$ [6]. Algorithm using this cut may be considered as an adaptation of the Kelley's algorithm [10].

ALGORITHM 1.

1. $k = 0$, $\varepsilon > 0$.
2. Solve the subproblem

$$\min \{F(\boldsymbol{y}) \mid \boldsymbol{y} \in \boldsymbol{Y}_k\}.$$

and let y^k be its optimal solution.

3. Let α_k be the smallest eigenvalue of $Q(\boldsymbol{y}^k)$. If $\alpha_k > -\varepsilon$, then terminate. Otherwise let $\boldsymbol{g}_k$ be the eigenvector of $\boldsymbol{Y}^k$ corresponding to α_k and

$$\boldsymbol{Y}_{k+1} = \boldsymbol{Y}_k \cap \{\boldsymbol{y} \mid \boldsymbol{g}_k^T Q(\boldsymbol{y}) \boldsymbol{g}_k \geqslant 0\}$$

$k = k + 1$ and go to 2. □

THEOREM 1. *If the feasible region of* (4) *is bounded, then any accumulation point* $\boldsymbol{y}^*$ *of the sequence* $\{\boldsymbol{y}^k\}$ *is an* ε*-optimal solution of* (4).

Proof. Let $(\boldsymbol{y}^{k_j}, \boldsymbol{g}_{k_j})$ be a sequence converging to $(\boldsymbol{y}^*, \boldsymbol{g}^*)$. If $\boldsymbol{y}^*$ is not an ε-optimal solution, then $\boldsymbol{g}^{*T} \boldsymbol{Q}(\boldsymbol{y}^*)\boldsymbol{g}^* < -\varepsilon$. This is a contradiction, since $\boldsymbol{g}_{k_j}^T \boldsymbol{Q}(\boldsymbol{y}^{k_{j+1}})\boldsymbol{g}_{k_j} \geqslant 0$ for all j, so that $\boldsymbol{g}^{*T} \boldsymbol{Q}(\boldsymbol{y}^*)\boldsymbol{g}^* \geqslant 0$. □

If the feasible region of (4) is unbounded, then we need to impose some technical condition to prove convergence of the algorithm. Coersiveness of F is one such condition.

THEOREM 2. *Under the coerciveness assumption, either Algorithm* 1 *terminates, yielding an* ε*-optimal solution, or it generates a bounded sequence of solutions* $\{\boldsymbol{y}^k\}$*, any accumulation point of which is an* ε*-optimal solution of* (P).

Proof. If the algorithm generates an unbounded sequence $\{\boldsymbol{y}^k\}$ then by coerciveness of $F(\cdot)$ we must have $\mid F(\boldsymbol{y}^k) \mid \to +\infty$, conflicting with the fact that $F(\boldsymbol{y}^0) \leqslant F(\boldsymbol{y}^1) \leqslant \cdots \leqslant F(\boldsymbol{y}^k) \leqslant F(\boldsymbol{a})$ for any $\boldsymbol{a} \in \boldsymbol{Y}$. Therefore, the sequence $\{\boldsymbol{y}^k\}$, if infinite, is bounded. The convergence of the algorithm can then be established by a standard argument (see, e.g., [10]). □

Although this algorithm works well when $F(\cdot)$ is linear or convex and n is small, it may not be fast enough.

To accelerate convergence, let us consider the supporting hyperplane method of Veinott [20].

Let $\boldsymbol{y}^0$ be a point in the interior of the constraint set $\boldsymbol{D} := \{\boldsymbol{y} \mid Q(\boldsymbol{y}) \succeq \boldsymbol{0}\}$, i.e., such that $Q(\boldsymbol{y}^0) \succ \boldsymbol{0}$, and define

$$\left|\begin{array}{l} \lambda_k = \operatorname{argmax}\{\lambda \mid Q(\boldsymbol{y}^0 + \lambda \boldsymbol{y}^k) \succeq \boldsymbol{0}\} \\ \quad \tilde{\boldsymbol{y}}^k = \boldsymbol{y}^0 + \lambda_k \boldsymbol{y}^k \end{array}\right. \tag{9}$$

Then $\tilde{\boldsymbol{y}}^k$ lies on the boundary of $\boldsymbol{D}$.

THEOREM 3. *Let*

$$\tilde{\boldsymbol{g}}_k = \text{argmin}\{\boldsymbol{g}^T Q(\tilde{\boldsymbol{y}}^k)\boldsymbol{g} \mid \boldsymbol{g} \in B_n \setminus \{\boldsymbol{0}\}\},$$

then the equation

$$\tilde{\boldsymbol{g}}_k{}^T Q(\boldsymbol{y} - \tilde{\boldsymbol{y}}^k)\tilde{\boldsymbol{g}}_k = 0$$

defines a supporting hyperplane of $\boldsymbol{D}$ *at* $\tilde{\boldsymbol{y}}^k$.

Proof. From (8) we must have $Q(\boldsymbol{y}^k) \succeq \boldsymbol{0}$ but $Q(\tilde{\boldsymbol{y}}^k) \not\succeq \boldsymbol{0}$, consequently, $\min\{\boldsymbol{g}^T Q(\tilde{\boldsymbol{y}}^k)\boldsymbol{g} | \boldsymbol{g} \in B_n\} = 0$. Thus $\tilde{\boldsymbol{g}}_k^T Q(\boldsymbol{y})\tilde{\boldsymbol{g}}_k \geqslant 0 = \tilde{\boldsymbol{g}}_k^T Q(\tilde{\boldsymbol{y}}^k)\tilde{\boldsymbol{g}}_k$ and hence, $\tilde{\boldsymbol{g}}_k^T Q(\boldsymbol{y} - \tilde{\boldsymbol{y}}^k)\tilde{\boldsymbol{g}}_k \geqslant 0$ for every $\boldsymbol{y} \in \boldsymbol{D}$. □

ALGORITHM 2.

1. Same as Algorithm 1.
2. Same as Algorithm 1.
3. Let α_k be the smallest eigenvector of $Q(\boldsymbol{y}^k)$. If $\alpha_k > -\varepsilon$, then terminate. Otherwise, calculate $\tilde{\boldsymbol{g}}^k$ by (8) and $\tilde{\boldsymbol{y}}_k$ by (9).

$$\boldsymbol{Y}_{k+1} = \boldsymbol{Y}_k \cap \{\boldsymbol{y} \mid \tilde{\boldsymbol{g}}_k{}^T Q(\boldsymbol{y})\tilde{\boldsymbol{g}}_k \geqslant 0\}$$

$k = k + 1$ and goto 1.

THEOREM 4. *Under the coerciveness assumption, either Algorithm* 2 *terminates, yielding an* ε*-optimal solution, or it generates a bounded sequence of solutions* $\{\boldsymbol{y}^k\}$*, any accumulation point of which is an* ε*-optimal solution of* (P).

Proof. As previously, if the sequence $\{\boldsymbol{y}^k\}$ is infinite it must be bounded. The convergence follows, then, by the same argument as used in Veinott [20]. □

REMARK 2. *If we define* $\varphi(\boldsymbol{y}) = -min\{\boldsymbol{g}^T Q(\boldsymbol{y})\boldsymbol{g} | \boldsymbol{g} \in B_n\}$ *then* $\varphi(\cdot)$ *is a convex function (pointwise maximum of a family of affine functions), and the constraint* $Q(\boldsymbol{Y}) \succeq \boldsymbol{0}$ *is equivalent to* $\varphi(\boldsymbol{y}) \leqslant 0$. *Since* B_n *is a compact set, it follows from a well known result of convex analysis ([9], Chapter* 4, *Theorem* 3*) that* $\boldsymbol{g}_k$ *in Step* 3 *of Algorithm* 1 *is a subdifferential of the convex function* $\varphi(\boldsymbol{y})$ *at* $\boldsymbol{y}^k$*, while* $\tilde{\boldsymbol{g}}_k$ *in Step* 3 *of Algorithm* 2 *is a subdifferential of the same function at* $\tilde{\boldsymbol{y}}^k$.

THEOREM 5. *If* $f(\boldsymbol{X})$ *is concave in the above problem, algorithm converges under a weaker assumption than coerciveness. Namely, it suffices to assume that for some* $\alpha < F(\boldsymbol{y}^0)$ *the set* $\{\boldsymbol{y} | F(\boldsymbol{y}) = \alpha\}$ *is bounded.*

Proof. See Tuy ([17], or [8], Theorem VI.2). □

3. Applications

In this section, we will present three practical problems formulated as (P) and report the results of computational experiments. All experiments were conducted on Pentium III Processor (500 MHz) using C/C++, OS:Vine Linux and used Householder's algorithm to calculate the minimal eigenvalue of a symmetric matrix. Also, linear subproblem was solved by CPLEX6.5 and convex subproblem was solved by NUOPT 5.0.

3.1. SEPARATION OF MULTI-DIMENSIONAL DATA BY AN ELLIPSOIDAL SURFACE

Let=$\{\boldsymbol{a}_1, \cdots, \boldsymbol{a}_m\} \subset R^n, \{\boldsymbol{b}_1, \cdots, \boldsymbol{b}_l\} \subset R^n$ be two sets of data in R^n and let us consider the problem of finding a convex quadratic surface (ellipsoid or paraboloid) separating these two sets.

One such example is reported in [12, 13], where $\boldsymbol{a}_i$'s are financial data associated with ith ongoing companies and $\boldsymbol{b}_j$'s are financial data associated with failed companies. There is a good reason to assume(See [13] for details) that $\boldsymbol{a}_i$'s are distributed within certain ellipsoid (or paraboloid) and that $\boldsymbol{b}_j$'s are distributed outside of such an ellipsoid.

If there exists $\boldsymbol{D} \in R^{n\times n}, \boldsymbol{c} \in R^n, c_0 \in R^1$ such that $\boldsymbol{D} \succeq \boldsymbol{0}$ satisfying

$$\begin{aligned} &\boldsymbol{a}_i^T \boldsymbol{D}\boldsymbol{a}_i + \boldsymbol{c}^T \boldsymbol{a}_i < c_0, \quad i = 1, \cdots, m, \\ &\boldsymbol{b}_j^T \boldsymbol{D}\boldsymbol{b}_j + \boldsymbol{c}^T \boldsymbol{b}_j > c_0, \quad j = 1, \cdots, l, \end{aligned}$$

then

$$E = \{\boldsymbol{x} \in R^n \mid \boldsymbol{x}^T \boldsymbol{D}\boldsymbol{x} + \boldsymbol{c}^T \boldsymbol{x} = c_0\}$$

is a separating ellipsoid (or paraboloid).

In general, such an ellipsoid may not exist. In such a case, we consider the following semi-definite programming problem (10) in view of the remarkable success reported in [4, 16]:

$$\left|\begin{array}{ll} \text{minimize} & \lambda \dfrac{1}{m}\displaystyle\sum_{i=1}^{m} y_i + (1-\lambda)\dfrac{1}{l}\sum_{j=1}^{l} z_j \\ \text{subject to} & \boldsymbol{a}_i^T \boldsymbol{D}\boldsymbol{a}_i + \boldsymbol{c}^T \boldsymbol{a}_i - y_i \leqslant c_0 - 1, \ i = 1, \cdots, m, \\ & \boldsymbol{b}_j^T \boldsymbol{D}\boldsymbol{b}_j + \boldsymbol{c}^T \boldsymbol{b}_j + z_j \geqslant c_0 + 1, \ j = 1, \cdots, l, \\ & \boldsymbol{D} \succeq \boldsymbol{0}, \ \ y_i \geqslant 0, \ i = 1, \cdots, m, \ \ z_j \geqslant 0, \ j = 1, \cdots, l, \end{array}\right. \tag{10}$$

where y_i represents the distance of $\boldsymbol{a}_i$ from E_- into the failure region and z_j represents the distance of $\boldsymbol{b}_j$ from E_+ into the ongoing region (see Figure 1). Also $\lambda \in (0, 1)$ is the weight representing the relative importance of the two types of classification error.

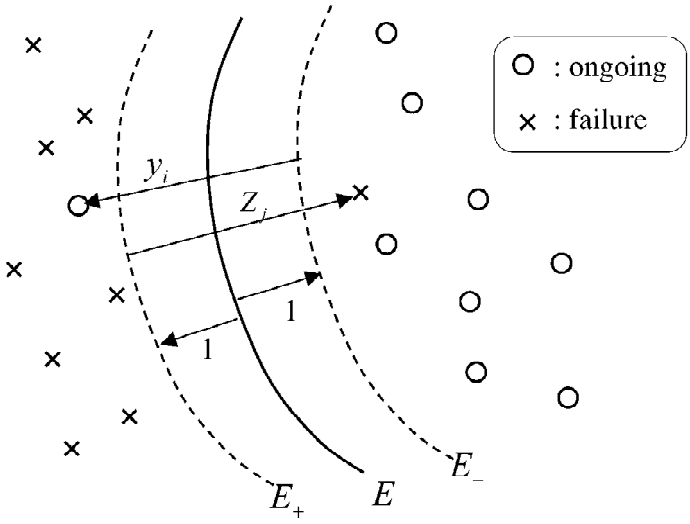

Figure 1. Ellipsoidal surface.

Table 1. CPU time (sec.)

No. of attributes		Data Set 1			Data Set 2
1st iteration		$n=3$	$n=6$	$n=9$	$n=7$
		0.05	1.20	1.41	1.58
ε	10^{-3}	0.11	2.22	16.68	19.92
	10^{-4}	0.12	2.71	23.04	25.04
	10^{-5}	0.14	3.17	29.57	31.11
	10^{-6}	0.16	3.66	36.87	36.48

The term ± 1 in the right hand side of the inequality constraints of (10) are added for normalization purpose.

We solved the problem (10) by using two different data sets. The first set consists of 428 data out of which 40 belong to failure group. The second set consists of 1701 companies out of which 119 belong to failure group. We used Kelley's cutting plane algorithm(Algorithm 1) since it is not easy to find a positive definite matrix satisfying the constraints. Table 1 shows their computation time for solving these problems.

This method was used for predicting the failure of companies during the next year. We compared the rate of wrong prediction of this model with linear and general quadratic model and found that the semi-definite model leads to significant improvement over other methods.

Let us note that we can solve problems up to $n = 9$ without too much difficulty when the number of data is 428 (Data Set 1). The efficiency depends upon the very efficient dual simplex procedure of CPLEX6.5. When, however the number of data is 1701 (Data Set 2), considerably more computation time is required. In fact, the problem with (n, m)=(7, 1701) requires almost the same amount of time for solving the problem with (n, m)=(9, 408). This is because solving linear subproblems tends to become more difficult as n increases. The maximal conceivable size of the real world failure discrimination problem is $n = 9$ and $m = 3000 \sim 5000$, so that it is within reach of the cutting plane algorithm.

Let us note here that a cutting plane algorithm turned out to be over 100 times faster than $SDPA$ 5.0, a state-of-the art interior point software. For more detailed discussions about failure discriminant analyses and the advantage of semi-definite separation over other methods, readers are referred to [12].

Let us note that the objective function is not coercive, but an alternative proof of convergence of this method without assuming coerciveness is given in [11].

3.2. SEMI-DEFINITE REGRESSION

Let us consider the following linear regression model:

$$Y = d_0 + d_1 X_1 + \cdots + d_m X_m + \varepsilon, \tag{11}$$

where $X_j \in R^1,\ j = 1, \cdots, m$ and Y are sets of $m+1$ variables and ε is a random variable. Also $d_j,\ j = 0, \cdots, n$ are constants to be estimated.

Given T data sets $\{(Y_t,\ X_{1t}, \cdots, X_{mt}),\ t = 1, \cdots, T\}$, the least square estimate of constants $d_j,\ i = 0, 1, \cdots, m$ is given by minimizing the sum of the squares of the residual:

$$S = \sum_{t=1}^{T} \left\{ Y_t - \left(d_0 + \sum_{j=1}^{m} d_j X_{jt} \right) \right\}^2. \tag{12}$$

When the fitting is not good enough, we sometimes use quadratic model:

$$Y = d_0 + \sum_{j=1}^{m} d_j X_j + \sum_{j=1}^{m} \sum_{k=1}^{m} d_{jk} X_j X_k + \varepsilon, \tag{13}$$

where $d_{jk} = d_{kj}, {}^{\forall} j, k$.

Least square estimates of $d_j,\ j = 0, \cdots, m$, and $d_{jk},\ j, k = 1, \cdots, m$ can be obtained by minimizing

$$\sum_{t=1}^{T} \left\{ Y_t - \left(d_0 + \sum_{j=1}^{m} d_j X_{jt} + \sum_{j=1}^{m} \sum_{k=1}^{m} d_{jk} X_{jt} X_{kt} \right) \right\}^2 \tag{14}$$

Table 2. (Algorithm 1): CPU time (s) (Iteration)

ε_n	5	6	7	8
10^{-3}	0.55(19)	1.12(24)	1.92(27)	4.79(40)
10^{-4}	0.7(23)	1.82(35)	3.65(43)	8.74(60)
10^{-5}	0.98(30)	2.87(47)	6.4(61)	15.65(83)
10^{-6}	1.33(37)	4.07(58)	10.83(81)	28.93(113)

However, this model often leads to overfitting because we have introduced $m(m+1)/2$ new parameters d_{jk}. To obtain a good fitting while escaping the risk of overfitting, we will impose a condition that the surface

$$S = \left\{ (X_1, \cdots, X_m) \mid d_0 + \sum_{j=1}^{m} d_j X_j + \sum_{j=1}^{m}\sum_{k=1}^{m} d_{jk} X_j X_k = const \right\} \tag{15}$$

is convex, i.e., an ellipsoid or paraboloid. The degree of freedom of the ellipsoidal regression model is significantly smaller than general quadratic regression model.

The resulting problem is

$$\left| \begin{array}{ll} \text{minimize} & \displaystyle\sum_{t=1}^{T} \left\{ Y_t - \left(d_0 + \sum_{j=1}^{m} d_j X_{jt} + \sum_{j=1}^{m}\sum_{k=1}^{m} d_{jk} X_{jt} X_{kt} \right) \right\}^2 \\ \text{subject to} & \boldsymbol{D} = (d_{jk}) \succeq \mathbf{0}. \end{array} \right. \tag{16}$$

We can choose any positive definite matrix $\boldsymbol{D}_0$ in the supporting hyperplane algorithm.

We used up to 1142 daily data of the TOPIX index, as $Y_t,\ t = 1, \cdots, 1142$ and solved problem (16) using up to eight explanatory variables X_j. Table 2 shows the computational result for Algorithm 1. Also Table 3 shows the result of Algorithm 2 when we choose $D_0 = I$.

Let us note that the objective function in this case is coercive.

We see from these tables that Algorithm 2 performs better than Algorithm 1, as expected. However, both take more computation time than for the linear case. This is due to the fact that we had to solve QP subproblems from scratch every time. Computation time would have been much less if we implement more efficient dual type algorithm for solving QP subproblem with one additional linear constraint.

Residual error associated with semi-definite regression model was about 30% less than that of linear regression model in this case.

Table 3. (Algorithm 2): CPU time (s) (Iteration)

ε_n	5	6	7	8
10^{-3}	0.35(19)	0.73(24)	1.25(27)	3.27(41)
10^{-4}	0.45(23)	1.09(33)	2.57(44)	6.58(63)
10^{-5}	0.66(30)	1.91(47)	5.32(66)	12.14(86)
10^{-6}	0.99(39)	2.71(57)	8.21(81)	19.61(107)

3.3. SEMI-DEFINITE LOGIT MODEL FOR ESTIMATING THE FAILURE PROBABILITY

Financial institutions are often required to estimate the failure probability of enterprises based upon the information about their financial data.

Let $x_i \in R^n$ be the financial data associated with enterprises $A_i, i = 1, \cdots, m$ out of which $A_1, \cdots, A_k$ are ongoing and $A_{k+1}, \cdots, A_m$ are failed enterprises. Linear logit model is one very popular method for estimating the failure probability of ongoing enterprises.

Let

$$f(\boldsymbol{x} : \boldsymbol{\beta}, \beta_0) = \frac{\exp\{\boldsymbol{\beta}^T \boldsymbol{x} + \beta_0\}}{1 + \exp\{\boldsymbol{\beta}^T \boldsymbol{x} + \beta_0\}} \tag{17}$$

be the failure probability corresponding to $\boldsymbol{x} \in R^n$, where $\boldsymbol{\beta} \in R^n, \beta_0 \in R^n$ are parameters to be estimated.

Let $L(\boldsymbol{\beta}, \beta_0)$ be the likelihood function associated with $\boldsymbol{x}_i, i = 1, \cdots, m$. Then

$$L(\boldsymbol{\beta}, \beta_0) = \prod_{i=1}^{m} f(x_i; \boldsymbol{\beta}, \beta_0) \tag{18}$$

$$= \prod_{i=1}^{k} \frac{\exp\{\boldsymbol{\beta}^T x_i + \beta_0\}}{1 + \exp\{\boldsymbol{\beta}^T x_i + \beta_0\}} \prod_{i=k+1}^{m} \frac{1}{1 + \exp(\boldsymbol{\beta}^T x_i + \beta_0)}. \tag{19}$$

Standard method can be applied to maximize $L(\boldsymbol{\beta}, \beta_0)$ since $lnL(\boldsymbol{\beta}, \beta_0)$ is a concave function.

This model is based on the assumption that the failure probability is a monotone function of each financial data. In fact,

$$\frac{\partial}{\partial x_j} lnf(\boldsymbol{x}; \boldsymbol{\beta}, \beta_0) = \frac{\beta_j}{1 + \exp\{\boldsymbol{\beta}^T \boldsymbol{x} + \beta_0\}} \tag{20}$$

so that $f(\boldsymbol{x}; \boldsymbol{\beta}, \beta_0)$is an increasing (decreasing) function depending upon the sign of β_j.

However, failure probability need not be monotone for all variables. Associated with some variable is certain 'desirable' range in which enterprises are considered to perform well. Linear logit model is not valid in this case.

In [15], Konno and Wu introduced a semi-definite logit model to handle non-monotonic case. Let

$$\tilde{f}(\boldsymbol{x}; \boldsymbol{D}, \boldsymbol{\beta}, \beta_0) = \frac{\exp\{\boldsymbol{x}^T \boldsymbol{D}\boldsymbol{x} + \boldsymbol{\beta}^T \boldsymbol{x} + \beta_0\}}{1 + \exp\{\boldsymbol{x}^T \boldsymbol{D}\boldsymbol{x} + \beta^T \boldsymbol{x} + \beta_0\}} \tag{21}$$

where $\boldsymbol{D} = (d_{ij}) \succeq \boldsymbol{0}$. Let us note that f need not be monotonic.

The maximum likelihood estimation problem reduces to the following convex minimization problem under semi-definite constraint:

$$\left|\begin{array}{ll} \text{minimize} & \displaystyle\sum_{i=1}^{m} ln\{1 + \exp(\boldsymbol{x}_i^T \boldsymbol{D}\boldsymbol{x}_i + \boldsymbol{\beta}^T \boldsymbol{x}_i + \beta_0)\} \\ & \displaystyle -\sum_{i=1}^{k} \{\boldsymbol{x}_i^T \boldsymbol{D}\boldsymbol{x}_i + \boldsymbol{\beta}^T \boldsymbol{x}_i + \beta_0\} \\ \text{subject to} & \boldsymbol{D} \succeq \boldsymbol{0} \end{array}\right. . \tag{22}$$

We solved problem (22) using 865 data out of which 74 belong to the failure group. We used NUOPT5.0 for solving linearly constrained convex minimization problems. Figure 2 shows the computation time for solving (22) as a function of n. Table 4 shows the amount of computational time as a function of ε.

Let us note that the algorithm converged without fail for all test problems though the objective function is not coercive.

From the practical point of view, $\varepsilon = 10^{-4}$ is good enough since we observe no meaningful difference in the solution when we decrease ε further. Computation time increases almost linearly as we increase the number of data, as expected. For details about this model readers are referred to [15].

4. Conclusions and Future Direction of Research

We proposed (cutting plane) algorithms for minimizing a (not necessarily convex) function subject to linear and semi-definite constraints and showed that the algorithm converges under coerciveness condition and under somewhat weaker condition when the objective function is concave. Also, computational results for three practical problems show that these algorithms can solve them within a practical amount of time.

Let us note that the efficiency of the algorithm crucially depends upon the following facts

(i) The dimension of the semi-definite matrix $\boldsymbol{X}$ is small, i.e., less than say 10.

(ii) Subproblem (8) can be solved reasonably fast.

The class of objective function f amenable to our approach include, among others convex functions and low rank nonconvex functions (See [14]).

When the dimension of the matrix is over 15, computation time will explode as observed for other outer approximation algorithm. Therefore, we need to use other

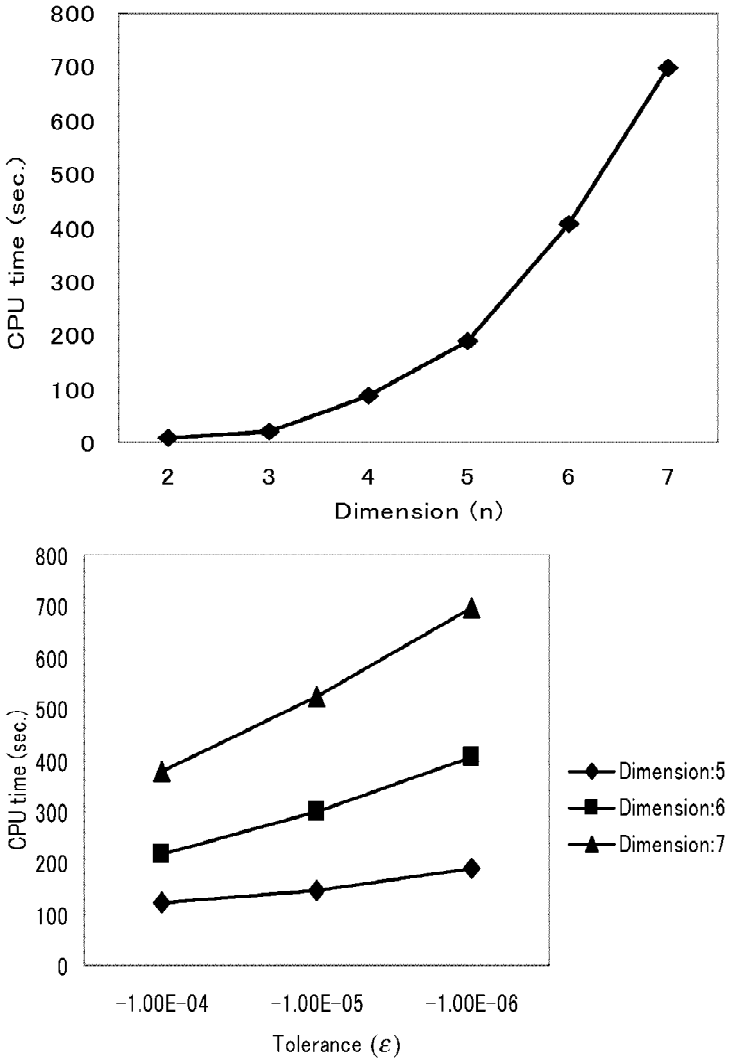

Figure 2.

algorithms less sensitive to the dimension of $\boldsymbol{X}$. Such examples are interior point algorithms [7, 21].

However, as demonstrated in [11–13], outer approximation method is much superior to standard interior point algorithm such as $SDPA$ 5.0, when the objective function is linear (Section 3.1). There may be other interior point based codes which can solve problems of Section 3.2 and 3.3 faster, but it is usually not easy to choose appropriate value of parameters to obtain good results, whereas our algorithm is much easier to implement.

In addition to the problems discussed in this paper, there exists a number of important problems which may be solved by our algorithms. One recent example is 'optimal fitting of volatility matirix' discussed in [3].

As a final remark, our algorithm can be applied to a problem with linear variables in addition to semi-definite variables. The efficiency of algorithm does not depend on the number of linear variables.

Acknowledgements

The research of the first author is partly supported by Grant-in-Aid for Scientific Research of the Ministry of Education, Science and Culture B(2) 11558046 and B(2) 12480105. Also, the authors acknowledge the support of the Hitachi Research Institute, Co.

References

1. Altman, E.I. and Nelson, A.D. (1968), Financial ratios, discriminant analysis and the prediction of corporate bankruptcy. *Journal of Finance* **23**: 589–609.
2. Bertsimas, D. and Popescu, I. (1998), On the relation between option and stock prices: a convex optimization approach, Technical Paper, Sloan School of Managemanet, MIT.
3. Brace, A. and Womersley. R. S., Exact fit to the swaption volatility matrix using semidefinite programming. to appear in *Mathematical Finance*.
4. Bradley, P. S., Fayyad, U. M. and Mangasarian, O. L. (1999), Mathematical programming for data mining: formulations and challenges. *INFORMS J. of Computing* 11: 217–238.
5. Fujisawa, K. Kojima, M. and Nakata, K. (1999), SDPA User's Manual-Version 5.0, Research Paper on Mathematical and Computing Science, Tokyo Institute of Technology.
6. Gantmacher, F. R. (1959), *The Theory of Matrices*, Chelsea Publishing Co.
7. Helmberg, C., Rendl, F., Vanderbei, R. J. and Wolkowitz, H. (1996), An interior point method for semi-definite programming. *SIAM J. on Optimizaiton* 6: 342–361.
8. Horst, R. and Tuy, H. (1996), *Global Optimization: Deterministic Approaches*, 3rd ed. Springer, Berlin.
9. Ioffe, A.D., and Tihomirov, V.M. (1979), *Theory of External Problems*. North-Holland, Amsterdam.
10. Kelley, J.E. (1960), The cutting-plane method for solving convex programs. *J. of the Society of Industrial Applied Mathematics*, 8: 703–712.
11. Konno, H., Gotoh, J. and Uno, T. (2000), A cutting plane algorithm for semi-definite programming problems with applications. WP 00-05, Center for Research in Advanced Financial Technology. Tokyo Institute of Technology. (to appear in *J. of Computational and Applied Mathematics*)
12. Konno, H., Gotoh, J. and Uryasev, S. and Yuuki, A. (2001), Failure discrimination by semi-definite programming. WP01-02, Center for Research in Advanced Financial Technology, Tokyo Institute of Technology. (to appear in *FEES 2001* (P. Pardalos, ed.))
13. Konno, H. and Kobayashi, H. (2000), Failure discrimination and rating of enterprises by semi-definite programming, *Asia-Pacific Financial Markets*,7: 261–273.
14. Konno, H., Thach, P.T. and Tuy, H. (1997), *Optimization over Low Rank Nonconvex Structures*, Kluwer Academic Publishers, Dordrecht.
15. Konno, H. and Wu, D. (2001), Estimation of failure probability using semi-definite logit model. WP 01-04, Center for Research in Advanced Financial Technology.

16. Mangasarian, O., Street, W. and Wolberg, W. (1995), Breast cancer diagnoses and prognoses via linear programming. *Operations Research*, 43: 570–577.
17. Tuy, H. (1983), Outer approximation methods for solving concave minimization problems. *Acta Mathematica Vietnamica*, 8: 3–34.
18. Vandenberghe, L. and Boyd, S. (1998), Connections between semi-definite and semi-infinite programming. *Semi-infinite Programming*, 277–294, Kluwer Academic Publishers, Boston.
19. Vandenberghe, L., and Boyd, S. (1996), Semi-definite programming. *SIAM Review* 38: 49–95.
20. Veinott., A. F., Jr. (1967), The supporting hyperplane method for unimodal programming. *Operations Research*, 15: 147–152.
21. Wolkowitz, H., Saigal, R. and Vandenberghe, L. (2000), *Handbook of Semi-Definite Programming: Theory, Algorithm and Application*, Kluwer Academic Publishers, Dordrecht.