TWO PROBLEMS IN CONVEX CONIC OPTIMIZATION

ZHAI XIAOJUN

(B.Sc., Northwestern Polytechnical University)

A THESIS SUBMITTED

FOR THE DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF MATHEMATICS

NATIONAL UNIVERSITY OF SINGAPORE

2007

Acknowledgements

I would like to express my sincere gratitude to my graduate advisor Professor Toh Kim-Chuan for his constant support, guidance and motivation. Remember two years ago, he said to me that he was sure that I would learn something. After two years' study, today finally I can say, "Yes, I did learn a lot from you, Sir." I am also grateful to the discussions with Professor Sun Defeng, he is a good researcher with keen eyes.

I would like to thank all my colleagues in Department of Mathematics, NUS, especially for Li Lu who offered me a lot of helpful discussions on the thesis.

At last I will thank for my family, my parents who support me all the time.

Abstract

In the first part of the thesis we focus on smallest enclosing ball problem. The smallest enclosing ball of balls problem (SEBB) is to find a ball that encloses a set of balls in R^n. In this part, first we use a smoothing log-exponential aggregation function to smooth the objective function, then we apply Nesterov's efficient gradient method to the resulting Lipschitz continuous smooth convex optimization problem. Further, we improve Nesterov's algorithm by adaptively decreasing the smoothing parameter and modified backtracking line search. From the viewpoint of efficiency estimates, we are able to solve a non-smooth convex minimization problem with $O(\frac{1}{\epsilon})$ iterations to achieve ϵ-accuracy in the approximate solution, compared to the traditional bound $O(\frac{1}{\epsilon^2})$ given by the subgradient method. At last, we present a numerical comparison of the performance of different algorithms in solving the SEBB problem. We show that the improved Nesterov's algorithm is able to solve large SEBB problem (in order of thousands) in moderate time.

The second part of the thesis is on the generalized convex quadratic semidefinite program (GQSDP). We use the primal-dual path following method to solve (GQSDP). To deal with the ill-conditioned linear system, we design two types of preconditioners and prove their effectiveness. At last, we show some numerical results on the primal-dual path following method of solving (QSDP).

Contents

Part I

Smallest Enclosing Ball Problem

Chapter 1

Introduction

1.1 Smallest Enclosing Ball Problem

In this part, we study the problem of finding the closed ball of smallest radius that contains a given set of n closed balls in d-dimensional Euclidean space. This problem, which we denote by SEBB, generalizes the well understood problem of finding the smallest enclosing ball of n given points, which is called SEBP. First we define a ball in n dimensional space. A ball B_i in R^n with center $c_i \in R^n$ and radius $r_i > 0$ is the closed set $B_i = \{x \in R^n : ||x - c_i|| \leq r_i\}$. From now on, we denote the ball B_i with center c_i and radius r_i as $B_i(c_i, r_i)$. Then we introduce smallest enclosing ball of points problem.

Problem 1 (Smallest Enclosing Ball of Points)**.** *Given a set of points $\mathcal{P} = \{p_1, p_2, ..., p_m\}$ in R^n, we want to find a ball $B^* = B^*(c^*, r^*)$ with the smallest radius r^*, such that $\mathcal{P} \subseteq B$.*

If the points in $\mathcal{P}$ become balls, then we get smallest enclosing ball of balls problem, SEBB.

Problem 2 (Smallest Enclosing Ball of Balls)**.** *Given a set of balls $\mathcal{B} = \{B_1, B_2, ..., B_m\}$ in R^n, we want to find a ball $B^* = B^*(c^*, r^*)$ with the smallest radius r^* such that $\forall B_i \in \mathcal{B}$, $B_i \subseteq B^*$.*

In n-dimensional space, suppose the center of our smallest enclosing ball is c, then in order to enclose the ball $B_i(c_i, r_i)$, the smallest enclosing ball must have a radius equal to $||c - c_i|| + r_i$.

From now on, we denote the distance between a point x_i and a ball $B(c_i, r_i)$ to be $D(x_i, B) = ||x_i - c_i|| + r_i$. Since we are trying to find the enclosing ball with the smallest radius, the smallest enclosing ball problem can be formulated as the following convex optimization problem:

$$\min_{x \in R^n} \max_{1 \leq i \leq m} \{||x - c_i|| + r_i\} \tag{1.1}$$

If we denote

$$f_i(x) = ||x - c_i|| + r_i, i = 1, 2, ...m$$

and

$$f(x) = \max_{1 \leq i \leq m} \{f_i(x)\} \tag{1.2}$$

then the problem (1.1) can be rewritten as

$$\min_{x \in R^n} f(x) \tag{1.3}$$

We claim that the solution to (1.1) exists and is unique. First, for a smallest enclosing ball problem with balls $\mathcal{B} = \{B_1, B_2, ..., B_m\}$, we fix one point c, and radius

$$r = \max_{i=1 \cdots m} D(c, B_i)$$

then $B(c, r)$ enclose all the balls in $\mathcal{B}$. Then we claim that the center of the smallest enclosing ball is in the ball $B(c, r)$, (see Proposition 4). In this close set $B(c, r)$, $f(x)$ is continuous, thus must have a minimum. So the solution to (1.1) exists. Second, the solution is unique. Otherwise there would exist two different balls, B_1 and B_2, of the same radius, with all balls in $\mathcal{B}$ are in B_1 and B_2. Then we can always construct a smaller ball $\bar{B} \supseteq B_1 \cap B_2$, and $\bar{B}$ contains all the balls in $\mathcal{B}$.

1.2 Application

Problem (1.1) arises in applications such as location analysis and military operations and now becomes an interesting topic in computational geometry. The applications include collision detection [20], the computation of bounding sphere hierarchies for clustering or efficient rendering of complex scenes, culling (e.g. for visualization of molecular models [8]), farthest neighbour approximation [1], automated manufacturing [23], and similarity search in feature spaces [21].

1.3 Related Works and Outline

Many algorithms have been developed for SEBP and SEBB problem. For SEBP problem, Megiddo [17] presented a deterministic $O(m)$ algorithm for the case $n < 3$. Welzl [5] developed an algorithm which is linear in m when n is small. For SEBB problem, a C++ program based on Welzl's algorithm and Gärtner's implementation [3] is developed by David White [4]. Software packages, like [3] [4], are only efficient when $n < 30$. Recently, Zhou et al. [7] have designed several efficient algorithms to solve SEBB problem. They smooth the function $f(x)$, reformulate SEBB as an SOCP problem. Their algorithms can be used to solve problems where n and m are large (in order of thousands).

In this part, we developed a new approach to solve the SEBB problem. After applying Nesterov's algorithm to the smoothed function $f(x)$, we improve the algorithm by allowing a new backtracking line search scheme, and a gradually decreasing smoothing parameter. Both of the techniques make our program faster. With the improved algorithm, we can solve large size problems in moderate time with m and n larger than 1000.

Problem(1.1) is a non-differentiable, non-smooth convex optimization problem.

Because of the non-differentiability of the objective function, regular gradient methods cannot be used to solve the problem. In order to overcome this difficulty, in Chapter 2 we smooth the function $f(x)$ by a log-exponential aggregation function. Then, we discuss the Nesterov's algorithm and its implementation to the smoothed SEBB problem in Chapter 3. In Chapter 4 we improve Nesterov's algorithm by allowing a new backtracking line searching scheme and smoothing parameter decreasing scheme in the algorithm. In Chapter 5 we report some numerical results, which show a comparison of the algorithms proposed here.

Notation: We denote by $C_L^{k,p}(Q)$ the class of functions with the following properties

- Any $f(x) \in C_L^{k,p}(Q)$ is k times continuously differentiable on Q
- Its pth derivative is Lipschitz continuous on Q with the constant L such that

$$||f^{(p)}(x) - f^{(p)}(y)|| \leq L||x - y||$$

for all $x, y \in Q$

Chapter 2

Smooth Approximation

Nesterov's algorithm is a gradient method for smoothed, first order Lipschitz continuous convex function. Since $f(x)$ is not smooth, in order to apply this algorithm to SEBB problem, we have to smooth $f(x)$ first. For any $p > 0$, define the smoothing log-exponential aggregation function $f(x;p)$ for $f(x)$ in (1.2) as

$$f(x;p) = p \ln \left(\sum_{i=1}^{m} \exp(g_i(x;p)/p) \right) \quad \text{where} \quad g_i(x;p) = r_i + \sqrt{||x - c_i||^2 + p^2} \tag{2.1}$$

After smoothing, we can see that $f(x;p)$ is a smooth function with smoothing parameter p. The following proposition lists some of the important properties of $f(x;p)$.

Proposition 1. *The function $f(x;p)$ has the following properties:*

1. *For any $x \in R^n$, and p_1, p_2 satisfying $0 < p_1 < p_2$, we have*

$$f(x, p_1) < f(x; p_2)$$

2. *For any $x \in R^n$ and $p > 0$, $f(x) \leq f(x;p) \leq f(x) + p(1 + \ln m)$.*

3. *For any $p > 0$, $f(x;p)$ is continuously differentiable and strictly convex.*

Proof. 1. For any $x \in R^n$ and p_1,p_2 satisfying $0 < p_1 < p_2$, by Jensen's inequality,

$$\left[\sum_{i=1}^{m}(\exp(g_i(x;p_2)))^{1/p_2}\right]^{p_2} > \left[\sum_{i=1}^{m}(\exp(g_i(x;p_2)))^{1/p_1}\right]^{p_1} > \left[\sum_{i=1}^{m}(\exp(g_i(x;p_1)))^{1/p_1}\right]^{p_1}$$

Hence, $f(x;p_1) < f(x;p_2)$.

2. Fix $p = p_2$ and let $p_1 \to 0$ in 1, we have $f(x) < f(x;p)$. Let

$$g_\infty(x;p) = \max_{1\le i\le m}\{g_i(x;p)\}$$

It is readily proven that $f(x) \le g_\infty(x;p) \le f(x) + p$. Thus, from (2.1), we have

$$f(x;p) = g_\infty(x;p) + p\ln\sum_{i=1}^{m}\exp[(g_i(x;p) - g_\infty(x;p))/p] \le g_\infty(x;p) + p\ln m$$

Hence,

$$f(x) \le f(x;p) \le f(x) + p(1 + \ln m)$$

3. For any $p > 0$, clearly, $f(x;p)$ is continuously differentiable. Now we prove that $f(x;p)$ is strictly convex. From (2.1), we know

$$\nabla f(x;p) = \sum_{i=1}^{m}\frac{\lambda_i(x;p)}{h_i(x;p)}(x - c_i) \tag{2.2}$$

where

$$h_i(x;p) = \sqrt{||x - c_i||^2 + p^2}, \quad \tau(x;p) = \sum_{i=1}^{m}\exp(g_i(x;p)/p) \tag{2.3}$$

$$\lambda_i(x;p) = \frac{\exp(g_i(x;p)/p)}{\tau(x;p)} \tag{2.4}$$

Further, we define

$$Q_{ij} = \frac{(x - c_i)(x - c_j)^T}{h_i(x;p)h_j(x;p)}$$

From (2.2), we can get

$$\nabla^2 f(x;p) = \sum_{i=1}^{m}\left[\frac{\lambda_i(x;p)}{h_i(x;p)}(I_n - Q_{ii}) + \frac{\lambda_i(x;p)}{p}Q_{ii} - \sum_{j=1}^{m}\frac{\lambda_i(x;p)\lambda_j(x;p)}{p}Q_{ij}\right]$$

For any $z \in R^n$ with $z \neq 0$, by the Cauchy-Schwartz inequality,

$$||z||^2 - z^T Q_{ii} z \geq ||z||^2 - ||z||^2 ||(x - c_i)/h_i(x;p)||^2 > 0 \quad \forall i = 1, ...m.$$

Thus,

$$\begin{aligned} z^T \nabla^2 f(x;p) z &= \sum_{i=1}^{m} \frac{\lambda_i(x;p)}{h_i(x;p)} (||z||^2 - z^T Q_{ii} z) \\ &+ \sum_{i=1}^{m} \left[\frac{\lambda_i(x;p)}{p} z^T Q_{ii} z - \sum_{j=1}^{m} \frac{\lambda_i(x;p)\lambda_j(x;p)}{p} z^T Q_{ij} z \right] \\ &> \sum_{i=1}^{m} \left[\frac{\lambda_i(x;p)}{p} z^T Q_{ii} z - \sum_{j=1}^{m} \frac{\lambda_i(x;p)\lambda_j(x;p)}{p} z^T Q_{ij} z \right] \\ &= \frac{1}{p} \sum_{i=1}^{m} \lambda_i(x;p) a_i^2 - \frac{1}{p} \left(\sum_{i=1}^{m} \lambda_i(x;p) a_i \right)^2 \geq 0 \end{aligned} \tag{2.5}$$

where

$$a_i = z^T (x - c_i)/h_i(x;p) \tag{2.6}$$

This shows that $\nabla^2 f(x;p)$ is positive definite. Therefore, $f(x;p)$ is strictly convex. Note that the inequality (2.5) follows from the fact that

$$|\sum_{i=1}^{m} \lambda_i(x;p) a_i| \leq \sqrt{\sum_{i=1}^{m} \lambda_i(x;p)} \sqrt{\sum_{i=1}^{m} \lambda_i(x;p) a_i^2}$$

and $\sum_{i=1}^{m} \lambda_i(x;p) = 1, \quad \lambda_i(x;p) \geq 0$ for $i = 1, ...m$.

□

Now we want to look at the Lipschitz continuous property of $f(x)$ and get the Lipschitz constant, first we introduce a lemma.

Lemma 1. *For any function $f(x), x \in R^n$ which is twice continuous differentiable, the first derivative of $f(x)$ is Lipschitz continuous with constant L*

$$||\nabla f(x) - \nabla f(y)|| \leq L||x - y|| \qquad \forall x, y \in R^n$$

if

$$||\nabla^2 f(\omega)|| \leq L \qquad \forall \omega \in R^n \tag{2.7}$$

Proof. For any $x, y \in R^n$, we have

$$\begin{aligned}\nabla f(y) &= \nabla f(x) + \int_0^1 \nabla^2 f(x + t(y-x))(y-x)dt \\ &= \nabla f(x) + \left(\int_0^1 \nabla^2 f(x + t(y-x))dt\right)(y-x)\end{aligned}$$

If (2.7) is satisfied, then

$$\begin{aligned}||\nabla f(y) - \nabla f(x)|| &= \left|\left|\left(\int_0^1 \nabla^2 f(x + t(y-x))dt\right) \cdot (y-x)\right|\right| \\ &\leq \left|\left|\int_0^1 \nabla^2 f(x + t(y-x))dt\right|\right| \cdot ||y-x|| \\ \leq \int_0^1 ||\nabla^2 f(x + t(y-x))||dt \cdot ||y-x|| &\leq L||y-x||\end{aligned}$$

□

Proposition 2. *The first derivative of $f(x;p)$ in (2.1) is Lipschitz continuous with constant $\frac{2}{p}$.*

Proof. From Lemma 1 we know that if we can prove $||\nabla^2 f(x)|| \leq L, \forall x \in R^n$ then L is the Lipschitz constant. From Proposition 1 we know that $f(x;p)$ is strictly convex, which means the Hessian matrix $\nabla^2 f(x)$ should be symmetric and positive definite. By the definition of induced matrix 2-norm, we have

$$||\nabla^2 f(x)|| = \max_{||z||=1} ||z^T \nabla^2 f(x) z||$$

So, it is equivalent to proving that

$$\max_{||z||=1} z^T \nabla^2 f(x;p) z \leq \frac{2}{p} \tag{2.8}$$

By the definition (2.3) and (2.4), we know that

$$\lambda_i(x;p) \geq 0. \quad h_i(x;p) \geq 0. \quad \frac{1}{h_i} = \frac{1}{\sqrt{||x - c_i||^2 + p^2}} \leq \frac{1}{p}$$

Further,

$$\sum_{i=1}^m \frac{\lambda_i(x;p)}{h_i(x;p)}(z^T Q_{ii} z) = \sum_{i=1}^m \frac{\lambda_i(x;p)}{h_i(x;p)} \cdot \frac{[z^T(x - c_i)]^2}{h_i(x;p)^2} \geq 0$$

$$\frac{[z^T(x-c_i)]^2}{h_i^2} \leq \frac{||x-c_i||_\infty^2}{||x-c_i||^2+p^2} \leq 1$$

and

$$\sum_{i=1}^{m}\sum_{j=1}^{m}\frac{\lambda_i(x;p)\lambda_j(x;p)}{p}z^TQ_{ij}z = \frac{1}{p}\left(\sum_{i=1}^{m}\lambda_i(x;p)a_i\right)^2 \geq 0$$

where a_i is defined by (2.6).

We can now prove (2.8). From (2.5) we know that

$$\begin{aligned}
z^T\nabla^2 f(x;p)z &= \sum_{i=1}^{m}\frac{\lambda_i(x;p)}{h_i(x;p)}(||z||^2 - z^TQ_{ii}z) \\
&+ \sum_{i=1}^{m}\left[\frac{\lambda_i(x;p)}{p}z^TQ_{ii}z - \sum_{j=1}^{m}\frac{\lambda_i(x;p)\lambda_j(x;p)}{p}z^TQ_{ij}z\right] \\
&\leq \sum_{i=1}^{m}\frac{\lambda_i(x;p)}{h_i(x;p)}||z||^2 + \sum_{i=1}^{m}\left[\frac{\lambda_i(x;p)}{p}z^TQ_{ii}z\right] \\
&= \sum_{i=1}^{m}(\frac{1}{p}\cdot\frac{[z^T(x-c_i)]^2}{h_i^2} + \frac{1}{h_i})\cdot\lambda_i(x;p) \\
&\leq (2/p)\sum_{i=1}^{m}\lambda_i(x;p) = \frac{2}{p} \qquad (2.9)
\end{aligned}$$

Note that (2.9) follows from the fact that $\sum_{i=1}^{m}\lambda_i(x;p) = 1$. □

Chapter 3

Nesterov's Algorithm and Implementation for SEBB

3.1 Nesterov's Algorithm

3.1.1 Algorithm Analysis

In [24] Nesterov discussed an efficient algorithm to solve the smooth convex minimization problem with Lipschitz continuous first derivative. Suppose we have a function $f(x)$ with continuous first derivative, defined on a convex subset Q of R^n, i.e. $f(x) \in C_L^{(1,1)}(Q)$ and we have

$$||\nabla f(x) - \nabla f(y)||^* \leq L||x - y|| \qquad \forall x, y \in Q$$

(we are using the Euclidean norm, so $||\cdot||_2^* = ||\cdot||_2$)
Since $f(x)$ is convex, we have the following inequality

$$f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{1}{2} L||y - x||^2 \tag{3.1}$$

Further, we take $T_Q(x) \in Q$ to be the optimal solution of the following minimization problem

$$\min_y \left\{ \langle \nabla f(x), y - x \rangle + \frac{1}{2} L||y - x||^2, \quad y \in Q \right\} \tag{3.2}$$

Substitute (3.2) to (3.1) we can get

$$f(T_Q(x)) \leq f(x) + \min_y\{\langle \nabla f(x), y - x\rangle + \frac{1}{2}L||y - x||^2 : y \in Q\} \tag{3.3}$$

While we are updating our solution from x to $T_Q(x)$, (3.3) offers us an one step minimization bound in Nesterov's algorithm. Further, we need a prox-function to implement Nesterov's Algorithm , which can be defined as follows.

Define $d(x)$ to be a function that is continuous and strongly convex on the closed convex set $Q \in R^n$ with convexity parameter $\sigma > 0$. Let x_0 be the center of the set Q.

$$x_0 = arg\min_x\{d(x) : x \in Q\}$$

Without loss of generality assume that $d(x_0) = 0$. Thus, for any $x \in Q$ we have

$$d(x) \geq \frac{1}{2}\sigma||x - x_0||^2 \tag{3.4}$$

Now we are ready to use the Nesterov's algorithm to solve the problem

$$\min_x\{f(x) : x \in Q\} \tag{3.5}$$

where Q is a closed convex set and $f(x) \in C_L^{1,1}(Q)$.

Algorithm A

For $k \geq 0$, do the following until some terminating criteria is satisfied

1. Compute $f(x_k)$ and $\nabla f(x_k)$
2. Find $y_k = T_Q(x_k)$
3. Find
$$z_k = arg\min_x\{\frac{L}{\sigma}d(x) + \sum_{i=0}^{k}\frac{i+1}{2}(f(x_i) + \langle \nabla f(x_i), x - x_i\rangle) : x \in Q\}$$
4. Set $x_{k+1} = \frac{2}{k+3}z_k + \frac{k+1}{k+3}y_k$

Basically, The algorithm is trying to update recursively two sequences of points $\{x_k\}_{k=0}^{\infty}$ and $\{y_k\}_{k=0}^{\infty}$ until y_k is close enough to the optimum.

3.1.2 Complexity Analysis of Nesterov's Algorithm

In [24] Nesterov proves a theorem to measure the complexity of his algorithm.

Theorem 1. *Let the sequence* $\{x_k\}_{k=0}^{\infty}$ *and* $\{y_k\}_{k=0}^{\infty}$ *be generated by Nesterov's Algorithm, then for any* $k \geq 0$*, we have*

$$\frac{(k+1)(k+2)}{4} f(y_k) \leq \min_x \left\{ \frac{L}{\sigma} d(x) + \sum_{i=0}^{k} \frac{i+1}{2} [f(x_i) + \langle \nabla f(x_i), x - x_i \rangle] : x \in Q \right\}$$

Therefore,

$$f(y_k) - f(x^*) \leq \frac{4Ld(x^*)}{\sigma(k+1)(k+2)} \tag{3.6}$$

where x^* *is an optimal solution to the problem (3.5).*

Define the absolute difference between current objective value and the optimum value as d, given $\epsilon > 0$, **efficiency estimate** for an algorithm is the order of steps required to achieve $\epsilon < d$.

Proposition 3. *Nesterov's Algorithm have an efficiency estimate of the order of* $O\left(\sqrt{\frac{L}{\epsilon}}\right)$.

Proof. This is a natural result implied by Theorem 1. □

3.2 Apply Nesterov's Algorithm to SEBB

We are going to apply Nesterov's algorithm to the smoothed SEBB, namely, we are going to use Nesterov's Algorithm to solve the problem

$$\min_x \{ f(x; p) : x \in Q \} \tag{3.7}$$

where Q is a closed convex set, $f(x; p)$ is defined by (2.1).

3.2.1 Determine the Closed Convex Set Q

Nesterov's algorithm requires the problem to be defined on a closed, convex set Q.

Proposition 4. *In problem (3.7), suppose the set of balls we are given is*

$$\mathcal{B} = \{B_1(c_1, r_1), \cdots, B_m(c_m, r_m)\}$$

then the smallest enclosing ball of set $\mathcal{B}$ is enclosed in the closed convex set given by

$$B(0, \max_{i=1\cdots m} D(0, B_i)).$$

Proof. Here we give the proof for the 2-dimensional case. First define

$$\max_{i=1\cdots m} D(0, B_i) = r^*$$

See figure 3.1, O is the origin. We want to prove that the smallest enclosing ball for $\mathcal{B}$ is enclosed in $B(O, r^*)$. Suppose there exists a smallest enclosing ball $B(O^{'}, r^{'})$ with center $O^{'}$ outside $B(0, r^*)$, radius $r^{'} \leq r^*$, i.e. $B(O^{'}, r^{'}) \nsubseteq B(O, r^*)$. Then we can find an enclosing ball $B(O^{''}, r^{''})$ for $\mathcal{B}$ with smaller radius than $r^{'}$. The center $O^{''} \in B(0, r^*)$, $O^{''}$ is on the surface of $B(0, r^*)$, and in a line with $O^{'}$ and O. We will show that for all $B_i(c_i, r_i)$ in $\mathcal{B}$, $D(O^{''}, B_i) \leq D(O^{'}, B_i)$.

Actually we can suppose X is the center of any ball in $\mathcal{B}$, then X should in the intersection of $B(O, r^*)$ and $B(O^{'}r^{'})$, in the triangle $\triangle O^{'}O^{''}X$, $\angle XO^{''}O^{'} \geq \frac{\pi}{2}$, so we always have $||O^{''}X|| \leq ||O^{'}X||$. Thus $O^{''} \in B(0, r^*)$ is the new smaller center of enclosing ball.

Therefore,the smallest enclosing ball for $\mathcal{B}$ is enclosed in $B(O, r^*)$. □

3.2.2 Determine $d(x)$

We choose the prox-function in Nesterov's algorithm to be

$$d(x) = \frac{1}{2}||x||^2$$

which means $\sigma = 1$ and $x_0 = 0$.

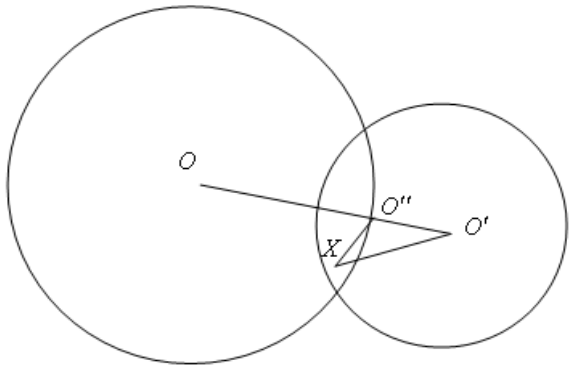

Figure 3.1: Determine closed set Q

3.2.3 Further Analysis of Implementation

In order to apply the Nesterov's algorithm to smallest enclosing ball problem, we have to do some more analysis on details of the algorithm. At the second step of the Nesterov's algorithm, we have to find

$$T_Q(x) = arg\min_x\{\langle\nabla f(x;p), y-x\rangle + \frac{1}{2}L||y-x||^2\} \tag{3.8}$$

The objective function is differentiable and it is easy to see that the minimum is achieved when the gradient of the above function reaches 0. Thus, optimal solution is

$$y = x - \frac{\nabla f(x;p)}{L} \tag{3.9}$$

At the third step of the algorithm, we have to find

$$arg\min\left\{\frac{Ld(x)}{\sigma} + \sum_{i=0}^{k}\frac{i+1}{2}(f(x_i;p) + \langle\nabla f(x_i;p), x-x_i\rangle)\right\} \tag{3.10}$$

Similarly to (3.8), the solution of the (3.10) is

$$z_k = -\sum_{i=0}^{k}\frac{i+1}{2L}\nabla f(x_i;p)$$

For computation efficiency, we use the recursive expression

$$z_k = z_{k-1} - \frac{k+1}{2L}\nabla f(x_k;p)$$

3.2.4 Complexity Analysis of Nesterov's Algorithm for SEBB

Proposition 5. *By applying Nesterov's Algorithm to problem 3.7, we can get efficiency estimate of the order* $O(\frac{1}{\epsilon})$.

Proof. By applying the Nesterov's algorithm to SEBB, we have two types of errors. First type of error arises from the smoothing approximation, i.e. replacing $f(x)$ with $f(x;p)$, we denote this error by ϵ_1. The other type of error is from Nesterov's algorithm, we denote this type of error as ϵ_2.
Given $\epsilon > 0$, we choose $p = \frac{\epsilon}{1+\ln m}$, by Proposition 1

$$f(x;p) \leq f(x) + p(1 + \ln m)$$

this makes

$$\epsilon_1 = f(x;p) - f(x) \leq \epsilon$$

We also set $\epsilon_2 = c \cdot \epsilon \leq \epsilon (c$ is a fixed scalar which $\in (0,1))$. By Proposition 3, Nesterov's Algorithm has efficiency estimate of order $O\left(\sqrt{\frac{L}{\epsilon_2}}\right)$, which is equivalent to $O(\sqrt{\frac{L}{\epsilon}})$. Further, we know $L = \frac{2}{p}$, which means

$$L = \frac{2}{p} = \frac{1 + \ln m}{\epsilon}$$

Combine these two types of errors, we get the final result that Nesterov's algorithm has the efficiency estimate of $O\left(\frac{1}{\epsilon}\right)$ on smoothed SEBB problem. □

Chapter 4

Improvement on Nesterov's Algorithm

4.1 Adaptively Decreasing the Smooth Parameter

In the Chapter 2, we use the log-exponential function to smooth $f(x)$ and introduce the smoothing parameter p. By the Proposition 1, we know that the smaller p is, the smaller our approximation error is. However, if we use a small p throughout the algorithm, it will result in a very big Lipschitz constant at the beginning. In Nesterov's algorithm, this will cause the step length $1/L$ to be very small, which means that the improvement in each step is small. This may affect the efficiency of our algorithm. The later numerical experiment also tells us that if we choose a small p throughout the algorithm, our program will stagnate. One solution to this problem is to allow the smoothing parameter p to decrease gradually to the target value. We use an algorithm as follows (see [7].)

Algorithm B

Let $\sigma \in (0,1)$, $x_0 \in R^n$ and $p_0 > 0$, $\epsilon_1, \epsilon_2 > 0$ be given,and set $k := 0$.
For $k = 0, 1, 2 \cdots$, until $p_k \leq \epsilon_1$ **do**

1. Use an unconstrained minimization method to solve

$$\min_{x \in R^n} f(x; p_k)$$

   approximately, and obtain an x_k such that $||\nabla f(x_k, p_k)|| \leq \epsilon_2$

2. set $p_{k+1} = \sigma p_k$, increment k by 1, and return to step 1.

Then we introduce a theorem to ensure that by using a decreasing smoothing parameter scheme, the algorithm stills goes to the optimal solution of original problem. (see [7])

Theorem 2. *Let $\{x_k\}_{k \geq 1}$ be the sequence of points produced by Algorithm B, x^* be the unique optimal solution of the problem (1.3) i.e.*

$$\min_{x \in R^n} f(x)$$

Then

$$\lim_{k \to +\infty} x_k = x^*$$

4.2 Backtracking Line Search Scheme

Nesterov's algorithm simultaneously generates two sequences toward the optimum point. One of them (equation 3.9) can be described as

$$y = x - \frac{\nabla f(x; p)}{L}$$

This is nothing but a line search process along the negative gradient direction. If we consider the above equation as a gradient decent approach, then the line search step $1/L$ is the step length we go along the $-\nabla f(x)$ direction. The questions here are: Is $1/L$ the optimal step length? Can we take larger steps? Actually we can always find a line search scheme to do better by taking a larger step length along the negative gradient direction, while still maintaining the convergence of the algorithm, for example, by backtracking.

4.2.1 Implementation of Traditional Backtracking

The traditional back-tracking line search can be described as follows.

Backtracking Line Search Algorithm

Given $0 < \alpha < 0.5, 0 < \beta < 1$, and a descent direction $-\nabla f(x)$ for f:
$t := 1$
while $f(x - t\nabla f(x) > f(x) - \alpha t||\nabla f(x)||^2 \qquad t := \beta t$
end

Instead of using the constant step length $1/L$, we can use the above algorithm to generate step length in each step. Before implementating the line search algorithm, let us do some theoretical analysis first.

First we look at Nesterov's algorithm. Substitute (3.9) to (3.1) gives us

$$f(y) = f(x - \frac{1}{L}\nabla f(x)) \leq f(x) - \frac{1}{2L}||\nabla f(x)||^2 \tag{4.1}$$

which means in each step, Nesterov's algorithm chooses a constant step length $1/L$, and at each step it decreases the objective function by at least $\frac{1}{2L}||\nabla f(x)||^2$.

On the other hand, for the above backtracking line search algorithm, the exit condition is given by

$$f(x - t\nabla f(x)) \leq f(x) - \alpha t||\nabla f(x)||^2 \tag{4.2}$$

Proposition 6. *Backtracking condition (4.2) is always satisfied whenever* $0 \leq t \leq 1/L$ *and* $0 < \alpha \leq 1/2$.

Proof. See [22]. □

The above proposition means that if we use backtracking by choosing $\alpha = 0.5$ and start to increase length step from $t = 1/L$, then the step length t should always greater than $1/L$. So for the backtracking line search, we have

$$f(y) = f(x - t\nabla f(x)) \leq f(x) - \frac{1}{2L}||\nabla f(x)||^2 \tag{4.3}$$

Theoretically, the bound given by backtracking is the same as Nesterov's algorithm (4.1). However, in computational practice, at certain cost of backtracking, we can normally get much larger search steps than $1/L$.(See Figure 4.1.)That means we can reduce the objective function by a larger amount in each step.

4.2.2 Modified Backtracking

It seems that implementing the backtracking line search in Nesterov's algorithm is promising, however, if we implement the traditional backtracking line search in Nesterov's algorithm directly, we will fail. The new algorithm is even slower (in terms of cpu time) than the original Nesterov's algorithm. By analyzing the profile of our algorithm, we found that we took too much time in backtracking to determine the step length t in each step.

At the same time we observed that while the smoothing parameter p is decreasing as the solution approaches the optimum, the step length t will first increase sharply, and then decrease gradually. Throughout the whole process, the real step we can take is always much larger than $1/L$, (See Figure 4.1). We also observed that the

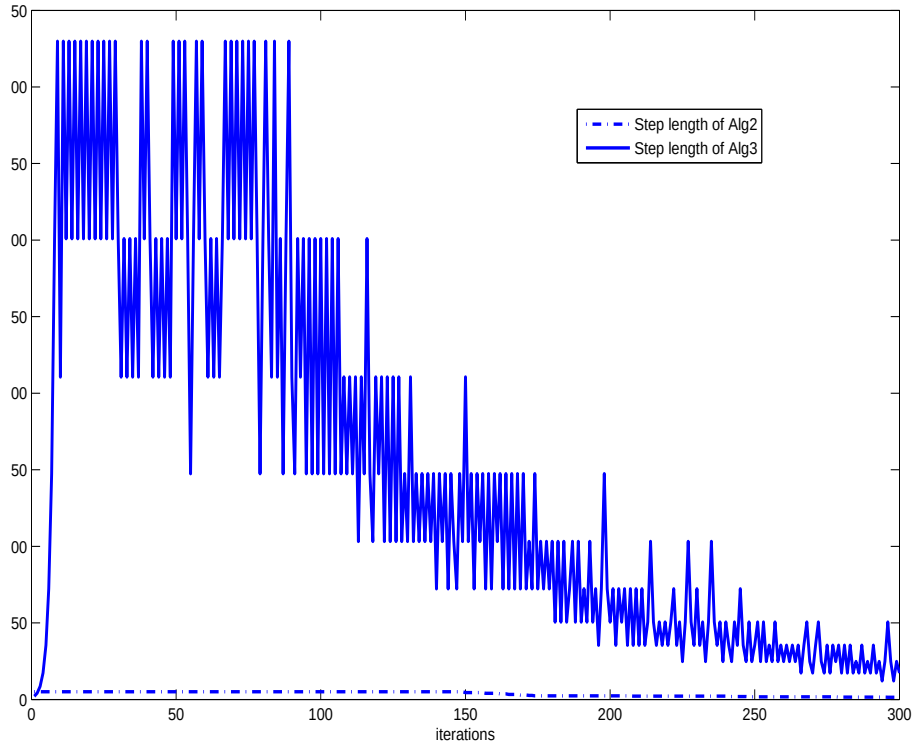

Figure 4.1: Step length of Alg2 and Alg3. Solid Curve–Alg3, Dashed Curve–Alg2 (for the definitions of algorithms Alg2 and Alg3, see Chapter 5)

current step length is similar to the last previous step length. This means that it is not efficient enough if we start backtracking each time with the same initial step length of $t = 1$, we are actually wasting a lot of time by doing that.

With the above discussions taken into account, we propose a modified backtracking line search algorithm which is more suitable for our problem.

Modified Backtracking Line Search Algorithm

Given $0 < \alpha \leq 0.5, 0 < \beta < 1$.

At step k, given a descent direction $-\nabla f(x_k)$ for f:

$t_k := t_{k-1}$

for $i = 1 : 2$

if $f(x_k - t\nabla f(x_k) > f(x_k) - \alpha t||\nabla f(x_k)||^2 \qquad t_k := \beta t_k$

else $\qquad t_k = t_k/\beta$

end if

end for

Chapter 5

Numerical Results

We have many ways of solving (3.7), which is a smooth, convex problem. According to the analysis above, we present 4 algorithms here.

- **Alg0**

  We fix p during the optimization process. At the same time we use Nesterov's algorithm. The step length is given by $1/L$, and L is obtained by Lemma 1, i.e $L = 2/p$.

- **Alg1**

  We fix p during the optimization process, We still use the Nesterov's framework, but instead of using $1/L$ as the step length, the step length is obtained from the modified backtracking line search.

- **Alg2**

  We decrease p as we described in Section 4.1. At the same time we use Nesterov's algorithm. The step length is given by $1/L$, and L is obtained by Lemma 1, i.e $L = 2/p$. Thus, the Lipschiz constant increases gradually as we are approaching the optimum.

- **Alg3**

  We decrease p as we described in Section 4.1. We still use the Nesterov's

framework, but instead of using $1/L$ as the step length, the step length is obtained from the modified backtracking line search.

Now we are going to implement the above 4 algorithms with Pentium IV Processor 3.0G personal computer with 1GB memory. All codes are written in Matlab 7.0. For all the 4 algorithms, $\epsilon_1 = 0.001$, $\epsilon_2 = 0.01$, eventually p will goes to $\frac{\epsilon_1}{1+\ln m}$. For Alg1 and Alg3 backtracking parameter $\alpha = 0.3$, $\beta = 0.7$. For Alg2, $\sigma = 0.9$. For Alg3, $\sigma = 0.99$.

The test problems are generated randomly. We use the following pseudo-random sequence:

$$\psi_0 = 7, \psi_{i+1} = (445\psi_i + 1) \mod 4096, \quad i = 1, 2, \cdots$$
$$\bar{\psi_i} = \frac{\psi_i}{40.96}, \quad i = 1, 2, \cdots \tag{5.1}$$

The elements of $c_i, i = 1, 2, \cdots, m$, are successively set to $\bar{\psi_1}, \bar{\psi_2}, \cdots$, in the order:

$$r_1, c_1(1), c_1(2), \cdots, c_1(n), r_2, c_2(1), \cdots, c_2(n), \cdots, r_m, c_m(1), \cdots, c_m(n)$$

The stopping criteria for all the 4 algorithms are

1. $p = 0.1\epsilon_1$ (for algorithms with a decreasing p)
2. $||\nabla f(x^k)||_2 \leq \epsilon_1$ or $\frac{f(x^k)-f(x^{k+1})}{f(x^k)} \leq \epsilon_1$

The maximum number of iterations is 5000.

The numerical results are obtained and summarized in Table 1 and Table 2. In these tables, **n** and **m** denote the dimension of the Euclidean space and the number of balls, respectively, **Obj Value** denotes the value of the objective function at the final iteration, **Iter** denotes the number of iterations, **Time** denotes the CPU time in second for solving each problem.

The numerical experiments in Table 1 and Table 2 show that Alg3 performs the

best among the 4 algorithms. And the cpu time used by Alg3 is growing almost linearly with n. Furthermore, we can see that the scheme of decreasing p plays a critical rule in the performance of the whole algorithm. With parameter p fixed, Alg0 and Alg1 both perform badly among all the algorithms. From our numerical results we can also say that backtracking line search scheme really helps since the algorithms with back-tracking scheme (Alg3) perform better than the one with just using the step length of $1/L$ (Alg2) in both accuracy and speed.

We also show the converge processes of Alg2 and Alg3 in Fugure5.1 with a relatively smaller problem size ($n = 400$, $m = 500$), it is clear that the convergence rate of Alg3 is faster.

Problem		Obj Value			
n	m	Alg0	Alg1	Alg2	Alg3
400	1000	1.143019442e3	0.753746632e3	0.679606446e3	0.679603838e3
800	1000	1.615030305e3	1.03045247e3	0.916976776e3	0.916975382e3
1200	1000	1.978822015e3	1.247183943e3	1.100685012e3	1.100687195e3
1600	1000	2.279018895e3	1.420075406e3	1.253338679e3	1.253330549e3
2000	1000	2.550887995e3	1.586512287e3	1.390647515e3	1.390646052e3

Table 1: Objective function value at the final iteration.

Problem		Alg0		Alg1		Alg2		Alg3	
n	m	Time	Iter	Time	Iter	Time	Iter	Time	Iter
400	1000	240.6	5000	396.6	5000	236.6	5000	103.1	1319
800	1000	460.8	5000	769.8	5000	459.9	5000	222.5	1453
1200	1000	681.4	5000	1150.9	5000	690.2	5000	402.5	1760
1600	1000	907.9	5000	1525.8	5000	911.8	5000	545.1	1797
2000	1000	1133.7	5000	1916.1	5000	1140.1	5000	818.4	2140

Table 2 : Performance comparison of the four algorithms in terms of total iterations and cpu time

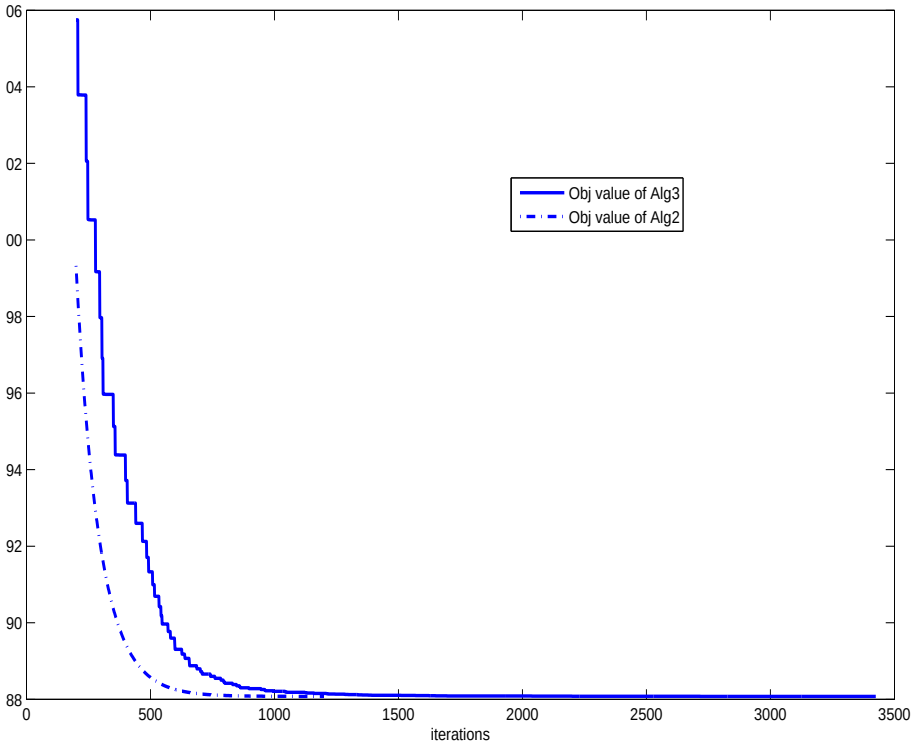

Figure 5.1: Objective Value of Alg2 and Alg3. Solid Curve–Alg3, Dashed Curve–Alg2

Part II

Generalized Convex Quadratic Semidefinite Programming

Chapter 6

Introduction

6.1 QSDP and GQSDP

First we introduce the following convex quadratic semidefinite program (QSDP)

$$(QSDP) \quad \min_X \frac{1}{2} X \cdot \mathcal{Q}(X) + C \cdot X$$
$$\mathcal{A}(X) = b, \quad X \succeq 0 \tag{6.1}$$

where $\mathcal{Q} : S^n \to S^n$ is a given self-adjoint positive semidefinite linear operator in S^n (the "$\cdot$" means the standard trace inner product in space S^n). $\mathcal{A} : S^n \to R^m$ is a linear map and $b \in R^m$. The notation $X \succeq 0$ means that X is positive semidefinte. The Lagrangian dual problem for (6.1) is as follows

$$(QSDD) \quad \max_{X,y,Z} -\frac{1}{2} X \cdot \mathcal{Q}(X) + b^T y$$
$$\mathcal{A}^T(y) - \mathcal{Q}(X) + Z = C, \quad Z \succeq 0 \tag{6.2}$$

Then we introduce another problem, the linearly constrained convex quadratic program (LCCQP), which has the following form

$$(LCCQP) \quad \min\{\frac{1}{2} x^T M x + d^T x : Bx = b, x \in R^n_+\} \tag{6.3}$$

where $x \in R^n_+$ means $x \in R^n$ and $x \geq 0$.

Combining (QSDP) and (LCCQP), we get a more general problem which we call

it generalized quadratic semidefinite program (GQSDP)

$$(GQSDP) \quad \min_{X,u} \frac{1}{2} X \cdot \mathcal{Q}(X) + C \cdot X + \frac{1}{2} u^T M u + d^T u$$
$$\mathcal{A}(X) + Bu = b$$
$$X \succeq 0, u \geq 0, X \in S_n, u \in R^p \tag{6.4}$$

6.2 Application

A good application of (QSDP) is the computation of nearest correlation matrix problem(see [15], [19], [14], [9], [11]). Given a data matrix $D \in S_n$ (not necessarily semidefinite) and a self-adjoint linear operator $\mathcal{L}$ on S_n, we want to solve

$$\min_X \{\frac{1}{2} ||\mathcal{L}(X - D)||_F^2 : \text{diag}(X) = e, X \succeq 0\} \tag{6.5}$$

Here e is a vector of ones. If we transform the above (6.5) to the standard (QSDP) formulation like (6.1), we get $\mathcal{Q} = \mathcal{L}^2$ and $C = -\mathcal{L}^2(D)$. Later in Chapter 9 we will solve this problem with an interior point method and show some numerical results.

Another application is the nearest Euclidean distance matrix (EDM) problem, it can also be formulated as (QSDP), (see [2]).

6.3 Related Work and Outline

After the formulation of (LCCQP), (QSDP) and (GQSDP), in this part we are going to focus on the computational issues of these problems, especially on (QSDP) and (GQSDP).

In [13] Luca discussed the preconditioning of the LCCQP. In [10] Toh proposed primal-dual path following algorithm to solve (QSDP). Toh also analyzed the structure of the augmented system arisen from (QSDP) and designed 3 preconditioners to help to solve the linear system.

The work in this part is based on the above materials. We are going to solve the (GQSDP) by a primal-dual path following method. In Chapter 7, we derive the augmented system for (GQSDP). Then in Section 8.1 we analyze the resulting augmented system. In order to solve the ill-conditioned system efficiently, in Section 8.2, we propose 2 preconditioners and prove some useful results about the spectral distribution after preconditioning. In Chapter 9, we give some numerical results based on the primal-dual path following algorithms in [10] to solve (QSDP).

Notation: We use the following notation and terminology. For an integer n, we let $\bar{n} = n(n+1)/2$. Given $U \in R^{q\times l}$, $V \in R^{q\times n}$, the symmetrized Kronecker product $U \circledast V$ is the linear map from $R^{n\times l}$ to S^q defined by $U \circledast V(M) = (VMU^T + UM^TV^T)/2$. For $U \in R^{q\times l}$, $V \in R^{q\times n}$, the Kronecker product $U \otimes V$ is the linear map from $R^{n\times l}$ to $R^{q\times p}$ defined by $U \otimes V(M) = VMU^T$. We use $U \circ V$ to denote the Hadamard product of two matrices U, V with same dimension. The set of symmetric positive semidefinite (definite) matrices is denoted by $\mathcal{S}^n_+(\mathcal{S}^n_{++})$. We use $||\cdot||_2$ (sometimes $||\cdot||$)to denote vector 2-norm or matrix 2-norm, $||\cdot||_F$ to denote the Frobenius norm. The notation $x = \Theta(\nu)$ means that there exist constants $c_1, c_2 > 0$ independent of ν such that $c_1\nu \le x \le c_2\nu$. We denote the identity matrix or operator of dimension d by I_d.

Given a self-adjoint linear operator $\mathcal{V}$ (or a matrix) defined on a finite dimensional inner product space, the set of eigenvalues of $\mathcal{V}$ is defined by $\mathrm{eig}(\mathcal{V})$. The largest and smallest eigenvalues of $\mathcal{V}$ in magnitudes are denoted by $\lambda_{\max}(\mathcal{V})$ and $\lambda_{\min}(\mathcal{V})$, respectively. For a linear map $\mathcal{T} : (\mathcal{X}, \bullet) \to (\mathcal{Y}, \bullet)$, where $\mathcal{X} = R^{k\times l}$ or $\mathcal{S}^l$ and $\mathcal{Y} = R^{p\times q}$ or $\mathcal{S}^q$, we define $||\mathcal{T}|| = \max\{||\mathcal{T}(M)||_F : ||M||_F \le 1\}$, the adjoint of $\mathcal{T}$ is defined by $\mathcal{T}^T$.

Chapter 7

Computation of Newton Direction in GQSDP

Recall the standard form of (GQSDP)

$$\begin{aligned}(GQSDP)\quad \min_{X,u} \frac{1}{2}X\cdot\mathcal{Q}(X)+C\cdot X+\frac{1}{2}u^TMu+d^Tu\\ \mathcal{A}(X)+Bu=b\\ X\succeq 0, u\geq 0, X\in S_n, u\in R^p\end{aligned}$$

The Lagrangian of (GQSDP) is given by

$$L(X,Z,u,y,t)=\frac{1}{2}X\cdot\mathcal{Q}(X)+C\cdot X+\frac{1}{2}u^TMu+d^Tu+y^T(b-\mathcal{A}(X)-Bu)-Z\cdot X-t^Tu$$

Note that with fixed y, Z and t, L is convex in X and u. In order to get the minimum of L, the derivative of X and u must vanish. Thus we have

$$\nabla_X L=\mathcal{Q}(X)+C-\mathcal{A}^T(y)-Z=0$$

$$\nabla_u L=Mu+d-B^Ty-t=0$$

The Lagrangian dual problem of (GQSDP) is given by

$$\begin{aligned}(GQSDD)\quad \max_{X,u,y,t} -\frac{1}{2}X\cdot\mathcal{Q}(X)-\frac{1}{2}u^TMu+b^Ty\\ \mathcal{Q}(X)+C-\mathcal{A}^T(y)=Z\\ Mu+d-B^Ty=t\\ Z\succeq 0, t\geq 0, Z\in S_n, t\in R^p, y\in R^m\end{aligned}\tag{7.1}$$

The interior point method we are using for (GQSDP) is a primal-dual path following method. It is based on the perturbed KKT conditions associated with the primal-dual pair (6.4) and (7.1), which are given by

$$\begin{aligned} -A^T(y) + Z - \mathcal{Q}(X) &= C \\ B^T y + t - Mu &= d \\ XZ &= \mu_1 I \\ Ut &= \mu_2 e \\ X \succeq 0, Z \succeq 0, M \succeq 0, u \geq 0, t \geq 0, X, Z \in S_n, u, t \in R^p & \end{aligned} \tag{7.2}$$

where U=diag$(u_1, u_2, \cdots, u_p)$, e is the vector of ones, μ_1 and μ_2 are parameters to be driven to 0 explicitly.

Let $\rho \geq 0$ be a given constant. By adding the condition

$$-\rho \mathcal{A}^T \mathcal{A}(X) = -\rho \mathcal{A}^T b \tag{7.3}$$

to the first condition in (7.2), we get an equivalent condition:

$$-A^T(y) + Z - \mathcal{Q}_\rho(X) = C_\rho$$

where $\mathcal{Q}_\rho(X) := \mathcal{Q} + \rho \mathcal{A}^T \mathcal{A}$ and $C_\rho := C - \rho \mathcal{A}^T b$. The motivation of considering such a transformation is from Remark 3.1 in [10].

Given the KKT equations for (GQSDP), we can derive the Newton direction now. At a given iterate (X, Z, y, u, t), the search direction $(\Delta X, \Delta Z, \Delta y, \Delta u, \Delta t)$ is the solution of the following Newton system:

$$\begin{aligned} -\mathcal{Q}_\rho(\Delta X) + \mathcal{A}^T(\Delta y) + \Delta Z &= R_{ds} := C_\rho - Z - \mathcal{A}^T(y) + \mathcal{Q}_\rho(X) \\ \mathcal{A}(\Delta X) + B\Delta u &= R_c := b - \mathcal{A}(X) - Bu \\ B^T \Delta y + \Delta t - M\Delta u &= R_{dl} := d - B^T y - t + Mu \\ \mathcal{F}_S \Delta X + \mathcal{F}_X \Delta S &= R_{cs} := \sigma \mu_1 I - H_K(XS) \\ U\Delta t + T\Delta u &= R_{cl} := \sigma \mu_2 e - Ut \end{aligned} \tag{7.4}$$

where $\mathcal{F}_X$ and $\mathcal{F}_S$ are linear operators on S^n that depend on the symmetrization scheme $H_K(\cdot)$ chosen, with K being the symmetrization matrix, and $\sigma \in (0,1)$ the centering parameter; for more details, see for example in [16].

By eliminating ΔZ and Δt in (7.4), we can get the following augmented equation:

$$\begin{pmatrix} -\mathcal{Q}_\rho - \mathcal{F}_X^{-1}\mathcal{F}_S & 0 & \mathcal{A}^T \\ 0 & -U^{-1}T - M & B^T \\ \mathcal{A} & B & 0 \end{pmatrix} \begin{pmatrix} \Delta X \\ \Delta u \\ \Delta y \end{pmatrix} = \begin{pmatrix} R_a \\ R_b \\ R_c \end{pmatrix} \tag{7.5}$$

where $R_a = R_{ds} - \mathcal{F}_X^{-1}R_{cs}$, $\quad R_b = R_{dl} - U^{-1}R_{cl}$, $\quad T = \mathrm{diag}(t_1, \cdots, t_p)$. In this thesis, we will consider only the Nesterov-Todd (NT) symmetrization scheme for which $\mathcal{F}_X^{-1}\mathcal{F}_S = W^{-1} \circledast W^{-1}$, where W is the unique symmetric positive semidefinite matrix satisfying $WZW = X$. For later convenience, we define

$$\mathcal{J}_\rho = \begin{pmatrix} -\mathcal{Q}_\rho - \mathcal{F}_X^{-1}\mathcal{F}_S & 0 & \mathcal{A}^T \\ 0 & -F & B^T \\ \mathcal{A} & B & 0 \end{pmatrix}$$

where $F = U^{-1}T + M$.

Chapter 8

Preconditioners for the Augmented Matrix

8.1 Partitioning the Augmented Matrix

We make the following assumptions on (6.4) and (7.1).

Assumption 1. *The problems (GQSDP) and (GQSDD) are strictly feasible and the map* $[\mathcal{A}, B] : S^n \times R^p \to R^m$ *defined by* $[\mathcal{A}, B](X, u) = \mathcal{A}(X) + Bu$ *is surjective.*

The Assumption 1 guarantees the existence and uniqueness of the system (7.5). Also the solution (X, Z, y, u, t) on the central path tends to the optimal solution when μ_1 and μ_2 tend to zero.

Assumption 2. *Strict complementarity holds for optimal solution* $(X^*, Z^*, y^*, u^*, t^*)$, *the ranks of* X^* *and* Z^* *sum to* n, *the ranks of* $U^* = \mathrm{diag}(u^*)$ *and* $T^* = \mathrm{diag}(t^*)$ *sum to* p.

At each step of the interior point method, if X and Z satisfy $XZ = \mu_1 I$, then X and Z commute. Thus there is an orthogonal matrix P that simultaneously diagonalizes X and Z, i.e.

$$X = P\Lambda P^T \quad Z = P\Sigma P^T$$

Further, we define $\mathcal{P} = P \circledast P$, $\tilde{\mathcal{A}} = \mathcal{A}\mathcal{P}$ and $\tilde{\mathcal{Q}}_\rho = \mathcal{P}^T \mathcal{Q}_\rho \mathcal{P}$. Assume the strict complementary condition holds, we partition P with P_1 and P_2, denoting the first r and the last $n - r$ columns of P respectively, where r and s are the ranks of optimal solution X^* and Z^*, see [10]. We further define $\mathcal{P}_1 = P_1 \circledast P_1 : S^r \to S^n$, $\mathcal{P}_2 = 2(P_1 \circledast P_2) : R^{r\times s} \to S^n$, and $\mathcal{P}_3 = P_2 \circledast P_2 : S^s \to S^n$. (Note that at the optimum $(X^*, Z^*, y^*, u^*, t^*)$, P becomes P^*, where P^* simultaneously diagnalizes X^* and Z^* in the sense that $X^* = P^*\Lambda^*(P^*)^T$ and $Z^* = P^*\Sigma^*(P^*)^T$, thus $\mathcal{P}^* = P^* \circledast P^*$ and $\mathcal{P}_1^* = P_1^* \circledast P_1^*$, etc.) Under the Assumptions 1 and 2, on the central path, the eigenvalue decomposition of W^{-1} must have the form

$$W^{-1} = PDP^T = P_1 D_1 P_1^T + P_2 D_2 P_2^T$$

It can be shown that $D_1 = \mathrm{diag}(d_1) \in R^{r\times r}$, $P_1^k \in R^{n\times r}$ correspond to the small eigenvalues of the order $\Theta(\sqrt{\mu}_1)$, and $D_2 = \mathrm{diag}(d_2) \in R^{s\times s}, P_2 \in R^{n\times s}$ correspond to the large eigenvalues of the order $\Theta(1/\sqrt{\mu}_1)$, see [10]. Recall that the notation $\gamma = \Theta(\sqrt{\mu}_1)$ means that there are constants $c_1, c_2 > 0$ such that $c_1\sqrt{\mu}_1 \le \gamma \le c_2\sqrt{\mu}_1$ for all μ_1 (we decrease μ_1 in every iteration). Further, the following decomposition also holds [16]

$$W^{-1} \circledast W^{-1} = (P \circledast P)(D \circledast D)(P \circledast P)^T =: \mathcal{P}\mathcal{D}\mathcal{P}^T \tag{8.1}$$

where $\mathcal{P} = P \circledast P$ and $\mathcal{D} = D \circledast D$. Let $\mathcal{D}_1 = D_1 \circledast D_1$, $\mathcal{D}_2 = D_2 \otimes D_1$ and $\mathcal{D}_3 = D_2 \circledast D_2$. Thus we can write $\mathcal{D} = \mathrm{diag}(\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3)$. Then it is easy that the following decomposition holds

$$\mathcal{J}_\rho = \begin{pmatrix} \mathcal{P} & 0 & 0 \\ 0 & L & 0 \\ 0 & 0 & I_m \end{pmatrix} \begin{pmatrix} -\mathcal{D} - \tilde{\mathcal{Q}}_\rho & 0 & \tilde{\mathcal{A}}^T \\ 0 & -\tilde{F} & \tilde{B}^T \\ \tilde{\mathcal{A}} & \tilde{B} & 0 \end{pmatrix} \begin{pmatrix} \mathcal{P}^T & 0 & 0 \\ 0 & L^T & 0 \\ 0 & 0 & I_m \end{pmatrix} \tag{8.2}$$

where $\tilde{F} = L^T(U^{-1}T + M)L$, $\tilde{B} = L^T B L$ and L is permutation matrix. We also define

$$\tilde{\mathcal{J}}_\rho = \begin{pmatrix} -\mathcal{D} - \tilde{\mathcal{Q}}_\rho & 0 & \tilde{\mathcal{A}}^T \\ 0 & -\tilde{F} & \tilde{B}^T \\ \tilde{\mathcal{A}} & \tilde{B} & 0 \end{pmatrix}$$

It is shown in [10] that $\mathcal{J}_\rho$ is generally an ill-conditioned matrix, thus constructing a preconditioner for $\mathcal{J}_\rho$ is the main task. First We focus on

$$\tilde{F} = L^T(M + U^{-1}T)L = L^TML + L^TU^{-1}L \cdot L^TTL$$

For future convenience, we define $\tilde{U} = L^TUL$, $\tilde{T} = L^TTL$ and $\tilde{M} = L^TML$. Thus we can choose the a permutation matrix L in (8.2) so that

$$\tilde{U} = \mathrm{diag}(\tilde{u}_1, \cdots, \tilde{u}_p) \quad \tilde{T} = \mathrm{diag}(\tilde{t}_1, \cdots, \tilde{t}_p)$$

satisfy

$$\tilde{u}_1 \geq \tilde{u}_2 \geq \cdots \geq \tilde{u}_p > 0 \qquad 0 < \tilde{t}_1 \leq \tilde{t}_2 \leq \cdots \leq \tilde{t}_p$$

Assume that at the optimum point, strict complementarity holds, then $\tilde{U}$ becomes $\tilde{U}^*$ and $\tilde{T}$ becomes $\tilde{T}^*$. We must have $\tilde{u}_i^*\tilde{t}_i^* = 0$ $(i = 1 \cdots p)$, and

$$\tilde{u}_1^* \geq \cdots \geq \tilde{u}_i^* > \tilde{u}_{i+1}^* = \cdots = \tilde{u}_p^* = 0 \qquad 0 = \tilde{t}_1^* = \cdots = \tilde{t}_{p-j}^* < \tilde{t}_{p-j+1}^* \leq \cdots \leq \tilde{t}_p^*$$

Here i and j are ranks of $\tilde{U}^*$ and $\tilde{T}^*$, respectively, and satisfy $i + j = p$. Thus we make the partition

$$\tilde{U} = \begin{pmatrix} \tilde{U}_1 & \\ & \tilde{U}_2 \end{pmatrix} \qquad \tilde{T} = \begin{pmatrix} \tilde{T}_1 & \\ & \tilde{T}_2 \end{pmatrix}$$

where $\tilde{U}_1 \in S_i$, $\tilde{U}_2 \in S_j$ and $\tilde{T}_1 \in S_j$, $\tilde{T}_2 \in S_i$. We can also determine that $\tilde{U}_1$ and $\tilde{T}_2$ correspond to the eigenvalues of the order of $\Theta(1)$, while $\tilde{U}_2$ and $\tilde{T}_1$ correspond to the eigenvalues of the order of $\Theta(\mu_2)$.

After the above partition, we have

$$\tilde{U}^{-1}\tilde{T} = \begin{pmatrix} \tilde{U}_1^{-1}\tilde{T}_1 & \\ & \tilde{U}_2^{-1}\tilde{T}_2 \end{pmatrix}$$

with $\tilde{U}_1^{-1}\tilde{T}_1$ correspond to the eigenvalues of the order of $\Theta(\mu_2)$ and $\tilde{U}_2^{-1}\tilde{T}_2$ correspond to the eigenvalues of the order of $\Theta(\frac{1}{\mu_2})$.

Thus we can write

$$\tilde{F} = L^T M L + \tilde{U}^{-1}\tilde{T} = \tilde{M} + \tilde{U}^{-1}\tilde{T} = \begin{pmatrix} \tilde{M}_1 + \tilde{U}_1^{-1}\tilde{T}_1 & \tilde{M}_2 \\ \tilde{M}_2^T & \tilde{M}_3 + \tilde{U}_2^{-1}\tilde{T}_2 \end{pmatrix}$$

where $\tilde{M} = L^T M L$ and $\tilde{M} = \begin{pmatrix} \tilde{M}_1 & \tilde{M}_2 \\ \tilde{M}_2^T & \tilde{M}_3 \end{pmatrix}$.

Now we consider the similar problem under the framework of (QSDP). First we define the corresponding matrices. In (QSDP), The corresponding part for $\tilde{\mathcal{J}}_\rho$ is $\tilde{\mathcal{B}}_\rho$, which can be defined as

$$\tilde{\mathcal{B}}_\rho = \begin{pmatrix} -\mathcal{D} - \tilde{\mathcal{Q}}_\rho & \tilde{\mathcal{A}}^T \\ \tilde{\mathcal{A}} & 0 \end{pmatrix} \tag{8.3}$$

We also define

$$\mathcal{B}_\rho = \begin{pmatrix} -\mathcal{F}_X^{-1}\mathcal{F}_S - \mathcal{Q}_\rho & \mathcal{A}^T \\ \mathcal{A} & 0 \end{pmatrix}$$

And thus we have

$$\mathcal{B}_\rho = \begin{pmatrix} \mathcal{P} & 0 \\ 0 & I_m \end{pmatrix} \tilde{\mathcal{B}}_\rho \begin{pmatrix} \mathcal{P}^T & 0 \\ 0 & I_m \end{pmatrix} \tag{8.4}$$

8.2 Preconditioners

8.2.1 Preconditioner I

It is easy to see that matrix $\mathcal{J}_\rho$ is ill-conditioned. In [10] Toh has proven that the $||\mathcal{B}_\rho|| \to \infty$ when $\mu_1 \to 0$. We can also see from the above analysis that when $\mu_2 \to 0$, it makes $\tilde{U}_1^{-1}\tilde{T}_1 \to 0$ and $\tilde{U}_2^{-1}\tilde{T}_2 \to \infty$, which shows that $\tilde{F}$ (thus F) is also badly conditioned. Therefore, in order to apply an iterative method to solve the Newton system, we have to rely on preconditioning technique. The first preconditioner we are going to propose is

$$\Omega_\rho = \begin{pmatrix} -\mathcal{P}\Gamma_\rho\mathcal{P}^T & 0 & \mathcal{A}^T \\ 0 & -L\Phi L^T & B^T \\ \mathcal{A} & B & 0 \end{pmatrix} \tag{8.5}$$

where $\Gamma_\rho = \mathrm{diag}(\mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3)$ satisfying the condition

$$\underline{\sigma}_1 I \preceq \mathcal{H}_1 \preceq \bar{\sigma}_1 I, \quad \underline{\sigma}_2 I \preceq \mathcal{H}_2 \preceq \bar{\sigma}_2 I, \quad \mathcal{D}_3 \preceq \mathcal{H}_1 \preceq \mathcal{D}_3 + \bar{\sigma}_3 I \tag{8.6}$$

and $\Phi = \mathrm{diag}(\Phi_1, \Phi_2)$ satisfying the condition

$$\underline{\lambda}_1 I \preceq \Phi_1 \preceq \bar{\lambda}_1 I, \quad \tilde{U}_2^{-1}\tilde{T}_2 \preceq \Phi_2 \preceq \bar{\lambda_2} I \tag{8.7}$$

Note that we have the following decomposition

$$\Omega_\rho = \begin{pmatrix} \mathcal{P} & 0 & 0 \\ 0 & L & 0 \\ 0 & 0 & I_m \end{pmatrix} \begin{pmatrix} -\Gamma_\rho & 0 & \tilde{\mathcal{A}}^T \\ 0 & -\Phi & \tilde{B}^T \\ \tilde{\mathcal{A}} & \tilde{B} & 0 \end{pmatrix} \begin{pmatrix} \mathcal{P}^T & 0 & 0 \\ 0 & L^T & 0 \\ 0 & 0 & I_m \end{pmatrix} \tag{8.8}$$

And we denote

$$\tilde{\Omega}_\rho = \begin{pmatrix} -\Gamma_\rho & 0 & \tilde{\mathcal{A}}^T \\ 0 & -\Phi & \tilde{B}^T \\ \tilde{\mathcal{A}} & \tilde{B} & 0 \end{pmatrix} \tag{8.9}$$

We are going to analyze the spectrum of the preconditioned matrix under this preconditioner, i.e to analyze the spectrum of the matrix $\Omega_\rho^{-1}\mathcal{J}_\rho$. From the decomposition (8.2) and (8.8), we can see that it is equivalent to analyzing the matrix $\tilde{\Omega}_\rho^{-1}\tilde{\mathcal{J}}_\rho$.

Lemma 2. *Suppose $U \in S^p$ is symmetric positive definite, and $V \in R^{m\times p}$ has full row rank. Let $G = [-U, V^T; V, 0]$. Suppose $\hat{U}$ is a symmetric positive definite approximation of U, and we consider $\hat{G} = [-\hat{U}, V^T; V, 0]$ as a preconditioner for G. Then $\hat{G}^{-1}G$ has $2m$ eigenvalues located at 1, and the remaining $p-m$ eigenvalues are those of the matrix $Z^T\hat{U}^{-\frac{1}{2}}U\hat{U}^{-\frac{1}{2}}Z$, where $Z \in R^{p\times(p-m)}$ is a matrix whose columns form an orthogonal basis of $\mathcal{N}(V\hat{U}^{-\frac{1}{2}})$.*

Proof. See Theorem 2 in [12]. □

The spectrum analysis of preconditioned matrix in (GQSDP) is based on the one with (QSDP), thus we will review some similar results in (QSDP) . First, the

preconditioner similar to (8.5) is

$$\Psi_\rho = \begin{pmatrix} -\mathcal{P}\Gamma_\rho\mathcal{P}^T & \mathcal{A}^T \\ \mathcal{A} & 0 \end{pmatrix}$$

We also have

$$\Psi_\rho = \begin{pmatrix} \mathcal{P} & 0 \\ 0 & I_m \end{pmatrix} \begin{pmatrix} -\Gamma_\rho & \tilde{\mathcal{A}}^T \\ \tilde{\mathcal{A}} & 0 \end{pmatrix} \begin{pmatrix} \mathcal{P}^T & 0 \\ 0 & I_m \end{pmatrix} \tag{8.10}$$

and

$$\tilde{\Psi}_\rho = \begin{pmatrix} -\Gamma_\rho & \tilde{\mathcal{A}}^T \\ \tilde{\mathcal{A}} & 0 \end{pmatrix}$$

In order to make further analysis, we make two assumptions on (QSDP).

Assumption 3. *The problem (QSDP) and (QSDD) are strictly feasible and that $\mathcal{A}$ is surjective.*

Assumption 4. *Strict complementary holds for (X^*, y^*, Z^*) in the sense of Alizadeh, Haeberly, and Overton [6].*

In [10], Toh gives the spectrum analysis for the preconditioned matrix $\Psi_\rho^{-1}\mathcal{B}_\rho$ for (QSDP), which can be stated by the following lemma.

Lemma 3. *Suppose that Assumptions 3 and 4 hold. Let $\{Z^l = [Z_1^l; Z_2^l; Z_3^l]\}_{l=1}^{\bar{n}-m}$ be an orthogonal set in $S^r \times R^{r\times s} \times S^s$ that form a basis of $\mathcal{N}(\tilde{\mathcal{A}}^k\Gamma_\rho^{-\frac{1}{2}})$.*
Consider the following matrix

$$G_\rho = \mathcal{Z}^T\Gamma_\rho^{-\frac{1}{2}}(\mathcal{D} + \tilde{\mathcal{Q}}_\rho)\Gamma_\rho^{-\frac{1}{2}}\mathcal{Z}$$

1. *Suppose $\mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3$ in Γ_ρ satisfy the condition (8.6) and $(\mathcal{P}_1^*)^T\mathcal{Q}_\rho\mathcal{P}_1^* \succ 0$. Then there exist positive constants c_1, c_2 such that for iteration number sufficiently large,*

$$\mathrm{eig}(G_\rho) \subset [c_1, c_2] \tag{8.11}$$

2. *Suppose $\mathcal{D} \preceq \Gamma_\rho \preceq \mathcal{D} + \tilde{\mathcal{Q}}_\rho$. Then*

$$\mathrm{eig}(G_\rho) \subset 1 + \left[0, ||Q_\rho|| \max\{\underline{\sigma}_1^{-1}, \underline{\sigma}_2^{-1}, \Theta(\mu_1)\}\right]. \tag{8.12}$$

3. *Suppose* $\underline{\beta}(\mathcal{D}+\tilde{\mathcal{Q}}_\rho) \preceq \Gamma_\rho \preceq \bar{\beta}(\mathcal{D}+\tilde{\mathcal{Q}}_\rho)$ *for some constants* $\underline{\beta}, \bar{\beta} > 0$*, Then*

$$\mathrm{eig}(G_\rho) \subset \left[\bar{\beta}^{-1}, \underline{\beta}^{-1}\right]$$

Proof. See Theorem 4.4 in [10]. □

And the following theorem states the spectrum property of preconditioned matrix $\Omega_\rho^{-1}\mathcal{J}_\rho$.

Theorem 3. *Suppose that Assumption 1 and 2 hold. Let* $\mathcal{Z}$ *be an orthogonal set that form a basis of* $\mathcal{N}(\tilde{\mathcal{A}}\Gamma^{-\frac{1}{2}}, \tilde{B}\Psi^{-\frac{1}{2}})$.

1. *The preconditioned matrix* $\Omega_\rho^{-1}\mathcal{J}_\rho$ *has* $2m$ *eigenvalues located at* 1*. The remaining* $p+\bar{n}-m$ *eigenvalues are those of the matrix*

$$\mathcal{V}_\rho = \mathcal{Z}^T \begin{pmatrix} \Gamma_\rho^{-\frac{1}{2}}(\mathcal{D}+\tilde{\mathcal{Q}}_\rho)\Gamma_\rho^{-\frac{1}{2}} & 0 \\ 0 & \Phi^{-\frac{1}{2}}\tilde{F}\Phi^{-\frac{1}{2}} \end{pmatrix} \mathcal{Z} \tag{8.13}$$

2. *Suppose* $\mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3$ *in* Γ_ρ *satisfy the condition (8.6) and* $(\mathcal{P}_1^*)^T\mathcal{Q}_\rho\mathcal{P}_1^* \succ 0$*. And* $\Phi = \mathrm{diag}(\Phi_1, \Phi_2)$ *satisfy the condition (8.7),* $\tilde{M}_1$ *converge to* $\tilde{M}_1^*$ *(*$\tilde{M}_1^* \succ 0$*). Then there exist positive constants* c_1 *and* c_2 *such that for iteration number sufficiently large, we have*

$$\mathrm{eig}(\mathcal{V}_\rho) \subset [c_1, c_2] \tag{8.14}$$

3. *Suppose* $\mathcal{D} \preceq \Gamma_\rho \preceq \mathcal{D}+\tilde{\mathcal{Q}}_\rho$*, and* $\tilde{U}^{-1}\tilde{T} \preceq \Phi \preceq \tilde{U}^{-1}\tilde{T}+M$*, denote* $b_1 = ||Q_\rho||\max\{\underline{\sigma}_1^{-1}, \underline{\sigma}_2^{-1}, \Theta(\mu_1)\}$ *and* $b_2 = ||M||\max\{\underline{\lambda}_1^{-1}, \Theta(\mu_2)\}$*. Then*

$$\mathrm{eig}(\mathcal{V}_\rho) \subset 1 + [0, \max(b_1, b_2)] \tag{8.15}$$

4. *Suppose* $\underline{\beta}(\mathcal{D}+\tilde{\mathcal{Q}}_\rho) \preceq \Gamma_\rho \preceq \bar{\beta}(\mathcal{D}+\tilde{\mathcal{Q}}_\rho)$ *for some constants* $\underline{\beta}, \bar{\beta} > 0$*,* $\underline{\gamma}(\tilde{U}^{-1}\tilde{T}+\tilde{M}) \preceq \Phi \preceq \bar{\gamma}(\tilde{U}^{-1}\tilde{T}+\tilde{M})$ *for some constants* $\underline{\gamma}, \bar{\gamma} > 0$*, Then*

$$\mathrm{eig}(\mathcal{V}_\rho) \subset [\min(\bar{\gamma}^{-1}, \bar{\beta}^{-1}), \max(\underline{\gamma}^{-1}, \underline{\beta}^{-1})] \tag{8.16}$$

Proof.

1. The result can be easily implied by Lemma 2.

2. It can be shown that $\mathcal{V}_\rho$ can be written as follows (see the proof of Lemma 3)

$$\mathcal{V}_\rho = \mathcal{Z}^T \begin{pmatrix} \mathrm{diag}(\mathcal{H}_1,\mathcal{H}_2)^{-\frac{1}{2}}\Upsilon_\rho \mathrm{diag}(\mathcal{H}_1,\mathcal{H}_2)^{-\frac{1}{2}} & 0 & 0 \\ 0 & I_{\bar{s}} & 0 \\ 0 & 0 & \Phi^{-\frac{1}{2}}\tilde{F}\Phi^{-\frac{1}{2}} \end{pmatrix} \mathcal{Z} + O(\sqrt{\mu_1}||\mathcal{Q}_\rho||)$$

where

$$\Upsilon_\rho = \begin{pmatrix} \mathcal{D}_1 + \mathcal{P}_1^T\mathcal{Q}_\rho\mathcal{P}_1 & \mathcal{P}_1^T\mathcal{Q}_\rho\mathcal{P}_2 \\ \mathcal{P}_2^T\mathcal{Q}_\rho\mathcal{P} & \mathcal{D}_2 + \mathcal{P}_2^T\mathcal{Q}_\rho\mathcal{P}_2 \end{pmatrix}$$

Also, $\Phi^{-\frac{1}{2}}\tilde{F}\Phi^{-\frac{1}{2}}$ can be written as follows

$$\Phi^{-\frac{1}{2}}\tilde{F}\Phi^{-\frac{1}{2}} = \begin{pmatrix} \Phi_1^{-\frac{1}{2}}(\tilde{U}_1^{-1}\tilde{T}_1 + \tilde{M}_1)\Phi_1^{-\frac{1}{2}} & \Phi_1^{-\frac{1}{2}}\tilde{M}_2\Phi_2^{-\frac{1}{2}} \\ \Phi_2^{-\frac{1}{2}}\tilde{M}_2^T\Phi_1^{-\frac{1}{2}} & \Phi_2^{-\frac{1}{2}}(\tilde{U}_2^{-1}\tilde{T}_2 + \tilde{M}_3)\Phi_2^{-\frac{1}{2}} \end{pmatrix}$$

$$= \begin{pmatrix} \Phi_1^{-\frac{1}{2}}(\tilde{U}_1^{-1}\tilde{T}_1 + \tilde{M}_1)\Phi_1^{-\frac{1}{2}} & 0 \\ 0 & I_j \end{pmatrix} + O(\sqrt{\mu_2}||M||)$$

Therefore, we have

$$V_\rho = \mathcal{Z}^T \begin{pmatrix} \Upsilon & 0 & 0 & 0 \\ 0 & I_{\bar{s}} & 0 & 0 \\ 0 & 0 & \Xi & 0 \\ 0 & 0 & 0 & I_j \end{pmatrix} \mathcal{Z} + O(\sqrt{\mu_2}||M||) + O(\sqrt{\mu_1}||\mathcal{Q}_\rho||)$$

where $\Upsilon := \mathrm{diag}(\mathcal{H}_1,\mathcal{H}_2)^{-\frac{1}{2}}\Upsilon_\rho\mathrm{diag}(\mathcal{H}_1,\mathcal{H}_2)^{-\frac{1}{2}}$, $\Xi := \Phi_1^{-\frac{1}{2}}(\tilde{U}_1^{-1}\tilde{T}_1 + \tilde{M}_1)\Phi_1^{-\frac{1}{2}}$. If condition 8.7 is satisfied, then

$$\Phi_1^{-\frac{1}{2}}(\tilde{U}_1^{-1}\tilde{T}_1 + \tilde{M}_1)\Phi_1^{-\frac{1}{2}} \succeq \bar{\lambda}_1^{-1}(\tilde{U}_1^{-1}\tilde{T}_1 + \tilde{M}_1) \tag{8.17}$$

From (8.17), as μ_2 is driven to 0, $\tilde{U}_1^{-1}\tilde{T}_1$ approaches 0, and $\tilde{M}_1$ converges to M_1^* ($M_1^* \succ 0$). So there exists a constant $d_1 > 0$ such that

$$\lambda_{\min}(\Phi_1^{-\frac{1}{2}}(\tilde{U}_1^{-1}\tilde{T}_1 + \tilde{M}_1)\Phi_1^{-\frac{1}{2}}) \geq d_1 > 0 \tag{8.18}$$

Similarly,

$$\Phi_1^{-\frac{1}{2}}(\tilde{U}_1^{-1}\tilde{T}_1+\tilde{M}_1)\Phi_1^{-\frac{1}{2}} \preceq \underline{\lambda}_1^{-1}(\tilde{U}_1^{-1}\tilde{T}_1+\tilde{M}_1) \tag{8.19}$$

and there exists constant $d_2 > 0$ such that

$$\lambda_{\max}(\Phi_1^{-\frac{1}{2}}(\tilde{U}_1^{-1}\tilde{T}_1+\tilde{M}_1)\Phi_1^{-\frac{1}{2}}) \leq d_2 < \infty \tag{8.20}$$

By Lemma 3, we know that there exist constant $c_1, c_2 > 0$

$$\text{eig}(\text{diag}(\mathcal{H}_1,\mathcal{H}_2)^{-\frac{1}{2}}\Upsilon_\rho \text{diag}(\mathcal{H}_1,\mathcal{H}_2)^{-\frac{1}{2}}) \subset [c_1, c_2] \tag{8.21}$$

Combine (8.18), (8.20) and (8.21), the result is easily followed.

3. We condider the partition $\mathcal{Z} = (\mathcal{Z}_1; \mathcal{Z}_2)$, then (8.13) can be written as

$$\mathcal{V}_\rho = (\mathcal{Z}_1^T, \mathcal{Z}_2^T)\begin{pmatrix} \Gamma_\rho^{-\frac{1}{2}}(\mathcal{D}+\tilde{\mathcal{Q}}_\rho)\Gamma_\rho^{-\frac{1}{2}} & 0 \\ 0 & \Phi^{-\frac{1}{2}}\tilde{F}\Phi^{-\frac{1}{2}} \end{pmatrix}\begin{pmatrix} \mathcal{Z}_1 \\ \mathcal{Z}_2 \end{pmatrix}$$

that is,

$$\mathcal{V}_\rho = (\mathcal{Z}_1^T\Gamma_\rho^{-\frac{1}{2}}(\mathcal{D}+\tilde{\mathcal{Q}}_\rho)\Gamma_\rho^{-\frac{1}{2}}\mathcal{Z}_1, \mathcal{Z}_2^T\Phi^{-\frac{1}{2}}\tilde{F}\Phi^{-\frac{1}{2}}\mathcal{Z}_2)$$

Since we have $\tilde{U}^{-1}\tilde{T} \preceq \Phi \preceq \tilde{U}^{-1}\tilde{T}+\tilde{M} = \tilde{F}$. We can deduct that

$$\mathcal{Z}_2^T\Phi^{-\frac{1}{2}}\tilde{F}\Phi^{-\frac{1}{2}}\mathcal{Z}_2 = I + \mathcal{Z}_2^T\Phi^{-\frac{1}{2}}(\tilde{F}-\Phi)\Phi^{-\frac{1}{2}}\mathcal{Z}_2$$

Since $\tilde{F}-\Phi \succeq 0$, it is clear that $\mathcal{Z}_2^T\Phi^{-\frac{1}{2}}\tilde{F}\Phi^{-\frac{1}{2}}\mathcal{Z}_2 \succeq I$. Further,

$$I + \mathcal{Z}_2^T\Phi^{-\frac{1}{2}}(\tilde{F}-\Phi)\Phi^{-\frac{1}{2}}\mathcal{Z}_2 = I + \mathcal{Z}_2^T\Phi^{-\frac{1}{2}}(\tilde{M}-(\Phi-\tilde{U}^{-1}\tilde{T}))\Phi^{-\frac{1}{2}}\mathcal{Z}_2$$

$$\preceq I + \mathcal{Z}_2^T\Phi^{-\frac{1}{2}}\tilde{M}\Phi^{-\frac{1}{2}}\mathcal{Z}_2 \preceq I + \mathcal{Z}_2^T\Phi^{-\frac{1}{2}}||M||\Phi^{-\frac{1}{2}}\mathcal{Z}_2$$

We denote $\mathcal{Y} = \Phi^{-\frac{1}{2}}\mathcal{Z}_2$, then

$$I = \mathcal{Z}^T\mathcal{Z} = \mathcal{Y}^T\Phi\mathcal{Y} = \begin{pmatrix} \mathcal{Y}_1^T & \mathcal{Y}_2^T \end{pmatrix}\begin{pmatrix} \Phi_1 & 0 \\ 0 & \Phi_2 \end{pmatrix}\begin{pmatrix} \mathcal{Y}_1 \\ \mathcal{Y}_2 \end{pmatrix} = \mathcal{Y}_1^T\Phi_1\mathcal{Y}_1 + \mathcal{Y}_2^T\Phi_2\mathcal{Y}_2$$

$$\succeq \underline{\lambda}_1\mathcal{Y}_1^T\mathcal{Y}_1 + \Theta(1/\mu_2)\mathcal{Y}_2^T\mathcal{Y}_2 \succeq \min(\underline{\lambda}_1, \Theta(1/\mu_2))\mathcal{Y}^T\mathcal{Y}$$

So we can see that

$$\mathcal{Z}_2^T\Phi^{-\frac{1}{2}}\tilde{F}\Phi^{-\frac{1}{2}}\mathcal{Z}_2 \preceq I + \left[0, ||M|| \max\{\underline{\lambda}_1^{-1}, \Theta(\mu_2)\}\right] I \tag{8.22}$$

From(8.12), we can imply that (see the proof of Lemma 3)

$$\mathcal{Z}_1^T \Gamma_\rho^{-\frac{1}{2}} (\mathcal{D} + \tilde{\mathcal{Q}}_\rho) \Gamma_\rho^{-\frac{1}{2}} \mathcal{Z}_1 \preceq I + \left[0, ||Q_\rho|| \max\{\underline{\sigma}_1^{-1}, \underline{\sigma}_2^{-1}, \Theta(\mu_1)\}\right] I \tag{8.23}$$

Combining (8.22) and (8.23), the result follows.

4. Since

$$\underline{\gamma}(\tilde{U}^{-1}\tilde{T} + \tilde{M}) \preceq \Phi \preceq \bar{\gamma}(\tilde{U}^{-1}\tilde{T} + \tilde{M})$$

it implies

$$\underline{\gamma}\Phi^{-\frac{1}{2}}(\tilde{U}^{-1} + \tilde{M})\Phi^{-\frac{1}{2}} \preceq I \preceq \bar{\gamma}\Phi^{-\frac{1}{2}}(\tilde{U}^{-1}T + \tilde{M})\Phi^{-\frac{1}{2}}$$

By multiplying $\mathcal{Z}_2$ and $\mathcal{Z}_2^T$, we get

$$\bar{\gamma}^{-1} I \preceq \mathcal{Z}_2^T \Phi^{-\frac{1}{2}} \tilde{F} \Phi^{-\frac{1}{2}} \mathcal{Z}_2 \preceq \underline{\gamma}^{-1} I \tag{8.24}$$

Combine with part 3 of Lemma 3, we are done.

□

8.2.2 Preconditioner II

The second preconditioner we are going to propose is

$$\Pi_\rho = \begin{pmatrix} -\mathcal{P}\Gamma_\rho \mathcal{P}^T & 0 & \mathcal{A}^T \\ 0 & -E & B^T \\ \mathcal{A} & B & 0 \end{pmatrix} \tag{8.25}$$

where $E = \mathrm{diag}(M) + U^{-1}T$. Note we have the following decomposition

$$\Pi_\rho = \begin{pmatrix} \mathcal{P} & 0 & 0 \\ 0 & I_p & 0 \\ 0 & 0 & I_m \end{pmatrix} \begin{pmatrix} -\Gamma_\rho & 0 & \tilde{\mathcal{A}}^T \\ 0 & -E & B^T \\ \tilde{\mathcal{A}} & B & 0 \end{pmatrix} \begin{pmatrix} \mathcal{P}^T & 0 & 0 \\ 0 & I_p & 0 \\ 0 & 0 & I_m \end{pmatrix} \tag{8.26}$$

Theorem 4. *Suppose that Assumption 1 and 2 hold. Let $\mathcal{Z}$ be an orthogonal set that form a basis of $\mathcal{N}(\tilde{\mathcal{A}}\Gamma^{-\frac{1}{2}}, BE^{-\frac{1}{2}})$.*

1. *The preconditioned matrix* $\Pi_\rho^{-1}\mathcal{J}_\rho$ *has* $2m$ *eigenvalues located at* 1. *The remaining* $p+\bar{n}-m$ *eigenvalues are those of the matrix*

$$\mathcal{V}_\rho = \mathcal{Z}^T \begin{pmatrix} \Gamma_\rho^{-\frac{1}{2}}(\mathcal{D}+\tilde{\mathcal{Q}}_\rho)\Gamma_\rho^{-\frac{1}{2}} & 0 \\ 0 & E^{-1}F \end{pmatrix} \mathcal{Z} \tag{8.27}$$

2. *Suppose* $\mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3$ *in* Γ_ρ *satisfy the condition (8.6) and* $(\mathcal{P}_1^*)^T\mathcal{Q}_\rho\mathcal{P}_1^* \succ 0$. *Also we require* $M \succ 0$. *Then there exist positive constants* c_1 *and* c_2 *such that for iteration number sufficiently large, we have*

$$\mathrm{eig}(\mathcal{V}_\rho) \subset [c_1, c_2] \tag{8.28}$$

Proof.

1. Note that E is a diagonal matrix. Then the statement follows from Lemma 2.

2. Note that we have the result (see Corollary 4.5 in [13])

$$\min\{\lambda_{\min}(N^{-1}M), 1\} \le \lambda_{\min}(E^{-1}F) \le \lambda_{\max}(E^{-1}F) \le \max\{\lambda_{\max}(N^{-1}M), 1\} \tag{8.29}$$

where $N = \mathrm{diag}(M)$. With the condition $M \succ 0$, it implies that there exist $d_1, d_2 > 0$ which

$$\lambda_{\min}(E^{-1}F) \subset [d_1, d_2] \tag{8.30}$$

Combine with (8.11), we are done.

□

We can see that Preconditioner II requires $M \succ 0$, which is stricter than the requirement of Preconditioner I (condition 8.7). However, Preconditioner II surely has a simpler form.

Chapter 9

Numerical Experiment for QSDP Solver

In this chapter we list some of the numerical results of solving the (QSDP) problem using an inexact primal-dual path following algorithm with different types of preconditioners. The program is written in Matlab, and based on SDPT3. We are running the program on an Intel Pentium PC 3.0 Hz with 512 RAM memory.
The problem we are going to solve in this chapter is the nearest correlation matrix problem (6.5). For the linear operator $\mathcal{Q}$, there are three different kind of products we are going to consider

- Symmetric Kronecker product, which means that $\mathcal{Q}(X) = U \circledast U(X)$, for a given $U \in S_n^+$. We denote this product with K.
- Hadamard product, which means that $\mathcal{Q}(X) = U \circ X$ for a given $U \in S_n$ and $U_{ij} \geq 0, \forall i, j$. We denote this product with H.
- A new product which is designed to be $\mathcal{Q}(X) = \frac{1}{2}(UX + XU), U \in S_n^+$ for a given U. We denote this product with N.

For the approximate correlation matrix D in (6.5), we simulate three different types of matrices using the method in [18]. The first type is a totally random correlation matrix which we denote by RD; The second type is a simulated correlation matrix from the $AR(1)$ model, we denote by AR1; The third type is a simulated correlation matrix from the compound symmetry model, we denote by CSM. We denote the

correlation matrix simulated by AR(1) model by Σ_{AR}, then

$$\Sigma_{AR}^{-1} = T' D^{-1} T$$

where $D = 0.01I$ and $T = (-\phi_{t,s})$, with $\phi_{t,t} = 1$, $\phi_{t+1,t} = 0.8$, and $\phi_{t,s} = 0$ otherwise. Define the correlation matrix generated by compound symmetry model by Σ_{CSM}, then

$$\Sigma_{CSM}^{-1} = T' D^{-1} T$$

where $D = \text{diag}(\sigma_1^2, \cdots, \sigma_n^2)$ with

$$\sigma_t^2 = \sigma^2 \left\{ 1 - \frac{(t-1)\rho^2}{1 + (t-1)\rho} \right\} \quad (t \geq 1)$$

and $T = (-\phi_{t,s})$ with $\phi_{t,t} = 1$, $\phi_{t,j} = \rho\{1 + (t-1)\rho\}^{-1}$, for $t \geq 2, j = 1, \cdots, t-1, \sigma = 1$ and $\rho = 0.5$. After generating the correlation matrix, we perturb the correlation matrix with another matrix which has Frobenius norm equal to 1, then assign the resulting matrix to D in (6.5). For the three preconditioners we are using, we denote them with P1, P2 and P3, for details of the (QSDP) preconditioners, please see [10]. In the experiment, we also change ρ in $\mathcal{Q}_\rho$ to see the difference.

K product, P1	$\rho = 0$		ρ=1		ρ=10	
(n=100)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	N/A	N/A	81.5	18	82.3	19
AR1	2.9	9	3.1	10	3.9	13
CSM	2.7	9	3.3	11	4.3	14
(n=200)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	402.9	16	322.9	18	316.5	20
AR1	8.6	9	10.3	11	11.8	13
CSM	8.2	9	9.9	11	11.6	13
(n=400)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	2373.0	16	2227.0	19	1670.7	20
AR1	51.8	10	57.2	11	66.5	13
CSM	49.6	10	54.6	11	65.2	13

H product, P1	$\rho = 0$		ρ=1		ρ=10	
(n=100)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	10.7	12	8.1	14	10.6	16
AR1	4.1	8	5.5	10	6.0	12
CSM	6.7	10	7.1	12	7.2	14
(n=200)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	38.0	12	45.2	16	45.5	18
AR1	19.9	9	19.1	10	24.3	13
CSM	21.2	10	26.7	13	26.7	14
(n=400)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	281.5	13	203.3	15	253.5	18
AR1	164.9	10	156.8	11	160.9	13
CSM	205.4	12	207.7	14	191.7	16

N product, P1	$\rho = 0$		ρ=1		ρ=10	
(n=100)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	22.6	13	20.0	16	22.6	18
AR1	7.3	8	9.5	11	9.0	12
CSM	7.2	9	7.8	11	8.8	13
(n=200)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	110.0	14	92.2	17	77.4	18
AR1	34.3	9	35.2	11	38.8	13
CSM	26.8	9	30.4	11	33.2	13
(n=400)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	506.3	14	463.2	17	447.7	19
AR1	173.0	9	196.0	11	217.5	13
CSM	159.4	10	157.6	11	197.0	14

K product, P2	$\rho = 0$		ρ=1		ρ=10	
(n=100)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	49.0	15	38.1	17	38.3	18
AR1	2.7	9	2.9	10	3.7	13
CSM	2.6	9	3.1	11	4.0	14
(n=200)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	190.4	16	191.2	18	187.3	20
AR1	8.5	9	10.2	11	11.9	13
CSM	8.1	9	9.7	11	11.7	13
(n=400)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	949.0	16	889.1	18	980.4	20
AR1	51.6	10	57.0	11	66.2	13
CSM	49.6	10	54.5	11	66.6	13

H product, P2	$\rho = 0$		ρ=1		ρ=10	
(n=100)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	15.0	12	15.8	15	16.0	16
AR1	4.0	8	5.4	10	6.0	12
CSM	6.8	10	7.1	12	7.1	14
(n=200)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	37.5	12	56.8	15	58.8	17
AR1	20.2	9	19.4	10	24.8	13
CSM	21.5	10	27.7	13	27.4	14
(n=400)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	251.0	13	233.9	15	250.5	17
AR1	164.1	10	156.2	11	160.2	13
CSM	205.3	12	207.8	14	190.9	16

N product, P2	$\rho = 0$		ρ=1		ρ=10	
(n=100)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	19.8	13	23.0	16	22.1	17
AR1	7.4	8	9.5	11	9.0	12
CSM	7.5	9	8.1	11	8.8	13
(n=200)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	82.1	14	73.2	16	78.7	18
AR1	35.0	9	35.8	11	40.6	13
CSM	26.6	9	30.6	11	33.3	13
(n=400)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	327.0	14	341.8	16	349.6	18
AR1	173.4	9	196.6	11	216.6	13
CSM	159.6	10	155.9	11	197.0	14

K product, P3	$\rho = 0$		ρ=1		ρ=10	
(n=100)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	69.8	15	65.9	17	48.0	18
AR1	2.7	9	3.0	10	3.9	13
CSM	2.7	9	3.2	11	4.1	14
(n=200)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	245.1	16	251.2	18	263.6	20
AR1	8.3	9	9.9	11	11.6	13
CSM	8.5	9	10.3	11	12.0	13
(n=400)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	1518.8	16	1518.6	18	1497.9	20
AR1	51.8	10	57.0	11	66.3	13
CSM	49.4	10	54.6	11	65.2	13

H product, P3	$\rho = 0$		ρ=1		ρ=10	
(n=100)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	6.8	11	7.5	14	8.0	15
AR1	4.4	8	5.6	10	6.1	12
CSM	6.7	10	7.1	12	7.3	14
(n=200)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	28.3	12	44.4	15	45.9	17
AR1	20.5	9	19.3	10	24.9	13
CSM	21.9	10	27.4	13	27.5	14
(n=400)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	201.7	13	183.5	15	214.3	17
AR1	165.3	10	156.2	11	160.7	13
CSM	204.9	12	208.3	14	191.5	16

N product, P3	$\rho = 0$		ρ=1		ρ=10	
(n=100)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	16.0	13	18.2	16	18.8	17
AR1	8.0	8	10.3	11	9.6	12
CSM	7.6	9	8.4	11	9.4	13
(n=200)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	81.6	15	73.4	16	73.6	18
AR1	34.8	9	35.4	11	39.6	13
CSM	26.7	9	29.8	11	33.4	13
(n=400)	cpu time	iteration	cpu time	iteration	cpu time	iteration
RD	417.6	15	408.8	17	445.5	19
AR1	173.8	9	196.4	11	217.2	13
CSM	160.3	10	156.2	11	197.8	14

(All the cpu time above is in term of second, N/A means the algorithm does not converge)

From the above tables, we notice that with different choices of correlation matrix, the time cost to solve the (QSDP) is dramatically different. The one with

random correlation matrix cost the most time, the simulated AR(1) and CSM matrix (which are more like real-world correlation matrix) cost less time. We can also see that with different choices of ρ, the time is also different, especially for the preconditioner P1. This phenomenon is interesting, it suggest that we may able to choose different ρ to different problem in order to reduce the computation time. However, to make this point clearer, it requires further analysis to the structure of the problem.

Bibliography

[1] A. Goel, P. Indyk, and K. Varadarajan. *Reductions among high dimensional proximity problems.* In Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms, pages 769-778. Society for Industrial and Applied Mathematics, 2001.

[2] A. Y. Alfakih, A. Khandani, and H. Wolkowicz, *Solving Euclidean distance matrix completion problems via semidefinite programming*, Computational Optimization and Applications, 12 (1999), pp. 13-30.

[3] B. Garter, *Smallest enclosing ball - fast and robust in C++*, http://www.inf.ethz.ch/personal/gaertner/miniball.html.

[4] D. White, Enclosing ball software, http://vision.ucsd.edu/ dwhite/ball.html.

[5] E. Welzl, *Smallest enclosing disks (balls and ellipsoids)*, H. Maurer, editor, New Results and New Trends in Computer Science, 359-370, Springer-Verlag, 1991.

[6] F. Alizadeh, J.-P. A. Haeberly, and M. L. Overton, *Complementarity and nondegeneracy in semidefinite programming*, Math. Programming, 77 (1997), pp. 111-128.

[7] Guanglu Zhou, Kim-Chuan Toh and Jie Sun. *Efficient algorithms for the smallest enclosing ball problem.* Comput. Optim. Appl,30(2):147-160, 2005.

[8] H.-M. Will. *Computation of additively weighted Voronoi cells for applications in molecular biology.* PhD thesis, ETH Zurich, 1998.

[9] H. Qi and D. Sun, *Quadratic convergence and numerical experiments of Newton's method for computing the nearest correlation matrix* , preprint, Department of Mathematics, National University of Singapore.

[10] Kim-Chuan Toh. *An inexact primal-dual path following algorithm for convex quadratic SDP*, Mathematical Programming, to appear.

[11] K. C. Toh, R. H. Tutuncu, and M. J. Todd, *Inexact primal-dual path-following algorithms for a special class of convex quadratic SDP and related problems*, Technical Report 1421, School of Operations Research and Industrial Engineering, Cornell University, Ithaca, New York, 2005.

[12] K. C. Toh, K. K. Phoon, and S. H. Chan, *Block preconditioners for symmetric indefinite linear systems*, Int. J. Numerical Methods in Engineering, 60 (2004), pp. 1361-1381.

[13] L. Bergamaschi, J. Gondzio and G. Zilli, *Preconditioning indefinite systems in interior point methods for optimization*, Computational Optimization and Applications, 28 (2004), pp. 149-171.

[14] J. Malick, *A dual approach to semidefinite least-squares problems*, SIAM J. Matrix Anal. Appl., 26 (2004), pp. 272-284.

[15] M. F. Anjos, N. J. Higham, P. L. Takouda, and H. Wolkowicz, *A semidefinite programming approach for the nearest correlation matrix problem*, Research Report, Department of Combinatorics and Optimization, University of Waterloo, 2003.

[16] M.J Todd, K.C Toh, and R.H Tütüncü.*On the Nesterov-Todd direction in semidefinite programming* SIAM J. Optimization,8 (1998), pp.769-796.

[17] N. Megiddo, *Linear-time algorithms for linear programming in ¡3 and related problems*, SIAM J. Comput., 12 (1983) 759-776.

[18] Naiping Liu, Mohsen Pourahmadi and Linxu Liu, *Covariance matrix selection and estimation via penalised normal likelihood*, Biometrika (2006), 93, 1, pp 85-98.

[19] N. J. Higham, *Computing the nearest correlation matrix problem from finance*, IMA J. Numerical Analysis, 22 (2002), pp. 329-343.

[20] P. M. Hubbard. *Approximating polyhedra with spheres for time-critical collision detection. ACM Trans. Graph.*, 15(3) : 179-210, 1996.

[21] R. Kurniawati, J. S. Jin, and J. A. Shepherd. *The SS+-tree: an improved index structure for similarity searches in a high-dimensional feature space.* In Proc. 5th Storage and Retrieval for Image and Video Databases SPIE, volume 3022, pages 110-120, 1997.

[22] Stephen Boyd, Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press 2004.

[23] T. H. Hopp and C. P. Reeve. *An algorithm for computing the minimum covering sphere in any dimension.* Technical Report NISTIR 5831, National Institute of Standards and Technology, 1996.

[24] Yu. Nesterov. *Smooth minimization of non-smooth functions*. Mathematical Programming, 103, pp. 127-152, 2005.

[25] Yu. Nesterov. *Introductory Lectures on Convex Programming*. 1996.

TWO PROBLEMS IN CONVEX CONIC OPTIMIZATION

ZHAI XIAOJUN

NATIONAL UNIVERSITY OF SINGAPORE

2007

TWO PROBLEMS IN CONVEX CONIC OPTIMIZATION ZHAI XIAOJUN 2007

TWO PROBLEMS IN CONVEX CONIC OPTIMIZATION ZHAI XIAOJUN 2007