Hybrid Methods for Optimization Problems with Positive Semi–Definite Matrix Constraints

By

Suliman Saleh Al–Homidan

PhD Thesis

Department of Mathematics and Computer Science

University of Dundee

Dundee

December 2003

Contents

List of Tables

List of Figures

Acknowledgements

I would like to express my sincere gratitude to my supervisor Professor Roger Fletcher for his kind supervision, suggestions and advice throughout the preparation of this thesis. His patience has been a constant source of inspiration throughout this work.

I would like to thank King Saud University for its financial support.

I would also like to thank the following:

My friends in the Department of Mathematics and Computer Science, University of Dundee, in particular George Mathai for his help in overcoming language difficulties.

My family and friends, with special regards to my beloved wife for her moral support.

Declaration

I declare that I am the author of this thesis; that I have consulted every reference cited except those for which I have indicated otherwise; that the work of which this thesis is a record has been done by myself, and that it has not been previously accepted for a higher degree.

Suliman Saleh Al–Homidan

Certification

This is to certify that Suliman Saleh Al–Homidan has complied with all the requirements for the submission of his PhD thesis to the University of Dundee.

Professor Roger Fletcher

Abstract

Three problems are handled in this thesis, all of which are involved with the positive semi–definite matrix as a convex constraint set. One problem is the Euclidean distance problem and the other two problems are different forms of the educational testing problem. Projection methods which solves least distance problems subject to the intersection of convex sets are used to solve these problems. It is found that the methods are globally convergent, but the rate of convergence is slow. However these methods do have the capability of determining the correct rank of the solution matrix, and this can be done in relatively few iterations. On the other hand there are conventional unconstrained and l_1 Sequential Quadratic Programming (SQP) methods which enable rapid convergence to be obtained. However, the correct rank is needed by these methods. Hence is is the purpose of this thesis to study hybrid methods. These hybrid methods have two different modes of operation. One is a projection method which provides global convergence and enables the correct rank to be determined. The other is either a quasi–Newton method or a nonlinear programming method, depending on the problem. An important feature concerns the interfacing of these modes of operation. Thus it has to be decided which method to use first, and when to switch between methods. Also it may not be straightforward, as we shall see here, to use the output of one method to start the other method. Difficulties such as these are addressed in the thesis. Many comparative numerical results are reported.

Chapter 0

Introduction

This thesis considers methods for solving certain optimization problems in which there are constraints on the variables. Many advances have taken place in this subject over the last forty years or so. There are now effective methods for situations in which the objective and constraint functions are smooth functions. Under reasonable assumptions, these methods can be shown to converge globally (that is from any starting point) to a point which satisfies optimality conditions for the problems. Also the rate of convergence can often be shown to be superlinear. Some progress has also been made for problems in which non–smooth functions occur. If these functions are a composition of a convex polyhedral function and a smooth function, then again globally and superlinear convergent methods have been suggested. This thesis addresses a rather more difficult situation in which some matrix, defined in terms of the problem variables, has to be positive semi–definite. One way to handle this problem is to impose a functional constraint in which the least eigenvalue of the matrix is non–negative. However, if there are multiple eigenvalues at the solution which is usually the case, such a constraint is usually non–smooth, and this non–smoothness cannot be modelled by a convex polyhedral composite function. An important factor is the determination of the multiplicity of the zero eigenvalues, or alternatively the *rank* of the matrix at the solution. If this rank is known it is usually possible to solve the problem by conventional techniques.

In this thesis the positive semi–definite matrix constraint is handled in a different way, by regarding it as a convex set. There are certain methods, known as *projection methods,* which can be used to solve least distance problems constrained by the intersection of convex sets. In this thesis the application of such methods to certain problems with positive semi–definite matrix constraint is considered. It is found that the methods are globally convergent, but the rate of convergence is linearly or slower. It is this latter feature that has probably contributed to the relatively little interest that has been shown in such methods. However it is demonstrated here that the methods do have the capability of determining the correct rank of the solution matrix, and this can be done in relatively few iterations.

Thus we are led to study hybrid methods in this thesis. The hybrid method has two different modes of operation. One is a projection method which provides global convergence and enables the correct rank to be determined. The other is either a quasi–Newton method or a conventional nonlinear programming method, depending on the problem, which enables rapid convergence to be obtained. An important feature concerns the interfacing of these modes of operation. Thus it has to be decided which method to use first, and when to switch between methods. Also it may not be straightforward, as we shall see here, to use the output of one method to start the other method. Difficulties such as these are addressed in the thesis. Hybrid methods have often been used successfully in optimization, for example Powell [1970], Hald and Madsen [1981] and Al–Baali and Fletcher [1985].

There are two main problems that are addressed in this thesis. Firstly, there is the *Euclidean distance matrix problem* which arises in many experimental sciences. The problem is to find the best Euclidean distance matrix which approximates a given non–Euclidean distance matrix. For solving this problem two methods are given. One is a projection method which is globally convergent. The other method for solving the Euclidean distance matrix problem is a quasi–Newton method, in particular the BFGS method. This method is superlinearly convergent but requires a knowledge of a certain characteristic rank. Hence new methods are established for solving this problem using the advantage of both methods by switching from one method to the other in a suitable way.

The second problem we going to study in this thesis is the *educational testing problem* which arises in statistics. In this problem there is given a symmetric positive definite matrix and it is required to determine how much can be subtracted from the diagonal of that matrix and still retain a positive semi–definite matrix. In the standard form the l_1–norm is used to measure the amount subtacted from the diagonal. Unfortunately this problem is not in the correct format for projection methods to be used directly. However there is an ingenious device due to Glunt [1991] which transforms this problem to a related one in which a least distance measure is used. We are therefore able to study the application of projection methods to the problem with the least distance measure. Then Glunt's transformation is used to enable the original educational testing problem to be solved.

These methods are again seen to be typified by being globally and slow convergent. When the correct rank for the matrix is known we are also able to use the l_1 *Sequential Quadratic Programming* (SQP) method to solve both problems, and this converges at second order. Subsequently hybrid methods are investigated to combine the advantageous features of both methods.

0.1 Outline of the thesis

Chapter 1 provides a general background to the optimization problem. This chapter includes a brief review of linear algebra and other various results. The concept of convex cones and normal cones with some important convex sets are also given. This chapter also introduces the concept of feasibility along with various expressions for feasible directions and describes optimality conditions relating to positive semi–definite matrix constraints. Finally, this chapter is concluded by a description of the Newton, quasi–Newton and Sequential Quadratic Programming (SQP) methods.

Chapter 2 provides a background about the projection methods for solving certain linear and least distance convex programming problems in which the feasible region is the intersection of a convex sets. Such optimization problems potentially arise in many practical situations, for example in linear programming problems, although projection methods are not the best for solving such problems. Here we are interested in the case where one of the convex cones is related to a positive semi–definite matrix cone. This chapter includes a description of the von Neumann [1950], Dykstra [1983] and Han [1988] projection methods for solving least distance convex programming problems and Glunt [1991] method for solving linear convex programming problems.

The aim of Chapter 3 is to find the best Euclidean distance matrix which approximates a given non–Euclidean distance matrix. Some applications of the above problem are given along with the definition of the Euclidean distance matrix and its characterization. Various methods for solving this problem are considered including a projection algorithm described by Glunt, Hayden, Hong and Wells [1990] and some new unconstrained methods based on using quasi–Newton methods. Other projection methods are also given and at the end of this chapter numerical comparisons of these methods are described.

In Chapter 4 some new methods for solving the Euclidean distance matrix problem are considered. These methods are developed from the methods of Chapter 3 using a hybrid method. A feature of some interest is how to move between the two methods. Numerical comparisons are also given in this chapter.

Chapter 5 considers a problem in which the objective function is a least distance function subject to a positive semi–definite matrix constraint where the diagonal of the matrix is allowed only to change. Two methods are developed for solving this problem. Firstly, a projection algorithm is given for solving this problem which converges globally. Secondly an implementation of the l_1 Sequential Quadratic Programming (SQP) method is used which converges quadratically. A transformation due to Fletcher [1985] is used to enable this method to be used. This chapter also includes a hybrid method between the projection method and the l_1 SQP method in a similar way to Chapter 4. Finally, numerical comparisons of these methods are carried out in the end of the chapter.

The problem to be considered in Chapter 6 is the educational testing problem. Previous attempts to solve the problem are described. The definition of the educational testing problem is given. This chapter also contains projection algorithm and l_1 SQP methods. At the end of this chapter numerical comparisons of these methods are given.

In Chapter 7 new methods for solving the educational testing problem are considered. The methods described here are similar to those in Chapter 4 and depend upon the two methods of Chapter 6 using a hybrid method. The projection method converges globally but often converges at very slow order. The l_1 SQP method converges quadratically but often requires the correct rank. Combining these two methods together produces a method with a better speed of convergence. Therefore this chapter describes two hybrid methods and also gives numerical comparisons.

The achievements of the thesis are summarized in Chapter 8 and suggestions for further research are discussed.

0.2 Notation

If $f(\mathbf{x})$ is continuously differentiable (C^1) then for any point $\mathbf{x}$ the vector of first partial derivatives, or gradient vector is referred to by $\mathbf{g}(\mathbf{x}) = \nabla f(\mathbf{x})$ and ∇ denotes the gradient operator $(\partial/\partial x_1\ ,\ \dots\ ,\ \partial/\partial x_n)^T$. If $f(\mathbf{x})$ is twice continuously differentiable (C^2) then there exists a matrix of second partial derivatives, or Hessian matrix, written $\nabla^2 f(\mathbf{x})$ which is square and symmetric.

Superscript "k" generally denotes quantities related to the kth iterate. For instance $f^{(k)} = f(\mathbf{x}^{(k)})$, $\mathbf{g}^{(k)} = \mathbf{g}(\mathbf{x}^{(k)})$, *etc*, and $f^* = f(\mathbf{x}^*)$, $\mathbf{g}^* = \mathbf{g}(\mathbf{x}^*)$, *etc.*

Throughout this thesis the lower case boldface letters such as $\mathbf{x}$, $\mathbf{y}, \mathbf{v}$ are used to denote vectors. Matrices are denoted by capital letters such as A, B, C and sometimes A written as $A = [a_{ij}]$.

Chapter 1

Optimization review

1.1 Introduction

The purpose of this chapter is to provide a general background to the optimization problem. This chapter includes some important concepts of optimization theory along with a description of the quasi–Newton method and the Sequential Quadratic Programming (SQP) method.

Section 1.2 contains a brief review of linear algebra and other various results. The concept of convex cones is given in Section 1.3. Also in that section two important convex cones are given. These are the cone of all $n \times n$ symmetric positive semi–definite matrices and the convex cone which is a subset of the positive semi–definite matrix cone. Section 1.3 also includes expressions for the normal cones of these convex cones. In Section 1.4 the concept of feasibility is described, along with various expressions for feasible directions. Section 1.5 describes optimality conditions relating to positive semi–definite matrix constraints. In Section 1.6 some details of how Newton and quasi–Newton methods work are given together with a proof of second order convergence. The SQP method is an efficient method for solving nonlinear programming problems when first and second derivatives are available. This method is Newton's method applied to find the stationary point of a Lagrangian function. The SQP method converges locally at second order. The global properties of the SQP method are improved by associating it with an exact penalty function. This method is described in Section 1.7.

1.2 Various results

The analysis of the optimization methods in this thesis requires results from linear algebra along with some definitions of rates of convergence. These are reviewed below.

Definition 1.2.1 (*Inner product*)

If $A,\ B \in \Re^{n\times n}$ then their inner product is defined by

$$\langle\, A\ ,\ B\,\rangle \;=\; \sum_{i,j=1}^{n} a_{ij}b_{ij} \;=\; tr(A^TB).$$

where $tr(A^TB)$ means the trace of the matrix A^TB which is the sum of the elements on the diagonal of A^TB.

Here $\Re^{n\times n}$ denotes the space of all real $n\times n$ matrices. Also we distinguish between $Diag\ A$ which denotes the diagonal matrix whose entries are the diagonal elements of A, and $diag\ \mathbf{a}$ which is the diagonal matrix whose entries are the elements of vector $\mathbf{a}$. The *null space* of A is defined by $N(A) = \{\mathbf{x} \in \Re^n : A\mathbf{x} = \mathbf{0}\}$.

Definition 1.2.2 (*Frobenius norm*)

A useful matrix norm in $\Re^{n\times n}$ is the Frobenius norm defined by

$$\|A\|_F \;=\; \langle\, A\ ,\ A\,\rangle^{\frac{1}{2}} \;=\; \{\sum_{i,j=1}^{n} |a_{ij}|^2\ \}^{\frac{1}{2}}$$

Definition 1.2.3 (*Householder matrix*)

A matrix $Q \in \Re^{n\times n}$ is said to be orthogonal if $Q^TQ = I$. A particular Householder matrix may be defined by

$$Q \;=\; I \;-\; \frac{2}{\boldsymbol{\nu}^T\boldsymbol{\nu}}\ \boldsymbol{\nu}\boldsymbol{\nu}^T, \quad \boldsymbol{\nu} \;=\; [1,\ \ldots,\ 1,\ 1+\sqrt{n}]^T. \tag{1.2.1}$$

This Householder matrix is a special case for which if $\mathbf{e} = [1,1,\ldots,1]^T$ then

$$Q\mathbf{e} \;=\; \begin{bmatrix} \mathbf{0} \\ -\|\mathbf{e}\|_2 \end{bmatrix} \;=\; \begin{bmatrix} \mathbf{0} \\ -\sqrt{n} \end{bmatrix}. \tag{1.2.2}$$

Definition 1.2.4 (*Irreducibly embeddable*)

If there exist n vectors $\mathbf{p}_1, \ldots, \mathbf{p}_n$ in $\Re^r$ $(r \le n-1)$ such that

$$a_{ij} = \| \mathbf{p}_i - \mathbf{p}_j \|_2^2 \quad (1 \le i,j \le n). \tag{1.2.3}$$

for set of vectors in $\Re^r$ but not in $\Re^{r-1}$ then the points $\mathbf{p}_1, \ldots, \mathbf{p}_n$ and the matrix $A = [a_{ij}]$ are said to be irreducibly embeddable in $\Re^r$.

Definition 1.2.5 (*Positive definite matrices*)

An $n \times n$ symmetric matrix A is said to be positive definite if

$$\mathbf{x}^T A\mathbf{x} > 0 \quad \forall \mathbf{x} \in \Re^n \qquad \mathbf{x} \neq \mathbf{0} \tag{1.2.4}$$

and is denoted by $A > 0$. If the inequality in (1.2.4) replaced by $\mathbf{x}^T A\mathbf{x} \ge 0$ then A is said to be *positive semi–definite* and is denoted by $A \ge 0$.

Positive definite matrices are an important class of matrices and arise naturally in many applications. The above definition cannot be checked numerically. Equivalent definitions which can be checked are the following

i. All eigenvalues of $A > 0$.

ii. There exists a unique lower triangular $L \in \Re^{n\times n}$ such that $LL^T = A$ with $l_{ii} > 0$ *(Choleski factors).*

iii. LDL^T factors exist with $l_{ii} = 1$ and $d_{ii} > 0$.

If A is a positive definite matrix, then the largest entry in A is on the diagonal and the diagonal elements are all positive.

Definition 1.2.6 (*First and second order convergence*)

Let $\mathbf{x}^*$ be a local minimum point with error defined as

$$\mathbf{h}^{(k)} = \mathbf{x}^{(k)} - \mathbf{x}^*.$$

If $\mathbf{h}^{(k)} \to \mathbf{0}$ we have convergence. If the errors behave as

$$\frac{\|\mathbf{h}^{(k+1)}\|}{\|\mathbf{h}^{(k)}\|^p} \to a$$

where $a > 0$ then the order of convergence is defined to be p order. The most important cases are where $p = 1$ *(first order or linear convergence)* in which $a < 1$ must hold, and $p = 2$ *(second order or quadratic convergence).* If $\frac{\|\mathbf{h}^{(k+1)}\|}{\|\mathbf{h}^{(k)}\|} \to 0$ then this is known as *superlinear convergence.* Often it is only possible to obtain bounds, for example

$$\frac{\|\mathbf{h}^{(k+1)}\|}{\|\mathbf{h}^{(k)}\|} \leq a$$

or

$$\mathbf{h}^{(k+1)} = O(\|\mathbf{h}^{(k)}\|)$$

for first order convergence and

$$\frac{\|\mathbf{h}^{(k+1)}\|}{\|\mathbf{h}^{(k)}\|^2} \leq a$$

or,

$$\mathbf{h}^{(k+1)} = O(\|\mathbf{h}^{(k)}\|^2).$$

for second order convergence.

1.3 Cones and normal cones

The concept of a convex cone and its properties are very useful when applying convex analysis, for instance the normal cone is important in the development of optimality conditions. In this section the notion of cones and normal cones is described.

Definition 1.3.1 (*Convex set and convex function*)

A subset C of $\Re^n$ is said to be a convex set if

$$\mathbf{x}_\lambda = (1-\lambda)\mathbf{x}_1 + \lambda\mathbf{x}_2 \in C$$

for all $\mathbf{x}_1, \mathbf{x}_2 \in C$ and $0 \leq \lambda \leq 1$. A convex function $f(\mathbf{x})$ on the domain C is defined by the condition that for any $\mathbf{x}_1, \mathbf{x}_2 \in C$ it follows that

$$f(\mathbf{x}_\lambda) \leq (1-\lambda)f(\mathbf{x}_1) + \lambda f(\mathbf{x}_2) \qquad \forall\ \lambda \in [0,1]$$

where $\mathbf{x}_\lambda = (1-\lambda)\mathbf{x}_1 + \lambda\mathbf{x}_2$.

Definition 1.3.2 (*Convex cone*)

A subset K of $\Re^n$ is called a convex cone if and only if $\mathbf{x}_1, \mathbf{x}_2 \in K, \alpha, \beta \geq 0$ implies that $\alpha \mathbf{x}_1 + \beta \mathbf{x}_2 \in K$.

The set of all $n \times n$ symmetric positive semi–definite matrices

$$K_{\Re} = \{A : A \in \Re^{n \times n}, \quad A^T = A \; and \; \mathbf{z}^T A \mathbf{z} \geq 0 \quad \forall \, \mathbf{z} \in \Re^n\} \tag{1.3.1}$$

is a convex cone of dimension $n(n+1)/2$. The dimension is the number of free parameters in a symmetric matrix A. Let $A, B \in K_{\Re}$ then $\mathbf{z}^T(\alpha A + \beta B)\mathbf{z} \geq 0 \quad \forall \, \mathbf{z} \in \Re^n$, and $\alpha, \beta \geq 0$. This is because $\alpha \mathbf{z}^T A \mathbf{z} \geq 0$, $\beta \mathbf{z}^T B \mathbf{z} \geq 0 \quad \forall \, \mathbf{z} \in \Re^n$, which implies that $\alpha A + \beta B \in K_{\Re}$. This proves that $K_{\Re}$ is a convex cone. (The subscript $\Re$ is used to distinguish this case from the restricted cone in the next paragraph).

Another convex cone which will be used for the projection method given in Chapter 3 is the set of all $n \times n$ symmetric positive semi–definite matrices with respect to M, where

$$M = \{ \, \mathbf{x} \in \Re^n : \; \mathbf{e}^T \mathbf{x} = 0 \, \} \tag{1.3.2}$$

and $\mathbf{e} = [1, 1, \ldots, 1]^T \in \Re^n$. $K_{\Re}$ is subset of this set which may be denoted by

$$K_M = \{A : A \in \Re^{n \times n}, \quad A^T = A \; and \; \mathbf{x}^T A \mathbf{x} \geq 0 \quad \forall \, \mathbf{x} \in M\} \tag{1.3.3}$$

which is a convex cone. Let $A, B \in K_M$ then

$$\mathbf{z}^T(\alpha A + \beta B)\mathbf{z} \geq 0 \quad \forall \, \mathbf{z} \in M, \;\; \alpha \geq 0 \; and \; \beta \geq 0. \tag{1.3.4}$$

This is because $\alpha \mathbf{z}^T A \mathbf{z} \geq 0$, $\beta \mathbf{z}^T B \mathbf{z} \geq 0 \quad \forall \, \mathbf{z} \in M$, which implies that $\alpha A + \beta B \in K_M$. Thus K_M is convex cone.

It is also convenient to define two other convex sets for the purposes of Chapters 5 and 6. If $F \in \Re^{n \times n}$ is any given symmetric positive definite matrix then define

$$K_{off} = \{A : A \in \Re^{n \times n}, \; A - Diag \; A = \bar{F}\}. \tag{1.3.5}$$

where $\bar{F} = F - Diag \; F$. This is the set of matrices whose off–diagonal elements are equal to those of F. Also, let $diag \; \mathbf{v} = Diag \; F$ then define

$$K_b = \{A : A \in \Re^{n \times n}, \quad A = \bar{A} + diag \, \mathbf{x}, \;\; x_i \leq v_i \;\; i = 1, 2, ...n\} \tag{1.3.6}$$

where $\bar{A} = A - Diag\, A$. This is the set of matrices that is obtained by reducing the diagonal of A. K_{off} and K_b are convex subspaces.

Next, the concept of the normal cone, denoted by ∂K, is introduced which is of importance when deriving optimality conditions for problems which involve any convex set. If $\mathbf{a}$ is on the boundary of K, then a vector $\mathbf{x}$ is said to be normal to a convex set K at $\mathbf{a}$, if $\mathbf{x}$ does not make an acute angle with any line segment in K emanating from $\mathbf{a}$. Therefore any vector $\mathbf{x} \in \partial K(\mathbf{a})$ must satisfy $\langle \mathbf{y} - \mathbf{a}, \mathbf{x} \rangle \leq 0$ for every $\mathbf{y} \in K$, (see Figure 1.3.1). The set of all vectors $\mathbf{x}$ normal to K at $\mathbf{a}$ is called the normal cone to K at $\mathbf{a}$, and denoted by

$$\partial K(\mathbf{a}) = \{\mathbf{x} : \mathbf{x} \in \Re^n, \quad \langle \mathbf{y} - \mathbf{a} , \mathbf{x} \rangle \leq 0 \qquad \forall\, \mathbf{y} \in K\}. \tag{1.3.7}$$

Equivalently the normal cone can be defined by

$$\partial K(\mathbf{a}) = \{\mathbf{x} : \mathbf{x} \in \Re^n, \quad \langle \mathbf{x}, \mathbf{a} \rangle = \sup_{\mathbf{y} \in K} \langle \mathbf{x}, \mathbf{y} \rangle\}. \tag{1.3.8}$$

It is convenient to define $\partial K(\mathbf{a}) = \{0\}$ if $\mathbf{a}$ is interior to K, and $\partial K(\mathbf{a}) = \emptyset$ (the empty set) if $\mathbf{a}$ is exterior to K, this is consistent with (1.3.7) and (1.3.8).

Let K_1 and K_2 be convex sets in $\Re^n$ whose relative interiors have a point $\mathbf{a}$ in common. Then

$$\partial(K_1 \cap K_2)(\mathbf{a}) = \partial K_1(\mathbf{a}) + \partial K_2(\mathbf{a}) \tag{1.3.9}$$

(see [Rockafellar 1970]).

In this thesis we consider the case of the convex cone in which the elements are matrices instead of vectors and we use the matrix inner product in Definition 1.2.1. It follows from (1.3.7) that

$$\partial K(A) = \{B : B \in \Re^{n \times n} \ \ and \ \langle Z - A , B \rangle \leq 0 \qquad \forall\, Z \in K\} \tag{1.3.10}$$

where K is a matrix cone.

It follows from (1.3.8) that the normal cone for (1.3.1) is

$$\partial K_{\Re}(A) = \{B : B \in \Re^{n \times n}, \langle A, B \rangle = \sup_{V \in K_{\Re}} \langle V, B \rangle\}.$$

However since unsymmetric matrices in $\partial K_{\Re}$ are not of interest here it is more convenient to define $\partial K_{\Re}$ by restricting it to the symmetric normal cone

Figure 1.3.1: The normal cone ∂K for a convex cone K at point $\mathbf{a}$.

$$\partial K_{\Re}(A) \;=\; \{B : B \;\in\; \Re^{n\times n}, B \;=\; B^T, \langle A, B\rangle \;=\; \sup_{V\in K_{\Re}} \langle V, B\rangle\}. \tag{1.3.11}$$

The most interesting case concerns the elements of the boundary of $K_{\Re}$, since $\partial K_{\Re}(A) = \{0\}$ when A is interior to $K_{\Re}$ $(A > 0)$.

In the following a theorem due to Fletcher [1985] is given to show how to find the normal cone $\partial K_{\Re}(A)$ at A, such that A belongs to the boundary of $K_{\Re}$

Theorem 1.3.3

If the columns of Z are an orthonormal basis for the null space of A, and Λ is any symmetric positive semi–definite matrix, then an equivalent form to (1.3.11) where A lies on the boundary of $K_{\Re}$ is the following

$$\partial K_{\Re}(A) \;=\; \{B : B \;\in\; \Re^{n\times n}, B \;=\; B^T, B \;=\; -\,Z\Lambda Z^T,$$

$$\Lambda \;=\; \Lambda^T,\; \Lambda \;\geq\; 0\}. \tag{1.3.12}$$

Proof

Consider $\sup_{V \in K_{\Re}} \langle V, B \rangle$ for fixed B, let $B = X \Omega X^T$ be the spectral decomposition of B with X being the orthogonal matrix of eigenvectors and $\Omega = diag\ [\omega_1, \omega_2, \ldots, \omega_n]$ the diagonal matrix of eigenvalues. Since A is positive semi–definite there exists $C = X^T V X$ which is positive semi–definite. Using Definition 1.2.1 of the inner product it follows that

$$\begin{aligned} \sup_{V \in K_{\Re}} \langle V, B \rangle &= \sup_{C \in K_{\Re}} \langle C, \Omega \rangle \\ &= \sup_{c_{ii} \geq 0} \sum c_{ii} \omega_i . \end{aligned}$$

This follows because

$$\begin{aligned} \langle V, B \rangle &= tr(VB) \\ &= tr(VXX^T BXX^T) \\ &= tr(X^T VXX^T BX) \\ &= tr(C\Omega). \end{aligned}$$

Hence

$$\sup_{V \in K_{\Re}} \langle V, B \rangle = 0 \qquad iff \ \ \omega_i \leq 0 \quad \forall\ i \tag{1.3.13}$$

and this is equivalent to $B \leq 0$ since ω_i are the eigenvalues of B. Hence an equivalent form to (1.3.11) is

$$\partial K_{\Re}(A) = \{B : B \in \Re^{n \times n}, B = B^T,\ \langle A, B \rangle = 0,\ B \leq 0\}. \tag{1.3.14}$$

Let $A = Y \Lambda_r Y^T$, with Λ_r being the diagonal matrix whose elements are the nonzero eigenvalues of A and the columns of Y are the corresponding orthonormal set of eigenvectors, so that $[Y \quad Z]$ is an orthogonal matrix. Express B as

$$B = [Y \quad Z] \begin{bmatrix} R & S \\ S^T & T \end{bmatrix} [Y \quad Z]^T. \tag{1.3.15}$$

Since $\langle A,\ B\rangle = 0$ then $tr(\Lambda_r Y^T BY) = 0$. The diagonal elements of $Y^T BY$ are zero because Λ_r is positive definite and diagonal. Also from (1.3.15) $Y^T BY = R$ so it follows that R has zero diagonal elements. Hence from (1.3.14) $B \leq 0$ implies that $R = \mathbf{0}$, and thus $T \leq 0$. Therefore from (1.3.15) $B = ZTZ^T$, and (1.3.12) follows since $\Lambda = -T$ $\square$.

Example 1.3.4

If $n = 2$ then the cone in (1.3.1) becomes

$$K_{\Re} = \{A : A = \begin{bmatrix} x & z \\ z & y \end{bmatrix} \ x \geq 0,\ y \geq 0,\ xy \geq z^2 \ \ and \ \ x, y, z \in \Re\}$$

and is illustrated in Figure 1.3.2. Clearly the matrices in the interior of the cone are positive definite, whereas those on the boundary are singular. For example the matrix

$$\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$$

on the boundary is positive semi–definite. Then $Z = [1 \quad 1]^T$ and $\Lambda = [\alpha] \geq 0$, so the normal cone (1.3.12) at this point is

$$\partial K_{\Re}(\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}) = \{B : B = -\alpha \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix},\ \alpha \geq 0\}.$$

The normal cone for $K_{off} \cap K_b$ is given in the following.

Theorem 1.3.5

Figure 1.3.2: The positive semi–definite matrix cone $K_{\Re}$.

Let $F \in \Re^{n\times n}$ be a given symmetric positive definite matrix and define K_{off} and K_b as in (1.3.5) and (1.3.6) respectively. Let $A \in K_{off} \cap K_b$. Then

$$\begin{aligned}\partial(K_{off} \cap K_b)\ (A) \ &= \ \{B : B \ \in \ \Re^{n\times n}, \\ &\left\{\begin{matrix} b_{ii} & \geq & 0 & if & x_i & = & v_i \\ b_{ii} & = & 0 & if & x_i & < & v_i \end{matrix}\right\} \ i = 1,...,n\}.\end{aligned} \tag{1.3.16}$$

where $A \ = \ \bar{A} + diag\ \mathbf{x}$.

Proof

From the normal cone definition (1.3.10) it is clear that

$$\partial K_{off}(\bar{A} + diag\ \mathbf{x}) \ = \ \{B : B \ = \begin{bmatrix} 0 & b_{21} & \dots & b_{n1} \\ b_{21} & 0 & \dots & b_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \dots & 0 \end{bmatrix}\} \tag{1.3.17}$$

because $\bar{Z} \ = \ \bar{A} \ = \ \bar{F}$ implies $Z - A = \mathbf{0} \quad \forall\, i \neq j,$ in (1.3.10) where $\bar{Z} \ = \ Z - Diag\, Z.$

Consider

$$\sup_{Z \in K_{off} \cap K_b} \langle B, Z \rangle$$

and assume for some i that $b_{ii} \ < \ 0$. Let $z_{ii} \ = \ a_{ii} \ - \ \beta$ then $Z \ \in \ K_{off} \cap K_b$. By making β sufficiently large we can make $\langle B, Z \rangle$ as large as we like. Thus, if $b_{ii} \ < \ 0$ for any i, we have that

$$\langle A, B \rangle \ = \ \sup_{Z \in K_{off} \cap K_b} \langle B, Z \rangle \ = \ \infty.$$

Now suppose $b_{ii} \ \geq \ 0 \quad \forall\, i.$ Then from the normal cone definition (1.3.8)

$$\begin{aligned} \partial K_{off} \cap K_b(\bar{A} + diag\ \mathbf{x}) \ = \ \{B : B \ \in \ \Re^{n \times n}, \\ \langle B, \bar{A} \ + \ diag\ \mathbf{x} \rangle \ = \ \sup_{Z \in K_{off} \cap K_b} \langle B, Z \rangle\}. \end{aligned}$$

Now since $\bar{Z} \ = \ \bar{A}, \ b_{ii} \ \geq \ 0 \quad \forall\, i$ and

$$\begin{aligned} \langle B, diag\ \mathbf{z} \rangle \ &= \ \sum_{i=1}^{n} b_{ii} z_{ii} \\ &\leq \ \sum_{i=1}^{n} b_{ii} v_i \\ &= \ \langle B, diag\ \mathbf{v} \rangle \end{aligned}$$

where $diag\, \mathbf{z} \ = \ Z \ - \ \bar{Z}$. Then

$$\sup_{Z \in K_{off} \cap K_b} \langle B, Z \rangle \ \leq \ \langle B, \bar{A} \ + \ diag\ \mathbf{v} \rangle$$

but since $\bar{A} \ + \ diag\ \mathbf{v} \ \in \ K_{off} \cap K_b$ then

$$\sup_{Z \in K_{off} \cap K_b} \langle B, Z \rangle \ = \ \langle B, \bar{A} \ + \ diag\ \mathbf{v} \rangle.$$

Thus

$$\langle B, \bar{A} \ + \ diag\ \mathbf{x} \rangle \ = \ \langle B, \bar{A} \ + \ diag\ \mathbf{v} \rangle \ = \ \langle A, \ B \rangle$$

Now

$$\sup_{Z \in K_{off} \cap K_b} \langle B, Z \rangle \ = \ \left\{ \begin{array}{lc} \infty & if \quad b_{ii} \ < \ 0 \quad for\ any\ i \\ \langle A, \ B \rangle & otherwise \end{array} \right\} \tag{1.3.18}$$

this implies from (1.3.8)

$$\begin{aligned} &\partial(K_{off} \cap K_b)\ (A) \ = \\ &\quad \{B : B \ \in \ \Re^{n \times n}, \ \langle B, \bar{A} \ + \ diag\ \mathbf{x} \rangle \ = \ \langle B, \bar{A} \ + \ diag\ \mathbf{v} \rangle\} \end{aligned} \tag{1.3.19}$$

which implies that

$$\sum_{i=1}^{n} b_{ii}(v_i \ - \ x_i) \ = \ 0. \tag{1.3.20}$$

Therefore if $x_i \ < \ v_i$ then $b_{ii} \ = \ 0$ since each term of (1.3.20) is nonnegative. $\Box$

In addition to the normal cone $\partial K_{\Re}$ another set of interest is the normal cone ∂K_M. This set is important when deriving the optimality conditions for the projection method given in Chapter 3. A theorem for the expression of the normal cone ∂K_M is stated and proved. Firstly though a theorem used in the proof is given. An example for the convex cone (1.3.3) when $n \ = \ 3$ is given later on.

The following theorem is based on Hayden and Wells [1988].

Theorem 1.3.6

Let Q be the Householder matrix in (1.2.1). If $A \ = \ A^T \in \Re^{n \times n}$ and M is given in (1.3.2), then

$$\mathbf{x}^T A \mathbf{x} \ \geq \ 0 \qquad \forall\ \mathbf{x} \ \in \ M \tag{1.3.21}$$

if and only if

$$QAQ \ = \ \begin{bmatrix} A_1 & \mathbf{a} \\ \mathbf{a}^T & \alpha \end{bmatrix}, \qquad A_1 \ \geq \ 0. \tag{1.3.22}$$

Proof

For all $\mathbf{x} \in M$ denote $\mathbf{y} = Q\mathbf{x}$, and it follows that $\mathbf{x} = Q\mathbf{y}$ since Q is orthogonal and symmetric. The condition $\mathbf{x} \in M$ is equivalent to $\mathbf{e}^T\mathbf{x} = 0$, or $\mathbf{e}^T Q\mathbf{y} = 0$, and hence to $\mathbf{e}_n^T\mathbf{y} = 0$ where $\mathbf{e}_n^T = [0,\ 0,\ \ldots,\ 0,\ 1]$. Thus (1.3.21) can be written as

$$(Q\mathbf{y})^T AQ\mathbf{y} \geq 0 \quad \forall\, \mathbf{y} \in \Re^n \ \ \textit{such that} \ \ y_n = 0. \tag{1.3.23}$$

Thus (1.3.22) follows. $\square$

In what follows we denote the rank of A_1 by r, and hence the spectral decomposition of A_1 can be expressed as

$$A_1 = U\Lambda U^T = U \begin{bmatrix} \Lambda_r & \mathbf{0} \\ \mathbf{0}^T & 0 \end{bmatrix} U^T \tag{1.3.24}$$

where U is an orthogonal matrix and $\Lambda_r > 0$ is an $r \times r$ diagonal matrix.

A theorem due to Glunt et. al. [1990] will be given to show how to find the normal cone $\partial K_M(A)$ at $A \in K_M$.

Theorem 1.3.7

Given any A, then the normal cone $\partial K_M(A)$ is given by

$$\partial K_M\,(A) = \{B : \ B = Q \begin{bmatrix} UGU^T & \mathbf{0} \\ \mathbf{0}^T & 0 \end{bmatrix} Q, \ \ H \leq 0\} \tag{1.3.25}$$

where

$$G = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0}^T & H \end{bmatrix},$$

U is an orthogonal matrix given by (1.3.24) and H is a symmetric matrix in $\Re^{(n-r-1)\times(n-r-1)}$ (The partitioning of G reflects that of A_1).

Proof

Let $B \in \partial K_M(A)$ and define B_1, $\mathbf{b}$ and β by

$$B = Q \begin{bmatrix} B_1 & \mathbf{b} \\ \mathbf{b}^T & \beta \end{bmatrix} Q.$$

Now let $X_1 \in \Re^{n-1 \times n-1}$ be any positive semi–definite matrix. Then for any $\mathbf{x}$, ξ by Theorem 1.3.6 the matrix

$$X = Q \begin{bmatrix} X_1 & \mathbf{x} \\ \mathbf{x}^T & \xi \end{bmatrix} Q$$

is in K_M. By (1.3.10)

$$\langle X - A, B \rangle \leq 0$$

and since Q is orthogonal we have

$$\langle QXQ, QBQ \rangle \leq \langle A, B \rangle$$

which implies that

$$\langle X_1, B_1 \rangle + 2\mathbf{x}^T\mathbf{b} + \xi\beta \leq \langle A, B \rangle. \tag{1.3.26}$$

Let either $\mathbf{b} \neq \mathbf{0}$ or $\beta \neq 0$. Choose $\mathbf{x} = \lambda\mathbf{b}$ and $\xi = \lambda\beta$ for sufficiently large $\lambda > 0$ then (1.3.26) is false (contradiction). This implies that $\mathbf{b} = \mathbf{0}$ and $\beta = 0$.

Following a similar strategy as in the previous proof (Theorem 1.3.3), let $V\Omega V^T$ be the spectral decomposition of B_1 with V being the orthogonal matrix of eigenvectors and $\Omega = diag\ [\omega_1,\ \omega_2,\ \ldots,\ \omega_{n-1}]$ the diagonal matix of eigenvalues. Since X_1 is positive semi–definite there exists a positive semi–definite matrix $C = V^T X_1 V$, and using (1.3.26)

$$\begin{aligned} \langle A, B \rangle \geq \langle X_1, B_1 \rangle &= \langle V^T X_1 V, \Omega \rangle = \langle C, \Omega \rangle \\ &= \sum_{j=1}^{n-1} c_{jj}\omega_j. \end{aligned}$$

Hence

$$\sup_{A \in K_M} \langle A, B \rangle \geq 0 \qquad iff\ \ \omega_i \leq 0 \qquad i = 1, \ldots, n-1 \tag{1.3.27}$$

and this is equivalent to $B_1 \leq 0$ since ω_i are the eigenvalues of B_1 and $c_{11},\ c_{22},\ \dots,\ c_{n-1\ n-1}$ are nonnegative scalars since C is positive semi–definite matrix. Therefore, if $B \in \partial K_M(A)$ it has the form

$$B = Q \begin{bmatrix} B_1 & 0 \\ 0 & 0 \end{bmatrix} Q, \quad B_1 \leq 0. \tag{1.3.28}$$

From (1.3.28) we have $\langle X, B\rangle \leq 0 \; \forall \; X \in K_M$ and since $\langle X - A, B\rangle \leq 0$, then

$$\begin{aligned} \langle A, B\rangle \leq \sup_{X \in K_M} \langle X, B\rangle & \leq 0 & (1.3.29) \\ & \leq \langle A, B\rangle. & (1.3.30) \end{aligned}$$

from (1.3.27). Thus from (1.3.29) and (1.3.30)

$$\langle A, B\rangle = 0. \tag{1.3.31}$$

Then (1.3.28), (1.3.31) and (1.3.22) imply that

$$\langle A_1, B_1\rangle = 0.$$

Then from the spectral decomposition A_1 in (1.3.24) we have

$$\langle \Lambda_r, U^T B_1 U\rangle = \langle \Lambda_r, G\rangle = \sum_{j=1}^{r} \lambda_j g_{jj} = 0 \tag{1.3.32}$$

where $G = U^T B_1 U \leq 0$. Now $G \in \Re^{(n-1 \times n-1)}$ has the following structure

$$G = \begin{bmatrix} N & S \\ S^T & H \end{bmatrix}$$

where $H \in \Re^{(n-r-1)\times(n-r-1)}$ but since $\lambda_j > 0$ and $g_{jj} \leq 0$ for $1 \leq j \leq r$ then from(1.3.32) $g_{jj} = 0$ for $1 \leq j \leq r$. Since G is negative semi–definite G has the form

$$G = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & H \end{bmatrix}, \quad H \leq 0.$$

Therefore

$$B_1 = U\begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & H \end{bmatrix}U^T, \quad H \le 0.$$

and B has the form of (1.3.25).

Conversely, if B is written in the form (1.3.25) then since $B \le 0$ and $X - A \ge 0 \quad \forall\, X \in K_M$, (since A is the nearest positive semi–definite matrix) then

$$\langle X - A, B\rangle \le 0 \quad \forall\, X \in K_M$$

which implies that $B \in \partial K_M(A)$ □

In the rest of this section an example of the convex cone (1.3.3) where $n = 3$ is given.

Example 1.3.8

For the example let $n = 3$, and

$$A = \begin{bmatrix} 0 & x & y \\ x & 0 & z \\ y & z & 0 \end{bmatrix}$$

It is convenient for what follows later to express the Householder matrix Q as

$$Q = \begin{bmatrix} a-c & b-d & -c-d \\ b-d & a-c & -c-d \\ -c-d & -c-d & -c-d \end{bmatrix}$$

where $a = 0.911$, $b = 0.244$, $c = 0.122$ and $d = 0.455$ accurate to 3 decimal places. Then

$$QAQ = \begin{bmatrix} bz - \frac{1}{3}x - ay & \frac{1}{3}(2x-y-z) & dz - \frac{1}{3}x - cy \\ \frac{1}{3}(2x-y-z) & by - \frac{1}{3}x - az & dy + cz - \frac{1}{3}x \\ dz - \frac{1}{3}x - cy & dy + cz - \frac{1}{3}x & \frac{2}{3}(x+y+z) \end{bmatrix}$$

and the matrix A_1 is given by

$$A_1 = \begin{bmatrix} bz - ay - \frac{1}{3}x & \frac{1}{3}(2x - y - z) \\ \frac{1}{3}(2x - y - z) & by - \frac{1}{3}x - az \end{bmatrix}.$$

Figure 1.3.3: The positive semi–definite matrix cone K_M in M.

Using (1.3.23) the cone K_M in (1.3.3) is defined by the inequalities

$$bz - ay - \tfrac{1}{3}x \geq 0$$

$$by - \tfrac{1}{3}x - az \geq 0$$

$$(bz - ay - \tfrac{1}{3}x)(by - \tfrac{1}{3}x - az) \geq [\tfrac{1}{3}(2x - y - z)]^2 \tag{1.3.33}$$

where inequality (1.3.33) implies that

$$z^2 - 2z(x + y) + (x - y)^2 \leq 0.$$

The matrix

$$A' = \begin{bmatrix} 0 & -1 & -1 \\ -1 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix}$$

is on the boundary of the cone K_M. Then

$$U = \begin{bmatrix} -0.2588 & 0.966 \\ 0.966 & 0.2588 \end{bmatrix} \quad \textit{and} \quad G = \begin{bmatrix} 0 & 0 \\ 0 & \lambda \end{bmatrix},$$

so the normal cone (1.3.25) at this point is

$$\partial K_M(A') = \{\lambda \begin{bmatrix} 0.5 & 0 & -0.5 \\ 0 & 0 & 0 \\ -0.5 & 0 & 0.5 \end{bmatrix}, \ \lambda \geq 0\}.$$

The cone for this example is illustrated in Figure 1.3.3.

1.4 The set of feasible directions

In this section results are given which are used subsequently to derive optimality conditions.

A *feasible point* $\mathbf{x}$ is a point which satisfies all the constraints in an optimization problem and the set of all such points is referred to as the *feasible region.* Here we consider problems in which the the feasible region is a convex set $K \subset \Re^n$. Let $\{\mathbf{x}^{(k)}\} \rightarrow \mathbf{x}$ where $\mathbf{x}^{(k)} \neq \mathbf{x} \ \ \forall \ k$ is an infinite sequence of feasible points. It is possible to express

$$\mathbf{x}^{(k)} - \mathbf{x} = \delta^{(k)}\mathbf{s}^{(k)} \quad \forall \ k \tag{1.4.1}$$

where $\delta^{(k)} > 0$ is a scalar. The sequence $\mathbf{x}^{(k)}$ is said to be a directional sequence if $\{\mathbf{s}^{(k)}\} \rightarrow \mathbf{s}$. The limiting vector $\mathbf{s}^{(k)}$ is referred to as a *feasible direction.* Then the set of feasible directions can be expressed as

$$\mathcal{F}(\mathbf{x}) = \{\mathbf{s} : \exists \ \{\mathbf{x}^{(k)}\} \ \textit{such that} \ \{\mathbf{x}^{(k)}\} \rightarrow \mathbf{x}, \{\mathbf{s}^{(k)}\} \rightarrow \mathbf{s}, \delta^{(k)} \rightarrow 0\}. \tag{1.4.2}$$

A related set of feasible directions which is easier to manipulate is the set

$$F(\mathbf{x}) \;=\; \{\mathbf{s} : \; \mathbf{s} \;\in\; \Re^n, \;\; \mathbf{s}^T\mathbf{g} \;\leq\; 0 \quad \forall \mathbf{g} \;\in\; \partial K(\mathbf{x})\}. \tag{1.4.3}$$

which is the set of feasible directions for the cone of all supporting hyperplanes at $\mathbf{x}$. For future reference it is important to prove that $\mathcal{F}(\mathbf{x}) \;\subseteq\; F(\mathbf{x})$.

Let $\mathbf{s} \;\in\; \mathcal{F}(\mathbf{x})$ then from (1.4.2) there exists a directional sequence $\mathbf{x}^{(k)} \;\rightarrow\; \mathbf{x}$ such that $\mathbf{s}^{(k)} \;\rightarrow\; \mathbf{s}$. Using (1.4.1) and dividing by $\delta^{(k)} \;>\; 0$ it follows that

$$\mathbf{s}^{(k)T}\mathbf{g} \;=\; \frac{(\mathbf{x}^{(k)} \;-\; \mathbf{x})^T\mathbf{g}}{\delta^{(k)}}. \qquad \forall\, \mathbf{g} \;\in\; \partial K(\mathbf{x}) \tag{1.4.4}$$

Now any vector $\mathbf{g} \;\in\; \partial K(\mathbf{x})$ satisfies $(\mathbf{z} \;-\; \mathbf{x})^T\mathbf{g} \;\leq\; 0 \quad \forall\, \mathbf{z} \;\in\; K$. Then since $\mathbf{x}^{(k)}$ are feasible points

$$(\mathbf{x}^{(k)} \;-\; \mathbf{x})^T\mathbf{g} \;\leq\; 0.$$

Hence taking limits in (1.4.4) as $k \;\rightarrow\; \infty$ and $\mathbf{s}^{(k)} \;\rightarrow\; \mathbf{s}$ implies that

$$\mathbf{s}^T\mathbf{g} \;\leq\; 0$$

or $\mathbf{s} \;\in\; F(\mathbf{x})$. Therefore this proves that $\mathcal{F}(\mathbf{x}) \;\subseteq\; F(\mathbf{x})$ for the general case.

For the positive semi–definite matrix cone (1.3.1) similar definitions to (1.4.2) and (1.4.3) hold. If S is a symmetric matrix which is equivalent to a feasible direction in (1.4.3), Z is a basis matrix for the null space of A and Λ is any symmetric positive semi–definite matrix, then using Theorem 1.3.3, (1.4.3) and the inner product Definition 1.2.1, it follows that

$$\begin{aligned} F(A) \;&=\; \{S : \; S \;=\; S^T, \; \langle B, S\rangle \;\leq\; 0 \quad \forall\, B \in \partial K_{\Re}(A)\} \\ &=\; \{S : \; S \;=\; S^T, \;\; \langle -Z\Lambda Z^T, S\rangle \;\leq\; 0 \quad \forall\, \Lambda \;\geq\; 0\} \\ &=\; \{S : \; S \;=\; S^T, \;\; \langle \Lambda, Z^TSZ\rangle \;\geq\; 0 \quad \forall\, \Lambda \;\geq\; 0\} \end{aligned}$$

and hence

$$F(A) \;=\; \{S : \; S \;=\; S^T, \;\; Z^TSZ \;\geq 0\}. \tag{1.4.5}$$

The following theorem is due to Fletcher [1985].

Theorem 1.4.1

For $A \;\in\; K_{\Re}$

$$\mathcal{F}(A) \equiv F(A) \tag{1.4.6}$$

Proof

In general we proved that $\mathcal{F}(A) \subseteq F(A)$ above. Now the converse is considered.

Take a direction $S \in F$ and let $X = [Y \;\; Z]$ be the eigenvector matrix for A which is described in Theorem 1.3.3 and Λ_r the diagonal matrix of nonzero eigenvalues.

There are two cases, first when $Z^TSZ \geq 0$ and singular, consider the trajectory

$$A_\epsilon = A + \epsilon S + \beta\epsilon^2 I \tag{1.4.7}$$

which gives

$$X^TA_\epsilon X = [Y \;\; Z]^TA + \epsilon S + \beta\epsilon^2 I[Y \;\; Z]$$

$$= \begin{bmatrix} Y^TAY + \epsilon Y^TSY + \beta\epsilon^2 Y^TY & Y^TAZ + \epsilon Y^TSZ + \beta\epsilon^2 Y^TZ \\ Z^TAY + \epsilon Z^TSY + \beta\epsilon^2 Z^TY & Z^TAZ + \epsilon Z^TSZ + \beta\epsilon^2 Z^TZ \end{bmatrix}.$$

Then, since $A = Y\Lambda_rY^T$ and Z is the basis matrix for the null space of A, it follows that

$$X^TA_\epsilon X = \begin{bmatrix} \Lambda_r + \epsilon Y^TSY + \beta\epsilon^2 & \epsilon Y^TSZ \\ \epsilon Z^TSY & \epsilon Z^TSZ + \beta\epsilon^2 \end{bmatrix}. \tag{1.4.8}$$

Now

$$\Lambda_r + \epsilon Y^TSY + \beta\epsilon^2 > 0 \tag{1.4.9}$$

and

$$\epsilon Z^TSZ + \beta\epsilon^2 - \epsilon^2 Y^TSZ(\Lambda_r + \epsilon Y^TSY + \beta\epsilon^2)^{-1}Z^TSY \geq 0 \tag{1.4.10}$$

are going to be proved. If $\beta > \|\Lambda_r^{-1}\|\|S\|^2$ is chosen and for ϵ sufficiently small, then clearly (1.4.9) and (1.4.10) are true by strength of $\Lambda_r > 0$ for (1.4.9) and $Z^TSZ > 0$ for (1.4.10). Hence there exist Choleski factors for (1.4.9) and (1.4.10) which enable us to construct a Choleski factor for (1.4.8). Therefore $X^TA_\epsilon X$ is positive semi–definite or equivalently A_ϵ is feasible.

For the second case when $Z^TSZ > 0$ consider the trajectory

$$A_\epsilon = A + \epsilon S. \tag{1.4.11}$$

Similar to the first case it gives

$$X^T A_\epsilon X \ = \ \begin{bmatrix} \Lambda_r \ + \ \epsilon Y^T SY & \epsilon Y^T SZ \\ \epsilon Z^T SY & \epsilon Z^T SZ \end{bmatrix}, \tag{1.4.12}$$

hence A_ϵ is feasible since

$$\Lambda_r \ + \ \epsilon Y^T SY \ > \ 0 \tag{1.4.13}$$

and

$$\epsilon Z^T SZ \ - \ \epsilon^2 Y^T SZ(\Lambda_r \ + \ \epsilon Y^T SY)^{-1} Z^T SY \ \geq \ 0. \tag{1.4.14}$$

Thus in both cases a direction $S \ \in \ \mathcal{F}(A)$ is constructed and if we take $\epsilon \ = \ \epsilon_k$ for any sequence $\epsilon_k \ \to \ 0$ then there exists a feasible directional sequence in $\mathcal{F}$. Therefore $F \ \subset \ \mathcal{F}$ proving that these sets are in fact equivalent. $\Box$

From this theorem we can deal with F which is easier to operate than $\mathcal{F}$. In this section expression (1.4.5) provides a characterization of a feasible direction of search. The benefit of this expression along with the normal cone expression (1.3.12) lies in their application to optimization problems. The expression for the normal cone plays the part of the subdifferential in the statement of optimality conditions. The expression for the feasible direction accommodates a characterization of a feasible direction of search which is easily verified.

1.5 First and second order conditions

The content of this section is useful in deriving the methods in Sections 5.3 and 6.4.

This section includes a useful theorem of first order conditions. Also at the end of this section second order conditions are stated. It is also shown how to compute a basis matrix Z for the null space of A in connection with the partial LDL^T factorization of A.

Consider the following problem

$$\begin{aligned} &\textit{minimize} \quad f(A) \\ &\textit{subject to} \ \ A \ \in \ K_{\Re}, \quad c_i(A) \ \leq \ 0. \quad i = 1, \ldots, m \end{aligned} \tag{1.5.1}$$

The problem of minimizing a convex function $f(A)$ on a general convex set K is said to be a convex programming problem. A special case of (1.5.1) occurs when $K = K_{\Re} \cup K_c$ where

$$K_c = \{A : A \in \Re^{n \times n} \ c_i(A) \leq 0. \quad i = 1, \ldots, m\}.$$

is a convex set (this is assured if the functions $c_i(A)$ are convex).

A *local solution* is a point at which, in a neighbourhood about that point, has no feasible point that gives a smaller value of the objective function.

Theorem 1.5.1

Every local solution $\mathbf{x}^*$ to a convex programming problem is a global solution.

Proof

Let A^* be a local but not global solution. Then $\exists\, A \in K$ such that $f(A) < f(A^*)$. By convexity of K

$$A_\lambda = (1 - \lambda)A^* + \lambda A.$$

By convexity of f

$$\begin{aligned} f(A_\lambda) &\leq (1 - \lambda)f(A^*) + \lambda f(A) \\ &= f(A^*) + \lambda(f(A) - f(A^*)) \\ &< f(A^*). \end{aligned} \tag{1.5.2}$$

Taking $\lambda \rightarrow 0$ in the limit there exists $f(A_\lambda)$ in the neighbourhood of $f(A^*)$ which contradictes the local solution property. Thus local solutions are global. □

In a convex programming problem every local solution is a global solution which has been proved above. If $f(A)$ and $c_i(A)$ $i = 1, \ldots, m$ are convex and nonsmooth then the first order necessary conditions can be given in the following theorem

Theorem 1.5.2 (*First order conditions*)

If A^* solves (1.5.1) and if the condition in Theorem 1.4.1 holds then A^* is feasible and there exist Lagrange multipliers $\Lambda^* \geq 0$ and $\boldsymbol{\pi}^* \geq 0$ satisfying the following

$$\sum_{i=1}^{m} \pi_i^* c_i^* = 0$$

and

$$\nabla_A \mathcal{L}(A^*, \Lambda^*, \boldsymbol{\pi}^*) = G^* + B^* + \sum_{i=1}^{m} \pi_i^* C_i^* = 0 \tag{1.5.3}$$

where $G^* \in \nabla f^*$, $B^* \in \partial K_{\Re}^*$ and $C_i^* = \nabla c_i^*$ $i = 1, \ldots, m$. (Note that the operator ∇ maps a scalar into a matrix).

Proof (see for example Rockafellar [1981] Chapter 5)

This theorem is related to the usual Kuhn–Tucker (KT) conditions (e.g. see Fletcher [1987]) and the π_i^* are KT multipliers. However an additional term derived from $\partial K_{\Re}^*$ also occurs.

The conditions in Theorem 1.5.2 are certainly sufficient when all feasible directions are strict ascent directions. However consider situation in which there exist feasible directions along which $f(A)$ has a zero directional derivative. Now higher order terms become significant. Second order information is required in order to provide algorithms that converge rapidly. Also, it is difficult to deal with the matrix cone constraint in (1.5.1), since it is not in the form of a functional constraint. An equivalent problem to (1.5.1) with equality and inequality constraints which are easier to manipulate is considered here. This formulation will enable us to derive algorithms with a second order rate of convergence.

Assume that r, the rank of A^*, $(1 < r < n)$ is known. Permuting rows and columns if necessary, then for A sufficiently close to A^* (which ensure that $D_1 > 0$) the partial factors

$$A = LDL^T \tag{1.5.4}$$

can be calculated, where

$$L = \begin{bmatrix} L_{11} & \\ L_{21} & I \end{bmatrix}, \quad D = \begin{bmatrix} D_1 & \\ & D_2 \end{bmatrix} \tag{1.5.5}$$

$L_{11} \in \Re^{r\times r}$ is unit lower triangular, $D_1 \in \Re^{r\times r}$ is diagonal and positive definite and $D_2 \in \Re^{n-r\times n-r}$. $D_2 = \mathbf{0}$ at the solution, in general we can calculate D_2 as follows, partitioning

$$A = \begin{bmatrix} A_{11} & A_{21}^T \\ A_{21} & A_{22} \end{bmatrix} \tag{1.5.6}$$

where $A_{11} \in \Re^{r\times r}$, from (1.5.5)

$$LDL^T = \begin{bmatrix} L_{11}D_1L_{11}^T & L_{11}D_1L_{21}^T \\ L_{21}D_1L_{11}^T & L_{21}D_1L_{21}^T + D_2 \end{bmatrix} \tag{1.5.7}$$

then

$$A_{22} = L_{21}D_1L_{21}^T + D_2 \tag{1.5.8}$$

and since

$$\begin{aligned} L_{21}D_1L_{21}^T &= (L_{21}D_1L_{11}^T)(L_{11}^{-T}D_1^{-1}L_{11}^{-1})(L_{11}D_1L_{21}^T) \\ &= A_{21}A_{11}^{-1}A_{21}^T, \end{aligned}$$

therefore

$$D_2(A) = A_{22} - A_{21}A_{11}^{-1}A_{21}^T. \tag{1.5.9}$$

Thus A is positive semi–definite if and only if $D_2 = \mathbf{0}$, thus the constraint $A \in K_\Re$ can be expressed as

$$D_2(A) = \mathbf{0} \tag{1.5.10}$$

This gives a ready expression which can be used to compute both first and second derivatives of the constraints with respect to the elements of A.

The orthonormal basis matrix Z for the null space of A^* can be calculated using (1.5.5). Define

$$V = L^{-T} = \begin{bmatrix} L_{11}^{-T} & -L_{11}^{-T}L_{21}^T \\ \mathbf{0} & I \end{bmatrix}$$

then using (1.5.7)

$$\begin{aligned} V &= \begin{bmatrix} L_{11}^{-T} & -A_{11}^{-1}A_{21}^T \\ \mathbf{0} & I \end{bmatrix} \\ &= \begin{bmatrix} V_{11} & V_{21} \\ \mathbf{0} & I \end{bmatrix}. \end{aligned} \tag{1.5.11}$$

Then

$$Z = \begin{bmatrix} V_{21} \\ I \end{bmatrix}.$$

From (1.5.9)

$$\begin{aligned} D_2(A) &= A_{22} - A_{21}A_{11}^{-1}A_{21}^T \\ &= [-A_{11}^{-1}A_{21}^T \quad I] \begin{bmatrix} A_{11} & A_{21}^T \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} -A_{11}^{-1}A_{21}^T \\ I \end{bmatrix} \\ &= Z^TAZ = \mathbf{0}. \end{aligned} \tag{1.5.12}$$

Then problem (1.5.1) can be expressed in the equivalent form

$$\begin{aligned} &\textit{minimize} \quad f(A) \\ &\textit{subject to} \;\; Z^TAZ = \mathbf{0} \quad c_i(A) \le 0. \quad i = 1, \ldots, m. \end{aligned} \tag{1.5.13}$$

It is convenient to introduce the Lagrangian function

$$\mathcal{L}(A, \Lambda, \boldsymbol{\pi}) = f(A) - \langle \Lambda, Z^TAZ \rangle + \sum_{i=1}^{m} \pi_i c_i(A) \tag{1.5.14}$$

in which Λ and $\boldsymbol{\pi}$ are Lagrange multipliers for the constraints (1.5.12) and $\mathbf{c}(A) \leq \mathbf{0}$ respectively. (Fletcher [1987] (Theorem 9.1.1)). Since $\langle \Lambda, Z^T A Z \rangle = \langle A, Z^T \Lambda Z \rangle$, then A^*, Λ^* and $\boldsymbol{\pi}^*$ satisfy

$$\nabla_A \mathcal{L}(A^*, \Lambda^*, \boldsymbol{\pi}^*) = \nabla_A f^* - Z^T \Lambda^* Z + \sum_{i=1}^{m} \pi_i^* \nabla_A c_i^* = \mathbf{0} \qquad (1.5.15)$$

This equation corresponds to (1.5.3). (Note that Λ and $\boldsymbol{\pi}$ not necessarily the same as the Λ and $\boldsymbol{\pi}$ in Theorem 1.5.2).

The matrix Λ that appears in the normal cone expression (1.3.12) can be defined as the Lagrange multiplier matrix for the constraints $D_2(A) = \mathbf{0}$ relative to the basis Z.

This treatment of second order conditions was given by Fletcher [1985].

1.6 Quasi–Newton methods

In this section the problem of finding a local solution to the problem

$$\underset{\mathbf{x}}{\text{minimize}} \ f(\mathbf{x}), \qquad \mathbf{x} \in \Re^n \qquad (1.6.1)$$

is considered. The function f is smooth and not necessary convex. In the previous section we dealt with optimization problems which have various types of constraint. However in this section the optimum value is sought of an objective function of many variables without any constraint. This type of problem will arise in Chapter 3.

The present section will be devoted to the study of quasi–Newton methods. First, the Newton method will be discussed in order to show how the quasi–Newton method is derived from it.

The idea behind Newton's method is to replace the function $f(\mathbf{x})$ in the equation to be minimized $(f(\mathbf{x}) = 0)$ by a quadratic model that approximates the function. The quadratic model is obtained from the first three terms of a Taylor series expansion of $f(\mathbf{x})$ about $\mathbf{x}^{(k)}$ as follows

$$f(\mathbf{x}^{(k)} + \boldsymbol{\delta}) \approx f^{(k)} + \mathbf{g}^{(k)T}\boldsymbol{\delta} + \tfrac{1}{2}\ \boldsymbol{\delta}^T G^{(k)} \boldsymbol{\delta} = q^{(k)}(\boldsymbol{\delta}) \tag{1.6.2}$$

where $\boldsymbol{\delta} = \mathbf{x} - \mathbf{x}^{(k)}$, and $q^{(k)}(\boldsymbol{\delta})$ is the resulting quadratic approximation for iteration k. With the requirement that the first and the second derivatives of $f(\mathbf{x})$ are known at any point, then the coefficients $f^{(k)}$, $\mathbf{g}^{(k)}$ and $G^{(k)}$ are also known. The kth iteration of Newton's method can be stated as follows:

Algorithm 1.6.1 (*Newton method*)

Let $G^{(k)}$ be an $n \times n$ positive definite matrix then the following algorithm computes the local minimum $\mathbf{x}^*$ for $f(\mathbf{x})$

```
i.   Select initial point $\mathbf{x}^{(0)} \in \Re^n$.
ii.  Solve $G^{(k)}\boldsymbol{\delta} = -\mathbf{g}^{(k)}$ for $\boldsymbol{\delta} = \boldsymbol{\delta}^{(k)}$
iii. Set $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \boldsymbol{\delta}^{(k)}$.
iv.  If $\mathbf{g}^{(k)} \approx \mathbf{0}$
          stop
     else
          go to (ii).
```

The following theorem by Fletcher [1987] gives the rate of convergence for Newton method.

Theorem 1.6.2

If f is twice continuously differentiable, $\mathbf{x}^{(k)}$ is sufficiently close to $\mathbf{x}^*$ for some k, G^* is nonsingular and $G(\mathbf{x})$ satisfies a *Lipschitz condition* $\|G(\mathbf{x}) - G(\mathbf{y})\| \leq \|\mathbf{x} - \mathbf{y}\|$ in a neighbourhood of a local minimizer $\mathbf{x}^*$, then $\lim_{k\to\infty} \mathbf{x}^{(k)} = \mathbf{x}^*$ and Newton's algorithm converges at second order.

Proof

Since f is differentiable, a Taylor series for $\mathbf{g}\,(\mathbf{x}^{(k)} + \mathbf{h}\,)$ about $\mathbf{x}^{(k)}$ exists and can be written as

$$\mathbf{g}(\mathbf{x}^{(k)} + \mathbf{h}) = \mathbf{g}^{(k)} + G^{(k)}\mathbf{h} + O(\|\mathbf{h}\|^2). \tag{1.6.3}$$

Denote $\mathbf{h}^{(k)} = \mathbf{x}^{(k)} - \mathbf{x}^*$. Letting $\mathbf{h} = -\mathbf{h}^{(k)}$ gives

$$\mathbf{0} = \mathbf{g}^* = \mathbf{g}(\mathbf{x}^{(k)} - \mathbf{h}^{(k)}) = \mathbf{g}^{(k)} - G^{(k)}\mathbf{h}^{(k)} + O(\|\mathbf{h}^{(k)}\|^2). \tag{1.6.4}$$

Multiplying equation (1.6.4) by $G^{(k)-1}$ gives

$$\mathbf{0} = -\boldsymbol{\delta}^{(k)} - \mathbf{h}^{(k)} + O(\|\mathbf{h}^{(k)}\|^2) = -\mathbf{h}^{(k+1)} + O(\|\mathbf{h}^{(k)}\|^2) \tag{1.6.5}$$

Hence, by definition of $O(\mathbf{h})$ (see Definition 1.2.6), there exists a constant $c > 0$ such that

$$\|\mathbf{h}^{(k+1)}\| \leq c\, \|\mathbf{h}^{(k)}\|^2. \tag{1.6.6}$$

Let $\mathbf{x}^{(k)}$ be in a neighbourhood of $\mathbf{x}^*$ with $\|\mathbf{h}\| \leq \alpha/c$, where $0 < \alpha < 1$, then this implies that $\|\mathbf{h}^{(k+1)}\| \leq \alpha\|\mathbf{h}^{(k)}\|$. Thus $\mathbf{x}^{(k)} \to \mathbf{x}^*$ since $\|\mathbf{h}^{(k)}\| \to 0$ and relation (1.6.6) shows that Newton's algorithm converges at second order. $\square$

The basic Newton method as it stands is not suitable for general purposes because $G^{(k)}$ may not be positive definite when $\mathbf{x}^{(k)}$ is far away from the solution $\mathbf{x}^*$. Sometimes even if $G^{(k)}$ is positive definite Newton's method may not converge, and $\{f^{(k)}\}$ may not even decrease.

Now, the concept of the *line search* is introduced. The line search algorithms have the following structure:

given an initial estimate $\mathbf{x}^{(0)}$ the basic structure of the kth iteration is

i. determine a direction of search $\mathbf{s}^{(k)}$

ii. find $\alpha^{(k)}$ to minimize $f(\mathbf{x}^{(k)} + \alpha^{(k)}\mathbf{s}^{(k)})$ with respect to $\alpha^{(k)}$

iii. set $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha^{(k)}\mathbf{s}^{(k)}$.

There are different methods which correspond to different ways of choosing $\mathbf{s}^{(k)}$. The line search subproblem in step (ii) is carried out by repeatedly sampling $f(\mathbf{x})$ and possibly its derivatives for different points $\mathbf{x} = \mathbf{x}^{(k)} + \alpha \mathbf{s}^{(k)}$ along the line. In practice step (ii) is solved approximately and the aim of the line search is to find a step $\alpha^{(k)}$ which gives a significant reduction in f on each iteration. (see Fletcher [1987] Sections 2.5 and 2.6 for more about the line search).

However the main difficulty in Newton's method arises from supplying the second derivative matrix G. Methods similar to Newton's method, and not requiring the second derivative, can be derived. Quasi–Newton methods are descent methods which approximate G^{-1} by a symmetric positive definite matrix $H^{(k)}$. The popularity of the most successful of these methods stems from the fact that they exhibit a fast rate of convergence while avoiding the second derivative calculations associated with Newton's method.

The quasi–Newton algorithm takes the following form.

Algorithm 1.6.3 (*quasi–Newton method*)

```
  i. Select initial point $\mathbf{x}^{(0)} \in \Re^n$.
 ii. Set $\mathbf{s}^{(k)} = - H^{(k)}\mathbf{g}^{(k)}$
iii. Line search along $\mathbf{s}^{(k)}$ giving $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha^{(k)}\mathbf{s}^{(k)}$
 iv. Update $H^{(k)}$ giving $H^{(k+1)}$ .
  v. If $\mathbf{g}^{(k)} \approx \mathbf{0}$
         stop
     else
         go to (ii).
```

The initial matrix $H^{(0)}$ is an arbitrary positive definite matrix, $H^{(0)} = I$ is the first choice if there is no better estimate. There are various possible formulas for updating the positive definite matrix H. An important formula was suggested independently by Broyden [1970], Fletcher [1970], Goldfarb [1970] and Shanno [1970], and is known as the BFGS formula

$$\begin{aligned} H^{(k+1)}_{BFGS} = H^{(k)} &+ \{1 + \frac{\boldsymbol{\gamma}^{(k)T} H^{(k)} \boldsymbol{\gamma}^{(k)}}{\boldsymbol{\delta}^{(k)T}\boldsymbol{\gamma}^{(k)}}\} \frac{\boldsymbol{\delta}^{(k)}\boldsymbol{\delta}^{(k)T}}{\boldsymbol{\delta}^{(k)T}\boldsymbol{\gamma}^{(k)}} \\ &- \frac{\boldsymbol{\delta}^{(k)}\boldsymbol{\gamma}^{(k)T} H^{(k)} + H^{(k)}\boldsymbol{\gamma}^{(k)}\boldsymbol{\delta}^{(k)T}}{\boldsymbol{\delta}^{(k)T}\boldsymbol{\gamma}^{(k)}}. \end{aligned} \tag{1.6.7}$$

where

$$\gamma^{(k)} = \mathbf{g}^{(k+1)} - \mathbf{g}^{(k)}.$$

and

$$\boldsymbol{\delta}^{(k)} = \alpha^{(k)}\mathbf{s}^{(k)} = \mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}.$$

There is growing evidence that the BFGS formula is the best general purpose quasi–Newton method currently available and it is an efficient technique for unconstrained optimization. Therefore, this formula will be used in this thesis. For more discussion about Newton and quasi–Newton methods with references see Fletcher [1987].

1.7 The l_1 SQP method

This section is devoted to constrained optimization in which additional constraints arise while in the previous section we had only objective functions. The methods in this section deal with constraints which are easier to handle than the constraints in previous sections. The constraints here are expressed in terms of equations and inequalities instead of sets and cones. Methods arising in this section are useful in deriving related methods in Sections 5.3 and 6.4.

This section will be devoted to the study of l_1 SQP method. First it will be shown how the l_1 SQP method is derived from the SQP method. The SQP method is also called the *Lagrange–Newton method.*

Consider the following equality constraint problem

$$\begin{aligned} &\underset{\mathbf{x}}{\text{minimize}} \quad f(\mathbf{x}) \\ &\textit{subject to} \;\; \mathbf{c}(\mathbf{x}) = \mathbf{0}. \end{aligned} \tag{1.7.1}$$

The idea behind the SQP method is to iterate on the basis of certain approximations to the problem function $f(\mathbf{x})$ and $\mathbf{c}(\mathbf{x})$ using a linear approximation to the constraint function $\mathbf{c}(\mathbf{x})$. This method is Newton's method applied to find the stationary point of the Lagrangian function

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) \;=\; f(\mathbf{x}) \;-\; \sum_i \lambda_i c_i(\mathbf{x}) \tag{1.7.2}$$

The variables in the Lagrangian function are $\mathbf{x}$ and $\boldsymbol{\lambda}$. The method generates a sequence of approximations $\mathbf{x}^{(k)}$ and $\boldsymbol{\lambda}^{(k)}$ to the solution vector $\mathbf{x}^*$ and the Lagrange multipliers $\boldsymbol{\lambda}^*$.

A Taylor series for $\nabla_{\mathbf{x},\boldsymbol{\lambda}}\, \mathcal{L}$ about $\mathbf{x}^{(k)}$, $\boldsymbol{\lambda}^{(k)}$ gives

$$\begin{aligned} \nabla_{\mathbf{x},\boldsymbol{\lambda}}\, \mathcal{L}(\mathbf{x}^{(k)} \;+\; \boldsymbol{\delta}\mathbf{x},\; \boldsymbol{\lambda}^{(k)} \;+\; \boldsymbol{\delta\lambda}) \;&=\; \nabla_{\mathbf{x},\boldsymbol{\lambda}}\, \mathcal{L}(\mathbf{x}^{(k)},\; \boldsymbol{\lambda}^{(k)}) \\ &+\; [\nabla^2_{\mathbf{x},\boldsymbol{\lambda}}\, \mathcal{L}(\mathbf{x}^{(k)},\; \boldsymbol{\lambda}^{(k)})] \begin{bmatrix} \boldsymbol{\delta}\mathbf{x} \\ \boldsymbol{\delta\lambda} \end{bmatrix} \;+\; \ldots \end{aligned}$$

where $\boldsymbol{\delta\lambda} \;=\; \boldsymbol{\lambda}^* \;-\; \boldsymbol{\lambda}^{(k)}$, $\boldsymbol{\delta}\mathbf{x} \;=\; \mathbf{x}^* \;-\; x^{(k)}$. Neglecting higher order terms

$$\nabla^2_{\mathbf{x},\boldsymbol{\lambda}}\, \mathcal{L}(\mathbf{x}^{(k)},\; \boldsymbol{\lambda}^{(k)}) \begin{bmatrix} \boldsymbol{\delta}\mathbf{x} \\ \boldsymbol{\delta\lambda} \end{bmatrix} \;=\; -\, \nabla_{\mathbf{x},\boldsymbol{\lambda}}\, \mathcal{L}(\mathbf{x}^{(k)},\; \boldsymbol{\lambda}^{(k)}). \tag{1.7.3}$$

Since $\nabla_{\mathbf{x},\boldsymbol{\lambda}}\, \mathcal{L}(\mathbf{x}^*,\; \boldsymbol{\lambda}^*) \;=\; \nabla_{\mathbf{x},\boldsymbol{\lambda}}\, \mathcal{L}(\mathbf{x}^{(k)} \;+\; \boldsymbol{\delta}\mathbf{x},\; \boldsymbol{\lambda}^{(k)} \;+\; \boldsymbol{\delta\lambda}) \;=\; \mathbf{0}$. Using (1.7.2) to find $\nabla_{\mathbf{x},\boldsymbol{\lambda}}\, \mathcal{L}$ and $\nabla^2_{\mathbf{x},\boldsymbol{\lambda}}\, \mathcal{L}$, gives the system

$$\begin{bmatrix} W^{(k)} & -A^{(k)} \\ -A^{(k)T} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\delta}\mathbf{x} \\ \boldsymbol{\delta\lambda} \end{bmatrix} \;=\; \begin{bmatrix} -\mathbf{g}^{(k)} \;+\; A^{(k)}\boldsymbol{\lambda}^{(k)} \\ \mathbf{c}^{(k)} \end{bmatrix} \tag{1.7.4}$$

where $\mathbf{g} \;=\; \nabla_{\mathbf{x}}\, f$, A is the *Jacobian matrix* of constraint $\mathbf{c}(\mathbf{x}^{(k)})$, and

$$W^{(k)} \;=\; \nabla^2_{\mathbf{x}} f(\mathbf{x}^{(k)}) \;-\; \sum_i \lambda_i^{(k)} \nabla^2_{\mathbf{x}} c_i(\mathbf{x}^{(k)}) \tag{1.7.5}$$

is the Hessian matrix $\nabla^2_{\mathbf{x}}\, \mathcal{L}(\mathbf{x}^{(k)}, \boldsymbol{\lambda}^{(k)})$. The system (1.7.4) is solved to give corrections $\boldsymbol{\delta}\mathbf{x}$ and $\boldsymbol{\delta\lambda}$.

An equivalent system to (1.7.4) is

$$\begin{bmatrix} W^{(k)} & -A^{(k)} \\ -A^{(k)T} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\delta}^{(k)} \\ \boldsymbol{\lambda}^{(k+1)} \end{bmatrix} \;=\; \begin{bmatrix} -\mathbf{g}^{(k)} \\ \mathbf{c}^{(k)} \end{bmatrix} \tag{1.7.6}$$

where $\boldsymbol{\lambda}^{(k+1)} = \boldsymbol{\lambda}^{(k)} + \boldsymbol{\delta\lambda}$ and $\boldsymbol{\delta}^{(k)} = \boldsymbol{\delta}\mathbf{x}$. System (1.7.6)is used to determine $\boldsymbol{\delta}^{(k)}$ and $\boldsymbol{\lambda}^{(k+1)}$, then $\mathbf{x}^{(k+1)}$ is given by

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \boldsymbol{\delta}^{(k)}. \tag{1.7.7}$$

This method requires initial approximations $\mathbf{x}^{(0)}$ and $\boldsymbol{\lambda}^{(0)}$, and uses (1.7.6) and (1.7.7) to generate the iterative sequence $\{\mathbf{x}^{(k)}, \boldsymbol{\lambda}^{(k)}\}$.

Similar to Newton's method in the previous section it is possible to restate this method in terms of one in which the subproblem involves the minimization of a quadratic function. Consider the subproblem

$$\begin{aligned} &\underset{\boldsymbol{\delta}}{\text{minimize}} \quad q^{(k)}(\boldsymbol{\delta}) = \tfrac{1}{2}\, \boldsymbol{\delta}^T W^{(k)} \boldsymbol{\delta} + \mathbf{g}^{(k)T}\boldsymbol{\delta} + f^{(k)} \\ &\textit{subject to } \mathbf{l}^{(k)}(\boldsymbol{\delta}) = A^{(k)T}\boldsymbol{\delta} + \mathbf{c}^{(k)} = \mathbf{0} \end{aligned} \tag{1.7.8}$$

This problem is the *quadratic programming subproblem* (QPS). Equations (1.7.6) gives the first order conditions for problem (1.7.8). If the reduced matrix $Z^{(k)T}W^{(k)}Z^{(k)}$ is positive definite then $\boldsymbol{\delta}^{(k)}$ minimizes (1.7.8), where $Z^{(k)}$ is the null matrix for $A^{(k)}$. Hence the following algorithm is suggested.

Algorithm 1.7.1

```
Given initial estimate $\mathbf{x}^{(0)}, \boldsymbol{\lambda}^{(0)}$

For $k = 1, 2, \ldots$
    i. Solve (1.7.8) to determine $\boldsymbol{\delta}^{(k)}$ and $\boldsymbol{\lambda}^{(k+1)}$ the vector of Lagrange multipliers of
       the linear constraints.
    ii. Set $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \boldsymbol{\delta}^{(k)}$.
```

This algorithm is known as the SQP algorithm.

Algorithm 1.7.1 suggests a generalization for solving the nonlinear inequality constraint problem

$$\underset{\mathbf{x}}{\text{minimize}} \quad f(\mathbf{x})$$

$$\textit{subject to} \ \ \mathbf{c}(\mathbf{x}) \ \geq \ \mathbf{0}. \tag{1.7.9}$$

Replacing $\mathbf{c}(\mathbf{x})$ by $\mathbf{l}^{(k)}(\boldsymbol{\delta})$ and $f(\mathbf{x})$ by $q^{(k)}(\boldsymbol{\delta})$ leads to the subproblem

$$\begin{aligned} &\underset{\boldsymbol{\delta}}{\text{minimize}} \quad q^{(k)}(\boldsymbol{\delta}) \ = \ \tfrac{1}{2}\, \boldsymbol{\delta}^T W^{(k)} \boldsymbol{\delta} \ + \ \mathbf{g}^{(k)T} \boldsymbol{\delta} \ + \ f^{(k)} \\ &\textit{subject to} \ \ \mathbf{l}^{(k)}(\boldsymbol{\delta}) \ = \ A^{(k)T} \boldsymbol{\delta} \ + \ \mathbf{c}^{(k)} \ \geq \ \mathbf{0}. \end{aligned} \tag{1.7.10}$$

This QPS can be used in an iterative scheme like Algorithm 1.7.1 in a similar way using (1.7.10) instead of (1.7.8) in step i.

The second order convergence of iteration (1.7.6) and (1.7.7) follows by using the technique of Theorem 1.6.2 applied to the system of $n+m$ equations $\nabla_{\mathbf{x},\boldsymbol{\lambda}} \ \mathcal{L}(\mathbf{x}, \ \boldsymbol{\lambda}) \ = \ \mathbf{0}$. The convergence of this method at a rapid rate can be proved when $\mathbf{x}^{(0)}$ and $\boldsymbol{\lambda}^{(0)}$ are sufficiently close to $\mathbf{x}^*$ and $\boldsymbol{\lambda}^*$ for some k. In fact a stronger result is given by Fletcher [1987] in the following theorem.

Theorem 1.7.2

If $\mathbf{x}^{(0)}$ is sufficiently close to $\mathbf{x}^*$, the Lagrangian matrix

$$\begin{bmatrix} W^{(k)} & -A^{(k)} \\ -A^{(k)T} & \mathbf{0} \end{bmatrix}$$

is non–singular, and if second order sufficient conditions hold at $\mathbf{x}^*, \boldsymbol{\lambda}^*$ with A^* having full rank, then the QPS iteration (1.7.6) and (1.7.7) converges at second order. If $\boldsymbol{\lambda}^{(k)}$ is sufficiently close to $\boldsymbol{\lambda}^*$, $\boldsymbol{\lambda}^{(0)}$ is suitably chosen and if (1.7.8) is solved uniquely by $\boldsymbol{\delta}^{(0)}$ then the SQP method converges at second order.

Proof (see Fletcher [1987] Chapter 12)

Globally the SQP method may not converge, especially when $\mathbf{x}^{(0)}$ is remote from $\mathbf{x}^*$. However the SQP method is usually modified by the l_1 exact penalty function. The l_1 exact penalty function associated with (1.7.9) is

$$\phi(\mathbf{x}) \;=\; \mu f(\mathbf{x}) \;+\; \sum_{i\in E} |c_i(\mathbf{x})| \;+\; \sum_{i \in I} \max(-c_i(\mathbf{x}), 0) \tag{1.7.11}$$

where E is the set of equality constraints and I the set of inequality constraints. For sufficiently small μ local solutions of the nonlinear programming problem (1.7.9) are equivalent to local solutions of (1.7.11) under wide asssumptions.

Various algorithms based on the use of (1.7.11) have been tried. The function (1.7.11) is not differentiable so it cannot be minimized by conventional methods. The most simple is Han's [1977] method which uses the solution of SQP subproblem (1.7.10) as a search direction, and the next point is accepted only if it significantly reduces the value of $\phi(\mathbf{x})$. An algorithm with better convergence properties is suggested by Fletcher [1981a] in which a different subproblem to (1.7.10) is solved, which takes into account the structure of (1.7.11), but uses the same approximating functions as in (1.7.10). The l_1 SQP method is a direct and efficient approach to nonlinear programming. Fletcher [1981a] shows how to use a step restriction (or trust region) so that the difficulties mentioned above are removed. The method can be explained easily as follows: instead of substituting the Taylor series approximations (1.7.10) into the nonlinear programming problems they are substituted directly into the l_1 exact penalty function (1.7.11), giving a piecewise quadratic approximating function $\psi^{(k)}(\boldsymbol{\delta})$ and hence a QP subproblem

$$\begin{aligned} &\underset{\boldsymbol{\delta}}{\text{minimize}} \quad \psi^{(k)}(\boldsymbol{\delta}) \\ &\textit{subject to} \;\; \|\boldsymbol{\delta}\| \;\le\; \rho^{(k)} \end{aligned} \tag{1.7.12}$$

where

$$\psi^{(k)}(\boldsymbol{\delta}) \;=\; q^{(k)}(\boldsymbol{\delta}) \;+\; \sum_{i\in E} |l_i^{(k)}(\boldsymbol{\delta})| \;+\; \sum_{i\in I} \max(-l_i^{(k)}(\boldsymbol{\delta}), 0). \tag{1.7.13}$$

The subproblem (1.7.12) is solved on each iteration which is of a similar to the QP subproblem (1.7.10). Subproblem (1.7.12) differs from (1.7.10) in that there are no explicit constraints derived from the linear approximations $\mathbf{l}^{(k)}(\boldsymbol{\delta})$. Thus there are no difficulties with an infeasible subproblem. The use of a trust region guarantees boundedness of the subproblem. The norm in (1.7.12) is arbitrary but either the $\|\cdot\|_\infty$ or the $\|\cdot\|_2$ is most likely choice since the subproblem (1.7.12) can then be solved by QP methods.

The radius $\rho^{(k)}$ is the step restriction which is adjusted adaptively in a customary way to be as large as possible subject to reasonable agreement between $\phi(\mathbf{x}^{(k)} + \boldsymbol{\delta})$ and $\psi^{(k)}(\boldsymbol{\delta})$, thus ensuring a significant decrease in the function $\phi(\mathbf{x})$. The *ratio* measures the extent to which ϕ and $\psi^{(k)}$ agree in neighbourhood of $\mathbf{x}^{(k)}$ is defined by

$$r^{(k)} = \frac{\nabla\phi(\mathbf{x}^{(k)})}{\nabla\psi^{(k)}(\boldsymbol{\delta}^{(k)})} \tag{1.7.14}$$

where

$$\nabla\phi(\mathbf{x}^{(k)}) = \phi(\mathbf{x}^{(k)}) - \phi(\mathbf{x}^{(k)} + \boldsymbol{\delta}^{(k)}) \tag{1.7.15}$$

is the *actual reduction* and

$$\nabla\psi^{(k)} = \phi(\mathbf{x}^{(k)}) - \psi^{(k)}(\boldsymbol{\delta}^{(k)}). \tag{1.7.16}$$

is the *predicted reduction.*

These features can be observed in the following algorithm from problem (1.7.9) given by Fletcher [1981a].

Algorithm 1.7.3

This algorithm solves problem (1.7.9)

i. Given $\mathbf{x}^{(k)}, \boldsymbol{\lambda}^{(k)}$ and $\rho^{(k)}$, calculate $f^{(k)}$, $\mathbf{g}^{(k)}$, $\mathbf{c}^{(k)}$, $A^{(k)}$ and $W^{(k)}$ which determine $\phi(\mathbf{x}^{(k)})$ and $\psi^{(k)}(\boldsymbol{\delta}^{(k)})$.

ii. Find a global solution $\boldsymbol{\delta}^{(k)}$ to (1.7.12).

iii. Evaluate $\phi(\mathbf{x}^{(k)} + \boldsymbol{\delta}^{(k)})$ and calculate $\nabla\phi(\mathbf{x}^{(k)}), \nabla\psi^{(k)}$ and $r^{(k)}$.

iv.

$$\begin{aligned} &If \quad r^{(k)} < 0.25 \quad set \quad \rho^{(k+1)} = \|\boldsymbol{\delta}^{(k)}\|/4 \\ &if \ \ r^{(k)} > 0.75 \ \ and \ \ \|\boldsymbol{\delta}^{(k)}\| = \rho^{(k)} \ \ set \ \ \rho^{(k+1)} = 2\rho^{(k)} \\ &otherwise \ \ set \quad \rho^{(k+1)} = \rho^{(k)}. \end{aligned}$$

v.

$$\begin{aligned} If \quad r^{(k)} \le 0 \qquad & set \quad \mathbf{x}^{(k+1)} = \mathbf{x}^{(k)}, \ \ \boldsymbol{\lambda}^{(k+1)} = \boldsymbol{\lambda}^{(k)} \\ & else \quad \mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \boldsymbol{\delta}^{(k)} \\ & \qquad \boldsymbol{\lambda}^{(k+1)} = multipliers \ \ from \ \ (1.7.12). \end{aligned}$$

The iteration based on (1.7.12) is guaranteed to converge to a Kuhn–Tucker point of (1.7.11) (Fletcher [1987]). Therefore, this algorithm will be used in this thesis (Chapters 5 and 6). For more about SQP and l_1 SQP methods see Fletcher [1987] Sections 12.4 and 14.5.

Chapter 2

Projection methods

2.1 Introduction

The purpose of this chapter is to provide a background about the projection methods for solving certain linear and least distance convex programming problems in which the feasible region is the intersection of a finite number of convex sets. The following least distance convex programming problem which arises in Chapters 3 and 5 is studied.

For a given point $\mathbf{f}$ find the point $\mathbf{x}^*$ which is the unique solution to the least distance problem

$$\begin{aligned} &\mathit{minimize} \;\; \| \mathbf{f} - \mathbf{x} \|_2 \\ &\mathit{subject\ to} \;\; \mathbf{x} \in \bigcap_{i=1}^{m} K_i \end{aligned} \qquad (2.1.1)$$

where $\bigcap_{i=1}^{m} K_i$ is the intersection of a finite number of convex sets K_1 , K_2 ,..., K_m. This minimization problem is one of a wide class of problems which arises in many applications.

Projection methods for solving this kind of problem were first given by von Neumann [1950], later improved by Dykstra [1983] and independently by Han [1988].

First some definitions relating to projections are introduced. If K is a subspace in Hilbert space H then define

$$K^{\perp} = \{\mathbf{x} \in H : \langle \mathbf{x}, \mathbf{y} \rangle = 0 \quad \forall\, \mathbf{y} \in K\}.$$

$K^{\perp}$ called the orthogonal complement of the set K.

Definition 2.1.1

If K is a subspace in Hilbert space, then the projection of $\mathbf{x} \in H$ onto K is $\mathbf{x}_1$ where $\mathbf{x} = \mathbf{x}_1 + \mathbf{x}_2$ and $\mathbf{x}_1 \in K$ and $\mathbf{x}_2 \in K^{\perp}$. The projection is denoted by $P_K(\mathbf{x}) = \mathbf{x}_1$ and $\mathbf{x}_1$ is unique.

To show that $\mathbf{x}_1$ is a unique point, let $\mathbf{x} = \mathbf{x}_1 + \mathbf{x}_2 = \mathbf{y}_1 + \mathbf{y}_2$ with $\mathbf{y}_1 \in K$ and $\mathbf{y}_2 \in K^{\perp}$.

Now since $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle = 0$, it follows that $\langle \mathbf{x}_1, \mathbf{y}_1 + \mathbf{y}_2 - \mathbf{x}_1 \rangle = 0$ and hence

$$\langle \mathbf{x}_1, \mathbf{y}_1 - \mathbf{x}_1 \rangle = 0$$

since $\langle \mathbf{x}_1, \mathbf{y}_2 \rangle = 0$. Likewise from $\langle \mathbf{y}_1, \mathbf{y}_2 \rangle = 0$ it follow that

$$\langle \mathbf{y}_1, \mathbf{x}_1 - \mathbf{y}_1 \rangle = 0.$$

These two equations show that $\langle \mathbf{x}_1 - \mathbf{y}_1, \mathbf{x}_1 - \mathbf{y}_1 \rangle = 0$ and hence

$$\mathbf{x}_1 = \mathbf{y}_1.$$

Theorem 2.1.2

A necessary and sufficient condition that P be a projection onto K is that

i. $\langle P_K(\mathbf{x}), \mathbf{y} \rangle = \langle \mathbf{x}, P_K(\mathbf{y}) \rangle \quad \forall\, \mathbf{x}, \mathbf{y} \in H$

ii. $P_K^2(\mathbf{x}) = P_K(\mathbf{x}) \quad \forall\, \mathbf{x} \in H$

Proof (see von Neumann [1950]).

If $\mathbf{y} \in H$ then the projection map is completely characterized by the condition

$$\langle \mathbf{x} - \mathbf{y}, \mathbf{x} - \mathbf{z} \rangle \leq 0 \qquad \forall\, \mathbf{z} \in K. \tag{2.1.2}$$

where $\mathbf{x} = P_K(\mathbf{y})$. Clearly we can observe from the normal cone definition (1.3.7) that

$$\mathbf{y} - \mathbf{x} \in \partial K(\mathbf{x}). \tag{2.1.3}$$

Also in this chapter the following linear convex programming problem is considered.

$$\begin{aligned} &minimize \quad \mathbf{e}^T\mathbf{x} \qquad \mathbf{x} \in \Re^n \\ &subject\ \ to \quad \mathbf{x} \in \bigcap_{i=1}^{m} K_i \end{aligned} \tag{2.1.4}$$

where $\mathbf{e} = [1, 1, \ldots, 1]^T \in \Re^n$.

Such optimization problems come up in many practical situations, for example in linear programming problem where K is the set of linear constraints, although projection methods are not the best for solving such problems. Here we are interested in the case where one of the K_i is a positive semi–definite matrix cone. An example of this is the educational testing problem in statistics.

In Section 2.2 the algorithms of von Neumann [1950], Dykstra [1983] and Han [1988] are described. The von Neumann algorithm simply iterates using succesive projections on to each K_i, while in the Dykstra and Han algorithms a more complex calculation is made. This section also includes some other important results. In Section 2.3 Glunt [1991] describes a projection method for solving the linear convex programming problem (2.1.4). His idea is to construct a hyperplane in $\Re^n$ and then carry out the method of alternating projections (von Neumann's method) between the convex set K and the hyperplane. His method converges globally. The general theory of Glunt's method for minimizing the linear function subject to a convex set in Hilbert space is given in that section.

2.2 The Dykstra algorithm

In this section the least distance convex programming problem given by (2.1.1) is considered.

The basic idea of the iterated projections was first discussed by von Neumann [1950]. He showed that if $m = 2$, K_1 and K_2 are subspaces of Hilbert space H and P_1 and P_2 are respectively the orthogonal projections onto K_1 and K_2, then the sequence of alternating projections is generated by the following algorithm:

Algorithm 2.2.1 (*von Neumann algorithm*)

Given a point $\mathbf{f}$, in each subsequent iteration 2 vectors are computed as follows :

```
Set   $\mathbf{x}_2^{(0)} = \mathbf{f}$
For   $k = 1, 2, ...$
      Set  $\mathbf{x}_0^{(k)} = \mathbf{x}_2^{(k-1)}$
      For  $i = 1, 2$
           $\mathbf{x}_i^{(k)} = P_i(\mathbf{x}_{i-1}^{(k)})$
      End
End.                                        (2.2.1)
```

The sequence in Algorithm 2.2.1 converges to $P_{K_1 \cap K_2}(\mathbf{f})$, which is the orthogonal projection onto the intersection of K_1 and K_2.

The von Neumann algorithm can be generalised for m subspaces in the following form

Algorithm 2.2.2

Given a point $\mathbf{f}$, subspaces $K_1,\ K_2,\ \ldots,\ K_m$ and the corresponding projections $P_1,\ P_2,\ \ldots,\ P_m$. In each subsequent iteration m vectors are computed as follows :

```
Set   $\mathbf{x}_m^{(0)} = \mathbf{f}$
For   $k = 1, 2, ...$
      Set  $\mathbf{x}_0^{(k)} = \mathbf{x}_m^{(k-1)}$
      For  $i = 1, 2, \ldots, m$
           $\mathbf{x}_i^{(k)} = P_i(\mathbf{x}_{i-1}^{(k)})$
      End
End.                                        (2.2.2)
```

Deutsch [1983] showed that the rate of convergence in Algorithm 2.2.1 decreases with the angle θ between the two subspaces, where $\theta \in [0,\ \frac{\pi}{2}]$ and is defined by

$$\theta \;=\; \cos^{-1}\{ \sup_{\mathbf{a}\in K_1,\mathbf{b}\in K_2} \frac{|\,\langle \mathbf{a} - P_{K_1\cap K_2}\mathbf{a},\ \mathbf{b} - P_{K_1\cap K_2}\mathbf{b}\rangle\,|}{\|\mathbf{a}\|\ \|\mathbf{b}\|}\}$$

where $P_{K_1\cap K_2}$ is the orthogonal projection on to $K_1 \cap K_2$. This result is derived for the case $m = 2$ and nothing is said about the rate of convergence in the general case. Also it is not easy to find the angle between the two subspaces in order to get the rate of convergence.

Cheney and Goldstein [1959] prove some important results. In one of these they showed that if in Algorithm 2.2.1 the subspaces are replaced by convex sets K_1 and K_2, and P_1 and P_2 are respectively the orthogonal projections onto K_1 and K_2, then they gave the following theorem

Theorem 2.2.3

Let K_1 and K_2 be two convex sets in Hilbert space H. Let P_1 and P_2 represent, respectively, the projections onto K_1 and K_2. Given any point $\mathbf{f} \in H$, then algorithm (2.2.1) generate sequences $\{\mathbf{x}_1^{(k)}\}$, $\{\mathbf{x}_2^{(k)}\}$. If one of the sets is compact or finite dimensional and if the distance between them is attained, then the sequences $\{\mathbf{x}_1^{(k)}\}$ and $\{\mathbf{x}_2^{(k}\}$ converge to points $\mathbf{x}_1$ and $\mathbf{x}_2$ respectively such that

$$\|\mathbf{x}_1 - \mathbf{x}_2\|_2 = \inf_{\mathbf{y}_1\in K_1,\ \mathbf{y}_2\in K_2} \|\mathbf{y}_1 - \mathbf{y}_2\|_2 \tag{2.2.3}$$

Proof (See Cheney and Goldstein [1959])

This result is useful in the next section.

Dykstra [1983] pointed out that if K_1 and K_2 are not subspaces then the von Neumann algorithm does not necessarily converge. Likewise, Han [1988] stated that the von Neumann algorithm cannot be applied successfully to problem (2.1.1) for general n. This can be seen from the following simple example in $\Re^2$.

Example 2.2.4

Figure 2.2.1: This example illustrates the failure of von Neumann algorithm to solve problem (2.1.1) for general n.

Let

$$K_1 = \{(x,y) : y \leq 0)\}$$

and

$$K_2 = \{(x,y) : x+y \leq 0)\},$$

then the straightforward projection method does not work for any point $\mathbf{f}$ outside K_1 and K_2 with $x \neq 0$ and $x \neq y$. For example if $\mathbf{x}_2^{(0)} = \mathbf{f} = (1, 1.5)$ then $\mathbf{x}_1^{(1)} = P_1((1, 1.5)) = (1, 0)$ and $\mathbf{x}_2^{(1)} = P_2P_1((1, 1.5)) = P_2((1, 0)) = (0.5, -0.5)$ hence $P_2P_1((0.5, -0.5)) = (0.5, -0.5)$ and Algorithm 2.2.1 stops at $P_2P_1(\mathbf{f}) = (0.5, -0.5)$ while $\mathbf{x}^* = (0, 0)$ (see Figure 2.2.1).

Dykstra's algorithm is based on an ingeniously simple modification of Algorithm 2.2.2. Han

[1988] independently discovered the same algorithm. In both algorithms the outer normal vector $\mathbf{y}_i^{(k)}$ of the set K_i at $\mathbf{x}_i^{(k)}$ is calculated and the previous outer normal $\mathbf{y}_i^{(k-1)}$ is added to $\mathbf{x}_{i-1}^{(k)}$ before projecting it to the set K_i. Therefore, by each projection an old outer normal vector is replaced by a new one and the sequence of normal vectors are intended to converge to a solution of a dual problem of (2.1.1). In the case when all K_i are subspaces then the addition of the normal is unneccessary for the corresponding projection and Algorithm 2.2.2 is recovered (Boyle et. al. [1986]). Dykstra's and Han's algorithm can be described as follows:

Algorithm 2.2.5 (*Dykstra–Han algorithm*)

Given a point $\mathbf{f}$, convex sets $K_1, K_2, ..., K_m$ and the corresponding projections $P_1, P_2, ..., P_m$. Set

$$\mathbf{y}_1^{(0)} = \mathbf{y}_2^{(0)} = ... = \mathbf{y}_m^{(0)} = 0$$

and

$$\mathbf{x}_m^{(0)} = \mathbf{f}$$

Each subsequent iteration will compute $2m$ vectors

$$\begin{array}{l} \mathbf{x}_1^{(k)},\ \mathbf{x}_2^{(k)},\ \ldots,\ \mathbf{x}_m^{(k)} \\ \mathbf{y}_1^{(k)},\ \mathbf{y}_2^{(k)},\ \ldots,\ \mathbf{y}_m^{(k)} \end{array}$$

as follows:

```
set  $\mathbf{x}_0^{(k)} = \mathbf{x}_m^{(k-1)}$
For  $k = 1, 2, ...$
  For  $i = 1, 2, ..., m$
    $\mathbf{z}_i^{(k)} = \mathbf{x}_{i-1}^{(k)} + \mathbf{y}_i^{(k-1)}$
    $\mathbf{x}_i^{(k)} = P_i(\mathbf{z}_i^{(k)})$
    $\mathbf{y}_i^{(k)} = \mathbf{z}_i^{(k)} - \mathbf{x}_i^{(k)}$
  End
End.                                              (2.2.4)
```

The following theorem by Dykstra [1983] gives the convergence result.

Theorem 2.2.6

The vectors $\mathbf{x}_i^{(k)}$ converge to the solution $\mathbf{x}^*$ of (2.1.1) as $k \to \infty$ for $i = 1, 2, \ldots, m.$

Proof (See Dykstra [1983])

The above algorithm is not easy to deal with and an algorithm which is easier to program and cheaper to run is the following

Algorithm 2.2.7

Given a point $\mathbf{f}$, convex sets $K_1, K_2, ..., K_m$ and the corresponding projections $P_1, P_2, ..., P_m$.

```
Let $\mathbf{f}^{(0)} = \mathbf{f}$
For $k = 1, 2, \ldots$
    $\mathbf{f}^{(k+1)} = \mathbf{f}^{(k)} + P_m \ldots P_1(\mathbf{f}^{(k)}) - P_1(\mathbf{f}^{(k)})$
End
```

A proof of how Algorithm 2.2.7 derived from Algorithm 2.2.5 using mathematical induction is now given.

First, we going to denote for $\mathbf{f}^{(0)} = \mathbf{f}$ and $\mathbf{f}^{(k-1)} = \mathbf{z}_1^{(k)}$. Then for $k = 1$

$$\mathbf{z}_1^{(1)} = \mathbf{f}^{(0)}, \quad \mathbf{x}_1^{(1)} = P_1(\mathbf{f}^{(0)}), \quad \mathbf{y}_1^{(1)} = \mathbf{f}^{(0)} - P_1(\mathbf{f}^{(0)}),$$

for $i > 1$

$$\begin{aligned}
\mathbf{z}_i^{(1)} &= \mathbf{x}_{i-1}^{(1)} + \mathbf{y}_i^{(0)} = P_{i-1} \ldots P_1(\mathbf{f}^{(0)}) \\
\mathbf{x}_i^{(1)} &= P_i(\mathbf{z}_i^{(1)}) = P_i \ldots P_1(\mathbf{f}^{(0)}) \\
\mathbf{y}_i^{(1)} &= \mathbf{z}_i^{(1)} - \mathbf{x}_i^{(1)} = P_{i-1} \ldots P_1(\mathbf{f}^{(0)}) - P_i \ldots P_1(\mathbf{f}^{(0)}).
\end{aligned}$$

Then for $k = 2$

$$\mathbf{f}^{(1)} = \mathbf{z}_1^{(2)} = \mathbf{x}_m^{(1)} + \mathbf{y}_1^{(1)} = P_m \ \dots\ P_1(\mathbf{f}^{(0)}) + \mathbf{f}^{(0)} - P_1(\mathbf{f}^{(0)}).$$

Assume it is true for some $k > 2$, where

$$\mathbf{f}^{(k-1)} := \mathbf{z}_1^{(k)} = P_m \ \dots\ P_1(\mathbf{f}^{(k-2)}) + \mathbf{f}^{(k-2)} - P_1(\mathbf{f}^{(k-2)})$$

and

$$\mathbf{z}_i^{(k)} = \mathbf{x}_{i-1}^{(k)} + \mathbf{y}_i^{(k-1)}$$

where

$$\mathbf{y}_i^{(k-1)} := \sum_{l=0}^{k-2} \{P_{i-1} \dots\ P_1(\mathbf{f}^{(l)}) - P_i \dots\ P_1(\mathbf{f}^{(l)})\}.$$

Then for $k+1$, it is clear that

$$P_i(\mathbf{y}_i^{(k-1)}) = 0$$

then

$$\begin{aligned} \mathbf{x}_1^{(k)} &= P_1(\mathbf{z}_1^{(k)}) = P_1(\mathbf{f}^{(k-1)}) \\ \mathbf{x}_i^{(k)} &= P_i(\mathbf{x}_{i-1}^{(k)} + \mathbf{y}_i^{(k-1)}) = P_i(\mathbf{x}_{i-1}^{(k)}) \\ &= P_i \dots\ P_1(\mathbf{f}^{(k-1)}). \qquad for\ i \geq 2 \end{aligned}$$

Also

$$\begin{aligned} \mathbf{y}_i^{(k)} &= \mathbf{z}_i^{(k)} - \mathbf{x}_i^{(k)} \\ &= \mathbf{x}_{i-1}^{(k)} + \mathbf{y}_i^{(k-1)} - \mathbf{x}_i^{(k)} \\ &= P_{i-1} \dots\ P_1(\mathbf{f}^{(k-1)}) + \\ &\qquad \sum_{l=0}^{k-2} \{P_{i-1} \dots\ P_1(\mathbf{f}^{(l)}) - P_i \dots\ P_1(\mathbf{f}^{(l)})\} - \\ &\qquad\qquad P_i \dots\ P_1(\mathbf{f}^{(k-1)}) \\ &= \sum_{l=0}^{k-1} \{P_{i-1} \dots\ P_1(\mathbf{f}^{(l)}) - P_i \dots\ P_1(\mathbf{f}^{(l)}). \end{aligned}$$

Therefore

$$\begin{aligned}
\mathbf{f}^{(k)} \;=\; \mathbf{z}_1^{(k+1)} &= \mathbf{x}_m^{(k)} + \mathbf{y}_1^{(k)} \\
&= P_m \ldots P_1(\mathbf{f}^{(k-1)}) + \mathbf{z}_1^{(k)} - \mathbf{x}_1^{(k)} \\
&= P_m \ldots P_1(\mathbf{f}^{(k-1)}) + \mathbf{f}^{(k-1)} - P_1(\mathbf{f}^{(k-1)}).
\end{aligned}$$

Which is Algorithm 2.2.7. □

In Algorithm 2.2.7 the vectors $\mathbf{y}_i$ for $i > 1$ are not used and saved from calculation which makes it cheaper. Using Boyle and Dykstra [1986] convergence result, we have the following theorem.

Theorem 2.2.8

Given $\mathbf{f}$ and the sequence $\{\mathbf{f}^{(k)}\}$ generated by Algorithm 2.2.7 then $P_i \ldots P_1(\mathbf{f}^{(k)}) \rightarrow \mathbf{d}^*$ the optimal solution of (2.1.1), for any $i \geq 1$.

Proof (See Boyle and Dykstra [1986])

In Example 2.2.4 Figure 2.2.2 shows how Algorithm 2.2.7 works successfully. In Figure 2.2.2 $\mathbf{f}^{(k)}$ is projected onto K_1 then onto K_2 and then subtracting $P_1(\mathbf{f}^{(k)})$ from $\mathbf{f}^{(k)} + P_1P_2(\mathbf{f}^{(k)})$ produces the new $\mathbf{f}^{(k+1)}$. It is clear from Figure 2.2.2 that $P_1(\mathbf{f}^{(k)})$ converges to the optimal solution $\mathbf{0}$ on the x–axis. Also $P_1P_2(\mathbf{f}^{(k)})$ converges to $\mathbf{0}$ on the $x = -y$ axis. Clearly $\mathbf{f}^{(k)}$ converges to $\mathbf{f}^* \neq \mathbf{0}$.

Dykstra [1983], shows that if the K_i are convex cones, then the $\mathbf{x}_i$ converge to the nearest point to the initial point $\mathbf{f}$ in the intersection of the K_i. This result has been extended by Boyle et. al. [1986] to the case where some of the K_i are convex sets. Han [1988] has shown that the algorithm works for general convex sets K_i given that the intersection has nonempty interior. Gaffke and Mathar [1989], show that the interior point condition may be omitted. The result of Boyle et. al. [1986] is enough for our application of projection method in Chapters 3, 5 and 6.

Figure 2.2.2: Illustrates the success of Dykstra–Han algorithm to solve problem (2.1.1) for general n.

2.3 A projection algorithm for linear convex programming problems

This section describes a projection method due to Glunt [1991] for solving the linear convex programming problem (2.1.4). He uses a particular choice of convex sets in an ingenious way. One important linear convex programming problem is the educational testing problem which will be solved by the method of this section in Chapter 6.

Glunt's idea is to take account of the function $\mathbf{e}^T\mathbf{x}$ by defining the hyperplane

$$L_\tau = \{\mathbf{y} \in \Re^n | \; f(\mathbf{y}) = \tau\} \tag{2.3.1}$$

where $f(\mathbf{y}) = \mathbf{e}^T\mathbf{y}$. If τ is chosen such that

$$\tau < \min_{\mathbf{x}\in K} f(\mathbf{x}) \tag{2.3.2}$$

then the sets K and L_τ are disjoint. Given $\mathbf{f} \in \Re^n$ Glunt then applies the von Neumann Algorithm 2.2.1 to the problem

$$\begin{aligned} &\underset{\mathbf{x}}{\text{minimize}} \quad \|\mathbf{f} - \mathbf{x}\|_2 \\ &\textit{subject to} \quad \mathbf{x} \in K \cap L_\tau \end{aligned} \tag{2.3.3}$$

which has no feasible solution. It follows from the Theorem 2.2.3 of Cheney and Goldstein [1959] that the iterates $\mathbf{x}_1^{(k)}$ and $\mathbf{x}_2^{(k)}$ will converge to points $\mathbf{x}_1^* \in L_\tau$ and $\mathbf{x}_2^* \in K$ such that $\|\mathbf{x}_1 - \mathbf{x}_2\|_2$ attains the minimum distance between K and L_τ. It can then be deduced from the relationship of L_τ and $\mathbf{e}^T\mathbf{x}$ (2.3.1), that $\mathbf{x}_2^*$ solves problem (2.1.4).

The von Neumann algorithm involves computing alternately the projections onto L_τ and K. That onto L_τ is straightforward. Glunt suggests that the projection on the $K = \bigcap_{i=1}^m K_i$ is computed by using an inner iteration based on the Dykstra algorithm. It follows from Theorem 2.2.6 that the resulting method is globally convergent.

The following is a statement of the outer (von Neumann) algorithm.

Algorithm 2.3.1

Given an arbitrary $\mathbf{g} \in \Re^n$, convex sets $K_1, K_2, ..., K_m$ and the corresponding projections $P_1, P_2, ..., P_m$.

```
Set    $\mathbf{x}_2^{(0)} = \mathbf{g}$
For    $k = 1, 2, ...$
          Set $\mathbf{x}_0^{(k)} = \mathbf{x}_2^{(k-1)}$
          $\mathbf{x}_1^{(k)} = P_{L_\tau}(\mathbf{x}_0^{(k)})$
          $\mathbf{x}_2^{(k)} = P_K(\mathbf{x}_1^{(k)})$          (2.3.4)
End
```

where $K = \bigcap_{i=1}^{m} K_i$

In this algorithm in every outer iteration $P_K(\mathbf{x}_1^{(k)})$ is calculated by solving the following problem

$$\begin{aligned} &\underset{\mathbf{x}}{\text{minimize}} \quad \|P_{L_\tau}(\mathbf{x}_0^{(k)}) - \mathbf{x}\|_2 \qquad \mathbf{x} \in \Re^n \\ &\textit{subject to} \quad \mathbf{x} \in K = \bigcap_{i=1}^{m} K_i. \end{aligned} \tag{2.3.5}$$

where $\mathbf{f}$ (in problem (2.1.1)) $= P_{L_\tau}(\mathbf{x}_0^{(k)}) = \mathbf{x}_1^{(k)}$ which is an initial point in every outer iteration. Problem (2.3.5) is solved using Algorithm 2.2.7.

The following three figures show how Algorithm 2.3.1 works under different circumstances. In Figure 2.3.1 the convex set K is nonsmooth at the solution and it is seen that Algorithm 2.3.1 terminates in 2 iterations with the point $\mathbf{x}_2^*$ as solution. In Figures 2.3.2 and 2.3.3 K is smooth at the solution and it seen that the solution point $\mathbf{x}_2^*$ is the limit point of the sequence $\{\mathbf{x}_2^{(k)}\}$. It can also be observed that if we make τ smaller (as in Figure 2.3.3 $(\tau = -3)$) then a more rapid rate of convergence is obtained. However a study of the numerical results indicates that the order of convergence is linear or slower. Similar features are observed when Glunt's method is applied to the educational testing problem as described in Chapter 6.

The following theorem, due to Glunt [1991], gives the convergence result for the linear convex programming problems.

Theorem 2.3.2

For any $\mathbf{g} \in \Re^n$, the sequences $\{\mathbf{x}_1^{(k)}\}$ and $\{\mathbf{x}_2^{(k}\}$ generated by Algorithm 2.3.1 converge to $\mathbf{x}_1$ and $\mathbf{x}_2$ respectively. Also the sequence $\{\mathbf{x}_2^{(k)}\}$ converges to the solution of the problem

$$\begin{aligned} &\textit{minimize} \quad f(\mathbf{x}) = \mathbf{e}^T\mathbf{x} \qquad \mathbf{x} \in \Re^n \\ &\textit{subject to} \quad \mathbf{x} \in K. \end{aligned} \tag{2.3.6}$$

The function values $f(\mathbf{x}_2^{(k)})$ decrease strictly monotonically to the minimal value.

Figure 2.3.1: Algorithm 2.3.1 terminates for a nonsmooth convex set.

Figure 2.3.2: Algorithm 2.3.1 converges for a smooth convex set.

Figure 2.3.3: Making τ smaller gives faster convergence.

Proof (Glunt [1991])

The convergence of the two sequences $\{\mathbf{x}_1^{(k)}\}$ and $\{\mathbf{x}_2^{(k}\}$ follows from Theorem 2.2.3. Set

$$\begin{aligned}\mathbf{x}_1^* &= \lim_{k\to\infty} \mathbf{x}_1^{(k)} \\ \mathbf{x}_2^* &= \lim_{k\to\infty} \mathbf{x}_2^{(k)}.\end{aligned}$$

Let $\mathbf{x}_2 = P_K(\mathbf{x}_1)$ then from the characterization of the projection map (2.1.2)

$$\langle \mathbf{x}_2 - \mathbf{x}_1, \mathbf{x}_2 - \mathbf{z} \rangle \leq 0. \qquad \forall \mathbf{z} \in K \tag{2.3.7}$$

Now $\mathbf{x}_1 = P_{L_\tau}(\mathbf{x}_2)$, and L_τ is a hyperplane with the unit vector $\mathbf{e}$, so $P_{L_\tau}(.)$ is easy to compute

$$P_{L_\tau}(\mathbf{x}_2) = \mathbf{x}_2 + \frac{\tau - \mathbf{e}^T\mathbf{x}_2}{\|\mathbf{e}\|^2}\mathbf{e}. \tag{2.3.8}$$

So from (2.3.7)

$$\langle \mathbf{x}_2 - (\mathbf{x}_2 + \frac{\tau - \mathbf{e}^T\mathbf{x}_2}{\|\mathbf{e}\|^2}\mathbf{e}, \mathbf{x}_2 - \mathbf{z} \rangle \leq 0 \qquad \forall \mathbf{z} \in K$$

$$\Rightarrow \quad \langle (\mathbf{e}^T\mathbf{x}_2 - \tau)\mathbf{e}, \mathbf{x}_2 - \mathbf{z} \rangle \leq 0 \qquad \forall \mathbf{z} \in K$$

$$\Rightarrow \quad (\mathbf{e}^T\mathbf{x}_2 - \tau) \langle (\mathbf{e}, \mathbf{x}_2 - \mathbf{z} \rangle \leq 0 \qquad \forall \mathbf{z} \in K$$

But

$$\tau < \min_{\mathbf{z} \in K} \mathbf{e}^T\mathbf{z}$$

hence $\mathbf{e}^T\mathbf{x}_2 - \tau \geq 0$. Therefore

$$\langle \mathbf{e}, \mathbf{x}_2 - \mathbf{z} \rangle \leq 0. \qquad \forall \mathbf{z} \in K$$

or

$$\mathbf{e}^T\mathbf{x}_2 \leq \mathbf{e}^T\mathbf{z}. \qquad \forall \mathbf{z} \in K$$

Thus $\mathbf{x}_2$ solves (2.3.6).

To demonstrate that $f(\mathbf{x}_2^{(k)})$ for $k = 1, 2, \ldots$ is monotonically decreasing, consider $\mathbf{x}_2^{(k)}$ and write

$$\begin{aligned} \mathbf{x}_1^{(k)} &= P_{L_\tau}(\mathbf{x}_2^{(k-1)}) \\ \mathbf{x}_2^{(k)} &= P_K(\mathbf{x}_1^{(k)}). \end{aligned}$$

Thus $\mathbf{x}_2^{(k)}$ is the nearest point in K to $\mathbf{x}_1^{(k)}$. Therefore unless $\mathbf{x}_2^* = \mathbf{x}_2^{(k-1)} = \mathbf{x}_2^{(k)}$,

$$\|\mathbf{x}_2^{(k)} - \mathbf{x}_1^{(k)}\| < \|\mathbf{x}_2^{(k-1)} - \mathbf{x}_1^{(k)}\|. \tag{2.3.9}$$

Similarly, $\mathbf{x}_1^{(k)}$ is the nearest point in L_r to $\mathbf{x}_2^{(k-1)}$, so unless $\mathbf{x}_1^* = \mathbf{x}_1^{(k-1)} = \mathbf{x}_1^{(k)}$,

$$\|\mathbf{x}_2^{(k)} - \mathbf{x}_1^{(k+1)}\| < \|\mathbf{x}_2^{(k)} - \mathbf{x}_1^{(k)}\|. \tag{2.3.10}$$

Thus from (2.3.9) and (2.3.10)

$$\|\mathbf{x}_2^{(k)} - \mathbf{x}_1^{(k+1)}\| < \|\mathbf{x}_2^{(k-1)} - \mathbf{x}_1^{(k)}\|. \tag{2.3.11}$$

But

$$\begin{aligned}
\mathbf{x}_2^{(k)} - \mathbf{x}_1^{(k+1)} &= \mathbf{x}_2^{(k)} - P_{L_\tau}(\mathbf{x}_2^{(k)}) \\
&= \mathbf{x}_2^{(k)} - (\mathbf{x}_2^{(k)} + \frac{\tau - \mathbf{e}^T\mathbf{x}_2^{(k)}}{\|\mathbf{e}\|^2}\mathbf{e}) \\
&= \frac{\mathbf{e}^T\mathbf{x}_2^{(k)} - \tau}{\|\mathbf{e}\|^2}\mathbf{e}.
\end{aligned}$$

Similarly

$$\mathbf{x}_2^{(k-1)} - \mathbf{x}_1^{(k)} = \frac{\mathbf{e}^T\mathbf{x}_2^{(k-1)} - \tau}{\|\mathbf{e}\|^2}\mathbf{e}.$$

Hence from (2.3.11)

$$\frac{\mathbf{e}^T\mathbf{x}_2^{(k)} - \tau}{\|\mathbf{e}\|^2}\mathbf{e} < \frac{\mathbf{e}^T\mathbf{x}_2^{(k-1)} - \tau}{\|\mathbf{e}\|^2}\mathbf{e}.$$

or

$$\mathbf{e}^T\mathbf{x}_2^{(k)} < \mathbf{e}^T\mathbf{x}_2^{(k-1)}$$

proving that $f(\mathbf{x})$ is monotonically decreasing. $\square$

Chapter 3

Algorithms for finding the nearest Euclidean distance matrix

3.1 Introduction

Symmetric matrices that have nonnegative offdiagonal elements and zero diagonal elements arise as data in many experimental sciences. This occurs when the values are measurements of distances between points in a Euclidean space. Such a matrix is referred to as a Euclidean distance matrix. Because of data errors such a matrix may not be exactly Euclidean and it is desirable to find the best Euclidean matrix which approximates the non–Euclidean matrix. The aim of this chapter is to study methods for solving this problem.

This chapter contains the projection algorithm described by Glunt, Hayden, Hong and Wells [1990]. This algorithm converges linearly or slower and globally using Algorithm 2.2.7. The disadvantage of the projection algorithm is the slow rate of convergence. This can be increased by using a quasi–Newton method which converges at superlinear order. Therefore, new unconstrained methods based on using quasi–Newton methods are described here.

Some applications of the above problem are given in Section 3.2 along with the definition of the Euclidean distance matrix and its characterization. The projection algorithm is given in

Section 3.3. In Section 3.4 various iterative schemes for an unconstrained programming problem are considered. In Section 3.5 other projection methods for solving the nearest Euclidean distance matrix problem are discussed. In Section 3.6 numerical comparisons of projection methods are carried out. Also numerical comparisons between the projection algorithm in Section 3.3 and unconstrained methods in Section 3.4 are carried out. In addition an example is given which gives more illustration of the unconstrained methods.

In Chapter 4 hybrid methods are considered. These methods take the advantage of both the above methods.

3.2 Euclidean distance matrix

Definition 3.2.1 (*Euclidean distance matrix*)

A matrix D is called a Euclidean distance matrix if it satisfies the following conditions:

i. D is a symmetric matrix : $d_{ij} = d_{ji} \quad \forall\, i,j = 1,\ldots,n$

ii. Diagonal elements are all zero: $d_{ii} = 0 \quad \forall\, i = 1,\ldots,n$

iii. There exist n points : $\mathbf{p}_1,\ldots,\mathbf{p}_n$ in $\Re^r$ $(r \leq n-1)$ such that

$$d_{ij} = \| \mathbf{p}_i - \mathbf{p}_j \|_2^2 \quad (1 \leq i,j \leq n).$$

The elements of D are the squared distances between pairs of points in r–dimensional Euclidean space. Now the Euclidean distance matrix problem can be expressed in the following form

Given a real symmetric matrix $F \in \Re^{n\times n}$, find the Euclidean distance matrix $D \in \Re^{n\times n}$ that minimizes

$$\| F - D \|_F. \tag{3.2.1}$$

The distance between A and B is defined by $\| A - B \|_F$.

A matrix F is an $n \times n$ symmetric data matrix with zero diagonal elements whose elements are regarded as approximate squared distances between pairs of points in a r–dimensional

Euclidean space. F is usually a distance matrix of squared distances f_{ij} between n points, e.g. atoms, stars, cities. Therefore, F must have certain obvious properties regardless of how distances are calculated, and how many spatial dimensions are allowed. The properties can be described as follows:

Definition 3.2.2 (*distance matrix*)

A matrix F is called a distance matrix if it satisfies the following conditions:

i. F is a symmetric matrix: $f_{ij} = f_{ji} \quad \forall\, i, j = 1, \ldots, n$

ii. Diagonal elements are all zero: $f_{ii} = 0 \quad \forall\, i = 1, \ldots, n$

iii. All off–diagonal elements are strictly greater than zero:

$$f_{ij} > 0, \qquad \forall\, i \neq j.$$

The motivation for this study arises from the statistical problems of multidimensional scaling and ordination. In multidimensional scaling, an observed matrix is to be approximated by a Euclidean distances in a specified dimension. The differences between observed and fitted distances are minimized. A discussion of types of multidimensional scaling may be found in De Leeuw et. al. [1980]. An application of multidimensional scaling has been applied in geography Colledge and Rushton [1972], cartography Gilbert [1974], genetics Lalouel [1977], archeology Kendall [1971] and biochemistry Crippen [1977,1978]. A broader review of scaling with application and algorithms is given by Young [1984]. A book by Meulman [1986] gives additional related applications in multivariate analysis.

Other applications arise in the conformation of molecular structures from nuclear magnetic resonance data. For a given data matrix a Euclidean distance matrix can be minimized to generate a molecular model in $\Re^3$ (see Havel et. al. [1983] and Crippen [1977,1978]). In conformation calculations Euclidean distance matrices are used to represent the squares of distances between the atoms of a molecular structure. Attempts to determine such a structure by nuclear–magnetic–resonance experiments give rise to a distance matrix F which because of data errors, may not be Euclidean.

Important characterizations for the Euclidean distance matrix which are used in the following sections are given in the following.

Schoenberg [1935] gave a modern characterization of Euclidean distance matrices. Young and Householder [1938] independently obtained similar result given in the following theorem. Schoenberg uses the fact that the first vector $\mathbf{p}_1$ in the Euclidean distance matrix definition can be translated to the origin.

Theorem 3.2.3

The distance matrix $D \in \Re^{n\times n}$ is a Euclidean distance matrix if and only if the $n-1 \times n-1$ symmetric matrix A defined by

$$a_{ij} = \tfrac{1}{2}[\, d_{1i} + d_{1j} - d_{ij}] \quad (2 \le i,j \le n) \tag{3.2.2}$$

is positive semi–definite, and D is irreducibly embeddable in $\Re^r$ $(r < n)$ where $r = rank(A)$.

Moreover, consider the spectral decomposition

$$A = U\Lambda U^T. \tag{3.2.3}$$

Let Λ_r be the matrix of non–zero eigenvalues in Λ and define X by

$$X = U_r\Lambda_r^{1/2}, \quad then \ A = XX^T \tag{3.2.4}$$

where $\Lambda_r^{1/2} \in \Re^{r\times r}$ and $U_r \in \Re^{n-1\times r}$ comprises the corresponding columns of U. Then the columns of X^T furnish coordinate choices for $\mathbf{p}_2,\ \mathbf{p}_3,\ \ldots,\ \mathbf{p}_n$ with $\mathbf{p}_1 = \mathbf{0}$.

Proof

First let D be Euclidean distance matrix and we aim to prove that A is positive semi–definite. Let $\mathbf{x} \in \Re^{n-1}$ then

$$\begin{aligned}\mathbf{x}^T A\mathbf{x} &= \tfrac{1}{2}\sum_{i,j=2}^{n}(d_{1i} + d_{1j} - d_{ij})x_{i-1}x_{j-1}\\ &= \sum_{i=2}^{n} d_{1i}x_{i-1}^2 + \sum_{\substack{i,j=2\\ i<j}}^{n}(d_{1i} + d_{1j} - d_{ij})x_{i-1}x_{j-1}.\end{aligned} \tag{3.2.5}$$

Also

$$\begin{aligned} d_{1i} + d_{1j} - d_{ij} &= \|\mathbf{p}_1 - \mathbf{p}_i\|^2 + \|\mathbf{p}_1 - \mathbf{p}_j\|^2 - \|\mathbf{p}_i - \mathbf{p}_j\|^2 \\ &= \sum_{k=1}^{r} p_{ik}^2 + \sum_{k=1}^{r} p_{jk}^2 - \sum_{k=1}^{r} (p_{ik} - p_{jk})^2 \\ &= 2 \sum_{k=1}^{r} p_{ik}\, p_{jk} \end{aligned} \tag{3.2.6}$$

where $\mathbf{p}_i^T = [p_{i1}, \ \ldots, p_{ir}]$. Hence from (3.2.5) and (3.2.6)

$$\begin{aligned} \mathbf{x}^T A\mathbf{x} &= \sum_{i=2}^{n} x_{i-1}^2 \sum_{k=1}^{r} p_{ik}^2 + 2 \sum_{\substack{i,j=2i \\ i<j}}^{n} x_{i-1}x_{j-1} \sum_{k=1}^{r} p_{ik}p_{jk} \\ &= \sum_{k=1}^{r} (x_1\, p_{2k} + x_2\, p_{3k} + \ldots + x_{n-1}\, p_{nk})^2 \\ &= \sum_{k=1}^{r} (\sum_{i=2}^{n} x_{i-1}p_{ik})^2 \end{aligned}$$

which is always nonnegative.

Conversly, let A be positive semi–definite, we shall prove that D is a Euclidean distance matrix. Let X be the orthogonal matrix defined by (3.2.4). Denote $\mathbf{p}_1 = 0$ and

$$\mathbf{p}_i = X\mathbf{e}_{i-1} \quad i = 2, \ldots, n$$

where $\mathbf{e}_i$ denotes columns of the unit matrix. Now

$$\|\mathbf{p}_1 - \mathbf{p}_i\|^2 = \|\mathbf{p}_i\|^2 = \mathbf{e}_{i-1}^T A\mathbf{e}_{i-1} = a_{ii} = d_{1i}.$$

Note that $A \in \Re^{n-1\times n-1}$ such that

$$A = \begin{bmatrix} a_{22} & \ldots & a_{2n} \\ \vdots & \ddots & \vdots \\ a_{2n} & \ldots & a_{nn} \end{bmatrix}.$$

Also,

$$\begin{aligned} \|\mathbf{p}_i - \mathbf{p}_j\|^2 &= (\mathbf{e}_{i-1} - \mathbf{e}_{j-1})^T A(\mathbf{e}_{i-1} - \mathbf{e}_{j-1}) \\ &= a_{ii} + a_{jj} - 2a_{ij} \\ &= d_{1i} + d_{1j} - 2[\tfrac{1}{2}(d_{1i} + d_{1j} - d_{ij})] \\ &= d_{ij} \end{aligned}$$

which shows that $\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_n$ are the n points satisfing the condition 3iii in Definition 3.2.1. Since $\mathbf{e}_i's$ are independent and $rank\ X = r$ then the $\mathbf{p}_i's$ are irreducibly embeddable in $\Re^r$. $\Box$

Now we want to gives an alternative characterization for the Euclidean distance matrix in the following theorem and corollory. This will be used later in Section 3.3.

Theorem 3.2.4

Let $D \in \Re^{n\times n}$ be a distance matrix; then D is a Euclidean distance matrix if and only if

$$\mathbf{x}^T(-D)\mathbf{x} \geq 0 \quad \forall\, \mathbf{x} \in M$$

where

$$M = \{\, \mathbf{x} \in \Re^n : \ \mathbf{e}^T\mathbf{x} = 0 \,\}. \tag{3.2.7}$$

Thus $-D \in K_M$.

Proof

Define

$$P = I - \mathbf{e}\mathbf{e}_1^T \tag{3.2.8}$$

where

$$\mathbf{e}_1^T = [\ 1 \quad 0 \quad 0 \quad \ldots \quad 0\ ]$$

then

$$P = \begin{bmatrix} 0 & 0 & 0 & \ldots & 0 \\ -1 & 1 & 0 & \ldots & 0 \\ -1 & 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -1 & 0 & 0 & \ldots & 1 \end{bmatrix}$$

which implies that

$$P(-D)P^T \;=\; \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ 0 & 2\,d_{21} & d_{21}+d_{31}-d_{32} & \dots & d_{21}+d_{n1}-d_{n2} \\ 0 & d_{21}+d_{31}-d_{32} & 2\,d_{31} & \dots & d_{31}+d_{n1}-d_{n3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & d_{21}+d_{n1}-d_{n2} & d_{31}+d_{n1}-d_{n3} & \dots & 2\,d_{n1} \end{bmatrix}$$

$$= \begin{bmatrix} 0 & \mathbf{0}^T \\ \mathbf{0} & A \end{bmatrix}. \tag{3.2.9}$$

Now we show that $P(-D)P^T \geq 0$ if and only if $-D$ is positive semi–definite in M. First let $P(-D)P^T$ be a positive semi–definite then we have

$$\mathbf{x}^T\,(P(-D)P^T)\,\mathbf{x} \;\geq\; 0 \qquad \forall\;\mathbf{x}.$$

Then when $\mathbf{x}^T\mathbf{e} = 0$ we have

$$\begin{aligned} 0 \leq \mathbf{x}^T\,(P(-D)P^T)\,\mathbf{x} &= (\mathbf{x}^T - \mathbf{x}^T\mathbf{e}\mathbf{e}_1^T)(-D)(\mathbf{x} - \mathbf{e}_1^T\mathbf{e}^T\mathbf{x}) \\ &= \mathbf{x}^T(-D)\mathbf{x} \end{aligned}$$

which implies that $-D$ is positive semi–definite in M.

Conversly, let $-D$ be a positive semi–definite in M then we have

$$\mathbf{y}^T\,(-D)\,\mathbf{y} \;\geq\; 0 \qquad \forall\;\mathbf{y}^T\mathbf{e} = 0.$$

We can express $\mathbf{x} = \lambda\,\mathbf{u} + \mathbf{y}$ where $\mathbf{u}$ is any vector such that $\mathbf{u}^T\mathbf{e} \neq 0$. Take $\mathbf{u} = \mathbf{e}_1$ then

$$\mathbf{x} \;=\; \mathbf{y} + \lambda\,\mathbf{e}_1^T$$

this implies that $\mathbf{e}^T\mathbf{x} = \lambda\,\mathbf{e}^T\mathbf{e}_1 = \lambda$ since $\mathbf{y}^T\mathbf{e} = 0$ and $\mathbf{e}^T\mathbf{e}_1 = 1$. It then follows that

$$\begin{aligned} \mathbf{y} &= \mathbf{x} - \lambda\,\mathbf{u} \\ &= \mathbf{x} - \mathbf{e}^T\mathbf{x}\,\mathbf{e}_1 \\ &= (I - \mathbf{e}_1\mathbf{e}^T)\mathbf{x} \\ &= P^T\mathbf{x}. \end{aligned}$$

For an arbitrary $\mathbf{x}$, this gives

$$\mathbf{x}^T(I - \mathbf{e}_1\mathbf{e}^T)^T(-D)\,(I - \mathbf{e}_1\mathbf{e}^T)\mathbf{x} \geq 0 \qquad \forall\ \mathbf{x}$$

$$\mathbf{x}^T P(-D)P^T\mathbf{x} \geq 0. \qquad \forall\ \mathbf{x}.$$

Using Theorem 3.2.3 the proof is established. □

Corollory 3.2.5

Let Q be the Householder matrix in (1.2.1) then the distance matrix $D = D^T \in \Re^{n\times n}$ is a Euclidean distance matrix if and only if the $n-1 \times n-1$ block D_1 in

$$Q(-D)Q = \begin{bmatrix} D_1 & \mathbf{d} \\ \mathbf{d}^T & \delta \end{bmatrix} \qquad (3.2.10)$$

is positive semi–definite.

Proof

Follows from Theorem 1.3.6 □.

The advantage of the formulation in (3.2.10) over that given in (3.2.9) is that it provides the basis for the construction of a projection algorithm.

In the rest of this section other characterizations for the Euclidean distance matrix are given.

The problem of characterizing the Euclidean distance matrices among the distance matrices was first solved by Menger [1931] who based his analysis on the determinant of the form:

$$A_k = det \begin{pmatrix} 0 & 1 & 1 & \dots & 1 & 1 \\ 1 & 0 & d_{12} & \dots & d_{1\ k-1} & d_{1k} \\ 1 & d_{21} & 0 & \dots & d_{2\ k-1} & d_{2k} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & d_{k-1\ 1} & d_{k-1\ 2} & \dots & 0 & d_{k-1\ k} \\ 1 & d_{k1} & d_{k2} & \dots & d_{k\ k-1} & 0 \end{pmatrix} \quad (k = 2, 3, \dots, n) \qquad (3.2.11)$$

now known as Cayley–Menger determinant. The Menger result is given in a theorem due to Blumenthal [1953, pp99–100] Four years later, Schoenberg [1935] gave his result in Theorem 3.2.3. Another characterization is given by Hayden et. al. [1988], who show that a distance matrix $D \in \Re^{n\times n}$ is a Euclidean distance matrix if and only if the bordered matrix:

$$A = \begin{bmatrix} -D & \mathbf{e} \\ \mathbf{e}^T & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & -d_{12} & \dots & -d_{1n} & 1 \\ -d_{21} & 0 & \dots & -d_{2n} & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ -d_{n1} & -d_{n2} & \dots & 0 & 1 \\ 1 & 1 & \dots & 1 & 0 \end{bmatrix}. \qquad (3.2.12)$$

has exactly one negative eigenvalue. Further, the n points represented by the matrix D (see Definition 3.2.1iii) are irreducibly embeddable in $\Re^r (r \leq n - 1)$ if and only if

$$r = n - 1 - dim\ N(D)$$

where $N(D)$ is the null space of D.

Other characterizations of the Euclidean distance matrix are also given in the literature.

3.3 The projection algorithm

Glunt et. al. [1990] give a description of an algorithm for computing the nearest Euclidean distance matrix, using the alternating projection method of Dykstra [1983] which guarantees convergence to the solution of problem (3.2.1). First an equivalent problem to (3.2.1) is given. This section includes a projection algorithm based on Dykstra's algorithm. The projection algorithm requires formulae, which are also given, for calculating the projection maps on to K_d (see (3.3.1)) and on to K_M. However in the first stage a formula for K_d and the normal cone of the intersection of K_d and K_M is given. The latter is used in problem (3.3.3) below.

Define

$$K_d = \{A : A \in \Re^{n\times n}, \quad A^T = A, \quad a_{ii} = 0, \quad i = 1, 2, \ \ldots, n\}, \tag{3.3.1}$$

then $K_M \cap K_d$ is a convex cone.

The normal cone $\partial K_M(A)$ at $A \in K_M$ is given in (1.3.25).

Let $A \in K_d$ then it is clear that

$$\partial K_d(A) = \{B : B = diag\,[b_1, b_2, \ \ldots, b_n]\}. \tag{3.3.2}$$

A general result for the normal cone of the intersection of two sets has been given in (1.3.9). Using this we have the following theorem.

Theorem 3.3.1

If $A \in K_M \cap K_d$ then

$$\partial(K_M \cap K_d)(A) = \partial K_M(A) + \partial K_d(A)$$

Proof (this is a special case of Rockafellar [1970])

Clearly from Theorem 3.2.4 $D \in K_M \cap K_d$ if and only if $-D$ is Euclidean distance matrix. Further, the matrices in K_M are characterized by (3.2.10). Thus, the minimization problem (3.2.1) is a special case of the following problem:

Given a distance matrix $-F \in \Re^{n\times n}$

$$\begin{aligned} &minimize \ \| F - D \|_F \\ &subject \ to \ D \in K_M \cap K_d. \end{aligned} \tag{3.3.3}$$

Note that in problem (3.2.1) F and D are different from F and D in this problem. Now $\partial K_M(A)$ and $\partial K_d(A)$ are given in (1.3.25) and (3.3.2) respectively. From Theorem 3.3.1 and (2.1.3) we can deduce that D^* solves problem (3.3.3) if and only if

$$F - D^* = Q \begin{bmatrix} U \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & H \end{bmatrix} U^T & \mathbf{0} \\ \mathbf{0}^T & \mathbf{0} \end{bmatrix} Q + B. \tag{3.3.4}$$

Then (3.3.4) is equivalently to

$$F \; = \; Q \; \begin{bmatrix} U \begin{bmatrix} \Lambda_r & \mathbf{0} \\ \mathbf{0} & H \end{bmatrix} U^T & \mathbf{d} \\ \mathbf{d}^T & \delta \end{bmatrix} Q \; + \; B$$

since

$$D^* \; = \; Q \; \begin{bmatrix} U \begin{bmatrix} \Lambda_r & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} U^T & \mathbf{d} \\ \mathbf{d}^T & \delta \end{bmatrix} Q \qquad (3.3.5)$$

this is true from Theorem 1.3.6 and (1.3.24) and $D^* \in K_M$ from (3.3.3).

Dykstra's algorithm depends crucially upon the computational complexity of the relevant projections. The minimization problem (3.3.3) is solved by applying Algorithm 2.2.7 to it. Problem (3.3.3) is to find the projection of a matrix to the intersection of two convex sets by a sequence of projections to the individual set successively. First we need definition for the projection maps $P_d(\cdot)$ and $P_M(\cdot)$, later formulae for them are obtained.

Definition 3.3.2

Let

$$K \; = \; \{A: \; A \in \Re^{n\times n}, A \; = \; A^T\},$$

then define the projection map $P_M(A)$ from K on to K_M and the projection map $P_d(A)$ from K on to K_d.

Since K_d is the subspace consisting of all real symmetric $n \times n$ matrices with zero diagonals then $P_d \, (F)$ is straightforwardly given by

$$P_d \; (F) \; = \; F \; - \; Diag \; (F) \qquad (3.3.6)$$

i.e., P_d maps F to the matrix obtained by replacing each diagonal element by zero.

The projection map $P_M(A)$ formula on to K_M is now introduced but first a theorem due to Higham [1988] is given which is used in proving the formula.

Theorem 3.3.3

Let $F_1 \in \Re^{n-1 \times n-1}$ be a symmetric matrix and $F_1 = U \Lambda U^T$ be its spectral decomposition, then $D_1 = U \Lambda^+ U^T$ solves the following problem

$$\begin{aligned} &\text{minimize } \; \|F_1 - D_1\|_F \\ &subject\ to \;\; D_1 \geq 0 \end{aligned}$$

where

$$\Lambda = diag\,[\lambda_1,\ \lambda_1,\ \ldots,\ \lambda_{n-1}]$$

and

$$\Lambda^+ = diag\,[\lambda_i :\ \lambda_i = \left\{ \begin{array}{ll} \lambda_i, & \lambda_i \geq 0 \\ 0, & \lambda_i < 0 \end{array} \right\} i = 1,\ \ldots,\ n-1\ ].$$

Proof

Take any $D_1 \geq 0$ express $D_1 = UYU^T$ where $Y \geq 0$ then

$$\begin{aligned} \|F_1 - D_1\|_F^2 &= \|\Lambda - Y\|_F^2 \\ &= \sum_{\substack{i,j=1 \\ i \neq j}}^{n-1} y_{ij}^2 + \sum_{i=1}^{n-1} (\lambda_i - y_{ii})^2 \\ &\geq \sum_{\lambda_i < 0} (\lambda_i - y_{ii})^2. \end{aligned}$$

It then follows because Y is positive semi–definite and $\lambda_i < 0$ that $-2\lambda_i y_{ii} \geq 0$ which implies that

$$\|F_1 - D_1\|_F^2 \geq \sum_{\lambda_i < 0} \lambda_i^2.$$

This lower bound is attained uniquely for the matrix $Y = \Lambda^+$ because

$$\begin{aligned}\|F_1 - D_1\|_F^2 &= tr(F_1 - D_1)(F_1 - D_1) \\ &= (\Lambda - \Lambda^+)(\Lambda - \Lambda^+) \\ &= \sum_{i=1}^{n-1} \lambda_i - \sum_{i=1}^{n-1} \lambda_i^+ \\ &= \sum_{i:\lambda_i<0} \lambda_i^2.\end{aligned}$$

that is $D_1 = U\Lambda^+U^T$. □

Now for calculating $P_M(F)$ we need to solve the following problem

$$\begin{aligned}&minimize\ \|F - D\|_F \\ &subject\ to\ D \in K_M.\end{aligned} \tag{3.3.7}$$

Let

$$F = Q \begin{bmatrix} F_1 & \mathbf{f} \\ \mathbf{f}^T & \zeta \end{bmatrix} Q \quad and \quad D = Q \begin{bmatrix} D_1 & \mathbf{d} \\ \mathbf{d}^T & \delta \end{bmatrix} Q$$

then $\|F - D\|_F$ is minimized by minimizing

$$\|F_1 - D_1\|_F \tag{3.3.8}$$

since

$$\|F - D\|_F = \|Q(F - D)Q\|_F = \left\| \begin{matrix} F_1 - D_1 & \mathbf{f} - \mathbf{d} \\ \mathbf{f}^T - \mathbf{d}^T & \zeta - \delta \end{matrix} \right\|_F .$$

Therefore from Theorem 3.3.3

$$P_M\ (F) = Q \begin{bmatrix} U\Lambda^+U^T & \mathbf{f} \\ \mathbf{f}^T & \zeta \end{bmatrix} Q, \tag{3.3.9}$$

is the solution of problem (3.3.7). Here Λ^+ corresponds to $\begin{bmatrix} \Lambda_r & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}$ in (3.3.5) such that

$$\begin{bmatrix} \Lambda_r & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \equiv \Lambda^+$$

In Chapter 2 von Neumann's algorithm was given which solves problem (3.3.3) in Algorithm 2.2.1. However, Example 2.2.4 illustrates the failure of von Neumann algorithm in solving problem (3.3.3) for general n. Moreover Algorithm 2.2.7 was given which successfully solves problem (3.3.3) for general n.

In particular, we have a matrix projections P_M and P_d given by (3.3.9) and (3.3.6) respectively. Using Algorithm 2.2.7 in case $m = 2$ with the projections $P_1 = P_M$ and $P_2 = P_d$ we have the following algorithm

Algorithm 3.3.4 (*projection algorithm*)

Given any distance matrix $-F \in \Re^{n\times n}$, let $F^{(0)} = F$

$$For \quad k = 1,\ 2,\ \ldots$$

$$F^{(k+1)} = F^{(k)} + [P_d P_M(F^{(k)}) - P_M(F^{(k)})] \tag{3.3.10}$$

This algorithm is given by Glunt et. al. [1990]. The convergence of this algorithm follows from Theorem 2.2.8 in which $P_1 = P_d$ and $P_2 = P_M$. Given any distance matrix $-F = -F^T \in \Re^{n\times n}$, then the sequences $\{P_M(F^{(k)})\}$ and $\{P_d P_M(F^{(k)})\}$ generated by Algorithm 3.3.4 converge in the Frobenius norm to the solution D^* of (3.3.3).

It is important to realize whether a distance matrix $-F$ is a Euclidean distance matrix or not before solving problem (3.3.3). This is because if $-F$ is a Euclidean distance matrix then $D^* = F$ and there is no problem to solve. In the following a test is given to indicate if the matrix $-D^{(k)}$ is Euclidean distance matrix or not.

It follows by induction that off–diagonal elements of $F^{(k)}$, $k = 1, 2, \ldots$ in Algorithm 3.3.4 are the same as $F^{(0)}(= F)$. Denote

$$\Delta^{(k)} = F^{(k)} - F \tag{3.3.11}$$

which is diagonal. Then (3.3.10) can now be written

$$\Delta^{(k+1)} = \Delta^{(k)} - Diag\ (D^{(k)}) \tag{3.3.12}$$

where $D^{(k)} = P_M\ (F + \Delta^{(k)})$. Therefore given any $F^{(k)}$ (or $\Delta^{(k)}$), the test

$$Diag\ (D^{(k)}) = 0 \tag{3.3.13}$$

where $D^{(k)} = P_M(F^{(k)}) = P_M(F + \Delta^{(k)})$ determines whether the matrix $-D^{(k)}$ is Euclidean distance matrix or not. This test is useful in Chapter 4.

3.4 Unconstrained methods

In this section we shall consider a different approach to the problem (3.3.3). The main idea is to replace the problem (3.3.3) by an unconstrained optimization problem in order to use the superlinearly convergent quasi–Newton methods. Three methods for solving problem (3.3.3) will be given along with an example showing how the points $\mathbf{p}_i'$s are represented in the space with different methods. In the end of this section a strategy is described of how to choose the initial matrix X and the rank r.

Schoenberg [1935] and Young et. al. [1938] independently formulated a characterization of the Euclidean distance matrix in Theorem 3.2.3. Using this theorem problem (3.2.1) can be expressed as:

$$\begin{aligned} &\underset{D}{\text{minimize}} \quad \phi \\ &\textit{subject to} \;\; A \geq 0 \end{aligned} \tag{3.4.1}$$

where

$$\phi = \|F - D\|_F^2,$$

$-F$ is a distance matrix and A is a function of D given by (3.2.2). (Note that the matrix $-D$ is the Euclidean distance matrix).

In the following analysis an equivalent unconstrained problem to (3.4.1) is derived.

From Definition 3.2.1 $-D$ is represented in the space $\Re^r$ by the following vectors

$$\mathbf{p}_1, \; \mathbf{p}_2, \; \ldots, \; \mathbf{p}_n. \tag{3.4.2}$$

It is always possible to put the first vector $\mathbf{p}_1$ at the origin by transforming each vector $\mathbf{p}_i$ to $\mathbf{p}_i - \mathbf{p}_1$ $i = 1, 2, \ldots, n$. Assume that the rank of A is known to be r ($1 \leq r < n$). The columns of X^T are the vectors $\mathbf{p}_2, \mathbf{p}_3, \ldots, \mathbf{p}_n$ (see Theorem 3.2.3). For quasi–Newton methods it will be convenient to store the matrix X as a one vector with $r(n-1)$ variables as follows

$$X^T = \begin{bmatrix} x_1 & x_2 & \dots & x_{n-1} \\ x_n & x_{n+1} & \dots & x_{2(n-1)} \\ \vdots & \vdots & \ddots & \vdots \\ x_{t_1} & x_{t_2} & \dots & x_{r(n-1)} \end{bmatrix}. \tag{3.4.3}$$

where $t_1 = (r-1)(n-1)+1$ and $t_2 = (r-1)(n-1)+2$. Hence

$$\begin{aligned} \mathbf{p}_2^T &= [x_1 \quad x_n \quad \dots \quad x_{t_1} \quad] \\ \mathbf{p}_3^T &= [x_2 \quad x_{n+1} \quad \dots \quad x_{t_2} \quad] \\ &\vdots \\ \mathbf{p}_n^T &= [x_{n-1} \quad x_{2(n-1)} \quad \dots \quad x_{r(n-1)}] \end{aligned} \tag{3.4.4}$$

and

$$X^T = [\mathbf{p}_2 \quad \mathbf{p}_3 \quad \dots \quad \mathbf{p}_n]. \tag{3.4.5}$$

The constraint $A \geq 0$ is equivalent to $A = XX^T$, which can be expressed as

$$A = XX^T = \begin{bmatrix} \mathbf{p}_2^T \cdot \mathbf{p}_2 & \dots & \mathbf{p}_2^T \cdot \mathbf{p}_n \\ \vdots & \ddots & \vdots \\ \mathbf{p}_n^T \cdot \mathbf{p}_2 & \dots & \mathbf{p}_n^T \cdot \mathbf{p}_n \end{bmatrix}. \tag{3.4.6}$$

Therefore satisfying the constraint in problem (3.4.1) is equivalent to expressing the elements of the matrix D as a function of X. Hence from Definition 3.2.1iii and using (3.4.4)

$$\begin{aligned} d_{ij} &= d_{ji} = \|\mathbf{p}_i - \mathbf{p}_j\|^2 \qquad i,j = 2,\dots,n \\ &= \sum_{k=0}^{r-1} (x_{i+km-1} - x_{j+km-1})^2, \qquad i,j = 2,\dots,n \end{aligned} \tag{3.4.7}$$

and

$$d_{1i} = d_{1j} = \sum_{k=0}^{r-1} x_{i+km-1}^2. \qquad i,j = 2,\dots,n \tag{3.4.8}$$

where $m = n - 1$ and $r = rank\ X$ (through out this section). Thus there exists a matrix X that satisfies (3.4.6) and hence the constraint $A \geq 0$ in problem (3.4.1).

An alternative way of deriving these expressions is the following, since $A = XX^T$ then

$$\begin{aligned} a_{ij} &= \sum_{k=1}^{r} x_{i+km-1} \cdot x_{j+km-1} \\ &= \sum_{k=1}^{r} (x_{i+km-1}^2 + x_{j+km-1}^2 - (x_{i+km-1} - x_{j+km-1})^2) \\ &= \tfrac{1}{2}[\, d_{1i} + d_{1j} - d_{ij}], \end{aligned}$$

$$d_{1i} = \sum_{k=0}^{r-1} x_{i+km-1}^2 \quad and \quad d_{1j} = \sum_{k=0}^{r-1} x_{j+km-1}^2 \quad i,j = 2,\ldots,n \tag{3.4.9}$$

and

$$d_{ij} = d_{ji} = \sum_{k=0}^{r-1} (x_{i+km-1} - x_{j+km-1})^2, \quad i,j = 2,\ldots,n \tag{3.4.10}$$

which is equivalent to (3.4.8) and (3.4.7).

Therefore, problem (3.4.1) above can be expressed in unconstrained form as follows:

$$\underset{X}{\text{minimize}} \quad \phi \tag{3.4.11}$$

where $\phi = \|F - D\|_F^2$, $-F$ is a distance matrix and the elements of the matrix D are given by (3.4.9) and (3.4.10).

Now three methods for solving problem (3.4.11) will be given. The quasi–Newton method (Algorithm 1.6.3) is used to solve problem (3.4.11) and the BFGS formula is used to update the Hessian matrix $H^{(k+1)}$. Quasi–Newton methods require only the function f and the first derivative $\mathbf{g}$ where f is ϕ and $\mathbf{g} = \nabla\phi$. However some difficulties arise, one of these is that the index $r\ (= rank(A))$ used in partitioning $A\ (= XX^T)$ is not known in advance. Fortunately it can be shown that by solving a sequence of problems for different r, each of which is well behaved, the correct value of r can be located. Excluding the function ϕ and the derivative $\nabla\phi$ which differ from one method to another, the procedures in the three methods are the same.

Method 3.4.1

Consider the matrix X with the vector $\mathbf{p}_1$ untranslated to the origin. Thus the vectors $\mathbf{p}_1,\ \mathbf{p}_2,\ \ldots\ ,\ \mathbf{p}_n$ are all vectors of variables, so that

$$\begin{aligned}
\mathbf{p}_1^T &= [x_1 \quad x_{n+1} \quad \ldots \quad x_{(r-1)n+1}] \\
\mathbf{p}_2^T &= [x_2 \quad x_{n+2} \quad \ldots \quad x_{(r-1)n+2}] \\
&\vdots \\
\mathbf{p}_n^T &= [x_n \quad x_{2n} \quad \ldots \quad x_{rn} \quad].
\end{aligned} \tag{3.4.12}$$

Then

$$X^T = \begin{bmatrix} x_1 & x_2 & \ldots & x_n \\ x_{n+1} & x_{n+2} & \ldots & x_{2n} \\ x_{2n+1} & x_{2n+2} & \ldots & x_{3n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{t_3} & x_{t_4} & \ldots & x_{rn} \end{bmatrix}. \tag{3.4.13}$$

where $t_3 = (r-1)n+1$ and $t_4 = (r-1)n+2$. So the elements of the matrix $D \in \Re^{n\times n}$ take the form

$$d_{ij} = d_{ji} = \sum_{k=0}^{r-1}(x_{i+kn} - x_{j+kn})^2 \tag{3.4.14}$$

then

$$\begin{aligned}
\phi &= \sum_{i,j=1}^{n}(d_{ij} - f_{ij})^2 \\
&= \sum_{i,j=1}^{n}\{\sum_{k=0}^{r-1}(x_{i+kn} - x_{j+kn})^2 - f_{ij})^2 \\
&= 2\sum_{\substack{i,j=1\\ i<j}}^{n}\{\sum_{k=0}^{r-1}(x_{i+kn} - x_{j+kn})^2 - f_{ij})^2
\end{aligned} \tag{3.4.15}$$

and

$$\nabla\phi = [\ \tfrac{\partial\phi}{\partial x_1} \quad \tfrac{\partial\phi}{\partial x_2} \quad \ldots \quad \tfrac{\partial\phi}{\partial x_{rn}}\ ]^T$$

where

$$\frac{\partial\phi}{\partial x_s} = 2\sum_{j=1}^{n}\{2[\sum_{k=0}^{r-1}(\ x_{l+kn}\ -\ x_{j+kn}\ )^2\ -\ f_{lj}]\ \ 2(\ x_s\ -\ x_{j+tn}\ )\}$$

$$= 8\sum_{j=1}^{n}\{[\sum_{k=0}^{r-1}(\ x_{l+kn}\ -\ x_{j+kn}\ )^2\ -\ f_{lj}](\ x_s\ -\ x_{j+tn}\ )\} \qquad (3.4.16)$$

for all $s = 1, \ldots, rn$ where $t = \frac{(s-l)}{n}$ and $l = mod\,(s,n)$ and if $l = 0$ then $l = n$.

Method 3.4.2

In this method, as explained earlier the first vector $\mathbf{p}_1$ is transformed to the origin (see Figures 3.4.1 and 3.4.2), so the number of variables is reduced from rn to $r(n-1)$. The matrix X^T is given in (3.4.3). The elements of the matrix D take the form

$$d_{i1} = d_{1j} = \sum_{k=0}^{r-1} x_{i+km-1}^2 \quad i = 2, \ldots, n \qquad (3.4.17)$$

$$d_{ij} = d_{ji} = \sum_{k=0}^{r-1}(x_{i+km-1}\ -\ x_{j+km-1}\ )^2 \quad i,j = 2, \ldots, n \qquad (3.4.18)$$

where $r = rank\ X$ and $m = n-1$. Hence

$$\phi = \sum_{i,j=1}^{n}(d_{ij}\ -\ f_{ij})^2$$

$$= 2\{\sum_{i=1}^{n}(d_{i1}\ -\ f_{i1})^2 + \sum_{\substack{i,j=2\\ i>j}}^{n}(d_{ij}\ -\ f_{ij})^2\}$$

$$= 2\{\sum_{i=1}^{n}(\sum_{k=0}^{r-1} x_{i+km-1}^2\ -\ f_{i1})^2 +$$

$$\sum_{\substack{i,j=2\\ i>j}}^{n}(\sum_{k=0}^{r-1}(x_{i+km-1}\ -\ x_{j+km-1}\ )^2\ -\ f_{ij})^2\} \qquad (3.4.19)$$

and

$$\nabla\phi \;=\; [\;\frac{\partial\phi}{\partial x_1}\quad \frac{\partial\phi}{\partial x_2}\quad \cdots \quad \frac{\partial\phi}{\partial x_{r(n-1)}}\;]^T.$$

where

$$\begin{aligned}\frac{\partial\phi}{\partial x_s} &= 8x_s\{\sum_{k=0}^{r-1} x_{l+km}^2 \;-\; f_{l+1\;1}\} \\ &+\; 8\{\sum_{j=1}^{m}[\sum_{k=0}^{r-1}(\;x_{l+km} \;-\; x_{j+km}\;)^2 \;-\; f_{l+1\;j+1}](\;x_s \;-\; x_{j+tm}\;)\}\end{aligned} \tag{3.4.20}$$

for all $s \;=\; 1,\;\ldots\;, r(n\;-\;1)$ where $t \;=\; \frac{(s\;-\;l)}{m}$ and $l \;=\; mod\;(s,m)$ and if $l \;=\; 0$ then $l \;=\; m$.

Method 3.4.3

In this method translation and rotation are used. First, the vector $\mathbf{p}_1$ transformed to the origin so the number of variables will be reduced to $r(n-1)$ as in the second method. Since $-D$ is a Euclidean distance matrix, it is always possible to make rotation about the origin, axes, planes and spaces depending on the dimension of r (e.g. if $r = 3$ 2 rotations, $r = 4$ 3 rotations etc.). Make a rotation in the second vector $\mathbf{p}_2$ around the origin until the vector $\mathbf{p}_2$ is located on one of the axes. Then the components of $\mathbf{p}_2$ are zeros except one, assume it is the first component. Similarly, rotate the vector $\mathbf{p}_3$ but this time around the axis where $\mathbf{p}_2$ located until $\mathbf{p}_3$ is located in one of the planes. Then the components of $\mathbf{p}_3$ are zeros except two; assume they are the first two.

Figures 3.4.1–6 shows an example of translation then rotation of vectors $\mathbf{p}_1$, $\mathbf{p}_2$ and $\mathbf{p}_3$ in the space $\Re^3$.

If we continue likewize with the rest of the vectors $\mathbf{p}_4$, $\mathbf{p}_5$, $\ldots$, $\mathbf{p}_r$ then matrix X^T has the following form

$$X^T \;=\; \begin{bmatrix} x_1 & x_2 & x_3 & \cdots & \cdots & \cdots & x_{n-1} \\ 0 & x_n & x_{n+1} & \cdots & \cdots & \cdots & x_{2n-3} \\ 0 & 0 & x_{2n-2} & \cdots & \cdots & \cdots & x_{3n-6} \\ \vdots & \vdots & \vdots & \ddots & & & \vdots \\ 0 & 0 & 0 & \cdots & x_p & \cdots & x_q \end{bmatrix} \tag{3.4.21}$$

where $p = (r-1)\,m - \frac{r(r-1)}{2}$ and $q = rm - \frac{r(r+1)}{2}$. Then the vectors $\mathbf{p}_1,\ \mathbf{p}_2,\ \dots,\ \mathbf{p}_n$ have the form

$$\begin{array}{lcllllll}
\mathbf{p}_1^T & = & [0 & 0 & 0 & \dots & 0 & 0] \\
\mathbf{p}_2^T & = & [x_1 & 0 & 0 & \dots & 0 & 0] \\
\mathbf{p}_3^T & = & [x_2 & x_n & 0 & \dots & 0 & 0] \\
\vdots & & & & & & & \\
\mathbf{p}_r^T & = & [x_{r-1} & x_{n+r-3} & x_{2n+r-4} & \dots & x_{p+r-m} & 0] \\
\mathbf{p}_{r+1}^T & = & [x_r & x_{n+r-2} & x_{2n+r-5} & \dots & x_{p+r-m+1} & x_p] \\
\vdots & & & & & & & \\
\mathbf{p}_n^T & = & [x_{n-1} & x_{2n-3} & x_{3n-6} & \dots & x_{p-1} & x_q]
\end{array} \tag{3.4.22}$$

In this method, the number of variables is reduced to $rm - \frac{r(r+1)}{2}$ where $m = n-1$. Thus the number of variables is reduced by $\frac{r(r+1)}{2}$ from Method 3.4.2. The elements of the matrix $D \in \Re^{n\times n}$ ($-D$ is the Euclidean distance matrix) take the form

$$d_{i1} = d_{1j} = \sum_{k=0}^{p-1} x_{i+t}^2 \qquad i = 2,\ 3,\ \dots,n \tag{3.4.23}$$

with $p = min\ (i-1, r)$ and $t = km - \frac{k(k+1)}{2} - 1$

$$d_{ij} = d_{ji} = \sum_{k=0}^{l-2}(x_{i+t} - x_{j+t})^2 + \sum_{k=l-1}^{p-1} x_{j+t}^2$$
$$for \quad i = 2,\ \dots,r \ \ and\ \ j = i+1,\ \dots,n \tag{3.4.24}$$

where $l = min\ (i,j)$, $p = min\ (j-1, r)$, $t = km - \frac{k(k+1)}{2} - 1$ and $m = n-1$. Also

$$d_{ij} = d_{ji} = \sum_{k=0}^{r-1}(x_{i+t} - x_{j+t})^2 \qquad \forall\, i,j = r+1,\ ,..,n \tag{3.4.25}$$

Thus

$$\begin{aligned}
\phi &= \sum_{i,j=1}^{n}[d_{ij} - f_{ij}]^2 \\
&= 2\{\sum_{i=2}^{n}[d_{i1} - f_{i1}]^2 \\
&\qquad + \sum_{i=2}^{r}\sum_{j=i+1}^{n}[d_{ij} - f_{ij}]^2 \\
&\qquad\qquad + \sum_{\substack{i,j=r+1\\ i>j}}^{n}[d_{ij} - f_{ij}]^2\} \\
&= 2\{\sum_{i=2}^{n}[\sum_{k=0}^{p}x_{i+t}^2 - f_{i1}]^2 \\
&\qquad + \sum_{i=2}^{r}\sum_{j=i+1}^{n}[\sum_{k=0}^{l-2}(x_{i+t} - x_{j+t})^2 + \sum_{k=l-1}^{p-1}x_{j+t}^2 - f_{ij}]^2 \\
&\qquad + \sum_{\substack{i,j=r+1\\ i>j}}^{n}[\sum_{k=0}^{r-1}(x_{i+t} - x_{j+t})^2 - f_{ij}]^2\} \qquad (3.4.26)
\end{aligned}$$

and the gradient vector $\nabla\phi$ can be calculated from Algorithm 3.4.4 where

$$\nabla\phi = [\ \frac{\partial\phi}{\partial x_1}\quad \frac{\partial\phi}{\partial x_2}\quad \ldots\quad \frac{\partial\phi}{\partial x_q}\ ]^T$$

with $q = rm - \frac{r(r-1)}{2}$.

Algorithm 3.4.4 (*gradient calculation* : $\nabla\phi$)

```
$q = 0$
$is = rm - \frac{r(r-1)}{2}$
$For \quad s = 1,\ 2,\ \ldots,\ is$
    $q = (s - 1 + \frac{q(q+1)}{2})/m$
```

```
        $l = s - qm + \frac{q(q+1)}{2}$
        $For \quad k = 1, 2, \ldots, n$
            $b_k = d_{l+1\ k} - f_{l+1\ k}$
            $p = min\ (\ r,\ q+1\ )$
            $If \quad k \leq p \quad Then$
                $b_k = 8\ b_k x_s$
            $Else$
                $t = mq - \frac{q(q+1)}{2} - 1$
                $b_k = 8\ b_k\ (\ x_s - x_{k+t})$
            $End\ If$
        $End\ For$
        $\frac{\partial \phi}{\partial x_s} = \sum_{k=1}^{n} b_k$
    $End\ For$
```

Example 3.4.5

As an example illustrating translation and rotation for Method 3.4.2 and Method 3.4.3, let

$$X^T = \begin{bmatrix} 2 & 6 & 4 \\ 3 & 5 & 4 \\ 2 & 4 & 5 \end{bmatrix}.$$

First transform $\mathbf{p}_1 = (2,\ 3,\ 2)$ to the origin then

$$X^{'T} = \begin{bmatrix} 0 & 4 & 2 \\ 0 & 2 & 1 \\ 0 & 2 & 3 \end{bmatrix}.$$

Figure 3.4.1: Transform the point $\mathbf{p}_1$ to the origin in order to reduce the number of variables from rn to $r(n-1)$.

Rotating $\mathbf{p}_2$ around the origin (see Figures 3.4.4 and 3.4.5) gives

$$X^{''T} = \begin{bmatrix} 0 & 4.99 & 3.26 \\ 0 & 0 & 1.58 \\ 0 & 0 & 1.41 \end{bmatrix}$$

whilst a second rotation around the x–axis of $\mathbf{p}_3$ located in the plane of y,z–axes (see Figures 3.4.5 and 3.4.6) gives

$$X^{'''T} = \begin{bmatrix} 0 & 4.99 & 3.26 \\ 0 & 0 & 1.83 \\ 0 & 0 & 0 \end{bmatrix}.$$

Figure 3.4.2: The location for each point after the translation.

An important consideration is the choice of the initial matrix X. In the following a procedure is given for finding a suitable initial matrix X. Let $-F$ be the given distance matrix, and let $r^{(0)}$ be the estimated rank. The initial matrix X for Method 3.4.2 can be calculated using Theorem 3.2.3 as follows:

Define the elements of A from F by

$$a_{ij} \;=\; -\tfrac{1}{2}[\, f_{1i} \;+\; f_{1j} \;-\; f_{ij}] \quad (2 \leq \; i,j \; \leq n). \tag{3.4.27}$$

Figure 3.4.3: Rotate the point $\mathbf{p}_2$ around the origin so that it is located on the x-axis. This removes $r-1$ variables.

Figure 3.4.4: The location for each point after the first rotation.

Figure 3.4.5: Rotate the point $\mathbf{p}_3$ around the x–axis so that it is located on the x,y–axis. This removes $r-1$ variables.

Figure 3.4.6: The final location for each point with variables reduced from 9 variables to only 3 variables.

Consider the spectral decomposition

$$A = U \, \Lambda \, U^T,$$

then the initial matrix X for the unconstrained Method 3.4.2 is given by

$$X = U \, \Lambda_r^{1/2} \tag{3.4.28}$$

where $\Lambda_r = diag\,[\lambda_1, \; \lambda_2, \ldots, \; \lambda_r]$, are the r largest eigenvalues in Λ.

The above equations can be used to form an initial matrix X for Method 3.4.1 and Method 3.4.3. However, the initial matrix X can be any independent vectors. The independence is important because if one of the vectors is dependent on the other vectors, error will occur in the minimizer of $\phi(X)$, and D will be embeddable in $\Re^{r-1}$ when it should be irreducibly embeddable in $\Re^r$. For example if

$$X^T = [\mathbf{p}_1, \; \mathbf{p}_2, \; \mathbf{p}_3]$$

and D^* is irreducibly embeddable in $\Re^2$, choose $\mathbf{p}_3$ such that it is dependent on $\mathbf{p}_2$ and $\mathbf{p}_1$ then the resultant matrix will be embeddable in $\Re^1$ which is not correct (see Figure 3.4.7–8).

Another important consideration for the unconstrained method is how the integer $r^* = rank(D_1^*) = X^*$ (D_1 given in (3.2.10)) can be identified correctly. Since r^* is not known in advance it is necessary to estimate it by an integer denoted by $r^{(k)}$. Any change to $r^{(k)}$ causes a change to $\phi(X)$, and the number of variables in $\phi(X)$. It is important to consider the effect of making a fixed incorrect estimate r to r^*. If $r^{(k)} < r^*$ then the methods described so far converges satisfactorily and ultimately at a superlinear rate of convergence to a minimizer of $\phi(X)$. Since r is too small the minimizer of $\phi(X)$ is not a solution of (3.4.11), however the matrix $-D^{(k)}$ is a Euclidean distance matrix but it is not the nearest Euclidean distance matrix to $F^{(0)}$. On the other hand if $r^{(k)} > r^*$ then the methods converges to the minimizer of $\phi(X)$, which is the solution of (3.4.11) but the rate of convergence is very slow because the number of variables in $\phi(X)$ are increased. It seems to be difficult to find an estimate of the rank r^* from the structure of the distance matrix $-F$.

Figure 3.4.7: Illustrates the dependence of $\mathbf{p}_1$, $\mathbf{p}_2$ and $\mathbf{p}_3$ which makes D embeddable in $\Re^1$.

Figure 3.4.8: Illustrates the independence of $\mathbf{p}_1$, $\mathbf{p}_2$ and $\mathbf{p}_3$ which makes D irreducibly embeddable in $\Re^2$.

A strategy has been selected to estimate r^*. The above observations suggest we should choose $r^{(0)}$ arbitrarily as a small integer. Subsequently $r^{(k)}$ is increased by one and $\phi(X)$ is minimized by the methods described above for each $r^{(k)}$. Let $-D^{(k)}$ denote the resulting Euclidean distance matrix. If $D^{(k)} = D^{(k+1)}$ then the algorithm terminates. Otherwise $r^{(k)}$ is increased by one which adds $n - 1$ new variables to problem (3.4.11), and it is necessary to add a new vector to the matrix X. This vector is determined randomly. It is important that the independence mentioned above is satisfied by the new vector. In Chapter 4 an alternative approach is studied in which the projection method is used to give a better estimate of the new vector. After adding one to $r^{(k)}$ the problem (3.4.11) is minimized again using one of the unconstrained methods and the above procedures are repeated. As $r^{(k)}$ can only be increased, the correct value r^* will be identified after a few repetitions of the iterative process.

Finally, an advantage of unconstrained method is that it allows the spatial dimensions to be chosen by the user. This is useful where the rank is already known. For example if the distance matrix are distances between cities then the dimension will be no more than $r = 2$. Likewize if the distance matrix are distances between atoms in a molecule or stars in space, then the maximum dimension is $r = 3$.

The disadvantage of Methods 3.4.1–3 is if the rank is unknown. The algorithm may have to be repeated many times before we find the correct rank. This makes convergence very slow. Therefore in Chapter 4 new methods will be introduced for solving problem (3.4.11) (or equivalently problem (3.3.3)) which avoid this disadvantage.

3.5 The Elegant algorithm

In the previous sections a complete description of the projection method and unconstrained methods have been given. The projection method along with Method 3.4.2 described in the previous section are used to construct the methods in Chapter 4.

In this section another method for solving problem (3.3.3) is given. The Elegant algorithm is described by Takane [1977] using a method related to the alternating least squares approach, later modified by Browne [1987].

Their methods are based on computing the gradient of

$$\phi \ = \ \|F \ - \ D\|. \tag{3.5.1}$$

It is then found that if

$$P \ = \ I \ - \ \frac{\mathbf{ee}^T}{n}$$

and

$$S(F) \ = \ diag \ [\sum_{j=1}^{n} f_{1j}, \quad \ldots, \sum_{j=1}^{n} f_{nj}]$$

then

$$\frac{\partial \phi}{\partial D} \ = \ (F \ - \ D \ - \ S(F) \ + \ S(D)) \ PDP \ = \ 0$$

which is a necessary condition for minimality.

Let $-F$ be a distance matrix, then this matrix can be transformed into a $n \times n$ matrix

$$A \ = \ -\tfrac{1}{2} \ PFP.$$

Now let $\Lambda_r \ = \ diag \ [\lambda_1, \ \lambda_2, \ \ldots, \ \lambda_r]$ be the diagonal matrix formed from the r positive eigenvalues of A, and U_r the $n \times r$ matrix of corresponding eigenvectors, and define

$$Y \ = \ U_r \Lambda_r^{1/2}. \tag{3.5.2}$$

Now the Elegant algorithm can be expressed as

Algorithm 3.5.1 (*Elegant algorithm*)

```
Given any distance matrix $-F \in \Re^{n\times n}$, choose $\alpha$, $0 < \alpha < 1$, let $F^{(0)} = F$
        $For \quad k = 1, 2, \ldots$
        $F^{(k+1)} = \alpha(F - S(F) + S(F^{(k)})) + (1-\alpha)Y^{(k)}Y^{(k)T}$
where $Y$ is given in (3.5.2).
```

To improve the rate of convergence, Browne [1987] added a penalty function to (3.5.1) and introduced an intermediate Newton–Raphson step, he called this method the Newton–Raphson method.

	NRA		EA		PA	
n	NI	CPU	NI	CPU	NI	CPU
4	18	0.12	21	0.17	26	0.16
8	31	0.25	17	0.29	19	0.22
16	71	1.94	16	1.62	30	0.78
32	175	32.0(0.05)	14	17.64(2.3)	36	5.01(0.08)
64	367	2189.4(5.23)	14	233.23(17.4)	56	53.46(2.9)
100	708	3095.0(540)	13	948.11(529.8)	68	241.56(16.0)

Table 3.6.1: Numerical comparisons between the three projection algorithms.

PA: Projection Algorithm 3.3.4.
EA: Elegant Algorithm 3.5.1.
NRA: Newton–Raphson algorithm.
NI: Average number of iteration.
CPU: Average CPU time in seconds.
(): Standard deviation in CPU time.

3.6 Numerical results

In this section numerical examples are given for unconstrained methods. First an example of order 4 is given in some detail and then another six examples are given showing how the unconstrained methods behave.

However in the first part comparisons between the projection algorithm and methods of Section 3.5 are considered. In Chapter 4 larger examples for both Algorithm 3.3.4 and Method 3.4.2 are given.

Table 3.6.1 given by Glunt et. al. [1990] compares the three algorithms: the projection Algorithm 3.3.2, the Elegant Algorithm 3.5.1 and the Newton–Raphson algorithm. In Elegant Algorithm $\alpha = \frac{1}{2}$ as long as $F^{(k)}$ is monotonically decreasing and then reducing α by a factor of $\frac{1}{2}$ at non–decreasing. All three algorithm converge to essentially the same values.

The matrices in Table 3.6.1 were randomly generated distance matrices. Table 3.6.1 shows that the projection method consumes less CPU time than the other methods. Therefore it will be used in Chapter 4.

In the rest of this section the numerical result for the quasi–Newton methods of Section 3.4 are discussed. A Fortran program has been written to solve problem (3.4.11) on a Sun computer. The results in this section are accurate to 7–8 decimal places in the distance between the given matrix and the Euclidean distance matrix.

Methods	Initial X		Optimal X		no. of variables	no. of line search	Distance
3.4.1	1	0	1.8104	0.5052	8	15	0
	0	1	0.4601	0.9253			
	0	0	0.0400	-0.4253			
	0	0	-1.3105	-0.0052			
3.4.2	1	0	-1.3682	-0.3581	6	19	0
	0	1	-1.7261	1.0100			
	0	0	-3.0943	0.6523			
3.4.3	1		-1.4142		5	22	0
	0	1	-1.4142	1.4142			
	0	0	-2.8284	1.4142			

Table 3.6.2: Results from example (3.6.1).

In the following an example is given in which $-F$ is a 4×4 Euclidean distance matrix given by

$$-F = \begin{bmatrix} 0 & 2 & 4 & 10 \\ 2 & 0 & 2 & 4 \\ 4 & 2 & 0 & 2 \\ 10 & 4 & 2 & 0 \end{bmatrix}. \tag{3.6.1}$$

This matrix is embedded in $\Re^2$ and F_1 is of rank 2 see Figure 3.6.1. Table 3.6.2 shows the results from the three methods of Section 3.4. They confirm that the programs work since the three distances are zero (see Table 3.6.2). In Table 3.6.2, Method 3.4.1 gives the optimal solution X which implies that the matrix D is the optimal matrix and it turns out to be the same as the matrix $-F$ in (3.6.1) since $-F$ is already a Euclidean distance matrix. This is also true for Methods 3.4.2 and 3.4.3. The distances in Figure 3.6.1 are squared before being stored in the matrix $-F$.

Another thing which is worth noting is that the matrix X is not unique. For example, X for Method 3.4.3 in Table 3.6.2

$$X^T = \begin{bmatrix} -1.41421 & -1.4142 & -2.8284 \\ & 1.4142 & 1.4142 \end{bmatrix}$$

Figure 3.6.1: The Euclidean distance matrix represented in $\Re^2$.

can be replaced by the matrix

$$X^T = \begin{bmatrix} 1.41421 & 1.4142 & 2.8284 \\ & 1.4142 & 1.4142 \end{bmatrix}$$

which gives the same result.

Finally the initial matrix X is chosen to be $[\mathbf{e}_1,\ \mathbf{e}_2,\ 0]$, but using equations (3.4.27–28) to reform the given distance matrix $-F$ to an initial matrix X for Method 3.4.2 reduces the number of line searches to zero. The superlinear convergence turns out to be true in this example.

In Table 3.6.3 six examples are chosen randomly to show how the three methods behave.

The initial matrix

$$X^T = [\mathbf{e}_1,\ \mathbf{e}_2,\ \ldots,\ \mathbf{e}_r,\ 0,\ \ldots,\ 0]$$

is used for all three methods. Probably the best method when n is large $(n \geq 9)$ is Method 3.4.2 because it has the least number of line searches. On the other hand, Method 3.4.1 is better when n is sufficiently small $(n < 9)$.

Method 3.4.3 is the worst because it takes the greatest number of line searches. Also, it fails with certain initial matrices and different initial matrix is needed to solve the problem. Specially when the given matrix is already a Euclidean distance matrix (see Table 3.6.3 $n = 11$).

In the first example ($n = 11$) the given matrix is a Euclidean distance matrix and Method 3.4.3 fails to find the optimal solution for many given initial vectors when $r > 5$ (marked by (*)). In the other methods sometimes the above initial matrix does not find the optimal solution and a different initial matrix is used (marked by (**)). Perhaps, the reason behind this is that Method 3.4.1 and Method 3.4.2 have more freedom to choose the optimal vectors $\mathbf{p}_i$'s near the initial vectors(The optimal vectors $\mathbf{p}_i$'s are not unique). In Method 3.4.3 because of rotation, the initial vectors have to search further for the optimal vectors because they are more specific. One can see this from Table 3.6.3, when r is small, where very few rotations occur, the number of line searches is almost similar with the other methods. When r is bigger the difference in the number of line searches becomes greater. It is clear that Method 3.4.2 is better than 3.4.1 because its number of variables is less than those of Method 3.4.1 by r variables.

In Table 3.6.3 there are two columns for Method 3.4.2. The first column for 3.4.2 has the above initial matrix as initial data every time we increase $r^{(k)}$. This make it comparable with the other methods. In the second column for Method 3.4.2 the initial matrix is reformed from F using equations (3.4.27–28), updating the initial matrix every time we increase $r^{(k)}$ using the previous result. This gives faster convergence for the method.

n	$r^{(k)}$	NL 3.4.1	NL 3.4.2	NL 3.4.2(+)	NL 3.4.3	NIP method	Dist–ance
11	1	23	23	14	23		25.573
	2	40	38	32	33		16.449
	3	49	50	59	63		10.090
	4	63	71	48	86		7.026
	5	80	84	50	198		5.289
	6	91	79	62	109(*)		3.967
	7	90	77	69	81(*)		2.638
	8	92	78	52	83(*)		1.684
	9	70	79	70	99(*)		0.961
	10*	66	72	58	72(*)	3	0
10	1	30	25	17	26		149.63
	2	41	37	31	43		71.407
	3*	45	41	37	47		62.3131
	4	52	51	38	65	47	62.3131
10	1	21	21	17	21(**)		856.302
	2	39	38	26	43		785.213
	3*	42	55	13	66		785.190
	4	55	62	16	79	56	785.190
10	1	29	30(**)	16	43		8107.56
	2	66(**)	48	40	62		6767.53
	3	53	59	47	76		6061.81
	4*	179(**)	77	26	103		5904.95
	5	84	113	52	141	64	5904.95
10	1*	29	35	5	35		990.88
	2	37	48	18	37	125	990.88
10	1	25	26	19	27(**)		34.021
	2	33	30	27	30		26.021
	3	43	40	34	41		24.172
	4*	46	48	40	62		23.973
	5	55	54	45	82	30	23.973

Table 3.6.3: Numerical comparisons between unconstrained methods and the projection algorithm.

(+): Using equations (3.4.27–28). Then updating the initial matrix every time we increase $r^{(k)}$.

NL: Number of line searches.

NIP: Number of iterations for projection method.

Chapter 4

Hybrid methods for finding the nearest Euclidean distance matrix

4.1 Introduction

In this chapter new methods for solving problem (3.3.3) are considered. The methods described here depend upon both projection and unconstrained methods using a hybrid method. The hybrid method works in two stages. First stage is the projection method which converges globally so is potentially reliable but often converges only at first order or slower which can be slow. Meanwhile in the second stage there is quasi–Newton method, in particular Method 3.4.2, which converges superlinearly if the correct rank r^* is given. The main disadvantage of the unconstrained methods are that they require the correct r^*. A hybrid method is one which switches between these methods and aims to combine their best features. To apply an unconstrained method requires a knowledge of the rank r^* and this knowledge can also be gained from the progress of the projection method. Hybrid methods can work well but there is one disadvantage. If the distance matrix have the same rank as the Euclidean distance matrix in which Method 3.4.2 works well, then most of the time will be taken up in the first stage, using the projection method. If this converges slowly then the hybrid method will not solve the problem effectively. Thus it is important to ensure that the second stage method is used to maximum effect. Hence in the algorithm of Section 4.4 the quasi–Newton method is applied first.

In Sections 4.3 and 4.4 two new methods are described. Firstly, there is the projection–unconstrained method, which starts with the projection method to determine the rank $r^{(k)}$ and continues with the unconstrained method. Secondly, the unconstrained–projection method is described, which solves the problem by the unconstrained method and uses the projection method to update the rank. In Section 4.2 a procedure of how to move from one method to another is given. A modified projection algorithm is given in this section, which involves an initial matrix to create a good starting point for the method. Also in this section a method is given, showing how to obtain an initial matrix for the unconstrained method from the result matrix from the projection method. Finally numerical results and comparisons between these hybrid methods and methods of Chapter 3 are given in Section 4.5.

4.2 Updating the result from the projection method to the unconstrained method and conversely

The methods in this chapter are constructed from both the projection method and unconstrained method, starting from one method and then alternating between the two methods at a specific iteration. These alternating methods perform without losing any information. This is because for every result coming from one method will be used to form the initial data for the other method at every alternation. This section shows how this can be done.

Since the rank in the unconstrained method is unknown, it is important to know if $D^{(k)}$ is a Euclidean distance matrix or not every time we rerun the unconstrained algorithm. To do this (3.3.13) is used to test if the matrix $D^{(k)}$ is Euclidean distance matrix or not.

First, consider updating the result data from the unconstrained method to obtain initial data for the projection method.

Let $-D^{(k)}$ be the Euclidean distance matrix obtained from the unconstrained method. If $D^{(k)}$ is a solution to (3.3.3) then there exists some $\Delta^{(k)}$ in

$$D^{(k)} = P_M\,(F^{(k)}) = P_M(F + \Delta^{(k)}) \qquad (4.2.1)$$

for some $\Delta^{(k)}$ and $\Delta^{(k)}$ in general is given by (3.3.11). Denote

$$F^{(k)} = Q \begin{bmatrix} F_1^{(k)} & \mathbf{f}^{(k)} \\ \mathbf{f}^{(k)T} & \zeta^{(k)} \end{bmatrix} Q, \tag{4.2.2}$$

and let $F_1^{(k)} = U^{(k)} \Lambda^{(k)} U^{(k)}$ be the spectral decomposition of $F_1^{(k)}$. By (3.3.9)

$$D^{(k)} = P_M\,(F^{(k)}) = Q \begin{bmatrix} U^{(k)}\Lambda^{(k)+}U^{(k)} & \mathbf{f}^{(k)} \\ \mathbf{f}^{(k)T} & \zeta^{(k)} \end{bmatrix} Q. \tag{4.2.3}$$

Hence

$$\begin{aligned}(D^{(k)} - F^{(k)})\mathbf{e} &= Q \begin{bmatrix} X^{(k)} & \mathbf{0} \\ \mathbf{0}^T & 0 \end{bmatrix} Q\mathbf{e} \\ &= Q \begin{bmatrix} X^{(k)} & \mathbf{0} \\ \mathbf{0}^T & 0 \end{bmatrix} \mathbf{e}_n = \mathbf{0}. \end{aligned} \tag{4.2.4}$$

Since $F^{(k)} = \Delta^{(k)} + F$ from (3.3.11), it follows from (4.2.4) that

$$(\,D^{(k)} - \Delta^{(k)} - F)\mathbf{e} = \mathbf{0} \tag{4.2.5}$$

or

$$\Delta^{(k)}\mathbf{e} = (D^{(k)} - F)\mathbf{e} \tag{4.2.6}$$

Because $\Delta^{(k)}$ is diagonal, (4.2.6) can be used to compute $\Delta^{(k)}$ from $D^{(k)}$. Now if $-D^{(k)}$ from unconstrained algorithm does have the correct rank, and $\Delta^{(k)}$ is computed from (4.2.6), then

$$\mathit{Diag}\ P_M\,(F + \Delta^{(k)}) = \mathbf{0} \tag{4.2.7}$$

and $\Delta^{(k)}$ can be identified as the solution to problem (3.3.3). If it does not have the correct rank then

$$\mathit{Diag}\ P_M\,(F + \Delta^{(k)}) \neq \mathbf{0}$$

and further iterations of the projection algorithm will take place. In this case the diagonal matrix $\Delta^{(k)}$ is used as starting matrix for the projection algorithm. Thus rewriting Algorithm 3.3.4 with this initial matrix gives

Algorithm 4.2.1

Let $-F \in \Re^{n \times n}$ be any distance matrix

$$F^{(0)} = F + \Delta^{(k)} \tag{4.2.8}$$

$$For \quad s = 1,\ 2,\ ...$$

$$F^{(s+1)} = F^{(s)} + [P_d P_M(F^{(s)}) - P_M(F^{(s)})]$$

Conversely, let $-D^{(k)}$ be a Euclidean distance matrix obtain during the projection method. Let r be the rank of the matrix $D_1^{(k)}$, (D_1 is given in (3.2.10)). The initial matrix X for Method 3.4.2 can be calculated using Theorem 3.2.3 as follows:

Define the elements A from $D^{(k)}$ by

$$a_{ij} = -1/2[\, d_{1i} + d_{1j} - d_{ij}] \quad (2 \leq i, j \leq n) \tag{4.2.9}$$

(the minus in (4.2.9) is because $-D^{(k)}$ is the Euclidean distance matrix). If the spectral decomposition of A is

$$A = U \, \Lambda \, U^T$$

then the initial matrix X^T for Method 3.4.2 is given by

$$X^T = \Lambda_r^{1/2} \; U_r^T \tag{4.2.10}$$

where $\Lambda_r = diag\,[\lambda_1, \lambda_2, \ldots, \lambda_r]$, the r largest eigenvalues in Λ and $U_r \in \Re^{n-1 \times r}$ comprises the corresponding columns of U.

4.3 Projection–unconstrained method

The main disadvantage of the unconstrained method is finding the exact rank r^*, since it is not known in advance it is necessary to estimate it by an integer $r^{(k)}$. It is suggested that the best estimate of the matrix rank $r^{(k)}$ is obtained by carrying out some iterations of the projection method. This is because the projection method is a globally convergent method.

Consider Λ_r in (4.2.10), then at the solution the number of eigenvalues in Λ_r is equal to the rank r^*. Thus

$$No.\ \Lambda_r^* = r^* \tag{4.3.1}$$

where $No.\ \Lambda$ is the number of positive eigenvalues in Λ. A similar equation to (4.3.1) is used to calculate an estimated rank $r^{(k)}$ given by

$$No.\ \Lambda_r^{(k)} = r^{(k)}.$$

where Λ_r is given by (4.2.10). The range of error is relatively small. Then the unconstrained method will be applied to solve the problem as described in Section 3.4.

The projection–unconstrained algorithm can be described as follows.

Algorithm 4.3.1

Given any distance matrix $-F$, let s be a positive integer. Then the following algorithm solves problem (3.3.3).

```
i. Let $F^{(0)} = F$
ii. Apply the projection method until
        $No.\ \Lambda_r^{(k)} = No.\ \Lambda_r^{(k+j)} \quad j = 1, 2, \ldots, s$        (4.3.2)
iii. $r^{(k)} = No.\ \Lambda_r^{(k)}$
iv. Find the initial matrix $X$ for the unconstrained method from the result matrix
    $D^{(k)}$ (see(4.2.10)).
v. Minimize $\phi$ in (3.4.11) using Method 3.4.2 to find $D^{(k)}$.
vi. Use (4.2.6) to calculate $\Delta^{(k)}$ from $D^{(k)}$.
        $If$
            $Diag\ P_M\ (F + \Delta^{(k)}) = \mathbf{0}$
        $Then$
            $D^* = D^{(k)}\ and\ terminate$
        $Endif$
```

vii. Apply one iteration of the modified projection method (Algorithm 4.2.1).

viii. Go to (iii).

The integer s in Algorithm 4.3.1 can be any positive number. If it is small then the rank $r^{(k)}$ may not be accurately estimated, however the number of iterations taken by projection method is small. In the other hand if s is large then a more accurate rank is obtained but the projection method needs more iterations.

The advantage of using the projection method as a first stage of the projection–unconstrained method is that if $-F^{(0)}$ is already a Euclidean distance matrix then the projection method terminates at the first iteration. Moreover it gives the best estimate to $r^{(k)}$. Sometimes using Method 3.4.1 instead of Method 3.4.2 for $n < 10$ gives fewer line searches. The test (4.2.7) is used to test if the matrix $-D^{(k)}$ is the nearest Euclidean distance matrix or not. If $-D^{(k)}$ is not the nearest Euclidean distance matrix then the rank $r^{(k)} \neq r^*$ and $r^{(k)}$ is updated using the projection algorithm. The problem (3.4.11) needs to be solved again using Method 3.4.2, also the initial matrix X is calculated from the matrix $D^{(k)}$ using (4.2.10).

Example 4.3.2

An example of this algorithm for $n = 4$

$$-F = \begin{bmatrix} 0 & 1 & 4 & 36 \\ 1 & 0 & 9 & 16 \\ 4 & 9 & 0 & 25 \\ 36 & 16 & 25 & 0 \end{bmatrix}.$$

After two iteration of the projection method equation (4.3.2) is satisfied with $s = 2$ and the number of positive eigenvalues in Λ_r is 2, therefore $r^{(2)} = 2$. The unconstrained method is then applied with initial matrix

$$X^T = \begin{bmatrix} 0.8367 & 1.8100 \\ -1.9001 & 1.2859 \\ 0.1567 & 5.8974 \end{bmatrix}$$

Method 3.4.2 gives the optimal matrix

$$-D^* = \begin{bmatrix} 0 & 3.9965 & 5.2387 & 34.8097 \\ 3.9965 & 0 & 7.7810 & 17.1714 \\ 5.2387 & 7.7810 & 0 & 25.4842 \\ 34.8097 & 17.1714 & 25.4842 & 0 \end{bmatrix}$$

and the test (4.2.7) is satisfied with

$$Diag\ P_M\ (F + \Delta^*) = \mathbf{0}.$$

Thus the matrix $-D^*$ is the nearest Euclidean distance matrix.

4.4 Unconstrained–projection method

Starting with the projection method has the advantage of knowing if the given matrix is a Euclidean distance matrix or not, and it gives the best estimate for the matrix rank $r^{(k)}$. However sometimes it takes many iterations before equation (4.3.2) is satisfied, since the projection method is a slowly convergent method. Also, in many distance matrices $-F$ with large n the rank $r^{(k)}$ estimated by the projection method is bigger then r^*, which mean slow convergence in the unconstrained method. In this method an algorithm starts with the unconstrained method with an arbitrary rank $r^{(k)}$. Then one iteration of the projection method will be calculated after every stage of the unconstrained–projection algorithm. In every stage of this algorithm the resulting matrix $D^{(k)}$ will be used as an initial matrix to the next stage, thus the matrix $D^{(k)}$ is updated at every stage from the previous one.

Now the unconstrained–projection algorithm can be described as follows.

Algorithm 4.4.1

If $-F$ is any distance matrix then the following algorithm solves problem (3.3.3)

i. Let $F^{(0)} = F$.

ii. Choose $r^{(k)}$.

iii. Minimize ϕ in (3.4.11) using Method 3.4.2 to find $D^{(k)}$.

iv. Calculate $Diag\ P_M\ (F + \Delta^{(k)})$ using (4.2.6) to calculate $\Delta^{(k)}$ from $D^{(k)}$ then

```
    $If$
        $Diag\ P_M\ (F + \Delta^{(k)}) = \mathbf{0}$
    $Then$
        $D^* = D^{(k)}$ $terminate$
    $Endif$
```

v. Calculate the diagonal matrix $\Delta^{(k)}$.

vi. Apply one iteration of the modified projection method (Algorithm 4.2.1).

vii. $r^{(k)} = No.\ \Lambda_r^{(k)}$.

viii. Find the initial matrix X for the unconstrained method from the result matrix $D^{(k)}$ (see(4.2.10)).

ix. Go to (iii).

$r^{(k)}$ in stage ii can be chosen using projection method or from the given distance matrix $-F^{(0)}$ using Λ_r in (4.2.10).

Another advantage of this algorithm is that if the rank is not correct then instead of adding one to $r^{(k)}$ it goes back to the projection method to provide a better estimate to $r^{(k)}$. This will increase or decrease $r^{(k)}$ nearer to r^*, therefore variables will be added to or subtracted from the problem. The new variables are estimated using the projection method. Another advantage is that at every stage only one iteration of projection method is used giving a faster converging algorithm.

Example 4.4.2

An example of this algorithm for $n = 5$

$$-F = \begin{bmatrix} 0 & 1 & 2 & 4 & 2 \\ 1 & 0 & 1 & 2 & 4 \\ 2 & 1 & 0 & 1 & 2 \\ 4 & 2 & 1 & 0 & 1 \\ 2 & 4 & 2 & 1 & 0 \end{bmatrix}.$$

If we choose $r^{(0)} = 2$, and minimize ϕ, we find that $Diag\ P_M\ (F + \Delta^{(k)}) \neq \mathbf{0}$ and we have

$$\Delta^{(k)} = \begin{bmatrix} -0.02140 & & & & \\ & -0.09230 & & & \\ & & 0.7068 & & \\ & & & -0.09230 & \\ & & & & -0.31140 \end{bmatrix}.$$

Apply one iteration of Algorithm 4.2.1 with starting diagonal matrix $\Delta^{(k)}$. This implies that $r^{(k)} = 3$. Finally minimize ϕ with starting matrix X derived from $D^{(k)}$ given by Algorithm 4.2.1. We find that

$$-D^* = \begin{bmatrix} 0 & 1.33 & 2 & 3.67 & 2.33 \\ 1.33 & 0 & 1 & 2.33 & 3.67 \\ 2 & 1 & 0 & 1 & 2 \\ 3.67 & 2.33 & 1 & 0 & 1.33 \\ 2.33 & 3.67 & 2 & 1.33 & 0 \end{bmatrix}$$

and hence we find that $Diag\ P_M\ (F + \Delta^{(k)}) = Diag\ P_M\ (F + \Delta^*) = \mathbf{0}$.

4.5 Numerical results

The algorithms of the Sections 4.3 and 4.4 are applied to solve problem (3.3.3). The numerical tests are a set of randomly generated distance matrices with values distributed between 10^{-3} and 10^3. The numerical result for unconstrained–projection method is given in Table 4.5.1 in more detail. Table 4.5.2 compares the four methods projection method, unconstrained Method 3.4.2, projection–unconstrained method and unconstrained–projection method using $\|F^{(k)} - F^{(k-1)}\| < 10^{-5}$ as a stopping criterion. All four algorithms converge to essentially the same values. A Fortran program has been written for these methods to solve problem(3.3.3). The eigenvalues for the projection method are solved using the NAG library.

The computations have been carried out on a Sun computer. In the unconstrained method, for most cases it is observed that fewer iterations are required to solve (3.3.3) as r increases.

For the unconstrained–projection method it is observed that fewer iterations are required as r is increased. This is because it has a good starting matrix updated from the projection method every time r increases. In the unconstrained method for large n we may increase r by 2 or more, this will reduce number of minimizing ϕ in (3.4.11) to half or more. The disadvantage is slow convergence when r exceeds r^*. The projection–unconstrained method and unconstrained–projection method are both very good, and need only a small number of iterations as is shown in both Table 4.5.1 and Table 4.5.2. In the projection–unconstrained method for example the unconstrained method converges very fast (with $n = 50$ only 13 line searches are used), this is because of the good starting initial matrix given by the projection method. Also in the unconstrained–projection method for $n = 50$ only 17 line searches are needed. In Table 4.5.2 for the unconstrained method $r^{(0)}$ is the initial rank then r is

increased by one and unconstrained method is repeated until we find the correct rank r^*.

	UPA			
n	$r^{(0)}$	NL in UA	$r^{(k)}$ from OPA	NL in UA
5	2*	12		
10	3	33	4*	11
15	4	63	5*	13
20	5	70	7*	11
25	6	94	8*	12
30	6	42	9*	10
35	6	98	9*	11
40	6	22	10*	16
45	6	46	11*	18
50	5	125	13*	17

Table 4.5.1: Result from unconstrained–projection Algorithm 4.4.1.

OPA: One iteration from projection Algorithm 4.2.1.
UA: Unconstrained algorithm (Method 3.4.2).
NL: Number of line searches.

	PA		UA			PUA			UPA	
n	r^*	NI	$r^{(0)}$	TNL	NV	NI	$r^{(k)}$	NL	$r^{(0)}$	TNL
5	2	21	2^*	12	8	2	2^*	7	2^*	12
10	4	46	3	80	36	2	4^*	15	3	44
15	5	64	4	140	70	4	$6(5^*)$	22	4	76
20	7	101	5	176	133	4	7^*	18	5	81
25	8	85	6	221	192	4	8^*	14	6	106
30	9	129	6	144	261	4	$10(9^*)$	19	6	52
35	9	115	6	382	306	8	9^*	23	6	109
40	10	168	6	161	390	7	$11(10^*)$	21	6	38
45	11	136	6	246	484	9	11^*	17	6	64
50	13	171	6	288	637	7	13^*	13	5	142

Table 4.5.2: Comparing the four methods.

PA: Projection Algorithm 3.3.4.
UA: Unconstrained algorithm (Method 3.4.2).
PUA: Projection–Unconstrained Algorithm 4.3.1.
UPA: Unconstrained–Projection Algorithm 4.4.1.
NI: Number of iteration in projection algorithm.
NL: Number of line searches in unconstrained algorithm.
TNL: Total number of line searches in unconstrained algorithm.
NV: Number of variables in unconstrained algorithm.

Chapter 5

Methods for minimizing least distance functions with semi–definite matrix constraints

5.1 Introduction

Minimizing a general function subject to semi–definite matrix constraint is a problem which arises in many practical situations, particularly in statistics where the semi–definite matrix constraint is usually a covariance matrix with varying elements. We are interested here in problems in which only the diagonal of the matrix is allowed to change, in the following way. Given a symmetric positive definite matrix $F \in \Re^{n\times n}$ then we consider the problem

$$\begin{aligned} &\textit{minimize} \quad f(\mathbf{x}) \\ &\textit{subject to} \quad \bar{F} + \textit{diag}\,\mathbf{x} \geq 0 \\ &\qquad\qquad x_i \leq v_i \qquad i = 1, ..., n \end{aligned} \tag{5.1.1}$$

where $\bar{F} = F - \textit{Diag}\ F$, $\textit{diag}\,\mathbf{v} = \textit{Diag}\ F$ and f is real valued function of $\mathbf{x}$. In Chapters 6 and 7, such problems are studied in which the objective function is linear.

In this chapter a least distance problem of the following type is solved. Given a symmetric positive semi–definite matrix $F \in \Re^{n\times n}$ then we consider

$$\begin{aligned} &\textit{minimize} \quad \mathbf{x}^T\mathbf{x} \qquad \mathbf{x} \in \Re^n \\ &\textit{subject to} \;\; \bar{F} + \textit{diag}\,\mathbf{x} \geq 0 \\ &\qquad\qquad x_i \leq v_i \qquad i = 1, ..., n \end{aligned} \tag{5.1.2}$$

where $\textit{diag}\ \mathbf{v} = \textit{Diag}\ F$. This kind of problem is important by itself and it is also used subsequently in Chapters 6 and 7. Problem (5.1.2) can be more general if we express it as

$$\begin{aligned} &\textit{minimize} \quad \|\mathbf{a} - \mathbf{x}\|_2^2 \qquad \mathbf{x} \in \Re^n \\ &\textit{subject to} \;\; \bar{F} + \textit{diag}\,\mathbf{x} \geq 0 \\ &\qquad\qquad x_i \leq v_i \qquad i = 1, ..., n \end{aligned} \tag{5.1.3}$$

where $\mathbf{a}$ is an initial point and then we have a different problem with every different $\mathbf{a}$. Problems of this type can be solved in a similar way to methods of this chapter.

Two methods are developed for solving problem (5.1.2). Firstly, a projection algorithm is given for solving problem (5.1.2) using Algorithm 2.2.7 which converges linearly or slower and globally. This method is described in Section 5.2. Subsequently this method is also used in Chapter 6. Secondly an implementation of the l_1 SQP method is used. Fletcher [1985] developed an algorithm for solving problem (5.1.1) in the case $f(\mathbf{x})$ is linear. It is the purpose of this chapter to follow his method but to apply it to problem (5.1.2). Various methods of this type are investigated in Section 5.3.

In Section 5.4 a hybrid method is described, which starts with the projection method to estimate the rank $r^{(k)}$ and continues with the l_1SQP method in a similar way to Section 4.3. Finally in Section 5.5 numerical comparisons of these methods are carried out.

5.2 The Projection algorithm

In this section we give a description of a projection algorithm for solving problem (5.1.2), using the alternating projection method of Algorithm 2.2.7. The constraints in problem (5.1.2) can be expressed as $\bar{F} + \textit{diag}\ \mathbf{x} \in K_{\Re} \cap K_{off} \cap K_b$ which gives an equivalent problem to (5.1.2) and can be expressed as

Given a symmetric positive definite matrix $F = F^T \in \Re^{n\times n}$

$$\begin{aligned} &minimize \quad \|\bar{F} - A\| \\ &subject\ to\ A \in K_{\Re} \cap K_{off} \cap K_b. \end{aligned} \tag{5.2.1}$$

The matrix norm here means the Frobenius norm given in Definition 1.2.2.

Then we follow Algorithm 2.2.7 with $m = 3$ and $K_1 = K_{\Re}$, $K_2 = K_{off}$ and $K_3 = K_b$ as given in (1.3.1), (1.3.5) and (1.3.6) respectively. Algorithm 2.2.7 is the projection algorithm used in this section, and guarantees global convergence to the solution of problem (5.1.2). The projection algorithm requires formulae, which are also given, for calculating the projection maps on to K_{off}, K_b and on to $K_{\Re}$. Subsequently two examples are given for solving problem (5.1.2) using the projection algorithm. Finally an interesting result relating normal cone of the intersection of K_{off}, K_b and $K_{\Re}$ to the solution of problem (5.2.1) is given.

Dykstra's algorithm depends crucially upon the computational complexity of the relevant projections. The minimization problem (5.2.1) is solved by applying Algorithm 2.2.7 to it. Problem (5.2.1) is to find the projection of a matrix to the intersection of three convex sets by a sequence of projections to the individual set successively. First we need definitions for the projection maps $P_{\Re}(\cdot)$, $P_{off}(\cdot)$ and $P_b(\cdot)$, later formulae for them are obtained.

Definition 5.2.1

Let

$$K = \{A : A \in \Re^{n\times n}, A = A^T\},$$

then define the projection map $P_{\Re}(A)$ from K on to $K_{\Re}$, the projection map $P_{off}(A)$ from K on to K_{off} and the projection map $P_b(A)$ from K on to K_b.

The projection map $P_{\Re}(A)$ formula on to $K_{\Re}$ for solving the following problem

$$\begin{aligned} &minimize \ \|F - A\|_F \\ &subject\ to\ A \in K_{\Re} \end{aligned} \tag{5.2.2}$$

is

$$P_{\Re}\ (F)\ = U\Lambda^{+}U^{T}. \tag{5.2.3}$$

where

$$\Lambda^{+}\ =\ \begin{bmatrix} \Lambda_r & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{5.2.4}$$

and $\Lambda_r\ =\ diag\ [\lambda_1,\ \lambda_2,\ \ldots,\ \lambda_r]$ is the diagonal matrix formed from the positive eigenvalues of F. The proof has been given in Theorem 3.3.3.

Since K_{off} consists of all real symmetric $n \times n$ matrices, in which the off–diagonal elements are fixed to F (the given matrix) then

$$P_{off}\ (A)\ =\ \bar{F}\ +\ Diag\ A. \tag{5.2.5}$$

Also, since K_b consisting of all real symmetric $n \times n$ matrices, in which the diagonal elements are not greater than $diag\ \mathbf{v} =\ Diag\ F$, we have

$$P_b\ (A)\ =\ \bar{A}\ +\ diag\ [h_1,\ h_2, ...,\ h_n]. \tag{5.2.6}$$

where

$$\mathbf{h}\ =\ \begin{Bmatrix} h_i\ =\ a_{ii} & if & a_{ii}\ \leq\ v_i \\ h_i\ =\ v_i & if & a_{ii}\ >\ v_i \end{Bmatrix}$$

We can now use projections $P_{\Re}$, P_{off} and P_b given by (5.2.3), (5.2.5) and (5.2.6) respectively to implement Algorithm 2.2.7 giving the following algorithm

Algorithm 5.2.2 (*projection algorithm*)

Given any positive definite matrix F, let $F^{(0)}\ =\ F$

```
$For\ \ k\ =\ 0,\ 1,\ 2,\ \ldots$
$F^{(k+1)}\ =\ F^{(k)}\ +\ [P_b P_{off} P_{\Re}(F^{(k)})\ -\ P_{\Re}(F^{(k)})]$
```

The convergence of this algorithm follows from Theorem 2.2.8 in which the sequences $\{P_{\Re}(F^{(k)})\}, \{P_{off}\ P_{\Re}(F^{(k)})\}$ and $\{P_b\ P_{off}\ P_{\Re}(F^{(k)})\}$ generated by Algorithm 5.2.2 converge in the Frobenius norm to the solution A^{*} of (5.2.1).

Example 5.2.3

An example of Algorithm 5.2.2 for $n = 3$, let

$$\bar{F} = \begin{bmatrix} 0 & 2 & 3 \\ 2 & 0 & 2 \\ 3 & 2 & 0 \end{bmatrix} \qquad \mathbf{v} = \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix}.$$

The solution is $\mathbf{x}^* = (3,\ 4/3,\ 3)$, no bounds are active, the rank of $F^* = \bar{F} + diag\,\mathbf{x}^*$ is $r = 1$.

Example 5.2.4

Another example for $n = 4$, let

$$\bar{F} = \begin{bmatrix} 0 & 1 & 2 & -2 \\ 1 & 0 & 3 & 2 \\ 2 & 3 & 0 & 1 \\ -2 & 2 & 1 & 0 \end{bmatrix} \qquad \mathbf{v} = \begin{bmatrix} 2 \\ 4 \\ 8 \\ 10 \end{bmatrix}.$$

The solution is $\mathbf{x}^* = (2,\ 2.6505,\ 4.1209,\ 6.3537)$. The bound $x_1 \le v_1$ is active. If v_1 is increased to $v_1 = 5$ then the bound $x_1 \le v_1$ is not active and the new solution for this modified problem is

$$\mathbf{x}^* = (3.4555,\ 3.1833,\ 3.1833,\ 3.4555).$$

The rank of $F^* = \bar{F} + diag\,\mathbf{x}^*$ is $r = 2$ in both cases.

In the rest of this section another result is developed giving conditions under which A^* solves (5.2.1). The normal cone $\partial K_{\Re}(A)$ at $A \in K_{\Re}$ is given in (1.3.12). Also the normal cone for $K_{off} \cap K_b$ is given in Theorem 1.3.5. This is based on normal cones for the relevant convex sets. A general result for the normal cone of the intersection of two sets has been given in (1.3.9). Therefore, as in Theorem 3.3.1, if $A \in K_{\Re} \cap K_{off} \cap K_b$ then

$$\partial(K_{\Re} \cap K_{off} \cap K_b)(A) = \partial K_{\Re}(A) + \partial(K_{off} \cap K_b)(A) \tag{5.2.7}$$

Now $\partial K_{\Re}(A)$ and $\partial(K_{off} \cap K_b(A))(A)$ are given in (1.3.12) and (1.3.16) respectively and we let Z, Λ and B denote the matrices that arise. From (5.2.7) and (2.1.3) we can deduce that A^* solves problem (5.2.1) if and only if

$$F - A^* \;=\; -Z\,\Lambda\,Z \;+\; B \;=\; U \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\Lambda \end{bmatrix} U^T \;+\; B \tag{5.2.8}$$

where $U \;=\; [Y \quad Z]$ as in (1.3.15). Then (5.2.8) is equivalently to

$$F \;=\; U \begin{bmatrix} \Lambda_r & \mathbf{0} \\ \mathbf{0} & -\Lambda \end{bmatrix} U^T \;+\; B$$

since

$$A^* \;=\; U \begin{bmatrix} \Lambda_r & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} U^T, \tag{5.2.9}$$

from the spectral decomposition of A^* since $A^* \in K_{\Re}$ from (5.2.1) and U is the same as U in (5.2.8) from Theorem 1.3.7.

5.3 The l_1SQP method

The main idea in this section is to find an algorithm which is globally convergent at a second order rate for solving problem (5.1.2). The idea of transforming the semi–definite matrix constraints in to the form $D_2(A) \;=\; \mathbf{0}$ given in (1.5.10) is used. The SQP methods in Section 1.7 are used in order to have the benefit of the ready availability of second derivatives of (1.5.10) which enables a second order rate of convergence to be achieved. At the end of this section a strategy is described of how to choose the rank r needed to determine D_2. Also two examples for solving problem (5.1.2) are given which are similar to Examples 5.2.3 and 5.2.4. However in the first part of this section we consider the normal cone and the feasible directions sets for the special case in which the positive semi–definite matrix cone $K_{\Re}$ is restricted to the diagonal elements of A (i. e. $A \in K_{\Re} \cap K_{off}$).

Now problem (5.2.1) can be expressed as

$$\begin{aligned} &\underset{\mathbf{x}}{\text{minimize}} \quad \mathbf{x}^T\mathbf{x} \qquad \mathbf{x} \in \Re^n \\ &\textit{subject to} \;\; \bar{A} \,+\, \textit{diag}\,\mathbf{x} \,\in\, K_{\Re} \cap K_{off}(A), \qquad \mathbf{x} \,\le\, \mathbf{v} \end{aligned} \tag{5.3.1}$$

where $diag\,\mathbf{v} = Diag\,F$ (the given matrix).

A useful form of $\partial(K_{\Re} \cap K_{off})(A)$ can be deduced using (1.3.12), let $\bar{B} = B - Diag\,B$ then

$$\partial(K_{\Re} \cap K_{off})(A) = \\ \{\acute{B} |\ Diag\,\acute{B} = B - \bar{B},\ B = -Z\Lambda Z^T,\ \Lambda = \Lambda^T,\ \Lambda \geq 0\} \tag{5.3.2}$$

that is the set of the vectors that are diagonal elements of all matrices of the form $-Z\Lambda Z^T$, where Λ is any symmetric positive semi–definite matrix and Z is the null space matrix.

Furthermore feasible directions for the set $K_{\Re} \cap K_{off}(A)$ can be deduced using (1.4.5)

$$\mathcal{F}(A) = F(A) = \{\bar{A} + diag\,\mathbf{s} |\ Z^T[diag\,\mathbf{s}]Z \geq 0\}. \tag{5.3.3}$$

Optimality conditions follow using Theorem 1.5.2. The first order necessary conditions for $\mathbf{x}^*$ to solve (5.3.1) are that $\mathbf{x}^*$ is feasible and there exist a matrix $\acute{B}^* \in \partial(K_{\Re} \cap K_{off})(A^*)$ and a vector $\boldsymbol{\pi}^* \geq 0$ $(\boldsymbol{\pi}^* \in \Re^n)$ such that

$$2\mathbf{x}^* + \mathbf{b}^* + \boldsymbol{\pi}^* = 0 \tag{5.3.4a}$$

$$\boldsymbol{\pi}^{*T}(\mathbf{v} - \mathbf{x}^*) = 0 \tag{5.3.4b}$$

where $diag\,\mathbf{b}^* = Diag\,\acute{B}^*$.

Now we going to use the second derivatives of (1.5.10) to solve problem (5.3.1).

Assume that the rank of A^* is known to be r $(1 \leq r < n)$. Permute the variables so that the bounds $x_i \leq v_i$ are inactive for $i = r+1, \ldots, n$, then (5.3.1) can be expressed as

$$\begin{aligned} &\underset{\mathbf{x}}{\text{minimize}} \quad \mathbf{x}^T\mathbf{x} \qquad \mathbf{x} \in \Re^n \\ &subject\ \ to\ \ D_2(\mathbf{x}) = 0, \qquad \mathbf{x} \leq \mathbf{v} \end{aligned} \tag{5.3.5}$$

where

$$D_2(\mathbf{x}) = D_2(\bar{A} + diag\,\mathbf{x}) = D_2(A)$$

and $D_2(A)$ is given by (1.5.9). The Lagrangian for problem (5.3.5) is

$$\mathcal{L}(\mathbf{x}, \Lambda, \boldsymbol{\pi}) = \mathbf{x}^T\mathbf{x} - <\Lambda, D_2(\mathbf{x})> + \boldsymbol{\pi}^T(\mathbf{x} - \mathbf{v}). \tag{5.3.6}$$

Also, the first order conditions for this problem are given by (5.3.4a) and (5.3.4b). From (5.3.2) $Diag\ \acute{B}$ is a diagonal matrix which has the same elements as the diagonal of the matrix $-Z^T\Lambda^*Z$ where $\Lambda^*(=[\lambda^*_{ij}] \quad i,j = r+1,\ldots,n)$ is the matrix of Lagrange multipliers for the constraints $D_2(\mathbf{x}) = 0$ and Z is the null space matrix for A^*. The elements of the Lagrange matrix Λ are indexed from $r+1,\ldots,n$ to correspond to the elements d_{ij} of D_2. Then using (1.5.9) in (5.3.6)

$$\frac{\partial \mathcal{L}}{\partial x_i} = 2x_i - \lambda_{ii} + \pi_i = 0. \qquad i = r+1,\ldots,n \tag{5.3.7}$$

The assumption that the bounds are inactive at the solution for $i > r$ i.e. $\pi_i = 0$ implies that

$$\lambda_{ii} = 2x_i. \qquad i = r+1,\ldots,n \tag{5.3.8}$$

To eliminate the variables x_i, $i = r+1,\ldots,n$ (1.5.9) is utilized by using the diagonal elements of $D_2(\mathbf{x})$

$$d_{ii}(\mathbf{x}) = x_i - \sum_{k,l=1}^{r} a_{ik}\ [A_{11}^{-1}]_{kl}\ a_{il} = 0 \qquad i = r+1,\ldots,n \tag{5.3.9}$$

where a_{ik} *and* a_{il} are elements in A_{21}. Therefore the unknown variables are reduced to $\mathbf{x} = [x_1,\ x_2,\ \ldots,\ x_r]^T \in \Re^r$. Then (5.3.5) reduces to

$$\begin{aligned} &\underset{\mathbf{x}}{\text{minimize}} \quad f(\mathbf{x}) = \sum_{k=1}^{r} x_k^2 + \sum_{i=r+1}^{n} x_i^2(\mathbf{x}) \\ &\textit{subject to } d_{ij}(\mathbf{x}) = 0, \quad i \neq j, \qquad i,j = r+1,\ldots,n \\ &\qquad\qquad \mathbf{x} \leq \mathbf{v} \end{aligned} \tag{5.3.10}$$

the alternative unknown vector is determined by (5.3.9). $x_i(\mathbf{x})$ denotes that x_i is the function of $\mathbf{x}$ given by

$$x_i(\mathbf{x}) = \sum_{k,l=1}^{r} a_{ik}\ [A_{11}^{-1}]_{kl}\ a_{il} \qquad i = r+1,\ldots,n \tag{5.3.11}$$

where $Diag\ A_{11} = diag\ \mathbf{x}$.

In (5.3.10) the constraints $d_{ij}(\mathbf{x}) = 0$ and $d_{ji}(\mathbf{x}) = 0$ are both equivalent, therefore in practice the constraints should be presented only for $i > j$ with $2\lambda_{ij}$ as the Lagrange multiplier for each constraint in this system. However in the rest of this section it is more convenient to refer to (5.3.10).

If

$$\Lambda = \begin{bmatrix} 2x_{r+1}(\mathbf{x}) & \cdots & \cdots & \lambda_{r+1\ n} \\ \vdots & \ddots & \vdots & \vdots \\ \lambda_{n-1\ r+1} & \cdots & \cdots & \lambda_{n+1\ n} \\ \lambda_{n\ r+1} & \cdots & \lambda_{n\ n-1} & 2x_n(\mathbf{x}) \end{bmatrix}$$

then (5.3.6) is the Lagrangian function for (5.3.10).

In the following expressions for ∇d_{ij} and $\nabla^2 d_{ij}$ will be derived where ∇ denotes the gradient operator $(\partial/\partial x_1, \ \ldots, \ \partial/\partial x_r)^T$. Differentiating $A_{11}A_{11}^{-1} = I$ gives

$$\frac{\partial A_{11}}{\partial x_s}A_{11}^{-1} + A_{11}\frac{\partial A_{11}^{-1}}{\partial x_s} = 0 \qquad s = 1, \ldots, r$$

$$\Rightarrow \qquad A_{11}\frac{\partial A_{11}^{-1}}{\partial x_s} = -\frac{\partial A_{11}}{\partial x_s}A_{11}^{-1}$$

then

$$\frac{\partial A_{11}^{-1}}{\partial x_s} = -\ A_{11}^{-1}\,\frac{\partial A_{11}}{\partial x_s}A_{11}^{-1},$$

but since

$$\frac{\partial A_{11}}{\partial x_s} = \mathbf{e}_s\mathbf{e}_s^T$$

where $\mathbf{e}_s = (0,\ 0,\ \ldots,\ 0,\ 1,\ 0,\ \ldots,\ 0)$ with one in the sth component, then

$$\frac{\partial A_{11}^{-1}}{\partial x_s} = -\ A_{11}^{-1}\,\mathbf{e}_s\mathbf{e}_s^T\,A_{11}^{-1}. \qquad (5.3.12)$$

Hence from (1.5.9)

$$\begin{aligned}\frac{\partial D_2}{\partial x_s} &= \frac{\partial}{\partial x_s}(A_{22} \;-\; A_{21}A_{11}^{-1}A_{21}^T) \\ &= 0 \;-\; A_{21}\,\frac{\partial A_{11}^{-1}}{\partial x_s}\,A_{21}^T \\ &= A_{21}A_{11}^{-1}\,\mathbf{e}_s\mathbf{e}_s^T\,A_{11}^{-1}A_{21}^T\end{aligned}$$

Using (1.5.11) gives

$$\frac{\partial D_2}{\partial x_s} = V_{21}^T\,\mathbf{e}_s\mathbf{e}_s^T\,V_{21}$$

and hence

$$\frac{\partial d_{ij}}{\partial x_s} = v_{si}\,v_{sj}. \qquad (5.3.13)$$

Furthermore differentiating (5.3.12)

$$\begin{aligned}\frac{\partial^2 A_{11}^{-1}}{\partial x_s \partial x_t} &= \frac{\partial}{\partial x_t}(-\,A_{11}^{-1}\,\mathbf{e}_s\mathbf{e}_s^T\,A_{11}^{-1}) \\ &= -[(-A_{11}^{-1}\,\mathbf{e}_t\mathbf{e}_t^T\,A_{11}^{-1})\,\mathbf{e}_s\mathbf{e}_s^T\,A_{11}^{-1} \;+\; A_{11}^{-1}\,\mathbf{e}_s\mathbf{e}_s^T\,(-A_{11}^{-1}\,\mathbf{e}_t\mathbf{e}_t^T\,A_{11}^{-1})] \\ &= A_{11}^{-1}(\mathbf{e}_t\mathbf{e}_t^T\,A_{11}^{-1}\,\mathbf{e}_s\mathbf{e}_s^T \;+\; \mathbf{e}_s\mathbf{e}_s^T\,A_{11}^{-1}\,\mathbf{e}_t\mathbf{e}_t^T)A_{11}^{-1}.\end{aligned}$$

So from (1.5.9)

$$\begin{aligned}\frac{\partial^2 D_2}{\partial x_s \partial x_t} &= -\,A_{21}A_{11}^{-1}(\mathbf{e}_t\mathbf{e}_t^T\,A_{11}^{-1}\,\mathbf{e}_s\mathbf{e}_s^T \;+\; \mathbf{e}_s\mathbf{e}_s^T\,A_{11}^{-1}\,\mathbf{e}_t\mathbf{e}_t^T)A_{11}^{-1}A_{21}^T \\ &= -\,V_{21}^T(\mathbf{e}_t\mathbf{e}_t^T\,A_{11}^{-1}\,\mathbf{e}_s\mathbf{e}_s^T \;+\; \mathbf{e}_s\mathbf{e}_s^T\,A_{11}^{-1}\,\mathbf{e}_t\mathbf{e}_t^T)V_{21}\end{aligned}$$

hence

$$\frac{\partial^2 d_{ij}}{\partial x_s \partial x_t} = -\,(v_{si}\,v_{tj} \;+\; v_{ti}\,v_{sj})[A_{11}^{-1}]_{st}. \qquad (5.3.14)$$

where $[A_{11}^{-1}]_{st}$ means the element of A_{11}^{-1} in st position.

For the SQP method the solution of the QP subproblem (1.7.8) is needed. In (1.7.8) $c_i = d_{ij}$ and $\nabla c_i = \mathbf{a}_i^T = \nabla d_{ij}^T$, which are given in (5.3.13). From (5.3.9), (5.3.10) and (5.3.11)

$$\nabla f = 2\mathbf{x} + 2 \sum_{i=r+1}^{n} x_i(\mathbf{x}) \nabla x_i(\mathbf{x})$$

then

$$\nabla f = 2\mathbf{x} - 2 \sum_{i=r+1}^{n} x_i(\mathbf{x}) \nabla d_{ii} \tag{5.3.15}$$

and

$$\nabla^2 f = 2I - 2 \sum_{i=r+1}^{n} [x_i(\mathbf{x}) \nabla^2 d_{ii} - (\nabla d_{ii})(\nabla d_{ii})^T] \tag{5.3.16}$$

Now in the QP subproblem (1.7.8) $W = \nabla^2 \mathcal{L}(\mathbf{x}, \Lambda, \boldsymbol{\pi})$ then from (5.3.6) and (5.3.16)

$$W^{(k)} = \nabla^2 \mathcal{L}(\mathbf{x}^{(k)}, \Lambda^{(k)}, \boldsymbol{\pi}^{(k)})$$

$$= 2I - 2 \sum_{i=r+1}^{n} [x_i(\mathbf{x}^{(k)}) \nabla^2 d_{ii}(\mathbf{x}^{(k)}) \tag{5.3.17}$$

$$- (\nabla d_{ii}(\mathbf{x}^{(k)}))(\nabla d_{ii}(\mathbf{x}^{(k)}))^T] - \sum_{\substack{i,j=r+1 \\ i \neq j}}^{n} \lambda_{ij}^{(k)} \nabla^2 d_{ij}(\mathbf{x}^{(k)}). \tag{5.3.18}$$

Including term (5.3.17) in the diagonal of the last term of (5.3.18) with $\lambda_{ii}^{(k)} = 2x_i(\mathbf{x}^{(k)})$ (from (5.3.8)) gives

$$W^{(k)} = 2I + 2 \sum_{i=r+1}^{n} [(\nabla d_{ii}(\mathbf{x}^{(k)}))(\nabla d_{ii}(\mathbf{x}^{(k)}))^T] - \sum_{i,j=r+1}^{n} \lambda_{ij}^{(k)} \nabla^2 d_{ij}(\mathbf{x}^{(k)}). \tag{5.3.19}$$

Now

$$\sum_{i=r+1}^{n} [(\nabla d_{ii}(\mathbf{x}^{(k)}))(\nabla d_{ii}(\mathbf{x}^{(k)}))^T = \begin{bmatrix} \sum_i v_{1i}^2 v_{1i}^2 & \cdots & \sum_i v_{1i}^2 v_{ri}^2 \\ \vdots & \ddots & \vdots \\ \sum_i v_{ri}^2 v_{1i}^2 & \cdots & \sum_i v_{ri}^2 v_{ri}^2 \end{bmatrix}$$

$$= UU^T \tag{5.3.20}$$

since $\partial d_{ii}/\partial x_s \;=\; v_{si}^2$, where $U \;=\; [V_{12}][V_{12}]$ and $[\;\;][\;\;]$ means the componentwise product. Rearranging (5.3.19) using (5.3.20) and (5.3.14) gives

$$\begin{aligned}
[W^{(k)}]_{st} &= [2I]_{st} \;+\; 2[UU^T]_{st} \;+\; 2[V_{12}\Lambda^{(k)}V_{12}^T]_{st}[A_{11}^{-1}]_{st} \\
&= [2I]_{st} \;+\; 2[UU^T]_{st} \;+\; 2[V_{12}\Lambda^{(k)}V_{12}^T]_{st}[V_{11}D_1^{-1}V_{11}^T]_{st} \qquad (5.3.21)
\end{aligned}$$

where $s,t \;=\; 1,\ldots,r$. V and D in (5.3.21) are calculated using (1.5.11) and (1.5.4–5).

From the above expressions the QP subproblem (1.7.8) can be expressed as

$$\begin{aligned}
&\underset{\boldsymbol{\delta}}{\text{minimize}} \quad f^{(k)} \;+\; \nabla f^{(k)}\boldsymbol{\delta} \;+\; \tfrac{1}{2}\,\boldsymbol{\delta}^T W^{(k)}\boldsymbol{\delta} \qquad\qquad \boldsymbol{\delta} \;\in\; \Re^r \\
&\textit{subject to} \;\; d_{ij}^{(k)} \;+\; \nabla\, d_{ij}^{(k)T}\boldsymbol{\delta} \;=\; 0 \qquad i \;\neq\; j \quad i,j = r+1,\ldots,n \\
&\qquad\qquad\qquad \mathbf{x}^{(k)} \;+\; \boldsymbol{\delta} \;\leq\; \mathbf{v} \qquad\qquad\qquad\qquad\qquad\qquad (5.3.22)
\end{aligned}$$

giving a correction vector $\boldsymbol{\delta}^{(k)}$, so that $\mathbf{x}^{(k+1)} \;=\; \mathbf{x}^{(k)} + \boldsymbol{\delta}^{(k)}$. Further the Lagrange multipliers of the equations in (5.3.22) become the elements $\lambda_{ij}^{(k+1)}$ for the next iteration.

The matrix W^* is positive semi–definite. This can be proved using (5.3.21) because

$$\mathbf{z}^T W^*\mathbf{z} \;=\; 2\mathbf{z}^T\mathbf{z} \;+\; 2\mathbf{z}^T UU^T\mathbf{z} \;+\; 2\mathbf{z}^T[V_{12}\Lambda^* V_{12}^T][V_{11}D_1^{-1}V_{11}^T]\mathbf{z}. \qquad (5.3.23)$$

Since $\mathbf{z}^T UU^T\mathbf{z} \;\geq\; 0$ and from (5.3.2) $\Lambda^* \;\geq\; 0$ then

$$\begin{aligned}
\mathbf{z}^T W^*\mathbf{z} &= 2\mathbf{z}^T\mathbf{z} \;+\; 2\mathbf{z}^T UU^T\mathbf{z} \;+\; 2\; tr(V_{12}\Lambda^* V_{12}^T[\textit{diag}\;\mathbf{z}]V_{11}D_1^{-1}V_{11}^T[\textit{diag}\;\mathbf{z}]) \\
&= 2\mathbf{z}^T\mathbf{z} \;+\; 2\mathbf{z}^T UU^T\mathbf{z} \\
&\qquad +\; 2\; tr(D_1^{-1/2}V_{11}^T[\textit{diag}\;\mathbf{z}]V_{12}\Lambda^* V_{12}^T[\textit{diag}\;\mathbf{z}]V_{11}D_1^{-1/2}) \\
&\geq 0 \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad (5.3.24)
\end{aligned}$$

since

$$\{D_1^{-1/2}V_{11}^T[\textit{diag}\;\mathbf{z}]V_{12}\}\;\Lambda^*\;\{V_{12}^T[\textit{diag}\;\mathbf{z}]V_{11}D_1^{-1/2}\}^T$$

is symmetric and positive semi–definite. Therefore if $\mathbf{x}^{(k)}$ is sufficiently close to $\mathbf{x}^*$ the basic SQP method converges and the rate is second order (see Section 1.7).

It is shown in Section 1.7 that the SQP method may not converge globally and it is usually modified by the l_1 exact penalty function. An equivalent form to (1.7.11) for problem (5.3.10) is

$$\begin{aligned}\phi(\mathbf{x}) \;=\; \sum_{k=1}^{r} x_k^2 \;+\; & \sum_{i=r+1}^{n} x_i^2(\mathbf{x}) \\ & +\; \sigma\{ \sum_{\substack{i,j=r+1\\ i\neq j}}^{n} |d_{ij}(\mathbf{x})| \;+\; \sum_{i=r+1}^{n} \max(v_i \;-\; x_i,\; 0)\}. \qquad (5.3.25)\end{aligned}$$

Since the bounds are inactive for $i \;>\; r,\;\; \pi_i$ is zero, implying that the max terms are zero if $\mathbf{x}^{(k)}$ is sufficiently close to $\mathbf{x}^*$. To guarantee that the minimizer $\mathbf{x}^*$ of (5.3.25) satisfies first order conditions for (5.3.10), the penalty parameter σ in (5.3.25) must satisfy

$$\sigma \;\geq\; \max_{ij} |\lambda^*_{ij}|. \qquad i,j \;=\; r+1,\ldots,n$$

Now since $\Lambda^* \;\geq\; 0$ and $\lambda^*_{ii} \;=\; 2x^*_i \quad i \;=\; r+1,\;\ldots,\; n$ then

$$\max_{ij} |\lambda^*_{ij}| \;\leq\; 2\;\max_{i}\; x^*_i. \qquad i,j \;=\; r+1,\ldots,n$$

Hence $\sigma \;\geq\; 2\;\max_i\; x^*_i$ must hold. However, since it is advantageous to choose σ as small as possible, the choice $\sigma \;=\; 2\;\max_i\; x^*_i$ is recommended. In practice if the unnecessarily redundant form of (5.3.10) is used with summation over indices $i \;>\; j$, then a similar summation is used in (5.3.25) and the choice $\sigma \;=\; 4\;\max_i\; x^*_i$ is recommended.

To ensure the descent property, it may be necessary to choose larger values of σ than $\sigma \;=\; 4\;\max_i\; x^*_i$ the choice

$$\sigma \;>\; \max_{ij} |\lambda^{(k+1)}_{ij}| \qquad i,j \;=\; r+1,\ldots,n$$

is sufficient. Unfortunately it has been observed that the resulting values of σ are very large and no successful algorithm of this type has been obtained. For more about how to choose the penalty parameter σ see Fletcher [1987] Chapter 12.

Algorithm 1.7.3 which has better convergence properties is now recommended. This differs from the formulation given in (5.3.25). An equivalent form to (1.7.12) is the following

$$\begin{aligned}&\underset{\boldsymbol{\delta}}{\text{minimize}} \quad \psi^{(k)}(\boldsymbol{\delta}) \\ &\textit{subject to}\;\; \mathbf{x}^{(k)} \;+\; \boldsymbol{\delta} \;\leq\; \mathbf{v} \\ &\qquad\qquad \|\boldsymbol{\delta}\|_\infty \;\leq\; \rho^{(k)} \qquad (5.3.26)\end{aligned}$$

where

$$\psi^{(k)}(\boldsymbol{\delta}) \;=\; f^{(k)} \;+\; \nabla f^{(k)T}\boldsymbol{\delta} \;+\; \tfrac{1}{2}\,\boldsymbol{\delta}^T W^{(k)}\boldsymbol{\delta} \;+\; \sigma\{\textstyle\sum_{\substack{i,j=r+1\\ i\neq j}}^{n} |d_{ij}^{(k)} \;+\; \nabla\; d_{ij}^{(k)T}\boldsymbol{\delta}|\} \qquad (5.3.27)$$

giving a correction vector $\boldsymbol{\delta}^{(k)}$, so that $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \boldsymbol{\delta}^{(k)}$. Also the Lagrange multipliers associated with each of the modulus terms in (5.3.27) become the elements of the matrix $\Lambda^{(k+1)}$ for the next iteration. The subproblem (5.3.26) can be solved by methods similar to those used in QP. The two methods (5.3.22) and (5.3.26) are equivalent when $\mathbf{x}^{(k)}, \Lambda^{(k)}$ are sufficiently close to $\mathbf{x}^*, \Lambda^*$ and σ is large enough.

In Han's method [1977] it is necessary for $\nabla^2\mathcal{L}^{(k)} \geq 0$ to hold, which excludes the possibility of an unbounded solution to (5.3.22). However in (5.3.26) it is not necessary to force $\nabla^2\mathcal{L}^{(k)} \geq 0$ to hold, and the choice $\sigma = 2\ \max_i\ x_i^{(k)}$ can be used.

The terms $|d_{ij}(\mathbf{x})|$ in (5.3.25) are not smooth and can cause slow convergence in practice. The second order correction is included to alleviate these difficulties. Let $\boldsymbol{\delta}^{(k)}$ be the solution of (5.3.26), then the second order correction is obtained by repeating (5.3.26) with some modification to (5.3.26), giving the subproblem

$$\begin{aligned} &\underset{\boldsymbol{\delta}}{\text{minimize}} \quad \psi^{(k)}(\boldsymbol{\delta}^{(k)}) \\ &\textit{subject to}\ \ \mathbf{x}^{(k)} + \boldsymbol{\delta} \leq \mathbf{v} \\ &\qquad\qquad\qquad \|\boldsymbol{\delta}\|_\infty \leq \rho^{(k)} \end{aligned} \qquad (5.3.28)$$

where

$$\psi^{(k)}(\boldsymbol{\delta}^{(k)}) \;=\; f^{(k)} \;+\; \nabla f^{(k)T}\boldsymbol{\delta} \;+\; \tfrac{1}{2}\,\boldsymbol{\delta}^T W^{(k)}\boldsymbol{\delta} \;+\; \sigma\{\textstyle\sum_{\substack{i,j=r+1\\ i\neq j}}^{n} |d_{ij}^{(k)} \;+\; \nabla\; d_{ij}^{(k)T}\boldsymbol{\delta} \;+\; \gamma^{(k)}|\}$$

and

$$\gamma^{(k)} \;=\; \tfrac{1}{2}\;\boldsymbol{\delta}^{(k)T}\nabla^2\; d_{ij}^{(k)}\boldsymbol{\delta}^{(k)} \qquad (5.3.29)$$

and $\boldsymbol{\delta}^{(k)}$ calculated from (5.3.26). The solution to (5.3.28) is denoted by $\tilde{\boldsymbol{\delta}}^{(k)}$. The modified algorithm solves (5.3.26) as before to get $\boldsymbol{\delta}^{(k)}$ then calculates $\gamma^{(k)}$ using (5.3.29) then recalculates $\tilde{\boldsymbol{\delta}}^{(k)}$ using (5.3.28) and revised Lagrange multipliers $\tilde{\Lambda}^{(k+1)}$. Now $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \tilde{\boldsymbol{\delta}}^{(k)}$ and $\tilde{\Lambda}^{(k+1)}$ is used in place of $\Lambda^{(k+1)}$. Using the second order correction takes advantage of the readily available second derivative matrices $\nabla^2\ d_{ij}\quad i,j = r+1,\ldots,n$.

An important constraint has been neglected up till now, that is the variables $\mathbf{x} \in \Re^r$ must permit the matrix $\bar{A} + \mathit{diag}\ \mathbf{x}$ to be factorized as in (1.5.4) with $D_1 > 0$. Therefore the restriction $D_1(\mathbf{x}) > 0$ on the feasible region of (5.3.10) is enforced. Also certain degenerate cases must be excluded. However if $\mathbf{x}^{(k)}$ is sufficiently close to $\mathbf{x}^*$ and r is identified correctly this restriction will usually be inactive at the solution. If $\mathbf{x}^{(k)}$ is remote from the solution then two constraints are introduced to avoid these disadvantages. Firstly the linearization of the constraint $d_{ii}(\mathbf{x}) \geq 0$

$$d_{ii}^{(k)} + \nabla d_{ii}^{(k)T} \boldsymbol{\delta} \geq 0. \quad i = r+1, \ldots, n \tag{5.3.30}$$

are added to the subproblems (5.3.22),(5.3.26) or (5.3.28). Secondly the linearization of the constraint $D_1(\mathbf{x}) > 0$ about $\mathbf{x}^{(k)}$

$$d_{ss}^{(k)} + \nabla d_{ss}^{(k)T} \boldsymbol{\delta} > 0. \quad s = 1, \ldots, r \tag{5.3.31}$$

However it is advisable not to allow $d_{ss}(\mathbf{x}^{(k)} + \boldsymbol{\delta})$ to become too close to zero, especially for small s which causes the factorization to fail ($D_1 \not> 0$). As a result the constraints

$$s\, d_{ss}^{(k)}/r + \nabla d_{ss}^{(k)T} \boldsymbol{\delta} \geq 0. \quad s = 1, \ldots, r \tag{5.3.32}$$

are also included to the subproblems (5.3.22),(5.3.26) or (5.3.28).

Even with these extra conditions it might be difficult to find a partial factor for the matrix $\bar{A} + diag\, \mathbf{x}$ in the form (1.5.5) for some iterates $\mathbf{x}^{(k)}$. In this case smaller radius for the trust region is chosen with $\rho^{(k+1)} = \rho^{(k)}/4$, $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)}$ and $\Lambda^{(k+1)} = \Lambda^{(k)}$ are chosen for the next iteration.

Another restriction on the variables $\mathbf{x} \in \Re^r$ of (5.3.10) is that the bounds $x_i \leq v_i$, $i = r+1, \ldots, n$ must remain inactive. This can be done by permuting the variables, although an acceptable permutation is not known in advance. Therefore the following procedure has been adopted. In the beginning of an iteration every variable is tested individually to reorganize the variables so that the active variables is first. As result of that the active bounds are those on variables $x_s^{(k)}$, $s = 1, \ldots, p$ where p is number of active bounds. This permutation makes a complete change to the factorization (1.5.9) so that the matrix D_2 and the basis matrix Z are redefined. The Lagrange multipliers are reset to zero since they are not suitable to the redefined basis. Also the function $\phi(\mathbf{x})$ in (5.3.25) is redefined. The number of permutations made during the course of the algorithm must be finite, this is because the above procedures conflict with the global convergence strategy of reducing $\phi(\mathbf{x}^{(k)})$ monotonically if the number of permutations are not finite.

Another important consideration for the l_1 SQP method is how the integer r^* can be identified correctly. Since r^* is not known in advance it is necessary to estimate it by an integer denoted by $r^{(k)}$. Any change to $r^{(k)}$ causes a change to $\phi(\mathbf{x})$, and the number of variables in $\phi(\mathbf{x})$. It is important to consider the effect of making a fixed incorrect estimate r to r^*. If $r^{(k)} < r^*$ then the l_1 SQP method converges satisfactorily at a second order rate to a minimizer $\phi(\mathbf{x})$. Since r is too small this minimizer is not a solution to (5.3.10) because $d_{ij}(\mathbf{x}) \neq 0$ for some indices $i \neq j \quad i, j = r+1, \ldots, n$, and also because $\Lambda \geq 0$ does not usually hold. On the other hand if $r^{(k)} > r^*$ then the l_1 SQP algorithm converges to the minimizer of $\phi(\mathbf{x})$, which is the solution of (5.3.10) but the rate of convergence is very slow because the number of variables in $\phi(\mathbf{x})$ are increased. The slow rate of convergence indicates that the nonsmooth nature of the problem is not accounted for. The initial idea is to increase or decrease $r^{(k)}$ as the iteration proceeds, using the fact that $\Lambda^{(k)} \not\geq 0$ to increase $r^{(k)}$, and the existence of an active constraint for $s = r$ in (5.3.31) to decrease $r^{(k)}$. The above idea by Fletcher [1982] was not in fact investigated, which it may be necessary to do for large problems. However the more simple strategy described in Section 5.5 below proved to be very reliable and reasonably efficient, especially for $n \leq 20$.

Two examples for problem (5.3.1) are given which are similar to Examples 5.3.2 and 5.3.3.

Example 5.3.1

Consider problem (5.3.1) where

$$\bar{F} = \begin{bmatrix} 0 & 2 & 3 \\ 2 & 0 & 2 \\ 3 & 2 & 0 \end{bmatrix} \qquad \mathbf{v} = \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix}.$$

The solution is $x^* = (3,\ 4/3,\ 3)$, no bounds are active i.e. $\boldsymbol{\pi}^* = 0$, and the set

$$K_{\Re} \cap K_{off}(\bar{F} + diag\, \mathbf{x}) = \{\bar{F} + diag\, \mathbf{x} | \begin{bmatrix} x_1 & 2 & 3 \\ 2 & x_2 & 2 \\ 3 & 2 & x_3 \end{bmatrix} \geq 0\} \tag{5.3.33}$$

is illustrated in the neighbourhood of $\mathbf{x}^*$ in Figure 5.3.1

It can be observed that $K_{\Re} \cap K_{off}(\bar{F} + \mathit{diag}\,\mathbf{x})$ is convex but not a cone and is nonsmooth at $\mathbf{x}^*$. The rank of $F^* = \bar{F} + \mathit{diag}\,\mathbf{x}^*$ is $r = 1$, and its partial factors are

$$D = \begin{bmatrix} 3 & & \\ & 0 & \\ & & 0 \end{bmatrix} \qquad L = \begin{bmatrix} 1 & & \\ 2/3 & 1 & \\ 1 & 0 & 1 \end{bmatrix}$$

$$L^{-1} = V = \left[\begin{array}{c:cc} 1 & -2/3 & -1 \\ & 1 & 0 \\ & & 1 \end{array}\right].$$

thus

$$Z = \begin{bmatrix} -2/3 & -1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

The vector $\mathbf{b}^* = -2\mathbf{x}^* = (-6, -8/3, -6)$ satisfies (5.3.4a) and the corresponding $B^* \in \partial K_{\Re}$ is generated by the matrix

$$\Lambda^* = \begin{bmatrix} 8/3 & -8/9 \\ -8/9 & 6 \end{bmatrix}$$

($\Lambda^* > 0$ as required), and

$$B^* = -Z\Lambda^* Z^T = -\begin{bmatrix} 6 & -8/9 & -146/27 \\ -8/9 & 8/3 & -8/9 \\ -146/27 & -8/9 & 6 \end{bmatrix}.$$

Example 5.3.2

Another example for $n = 4$, let

Figure 5.3.1: The boundary of the restricted cone $(K_{\Re} \cap K_{off})(\bar{F} + \mathit{diag}\,\mathbf{x})$ in (5.3.33) (contours of x_2).

$$\bar{F} = \begin{bmatrix} 0 & 1 & 2 & -2 \\ 1 & 0 & 3 & 2 \\ 2 & 3 & 0 & 1 \\ -2 & 2 & 1 & 0 \end{bmatrix} \qquad \mathbf{v} = \begin{bmatrix} 2 \\ 4 \\ 8 \\ 10 \end{bmatrix}.$$

The solution is $(2,\ 2.6505,\ 4.1209,\ 6.3537)^T$. The rank of $F^* = \bar{F} + \mathit{diag}\,\mathbf{x}^*$ is $r = 2$, and its partial factors are

$$D = \begin{bmatrix} 2 & & & \\ & 2 & & \\ & & 0 & \\ & & & 0 \end{bmatrix} \qquad L = \begin{bmatrix} 1 & & & \\ 0.5 & 1 & & \\ 1 & 1 & 1 & \\ -1 & 1.5 & 0 & 1 \end{bmatrix}$$

$$L^{-1} = V = \begin{bmatrix} 1 & -0.5 & \vdots & -2/3 & -1 \\ & 1 & \vdots & -1 & -1.5 \\ & & \vdots & 1 & 0 \\ & & \vdots & & 1 \end{bmatrix}.$$

thus

$$Z = \begin{bmatrix} -2/3 & -1 \\ -1 & -1.5 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

The bound $x_1 \le v_1$ is active and has a Lagrange multiplier $\pi_1^* = 55.37079$. The vector $\mathbf{b}^* + \boldsymbol{\pi}^* = -2\mathbf{x}^*$ satisfies (5.3.4a) and the corresponding $B^* \in \partial K_{\Re}$ is generated by the matrix

$$\Lambda^* = \begin{bmatrix} 8.2418 & -10.5108 \\ -10.5108 & 12.7074 \end{bmatrix}$$

If the bound v_1 is increased to $v_1 = 4$ for example, then the bound $x_1 \le v_1$ becomes inactive and the vector $(2,\ 2.6505,\ 4.1209,\ 6.3537)^T$ is feasible but not optimal with $\sqrt{\mathbf{x}^T\mathbf{x}} = 8.269$. This time the conditions (5.3.4a) and (5.3.4b) do not hold. The optimal solution to this modified problem is

$$\mathbf{x}^* = (3.4555,\ 3.1833,\ 3.1833,\ 3.4555)^T$$

with $\sqrt{\mathbf{x}^T\mathbf{x}} = 6.644$ and $r = 2$. Second order conditions are used in this modified problem.

5.4 A hybrid method

In this section a new method for solving problem (5.1.2) is considered. The method described here depends upon both projection and l_1 SQP methods using a hybrid method. The projection method which converges globally but often converges at very slow order. Meanwhile in the l_1 SQP method which converges at second order if the correct rank r^* is given. The main disadvantage of the l_1 SQP method are that they require the correct r^*. The projection–l_1 SQP method starts with the projection method to determine the rank $r^{(k)}$ and continues with the l_1 SQP method.

The method in this section follows a similar strategy to that in Section 4.3. Since r^* is not known in advance it is necessary to estimate it by an integer $r^{(k)}$. It is suggested that the best estimate of the matrix rank $r^{(k)}$ is obtained by carrying out some iterations of the projection method. This is because the projection method is a globally convergent method.

Consider Λ_r from (5.2.4) then at the solution the number of eigenvalues in Λ_r is equal to the rank of A^*. Thus

$$No.\ \Lambda_r^* \ = \ rank(A^*) \ = \ r^* \tag{5.4.1}$$

where $No.\ \Lambda$ is the number of positive eigenvalues in Λ. A similar equation to (5.4.1) is used to calculate an estimated rank $r^{(k)}$ and is given by

$$No.\ \Lambda_r^{(k)} \ = \ r^{(k)}.$$

where Λ_r is given by (5.2.4). The range of error is relatively small. The l_1 SQP method will be applied to solve the problem as described in Section 5.3.

The projection–l_1 SQP algorithm can be described as follows.

Algorithm 5.4.1

Given any matrix $F = F^T \in \Re^{n\times n}$, let s be a positive integer. Then the following algorithm solves problem (5.1.2)

i. Let $F^{(0)} = F$

ii. Apply the projection method until

$$No.\ \Lambda_r^{(k)} \ = \ No.\ \Lambda_r^{(k+j)} \qquad j \ = \ 1, 2, \ \ldots, s \tag{5.4.2}$$

iii. $r^{(k)} \ = \ No.\ \Lambda_r^{(k)}$

iv. Use the result vector $\mathbf{x}$ from projection method as an initial vector for the l_1 SQP method

v. Apply the l_1 SQP method for solving problem (5.1.2).

The integer s in Algorithm 5.4.1 can be any positive number. If it is small then the rank $r^{(k)}$ may not be accurately estimated, however the number of iterations taken by the projection method is small. In the other hand if s is large then a more accurate rank is obtained but the projection method needs more iterations.

The advantage of using the projection method as the first stage of the projection–l_1 SQP method is that if $F^{(0)}$ is positive semi–definite (singular) then the projection method terminates at the first iteration. Moreover it gives the best estimate to $r^{(k)}$.

It has been found difficult to produce an algorithm starting with l_1SQP method and then using the projection method to update the rank, in contrast to the method in Section 4.4.

A way of finding a lower bound on the rank $r^{(k)}$ is suggested by Fletcher [1985]. The number of free variables in problem (5.1.2) are at most n, and this can be reduced to $n - p$ if there are p active bounds at the solution. Since $D_2 \in \Re^{(n-r)\times(n-r)}$ and symmetric then the equation $D_2 = \mathbf{0}$ introduces $1/2\,(n - r + 1)(n - r)$ conditions, so except in degenerate cases it follows that

$$n - p \geq 1/2\,(n - r + 1)(n - r) \tag{5.4.3}$$

which imposes a significant restriction on the dimensions of D_2. For example if $n - p = 20$ and $n = 21$ then r can be no smaller than 14.

5.5 Numerical results and comparisons

In this section numerical examples are given for the projection algorithm l_1 SQP algorithm and Algorithm 5.4.1. First numerical examples for Algorithm 5.2.2 are given in some detail in Table 5.5.1 then the same numerical examples for l_1 SQP algorithm and Algorithm 5.4.1 are given in Table 5.5.2.

The numerical test problems are obtained from the data given in Table 6.2.1, by Woodhouse [1976].

The projection Algorithm 5.2.2, l_1 SQP algorithm and Algorithm 5.4.1 are applied to solve problem (5.1.2). The Woodhouse data set is a 64×20 data which corresponds to 64 students and 20 subtests. Various selections from the set of subsets of columns are used to give various test problems to form the matrix A. These subsets are those given in the first columns of Tables 5.5.1 and 5.5.2, the value of n is the number of elements in each subset.

The results obtained by the Algorithm 5.2.2 are tabulated in Table 5.5.1. Using $\|\mathbf{x}^{(k+1)}\| - \|\mathbf{x}^{(k)}\| < 10^{-8}$ as a stopping criterion it is estimated that the x_i are accurate to $4-5$ decimal places and $\|\mathbf{x}\|_2$ is accurate to $6-7$ decimal places. In Table 5.5.1 the column headed by NI gives the number of iterations used by the projection method. It is clear from Table 5.5.1 that when the bounds are active the number of iterations becomes very large. The x_i^* elements marked by $(*)$ are the active elements.

Moreover Table 5.5.1 gives the correct rank r^* for each particular problem. The order of convergence is very slow as can be seen from Table 5.5.1. Also in Table 5.5.1 the optimal x_i^* for $i = 1, 2, ..., n$ and $\|\mathbf{x}^*\|_2$ are given. Finally, the eigenvalues for the projection method are solved using the NAG library.

At the end of Section 5.3 a difficult strategy had been described for applying the l_1 SQP method. A more simple strategy has been adopted. Initially choosing $r^{(k)}$ as the smallest integer compatible with (5.4.3). Starting from $\mathbf{x}^{(0)} = \mathbf{v}$, $\Lambda^{(0)} = 0$ and $\rho^{(0)}$ supplied by the user then $\phi(\mathbf{x})$ is minimized by the iteration based on (5.3.26) as described in Section 5.3. Thus if $\|D_2(\mathbf{x})\| \leq \epsilon$ for some small ϵ, at the solution then the algorithm terminates. If not then $r^{(k)}$ is increased by one. Then a new variable x_{r+1} is adding to problem (5.3.10). This variable is estimated by adding the value of the l_1 norm of the first column of D_2 to the current value of x_{r+1} as given by (5.3.9). Then the partial factors of the new matrix are well–determined. Also increases in $r^{(k)}$ reduce the dimension of D_2. The Lagrange multiplier matrix is changed by deleting all the elements in the first row and column. The radius $\rho^{(k)}$ is reinitialized and finally the iteration based on (5.3.26) is used to solve this problem. After a few repetitions r^* will be identified.

In Table 5.5.2 three methods are compared: projection method (PM), l_1SQP algorithm and projection–l_1SQP algorithm (Pl_1SQP). The stopping criterion is $\|\mathbf{x}^{(k+1)}\| - \|\mathbf{x}^{(k)}\| < 10^{-8} = \epsilon$. It is estimated that x_i are accurate to $4 - 5$ decimal places and $\|\mathbf{x}\|$ is accurate to $6 - 7$ decimal places. In Table 5.5.2 the columns headed by NI give the number of iterations used by the projection method and the columns headed by NQP gives the number of times that the major l_1SQP problem (5.3.26) is solved. $r^{(0)}$ in the column headed by l_1SQP gives the initial rank for F using equation (5.4.3) and $r^{(0)}$ in the column headed by Pl_1SQP gives the initial rank for F using Algorithm 5.4.1. The three

Columns which determine F	r^*	NI	$x_i^* \quad i = 1, 2, .., n$				$\sqrt{\sum (x_i^{2*})}$
1,2,5,6	3	63	182.7042	146.9628	69.6629	45.8211	248.8602
1,3,4,5	2	115	235.0096	88.4015	189.1918	67.6986	321.5913
1,2,3,6,8,10	5	141	367.4156	273.0114	279.8192	50.4784	616.2334
			228.0582	193.2790			
1,2,4,5,6,8	4	881	317.4348	146.2721	244.8117	65.6893	491.7348
			4.1061	235.3253			
1–6	5	336	222.2243	282.8910	262.8245	238.0719	510.3758
			71.5195	14.2313			
1–8	6	387	369.8391	290.2214	255.5179	176.0771	640.5922
			56.6419	48.0679	223.0925	194.3380	
1–10	8	954	401.7844	299.7303	249.6374	194.1057	736.9839
			35.6192	50.3791	240.8572	214.9912	
			232.9831	171.9279			
1–12	10	1360	386.8981	286.8628	264.6721	195.7548	800.0756
			67.2526	39.7566	232.4680	227.8524	
			266.8375	187.5834	131.9821	252.7745	
1–14	12	854	404.4696	294.5210	265.8667	213.4180	882.7606
			73.4999	35.6596	254.5520	235.9188	
			250.0652	191.7257	161.8923	250.0233	
			267.8237	160.7042			
1–16	14	3663	407.5394(*)	290.8398	275.5972	215.0889	945.4555
			81.3601	33.5239	248.6281	244.9842	
			261.4713	197.1172	168.2075	258.6026	
			259.0489	159.3373	99.1123	294.4601	
1–18	15	30326	407.5394(*)	296.5150	265.6089	216.2863	1108.5326
			98.2078	44.7847	260.8753	246.8023	
			248.7318	185.1102	176.9004	270.7481	
			258.8518	160.6789	101.7151	308.4449	
			435.4937	358.0457			
1–20	18	11037	407.5394(*)	312.4666	258.1156	227.1807	1253.6603
			120.1546	49.2651	292.7023	272.3617	
			244.4578	201.3850	175.7458	279.3872	
			250.5748	158.5493	100.0581	310.8974	
			457.7386	356.8083	406.2569	327.4915	

Table 5.5.1: Results for problem (5.1.2) from projection Algorithm 5.2.2.

methods converge to approximately the same values.

In l_1SQP one of the variables in almost every test example is adjusted by a small unit (< 2.0) so that the matrix $\bar{A} + \mathit{diag}\, \mathbf{x}^*$ is exactly singular and positive semi–definite for all methods. The initial value of $\rho^{(0)}$ is 20.0. In l_1SQP most cases require a few iterations for solving (5.3.10) as r increases. For each value of r second order convergence of the iteration based on (5.3.26) and (5.3.28) is obtained.

The projection method is a very slowly convergent method especially when the bounds are active. Therefore it will be used only for estimating the rank r. In the Pl_1SQP algorithm the initial value of $\rho^{(0)}$ is 5.0.

Finally the projection method is not very successful in estimating the rank r^* especially when $n \geq 12$ and a more effective method is required to give a better estimate for r^* similar to those methods in Chapter 4.

Columns which determine A	r^*	PM NI	l_1SQP $r^{(0)}$	l_1SQP NQP	Pl_1SQP NI	Pl_1SQP $r^{(0)}$	Pl_1SQP NQP
1,2,5,6	3	63	2	10	5	3	4
1,3,4,5	2	115	2	16	6	2	5
1,2,3,6,8,10	5	141	3	11	10	4	9
1,2,4,5,6,8	4	881	3	20	8	4	7
1–6	5	336	3	22	12	5	9
1–8	6	387	5	18	13	5	11
1–10	8	954	6	19	7	8	7
1–12	10	1360	8	27	16	8	24
1–14	12	854	10	30	20	10	14
1–16	14	3663	11	35	27	10	33
1–18	15	30326	13	33	38	12	13
1–20	18	11037	15	45	55	15	27

Table 5.5.2: Numerical comparisons of methods of this chapter.

Chapter 6

Algorithms for solving the educational testing problem

6.1 Introduction

The problem to be considered here is the educational testing problem. Such optimization problems come up in many practical situations, particularly in statistics where we have a matrix F which is usually a covariance matrix with varying elements. The educational testing problem is; given a symmetric positive definite matrix F how much can be subtracted from the diagonal of F and still retain a positive semi–definite matrix this can be expressed as

$$\begin{aligned} &\mathit{maximize} \quad \mathbf{e}^T\boldsymbol{\theta} \qquad \boldsymbol{\theta} \in \Re^n \\ &\mathit{subject\ to} \;\; F - \mathit{diag}\,\boldsymbol{\theta} \geq 0 \\ &\qquad\qquad \theta_i \geq 0 \qquad i = 1, ..., n \end{aligned} \tag{6.1.1}$$

where $\mathbf{e} = (1, 1, ..., 1)^T$. An equivalent form to problem (6.1.1) is

$$\begin{aligned} &\mathit{minimize} \quad \mathbf{e}^T\mathbf{x} \qquad \mathbf{x} \in \Re^n \\ &\mathit{subject\ to} \;\; \bar{F} + \mathit{diag}\,\mathbf{x} \geq 0 \\ &\qquad\qquad x_i \leq v_i \qquad i = 1, ..., n \end{aligned} \tag{6.1.2}$$

where $\bar{F} = F - Diag\, F$, and $diag\, \mathbf{v} = Diag\, F$.

An early approach in solving the educational testing problem is due to Bentler [1972]. He writes $F - diag\, \boldsymbol{\theta} = CC^T$, where C is unknown and minimizes trace (CC^T) subject to certain conditions. He found that there are a large number of variables, and also it does not account for the bounds $\theta_i \geq 0 \;\; \forall\, i$. Furthermore, some difficulties in convergence to the optimum solution arise.

Woodhouse and Jackson [1977] have given a method for solving the problem by searching in the space of $\boldsymbol{\theta}$. However their method does not work efficiently and failed for particular examples.

Fletcher [1981b] has solved the problem in which the semi–definite constraint is reduced to an eigenvalue constraint and standard nonlinear programming techniques are used. But still some difficulties arise with the rates of convergence. Also the presumption that the eigenvalue constraint would be smooth at the solution, except in rare cases, is not correct and in fact the majority of such problems are nonsmooth at the solution.

In 1985 Fletcher developed a different algorithm for solving the educational testing problem. He gives various iterative methods for solving the nonlinear programming problem derived from the educational testing problem (6.1.2) using sequential quadratic programming techniques. One of these algorithms is the use of an l_1 exact penalty function. This algorithm works well with second order convergence and the function converging to the optimal solution. The only problem in these algorithms is the requirement to know the exact rank for the matrix $A^* = \bar{F} + diag\, \mathbf{x}^*$ where $\mathbf{x}^*$ solves (6.1.2).

Finally, Glunt [1991] describes a projection method for solving the educational testing problem. His idea is to construct a hyperplane L_τ in $\Re^n$ and then carry out the method of alternating projections (the von Neumann Algorithm 2.2.2) between the convex set $K = K_\Re \cap K_{off} \cap K_b$ and the hyperplane L_τ. His method converges globally and the order of convergence is very slow.

The statistical background involved in the educational testing problem is described in Section 6.2. In Section 6.3 the educational testing problem is solved using the theory developed in Section 2.3. Section 6.4 contains a brief description of the l_1 SQP method for solving problem (6.1.2). Finally in Section 6.5 numerical comparisons of these methods are carried out.

In Chapter 7 hybrid methods are considered. The projection method converges linearly or slower while the l_1 SQP method converges at second order but it requires the correct rank

r^* which can be gained from the projection method. Therefore, hybrid methods that take the advantage of both projection and l_1 SQP methods are described in Chapter 7.

6.2 The educational testing problem

This section explains the educational testing problem which arises from statistics. The problem is to find lower bounds for the reliability of the total score on a test (or subtests) whose items are not parallel using data from a single test administration. The educational testing problem consists of a number of student (N) taking a test or examination consisting of (n) subtests. The problem is to find how reliable is the students's total score in the sense of being able to

reproduce the same total on two independent occasions. Specifically it is required to know what evidence about reliability can be obtained by carrying out a test on one occasion only.

In this thesis we do not develop the entire theory (see Fletcher [1981b]) but just give enough information to construct the test problem (6.1.1). The data for the problem is an $N \times n$ table of scores $[X_{ij}]$ (see Table 6.2.1) such that X_{ij} gives the observed score of student i on subject j.

Define the mean observed score of subject j by

$$\bar{X}_j = \frac{1}{N} \sum_i X_{ij}. \tag{6.2.1}$$

Then the $n \times n$ matrix F given in (6.1.1) is constructed from an $N \times n$ data matrix $[X_{ij}]$ in the following way

$$f_{jk} = \frac{1}{N - 1} \sum_i (X_{ij} - \bar{X}_j)(X_{ik} - \bar{X}_k) \tag{6.2.2}$$

see Guttman [1945]. Then problem (6.1.1) is constructed with $\boldsymbol{\theta}$ as the unknown vector. For more about the statistical background for the educational testing problem and references see Fletcher [1981b].

15	25	20	28	35	50	21	18	22	28	28	12	15	40	18	23	14	16	15	10
21	27	32	32	41	42	30	35	33	32	64	16	24	38	34	13	15	17	28	18
23	35	40	22	55	48	36	40	46	18	38	18	26	37	24	24	17	20	19	26
23	29	50	36	42	52	44	32	24	19	32	24	20	46	32	23	11	12	40	28
34	37	42	19	36	46	17	26	35	28	39	54	21	47	29	42	18	18	30	20
36	60	70	45	55	54	32	30	32	29	41	28	20	47	36	28	20	20	18	24
36	35	46	27	50	40	60	34	39	46	48	63	20	48	40	19	21	24	40	22
38	70	44	50	45	42	20	28	29	16	55	40	22	49	42	25	23	26	30	28
39	46	52	24	37	60	53	30	46	43	54	54	23	46	44	22	35	22	48	30
40	74	65	60	72	41	33	36	24	52	64	36	28	50	46	26	25	23	30	30
40	48	32	23	58	52	23	40	37	24	58	38	29	54	44	28	11	27	41	34
41	12	24	50	47	48	41	42	37	28	56	57	32	51	43	25	17	24	32	39
46	52	76	48	70	58	20	50	28	42	76	58	28	58	45	34	27	35	18	56
46	73	84	63	38	57	33	56	42	18	72	77	31	52	48	32	35	32	32	19
47	42	74	28	60	57	36	42	48	52	63	46	36	53	49	53	29	30	42	40
47	82	72	70	39	64	21	25	44	26	44	44	37	51	46	33	38	35	37	42
47	40	42	50	48	61	40	40	26	29	61	44	30	56	47	52	46	37	48	40
48	70	65	48	42	57	35	58	50	46	60	32	34	58	54	35	36	31	16	18
49	65	60	55	62	56	52	50	52	28	50	48	34	58	53	41	45	40	38	52
50	30	35	28	62	54	41	46	50	21	65	33	32	58	54	38	50	44	43	50
52	42	54	33	42	64	40	40	56	44	64	38	34	60	44	34	38	30	44	38
52	72	70	65	72	68	62	38	56	44	58	46	36	58	46	36	55	20	48	47
52	44	64	72	44	62	35	44	56	46	62	39	30	61	46	38	40	42	24	80
53	25	42	28	68	52	41	45	44	26	28	43	51	62	47	35	42	48	50	40
54	48	60	58	36	51	63	41	64	29	63	49	32	58	47	39	43	58	48	49
55	64	62	30	42	57	34	47	52	34	57	37	43	63	48	38	47	20	54	65
58	30	24	62	51	51	44	36	43	25	36	54	41	65	48	43	40	35	50	42
58	16	40	45	42	58	44	42	58	36	58	52	40	64	49	36	18	45	53	28
58	44	56	51	68	68	46	48	72	38	62	34	32	68	49	42	47	47	28	70
59	58	58	50	74	52	36	58	60	28	44	56	34	72	51	33	48	58	54	58
60	32	35	48	40	56	52	32	40	37	72	57	36	61	52	51	42	46	17	51
60	78	80	62	52	54	58	47	80	32	64	39	45	66	53	42	70	40	50	18
60	38	55	66	42	52	30	54	62	42	90	38	38	63	56	46	62	48	55	44
61	48	64	68	70	53	42	40	38	45	73	56	50	64	54	46	45	42	50	22
62	86	94	50	49	62	48	56	74	33	84	36	52	67	52	47	41	70	57	53
63	35	38	55	38	58	46	59	63	48	62	58	38	68	53	47	48	75	44	25
64	79	65	76	68	57	32	33	52	46	72	62	52	54	54	48	55	35	60	54
64	50	52	35	60	56	52	64	76	36	63	44	48	56	52	49	63	38	48	46

Table 6.2.1:(to be continued in the next page)

65	37	42	70	50	58	58	62	66	34	53	64	62	72	53	50	74	45	62	57
65	82	74	63	36	62	60	58	60	38	57	63	58	70	51	51	55	55	64	58
67	44	46	54	52	58	55	68	80	40	28	65	50	71	54	52	65	70	74	68
67	48	56	80	44	64	62	35	70	62	58	72	56	72	60	41	78	38	75	68
68	62	78	56	50	53	62	54	80	64	62	48	54	74	62	76	58	45	80	36
69	39	30	42	38	62	40	32	68	56	68	71	58	73	56	43	79	47	73	70
70	52	20	76	69	61	64	56	69	54	64	66	58	78	52	72	32	47	71	71
72	54	72	38	54	51	66	65	76	72	54	49	60	74	57	42	68	62	22	46
72	42	48	70	70	57	42	40	68	53	62	74	60	78	58	52	55	48	72	75
74	64	66	70	42	60	40	78	53	48	69	67	76	77	59	68	58	55	16	60
78	68	62	63	35	56	63	80	74	43	71	78	62	76	59	68	70	75	72	75
79	37	42	28	64	52	40	38	72	72	56	52	58	82	60	61	75	66	58	58
80	62	30	65	59	51	68	57	74	48	78	71	42	54	61	62	78	69	78	58
82	85	80	52	44	57	70	69	64	71	85	76	64	56	63	67	44	70	60	26
82	40	74	52	52	64	74	64	76	46	64	46	51	80	63	68	65	55	70	67
84	42	76	70	55	61	90	80	56	41	58	82	72	72	67	73	85	60	76	78
84	42	54	60	42	58	42	60	52	70	77	68	68	74	59	71	79	65	44	76
86	85	88	80	37	63	80	72	79	42	73	42	68	82	65	73	85	62	80	82
87	53	51	62	68	56	60	42	78	42	62	74	62	84	65	75	63	75	68	83
89	41	60	40	60	54	88	88	83	57	84	64	64	80	62	65	90	78	88	52
90	73	78	77	52	42	56	50	58	59	72	84	70	84	62	63	82	85	78	87
90	81	74	64	48	38	86	52	80	63	66	68	60	62	63	74	75	81	84	94
96	85	88	90	72	44	58	62	70	74	64	74	72	64	64	84	85	78	88	54
97	56	55	35	68	70	78	76	56	72	83	69	65	86	65	76	82	89	92	90
99	75	65	88	54	42	80	90	88	58	78	88	70	88	63	82	72	71	98	80
100	65	75	70	70	60	83	85	70	62	72	90	72	84	64	78	88	80	80	72

Table 6.2.1: The Woodhouse [1976] data which corresponds to 64 students and 20 subtests.

6.3 A projection algorithm for solving the educational testing problem

In this section a projection algorithm for solving the educational testing problem is described. The method described here depends on Algorithm 2.3.1 developed in Section 2.3.

The constraints in problem (6.1.2) can be expressed as

$$\bar{F} + diag\,\mathbf{x} \in K_{\Re} \cap K_{off} \cap K_b.$$

Then problem (6.1.2) can be expressed as

$$\begin{aligned} &\underset{\mathbf{x}}{\text{minimize}} \quad \mathbf{e}^T\mathbf{x} \qquad \mathbf{x} \in \Re^n \\ &\textit{subject to } \bar{F} + diag\,\mathbf{x} \in K_{\Re} \cap K_{off} \cap K_b \end{aligned} \tag{6.3.1}$$

where $K_{\Re}$, K_{off} and K_b are given in (1.3.1), (1.3.5) and (1.3.6) respectively. Therefore problem (6.3.1) is a special case of problem (2.1.4) which can be solved by Algorithm 2.3.1. To solve (6.3.1) in this way we need the hyperplane L_τ given by (2.3.1) and we define $K = \bigcap_{i=1}^{m} K_i$ by $K = K_{\Re} \cap K_{off} \cap K_b$. However L_τ must be defined on the space of $n \times n$ matrices, and this can be done by

$$\begin{aligned} L_\tau &= \{Y = \bar{Y} + diag\,\mathbf{y} \in \Re^{n\times n} |\ \mathbf{e}^T\mathbf{y} = \tau\} \\ &= \{Y \in \Re^{n\times n} |\ tr(Y) = \tau\} \end{aligned} \tag{6.3.2}$$

where $Diag\,Y = diag\,\mathbf{y}$ and τ is chosen such that

$$\tau < \min_{\mathbf{x}\in K} \mathbf{e}^T\mathbf{x} \tag{6.3.3}$$

We also need the projection $P_{L_\tau}(Y)$, and following (2.3.8) with $\mathbf{e}$ replaced by I we can write

$$P_{L_\tau}(Y) = Y + \frac{\tau - tr(Y)}{n} I. \tag{6.3.4}$$

In Algorithm 2.3.1 the projection $P_K(.)$ in (2.3.4) is given. In particular for problem (6.1.2) we need the projection $P_K(A)$ where $K = K_{\Re} \cap K_{off} \cap K_b$ for any matrix A. In fact this projection was solved by Algorithm 5.2.2 and hence we just include Algorithm 5.2.2 as an inner iteration inside the following algorithm which is a special case of Algorithm 2.3.1. This algorithm solves the educational testing problem.

Algorithm 6.3.1

Given any positive definite matrix F, let $F^{(0)} = F$

```
For  $k = 1, 2, ...$
    $F^{(k+1)} = P_{L_\tau}(F^{(k)})$
    For  $l = 1, 2, ...$                                   (6.3.5)
```

```
            $A^{(0)} \;=\; F^{(k+1)}$
            $A^{(l+1)} \;=\; A^{(0)} \;+ P_b P_{off} P_{\Re}(A^{(0)}) \;-\; P_{\Re}(A^{(0)})$
        $End$                                                              (6.3.6)
        $F^{(k+1)} \;=\; F^{(0)} \;-\; diag\ F^{(0)} + diag\ P_{\Re}(A^*)$
    $End$
```

where A^* is the solution for the inner iteration and $P_{\Re}$, P_{off} and P_b are given in (5.2.3), (5.2.5) and (5.2.6) respectively.

From Theorem 2.3.2 $P_{\Re}(A^{(k)})$ and $P_b P_{off} P_{\Re}(A^{(k)})$ converges to the solution of problem (6.3.1). Also in Theorem 2.3.2 $\mathbf{x}_1^{(k)} \equiv F^{(k)}$ and $\mathbf{x}_2^{(k)} \equiv A^{(k)}$. Equations (6.3.5)–(6.3.6) are the inner loop and they are the same as Algorithm 5.2.2. In Section 6.5 numerical results for Algorithm 6.3.1 are given.

6.4 The l_1SQP method

This section contains a brief description of l_1SQP method for solving the educational testing problem. The l_1SQP methods in Section 1.7 are used. This method was given by Fletcher [1985].

The constraints in problem (6.1.2) can be expressed as

$$\bar{F} \;+\; diag\,\mathbf{x} \;\in\; K_{\Re} \;\cap\; K_{off} \;\cap\; K_b.$$

Then problem (6.1.2) can be expressed as

$$\begin{aligned} &\underset{\mathbf{x}}{\text{minimize}} \quad \mathbf{e}^T\mathbf{x} \qquad \mathbf{x} \in \Re^n \\ &subject\ to\ \bar{A} + diag\,\mathbf{x} \in K_{\Re} \cap K_{off},\ \ \mathbf{x} \le \mathbf{v} \end{aligned} \tag{6.4.1}$$

where $diag\,\mathbf{v} = Diag\ A^{(0)}$. This problem is similar to problem (5.3.1) in Section 5.3, therefore the method is given in detail in that section. Thus for solving problem (6.4.1) one follows the details of Section 5.3, changing only the definition of the objective function. However in this section we give a summary of what has been given in Section 5.3. The first order necessary conditions for $\mathbf{x}^*$ to solve (6.4.1) are similar to what given in (5.3.4) with the condition (5.3.4a) replaced by

$$\mathbf{e} \; + \; \mathbf{b}^* \; + \; \boldsymbol{\pi}^* \; = \; 0.$$

It is difficult to deal with the matrix cone constraints in (6.4.1), since it is not easy to specify if the elements are feasible or not. An equivalent problem to (6.4.1) with the constraint $D_2 \; = \; \mathbf{0}$ is considered. This problem is similar to problem (5.3.5) with the objective function $\mathbf{x}^T\mathbf{x}$ replaced by $\mathbf{e}^T\mathbf{x}$. This formulation will enable us to derive algorithms with a second order rate of convergence.

Now using the constraint $D_2 \; = \; \mathbf{0}$ in the form (5.3.9), this will produce an equivalent problem to (6.4.1). The number of variables in this new problem can be reduced to r variables which gives the new reduced problem

$$\begin{array}{ll} \underset{\mathbf{x}}{\text{minimize}} & f(\mathbf{x}) \; = \; \sum_{k=1}^{r} x_k \; + \sum_{i=r+1}^{n} x_i(\mathbf{x}) \\ \textit{subject to} & d_{ij}(\mathbf{x}) \; = \; 0, \quad i \; \neq \; j, \quad \mathbf{x} \; \leq \; \mathbf{v}. \quad i,j \; = \; r+1,\ldots,n \end{array} \tag{6.4.2}$$

The expressions for the derivatives $\frac{\partial d_{ij}}{\partial x_s}$ and $\frac{\partial^2 d_{ij}}{\partial x_s \partial x_t}$ given in (5.3.13) and (5.3.14) respectively enable us to finding expressions for $\nabla \, f$, $\nabla^2 \, f$ and $W^{(k)}$. Then using these expressions the QP subproblem

$$\begin{array}{ll} \underset{\boldsymbol{\delta}}{\text{minimize}} & f^{(k)} \; + \; \nabla f^{(k)}\boldsymbol{\delta} \; + \; \tfrac{1}{2} \; \boldsymbol{\delta}^T W^{(k)} \boldsymbol{\delta} \qquad \boldsymbol{\delta} \; \in \; \Re^r \\ \textit{subject to} & d_{ij}^{(k)} \; + \; \nabla \; d_{ij}^{(k)T} \boldsymbol{\delta} \; = \; 0 \qquad i \; \neq \; j \quad i,j = r+1,\ldots,n \\ & \mathbf{x}^{(k)} \; + \; \boldsymbol{\delta} \; \leq \; \mathbf{v} \end{array} \tag{6.4.3}$$

is defined. Thus the SQP method applied to (6.4.2) requires the solution of the QP subproblem (6.4.3). The matrix $W^{(k)}$ is positive semi–definite see Fletcher [1985].

It is shown in Section 1.7 that the SQP method may not converge globally and it is usually modified by the exact penalty function (5.3.25). Now a similar technique to what stated in Section 5.3 is followed to take over the problem of non–globality convergent using the l_1 exact penalty function (5.3.25).

This section is concluded by some restrictions and conditions in similar manner to those considered at the end of Section 5.3.

For more about the l_1 SQP methods for solving the educational testing problems see Fletcher [1985].

6.5 Numerical results and comparisons

In this section numerical problems are obtained from the data given in Table 6.2.1, by Woodhouse [1976]. The Woodhouse data set is a 64×20 data which corresponds to 64 students and 20 subtests. Various selections from the set of subsets of columns are used to give various test problems to form the matrix A. These subsets are those given in the first columns of Tables 6.5.1–2, the value of n is the number of elements in each subset. Equation (6.2.2) gives the formula for calculating the educational testing problems from Table 6.2.1.

In Algorithm 6.3.1 τ must satisfy the condition (6.3.3). Since $\mathbf{x}^*$ not known in advance and with elements $f_{ij} \overset{\approx}{>} 100$ then it is clear that the diagonal elements $\bar{F} + diag\ \mathbf{x}^{(k)}$ is greater than about 100 so $\mathbf{e}^T\mathbf{x} \overset{\approx}{>} 100n$ since F is positive definite. Therefore from (6.3.3) the choice $\tau = 100$ is recommended. In fact we recommend this choice since the elements f_{ij} are close to each either in magnitude. However, in general the off-diagonal elements can play a role in making a better estimate for τ. If τ chosen randomly and does not satisfy the condition (6.3.3) then the matrix $F - diag\ \mathbf{x}^{(k)}$ is not positive semi–definite and the method is rerun with different τ. In Chapter 7 more information is available and a different strategy is followed.

Glunt [1991] and Fletcher [1985] tested their methods on the twelve test problems originally due to Woodhouse [1976]. The same test problems are applied for the methods in this chapter. This section contains numerical results for the projection method given in Table 6.5.1. Numerical results for the l_1 SQP algorithm are given in Table 6.5.2. In all the tables of this section NOI gives the number of outer iteration when solved by Algorithm 6.3.1, TNII gives the total number of inner iteration used by Algorithm 5.2.2 in Algorithm 6.3.1 and $r^{(0)}$ gives the number of positive eigenvalues in the first iteration of Algorithm 6.3.1.

In Table 6.5.1 a comparison between $\tau = -100$ and $\tau = 100$ is given for the same test problems using Algorithm 6.3.1. We choose $\tau = -100$ for comparison purposes which shows that when τ is remote from condition (6.3.3) then the method takes more inner iterations. It is clear that with $\tau = 100$ the method takes fewer inner iterations in most of the examples. Because of Algorithm 5.2.2 the projection method is very slow and the number of iterations taken by the projection method is very large especially when the bounds are active. The results obtained by the l_1SQP method of Section 6.4 are tabulated in Table 6.5.2 as given by Fletcher [1985] and mentioned here for comparison purposes. The iterates converge to essentially the same values of $\mathbf{x}^*$ in both methods.

The projection method is very expensive in the sense that it consumed a large number of iterations whilst the l_1SQP method takes a very small number of iterations.

The NAG routine is used here to find the eigenvalues and eigenvectors for the matrix $\bar{F} + diag\ \mathbf{x}^{(k)}$. This matrix is reduced to a real symmetric tridiagonal matrix by Householder's method. Then the eigenvalues and eigenvectors are calculated using the QL algorithm. The amount of work required by these algorithms is approximately $\frac{4}{3}n^3$ multiplications per one inner iteration (Golub and Van Loan [1989]).

Again the NAG routine is used this time for solving the QP subproblem (6.4.3) which is one iteration of the SQP method. The method used by the NAG routine to solve the QP subproblem requires the solution for the system

$$Z^{(k)}\ W\ Z^{(k)T}\mathbf{p}^{(k)} \ = \ -\, Z^{(k)T}(\mathbf{c} \ + \ W\ \mathbf{x}^{(k)}) \tag{6.5.1}$$

where $\mathbf{c} = \nabla f$ and $Z^{(k)}$ is a matrix whose columns form a basis for the null space of $A^{(k)}$ (the matrix of coefficients of the bounds and active constraints). $\mathbf{p}^{(k)}$ is a search direction. The matrix $Z^{(k)}$ is obtained from the TQ factorization of $A^{(k)}$, in which $A^{(k)}$ is represented as

$$A^{(k)} \begin{bmatrix} Z^{(k)} \\ Q \end{bmatrix} \ = \ [\,\mathbf{0} \quad T^{(k)}\,]. \tag{6.5.2}$$

The Lagrange multipliers $\boldsymbol{\lambda}^{(k)}$ are defined as the solution of the system

$$A^{(k)}\ \boldsymbol{\lambda}^{(k)} \ = \ \mathbf{c} \ + \ W\ \mathbf{x}^{(k)}. \tag{6.5.3}$$

Eqautions (6.5.1) and (6.5.2) costs approximately $\frac{7}{3}n^3$ multiplications to solve and (6.5.3) costs approximately $\frac{8}{3}n^3$ multiplications to solve (Golub and Van Loan [1989]). Thus one iteration of the SQP method costs approximately $\frac{15}{3}n^3$ multiplications.

Thus one iteration of the SQP method costs about 6 times greater as one iteration of the projection method. Nonetheless the SQP method is much better than the projection method since the number of iterations taken by the projection method is about 60 times greater than the number of iterations taken by the SQP method. However in Chapter 7 hybrid methods are carried out which use even fewer iterations.

Columns which determine A	$\tau = -100$		$\tau = 100$	
	NOI	TNII	NOI	TNII
1,2,5,6	3	197	4	240
1,3,4,5	2	224	3	266
1,2,3,6,8,10	3	580	3	522
1,2,4,5,6,8	4	4994	4	4518
1–6	3	1351	3	1243
1–8	4	1948	4	1702
1–10	3	2918	3	2534
1–12	3	2403	3	2442
1–14	3	3196	3	3143
1–16	3	5215	3	4796
1–18	3	14043	3	14171
1–20	3	8255	3	7978

Table 6.5.1: Results for the educational testing problem from the projection Algorithm 6.3.1

Columns which determine A	$r^{(0)}$	r^*	NQP	$\sum \theta_i^*$
1,2,5,6	2	3	14	542.77356
1,3,4,5	2	2	12	633.15784
1,2,3,6,8,10	3	5	9	305.48170
1,2,4,5,6,8	3	4	13	564.46331
1–6	3	4	14	535.36227
1–8	5	6	29	641.83848
1–10	6	8	34	690.78040
1–12	8	9	29	747.48921
1–14	10	12	36	671.27506
1–16	11	14	42	663.46204
1–18	13	15	27	747.50574
1–20	15	18	39	820.34265

Table 6.5.2: Results for the educational testing problem from the l_1SQP method of Section 6.4.

Table 6.5.3 investigates the effect of varying τ. It shows the outcome from Algorithm 6.3.1 for the following example

$$\bar{F} = \begin{bmatrix} 0 & 1 & 2 & -2 \\ 1 & 0 & 3 & 2 \\ 2 & 3 & 0 & 1 \\ -2 & 2 & 1 & 0 \end{bmatrix} \qquad \mathbf{v} = \begin{bmatrix} 2 \\ 4 \\ 8 \\ 10 \end{bmatrix}$$

with different τ. From Table 6.5.3 it is clear that small τ increases the total number of iterations performed by Algorithm 5.2.2, whilst on the other hand bigger τ decreases the total number of inner iterations and increases the number of outer iterations which are very cheap to calculate using the projection (6.3.4) which costs approximately n multiplications while one inner iteration costs approximately $\frac{4}{3}n^3$ multiplications. Hence it is recommended to increase τ to be close to the boundary of the condition (6.3.3) which is compatible with the choice in Table 6.5.1.

τ	NOI	TNII	$\sum x_i^*$	$r^{(0)}$	r^*
-30.0	2	2679	15	0	2
-20.0	2	2215	15	1	2
-10.0	2	1734	15	2	2
-5.0	2	1571	15	2	2
0.0	2	1291	15	2	2
5.0	3	1308	15	2	2
10.0	3	960	15	2	2
14.0	6	787	15	2	2
14.9	15	891	15	2	2
15.0	30	792	15.0051	2	2

Table 6.5.3: Numerical comparisons for same example with different τ.

Chapter 7

Hybrid methods for solving the educational testing problem

7.1 Introduction

In this chapter new methods for solving the educational testing problem are introduced. The methods described here depend upon both projection and l_1 SQP methods using a hybrid method. The hybrid method works in two stages. First stage is the projection method which converges globally so is potentially reliable but often converges at slow order. Meanwhile in the second stage there is l_1 SQP methods, in particular the method described in Section 6.4, which converges at second order if the correct rank r^* is given. The main disadvantage of the l_1 SQP methods are that they require the correct r^*. A hybrid method is one which switches between these methods and aims to combine their best features. To apply an l_1 SQP method requires a knowledge of the rank r^* and this knowledge can also be gained from the progress of the projection method. Hybrid methods can work well but there is one disadvantage. If the positive definite matrix have the same rank as the optimal positive semi–definite matrix in which the l_1 SQP method works well, then most of the time will be taken up in the first stage, using the projection method. If this converges slowly then the hybrid method will not solve the problem effectively. Thus it is important to ensure that the second stage method is used to maximum effect. Hence in the algorithm of Section 7.3 the l_1 SQP method is applied first.

In Sections 7.2 and 7.3 two new methods are described. Firstly, there is the projection–l_1SQP method, which starts with the projection method to determine the rank $r^{(k)}$ and continues with the l_1 SQP method. Secondly, the l_1 SQP–projection method is described, which solves the problem by the l_1 SQP method and uses the projection method to update the rank. Numerical results and comparisons are given in Section 7.4.

As with the methods of Chapter 6 it is easy to move from one method to the other in either direction. This in contrast to Chapter 4 where some special techniques were developed to enable this to be done.

7.2 Projection–l_1SQP method

The main disadvantage of the l_1 SQP method is finding the exact rank r^*, since it is not known in advance it is necessary to estimate it by an integer $r^{(k)}$. It is suggested that the best estimate of the matrix rank $r^{(k)}$ is obtained by carrying out some iterations of the projection method given in Section 6.3. This is because the projection method is a globally convergent method.

The method in this section follows a similar strategy as that in Section 4.3.

Consider Λ_r in (5.2.4), then at the solution the number of eigenvalues in Λ_r is equal to the rank r^*. Thus

$$No.\ \Lambda_r^* \ = \ r^* \tag{7.2.1}$$

where $No.\ \Lambda$ is the number of positive eigenvalues in Λ. A similar equation to (7.2.1) is used to calculate an estimated rank $r^{(k)}$ given by

$$No.\ \Lambda_r^{(k)} \ = \ r^{(k)}.$$

where Λ_r is given by (5.2.4). The range of error is relatively small. Then the l_1 SQP method will be applied to solve the problem as described in Section 6.4.

Another consideration is τ how to be chosen, if τ is close to the boundary of the condition (6.3.4) then the equation

$$No.\ \Lambda_r^{(k)} \ = \ r^*$$

may satisfied in the first few iterations. Experiments proved this fact see Table 6.5.1.

The projection–l_1 SQP algorithm can be described as follows.

Algorithm 7.2.1

Given any positive definite matrix $F = F^T \in \Re^{n\times n}$, let s be a positive integer. Then the following algorithm solves the educational testing problem

```
i. Let $F^{(0)} = F$
ii. Choose $\tau$ to be close to the boundary of the condition (6.3.3).
iii. Apply Algorithm 6.3.1 until
        $No.\ \Lambda_r^{(k)} = No.\ \Lambda_r^{(k+j)} \quad j = 1, 2, \ldots, s$        (7.2.2)
iv. $r^{(k)} = No.\ \Lambda_r^{(k)}$
v. Use the result vector $\mathbf{x}$ from Algorithm 6.3.1 as an initial vector for $l_1$ SQP method
vi. Apply $l_1$ SQP method to solve the problem with $r = r^{(k)}$.
        $If$
            $\|D_2(\mathbf{x})\| \le \epsilon$ $for\ some\ small\ \epsilon$
        $Then$
            $F^* = F^{(k)},\ r^* = r^{(k)}$ $and\ terminate$
        $Endif$
vii. Apply one inner iteration of the Algorithm 6.3.1.
viii. Go to (iv).
```

The integer s in Algorithm 7.2.1 can be any positive number. If it is small then the rank $r^{(k)}$ may not be accurately estimated, however the number of iterations taken by projection method is small. In the other hand if s is large then a more accurate rank is obtained but the projection method needs more iterations.

The advantage of using the projection method as the first stage of the projection–l_1 SQP method is that if $F^{(0)}$ is positive semi–definite (singular) then the projection method terminates at the first iteration. Moreover it gives the best estimate to $r^{(k)}$.

Another way of estimating the rank $r^{(k)}$ is suggested by Fletcher [1985] and it was given in the end of Section 5.4, equation (5.4.3).

7.3 l_1SQP–Projection method

Starting with projection method has the advantage of knowing if the given matrix is a positive semi–definite (singular) or not, and it gives the best estimate for the matrix rank $r^{(k)}$. However sometimes it takes many iterations before equation (7.2.2) is satisfied, especially if τ is chosen to be small, this means slow convergence since the projection method is slow converges method. In this method an algorithm starts with the l_1SQP method with an estimated rank $r^{(k)}$ is considered. Then one iteration of the projection method will be calculated after every stage of the l_1SQP–projection algorithm the resulting vector $\mathbf{x}^{(k)}$ will be used as an initial vector to the next stage, thus the vector $\mathbf{x}^{(k)}$ is updated at every stage from the previous one.

The method in this section follows a similar strategy as that in Section 4.4.

Now the l_1SQP–projection algorithm can be described as follows.

Algorithm 7.3.1

Given any positive definite matrix $F = F^T \in \Re^{n\times n}$ the following algorithm solves the educational testing problem

i. Let $F^{(0)} = F$

ii. Choose $r^{(k)}$ (small as possible based on one of Section 7.2 strategies).

iii. Apply l_1 SQP method if $\|D_2(\mathbf{x})\| \leq \epsilon$ for some small ϵ, terminates.

iv. Use the result $\mathbf{x}^{(k)}$ as an initial vector for projection method (Algorithm 6.3.1).

v. Choose τ to be close to the boundary of the condition (6.3.3), $(\tau = \sum x_i^{(k)})$.

vi. Apply one iteration of the projection method.

vii. $r^{(k)} = No.\ \Lambda_r^{(k)}$.

viii. Use the result $\mathbf{x}^{(k)}$ as an initial vector for l_1 SQP method.

ix. Go to (iii).

Another advantage of this algorithm is that if the rank is not correct then instead of adding one to $r^{(k)}$ it goes back to the projection method to provide a better estimate to $r^{(k)}$. This will increase or decrease $r^{(k)}$ nearer to r^*, therefore variables will be added to or subtracted from the problem. The new variables are estimated using the projection method.

Another advantage is that at every stage only one iteration of projection method is used giving a faster converging algorithm.

Example 7.3.2

An example of this algorithm for $n = 5$

$$F = F^{(0)} = \begin{bmatrix} 0 & 5 & 4 & 3 & 1 \\ 5 & 0 & 6 & 3 & 3 \\ 4 & 6 & 0 & 6 & 4 \\ 3 & 3 & 6 & 0 & 5 \\ 1 & 3 & 4 & 5 & 0 \end{bmatrix} \qquad \mathbf{v} = \begin{bmatrix} 10 \\ 10 \\ 10 \\ 10 \\ 10 \end{bmatrix}.$$

If we use projection method to estimate $r^{(0)}$ as in the previous section with $\tau = -100$ then $r^{(0)} = 1$, apply the l_1SQP algorithm then it terminates with

$$\mathbf{x} = [10/3 \quad 7.5 \quad 4.8 \quad 2.7 \quad 0.3], \qquad \sum_{i=1}^{5} x_i = 18.6333$$

$\sum_{i=1}^{5} x_i = 18.6333$ and $D(\mathbf{x}) \ncong 0$. Applying one iteration from the projection method with $\tau = 18$ we find that $r^{(k)} = 3$ and

$$\mathbf{x} = [3.4671 \quad 7.5652 \quad 6.1089 \quad 4.5908 \quad 2.5492].$$

Apply this as an initial vector for l_1 SQP algorithm, after 15 iterations the l_1SQP algorithm terminates with $D(\mathbf{x}) \cong 0$,

$$\mathbf{x} = [13/3 \quad 9.0 \quad 6.0 \quad 9.0 \quad 13/3]$$

and $\sum \mathbf{x} = 32.6667$. If we use the initial $\tau = 18$ instead of $\tau = -100$ in the first stage of the projection method then $r^{(0)} = 3$ and the l_1SQP algorithm will terminate directly with the same result.

7.4 Numerical results and comparisons

In this section numerical problems are obtained from the data given in Table 6.2.1, by Woodhouse [1976]. The Woodhouse data set is a 64×20 data which corresponds to 64 students

Columns which determine F	τ	TNII	$r^{(0)}$	r^*	NQP	$\sum \theta_i^*$
1,2,5,6	400	4	3	3	11	542.77356
1,3,4,5	400	2	2	2	12	633.15784
1,2,3,6,8,10	600	11	4	5	8	305.48170
1,2,4,5,6,8	600	4	4	4	13	564.46331
1–6	600	6	4	4	10	535.36227
1–8	800	13	5	6	14	641.83848
1–10	1000	15	7	8	21	690.78040
1–12	1200	23	9	9	9	747.48921
1–14	1400	25	10	12	34	671.27506
1–16	1600	22	11	14	44	663.46204
1–18	1800	20	12	15	27	747.50574
1–20	2000	29	14	18	39	820.34265

Table 7.4.1: Results for the educational testing problem from the projection–l_1SQP method of Section 7.2.

and 20 subtests. Various selections from the set of subsets of columns are used to give various test problems to form the matrix F. These subsets are those given in the first columns of Tables 7.4.1–3, the value of n is the number of elements in each subset. Equation (6.2.2) gives the formula for calculating the educational testing problems from Table 6.2.1.

The result obtained by the new method of Section 7.2 are tabulated in Table 7.4.1. In Table 7.4.1 the columns headed by NQP give the number of times that the major l_1SQP is solved.

In the projection–l_1SQP method τ needs to be estimated very close to $\sum x_i^*$, this will give us a very good estimate of the rank. Since the average size of the educational testing problem elements are more than 100, $\tau = n \times 100$ is chosen as an initial value (see Section 6.5). In Table 7.4.1 it is clear that when $n > 10$ then τ becomes very small comparing with $\sum x_i^*$ which makes the projection method estimate $r^{(k)}$ very small comparing with the correct r^*.

The result obtained by the new method of Section 7.3 are tabulated in Table 7.4.2. In the l_1SQP–projection method $r^{(k)}$ updated using one iteration of the projection method. In the projection method τ estimated using the result from the l_1SQP method. In the 1–10 case the projection method estimated $r^{(k)} = 10$ instead of $r^{(k)} = 9$.

In both Tables 7.4.1 and 7.4.2 it can be seen that the results we have are exactly the same as Fletcher [1985]. Also one or two of the variables are adjusted so that the matrix $F - diag\ \boldsymbol{\theta}$ is exactly singular and positive semi–definite.

Finally in Table 7.4.3 the four methods are compared.

Columns which determine F	$r^{(0)}$	NQP	PM$r^{(k)}$	NQP	$\sum \theta_i^*$
1,2,5,6	2	5	3	6	542.77356
1,3,4,5	2	12			633.15784
1,2,3,6,8,10	3	4	5	5	305.48170
1,2,4,5,6,8	3	6	4	4	564.46331
1–6	3	7	4	4	535.36227
1–8	5	7	6	6	641.83848
1–10	6	9	8	11	690.78040
1–12	8	3	10	9	747.48921
1–14	10	6	12	9	671.27506
1–16	11	9	14	10	663.46204
1–18	13	7	15	16	747.50574
1–20	15	5	18	21	820.34265

Table 7.4.2: Results for the educational testing problem from the l_1SQP–projection method of Section 7.3.

PM$r^{(k)}$:rank r updated from the projection method.

Columns which		PM	l_1SQP		Pl_1SQP			l_1SQPP	
determine F	r^*	TNII	$r^{(0)}$	NQP	TNII	$r^{(0)}$	NQP	$r^{(0)}$	TNQP
1,2,5,6	3	197	2	14	4	3	11	2	11
1,3,4,5	2	224	2	12	2	2	12	2	12
1,2,3,6,8,10	5	580	3	9	11	4	8	3	9
1,2,4,5,6,8	4	4994	3	13	4	4	13	3	10
1–6	4	1351	3	14	6	4	10	3	11
1–8	6	1948	5	29	13	5	14	5	13
1–10	8	2918	6	34	15	7	21	6	20
1–12	9	2403	8	29	23	9	9	8	12
1–14	12	3196	10	36	25	10	34	10	15
1–16	14	5215	11	42	22	11	44	11	19
1–18	15	14043	13	27	20	12	27	13	23
1–20	18	8255	15	39	29	14	39	15	26

Table 7.4.3: Comparing the four methods.

Pl_1SQP: the projection–l_1SQP method.
l_1SQPP: the l_1SQP–projection method.
TNQP : total number of NQP.

Chapter 8

Conclusions and further work

In this thesis we have studied certain problems involving positive semi–definite matrix constraint. We have found that our implementations of the new unconstrained methods for solving the Euclidean distance matrix problem have performed well in comparison with the projection method. However, the hybrid methods in Chapter 4 performed even better, with very fast convergence, especially the projection–unconstrained method (Section 4.3) which is much better than the projection method and Method 3.4.2 from which it is composed. In determining the correct rank the projection method worked well and found the rank in a few iterations. Also we have successfully found methods for switching from one method to another.

A number of suggestions for further research about the methods that solves the Euclidean distance matrix problem are the following.

- It is clear that if the diagonal matrix $\Delta^{(k)}$ in (3.3.11) satisfies $P_d P_M(F + \Delta^{(k)}) = P_M(F + \Delta^{(k)})$ then $P_M(F + \Delta^{(k)})$ is the required solution where F is a given matrix. Possibly from the structure of the given matrix F that one can find the required diagonal matrix in one go. It is not clear how to do this but it might be worth trying.

- The unconstrained methods have a large number of variables $(\sim (r-1)n$ depending on the method) which means that the method takes a large number of line searches to solve the problem. Therefore it is worth trying to restate the problem with only the diagonal matrix $\Delta^{(k)}$ as variables and then finding methods for solving it.

- Method 3.4.3 needs more investigation because the number of variables is less than the other unconstrained methods, whilst the number of line searches is larger. However number of possible reasons have been given in Section 3.6.

For the least distance problem in Chapter 5 two methods are developed, that is the projection method and the l_1 SQP method. The l_1 SQP method has performed well in comparison with the projection method which takes a huge number of iterations to solve the problem. Also the projection– l_1 SQP method (Section 4.3) has worked well in solving the problem. The few iterations taken by the projection method to determine the rank saves a large number of iterations taken by the l_1 SQP method. The integer s in Algorithm 5.4.1 chosen to be small ($\sim$ 2) in Table 5.5.2. This reduces the number of iterations taken by projection method although the rank is not accurately estimated and the lower bound given by Fletcher [1985] has worked better in the case 1–18.

Two suggestions for further research about the methods that solves the least distance problem are given in the following.

- By looking at problem (5.4.3), there is a different problems with every different initial vector $\mathbf{a}$. The projection Algorithm 5.2.2 solves this problem with the initial vector zero replaced by $\mathbf{a}$. Extending the l_1 SQP method to solve problems of this type is worth investigation.
- A modified projection algorithm similar to Algorithm 4.2.1 is needed for the least distance problem this enable us to use the result matrix from the l_1 SQP method as an initial matrix for the projection method. Then a more effective hybrid method could be obtained.

Two methods have given for solving the educational testing problem. One is the l_1 SQP method by Fletcher [1985] the other is the projection method by Glunt [1991]. The hybrid methods developed in Chapter 7 have good rate of convergence specially the l_1 SQP–projection method (Section 7.3) as compared with the methods of Chapter 6. The projection method is not very effective in determining the rank when $n \geq 12$. This is because a small value of s is shosen in Algorithms 7.2.1 and 7.3.1. In the other hand if s is increased then a large number of iterations are consumed by the projection method. Hence a suitable way of chosing the integer s is needs some investigation. Various examples are solved in Sections 6.5 and 7.4 with different τ. The best way to choose τ is given there.

References

Al–Baali, M. and Fletcher, R. [1985]. Variational methods for nonlinear least squares, *J. Oper. Res. Soc., 36, pp. 405–421.*

Bentler, P. M. [1972]. A lower–bound method for the dimension–free measurement of internal consistency, *Social Sci. Res., 1, pp. 343–357.*

Blumenthal, L. M. [1953]. *Theory and Applications of Distance Geometry*, Oxford Univ. Press, London.

Boyle, J. P. and Dykstra, R. L. [1986]. A method for finding projections onto the intersection of convex sets in Hilbert space, in *Advances in Order Restricted Statistical Inference,* (Eds. R. Dykstra, T. Robertson, and F. T. Wright), Lecture Notes in Statistics 37, Springer–Verlag, Berlin, pp. 28–47.

Browne, M. W. [1987]. The Young–Householder algorithm and the least squre multidimensional scaling of squared distance, *J. of Classification, 4, pp. 175–190.*

Broyden, C. G. [1970]. The convergence of a class of double rank minimization algorithms, parts I and II, *J. Inst. Maths. Applns., 6, pp. 76–90 and 222–231.*

Cheney, W. and Goldstein, A. [1959]. Proximity maps for convex sets, *Proc. Amer. Math. Soc., 10, pp. 448–450.*

Colledge, R. G. and Rushton, G. [1972]. *Multidimensional scaling: review and geographical applications. Geographic technical papers series, no. 10.* Association of American geographers. Washington.

Crippen, G. M. [1977]. A novel approach to calculation of conformation: distance gemotry. *J. Computational Physics 24, pp. 96–107.* *

Crippen, G. M. [1978]. Rapid calculation of coordinates from distance measures. *J. Computational Physics 26, pp. 449–452.*

De Leeuw, J. and Heiser, W. [1980]. Multidimensional scaling with restrictions on the configuration, in *Multivariate Analysis V,* (Ed. P. R. Krishnaiah), North Holland Pub. Co., pp. 502–522.

Deutsch, F. [1983]. Von Neumann's alternating method: the rate of convergence, in *Approximation Theory IV,* (Eds. C. Chui, L. Schumaker and J. Ward), Academic Press, New York–London, pp. 427–434.

Dykstra, R. L. [1983]. An algorithm for restricted least squares regression, *J. Amer. Stat. Assoc. 78, pp. 839–842.*

Fletcher, R. [1970]. A new approach to variable metric algorithms, *Computer J., 13, pp. 317–322.*

Fletcher, R. [1981a]. Numerical experiments with an exact l_1 penalty function method, in *Nonlinear Programming 4,* (Eds. O. L. Mangasarian, R. R. Meyer and S. M. Robinson), Academic Press, New York.

Fletcher, R. [1981b]. A nonlinear programming problem in statistics (educational testing), *SIAM J. Sci. Stat. Comput., 2, pp. 257–267.*

Fletcher, R. [1982]. Semi–definite matrix constraints in optimization, *Dept. Mathematical Sciences, Numerical Analysis Report NA/61, Univ. Dundee, Scotland.*

Fletcher, R. [1985]. Semi–definite matrix constraints in optimization, *SIAM J. Control and Optimization, 23, pp. 493–513.*

Fletcher, R. [1987]. *Practical methods of optimization*, John Wiley and Sons, Chichester.

Gaffke, N. and Mathar, R. [1989]. A cyclic projection algorithm via duality, *Metrika, 36, pp. 29–54.*

Gilbert, E. N. [1974]. Distortion in maps, *SIAM Review, 16, pp. 47–62.*

Glunt, W. [1991]. An alternating projections method for linear convex programming problems, Ph.D. Thesis, University of Kentucky.

Glunt,W. Hayden, T. L. Hong, S. and Wells, J. [1990]. An alternating projections method for computing the nearest Euclidian distance matrix, *SIAM J. Matrix and App. , 4, pp. 589–600.*

Goldfarb, D. [1970]. A family of variable metric methods derived by variaional means, *Maths. Comp., 24, pp. 23–26.*

Golub, G. H. and Van Loan, C. F. [1989]. *Matrix Computations*, Johns Hopkins Universty Press, Baltimore, MD.

Guttman, L. [1945]. A basis for analyzing test–retest reliability, *Psychometrika, 10, pp. 255–282.*

Hald, J. and Madsen, K. [1981]. Combined LP and quasi–Newton methods for minmax optimization, *Math. Programming, 20, pp. 49–62.*

Han, S. P. [1977]. A globally convergent method for nonlinear programming, *J. Optim. Theory Appl., 22, pp. 297–309.*

Han, S. P. [1988]. A successive projection method, *Math. Programming, 40, pp. 1–14.*

Havel, T. Kuntz, I. and Crippen, G. [1983]. The theory and practice of distance geometry, *Bull. Math. Biol., 45, pp. 665–720.*

Hayden, T. L. and Wells, J. [1988]. Approximation by matrics positive semi–definite on a subspace, *Linear Alg. and Appl., 109, pp. 115–130.*

Higham, N. [1988]. Computing a nearest symmetric positive semi–definite matrix, *Linear Alg. and Appl., 103, pp. 103–118.*

Kendall, D. G. [1971]. Construction of maps from "odd" bits of information, *Nature, 231, pp. 158–159.*

Lalouel, J. M. [1977]. Linkage mapping from pairwise recombination data, *Heredity, 38, pp. 61–77.*

Menger, K. [1931]. New foundations of Euclidean geometry, *Amer. J. Math., 53, pp. 721–745.*

Meulman, J. [1986]. *A distance approach to nonlinear multivariate analysis*, DSWO Press, Leiden.

Powell, M. J. D. [1970]. A hybrid method for nonlinear equations, in *Numerical Methods for Nonlinear Algebraic Equations,* (Ed. P. Rabinowitz), Gordon and Breach, London.

Rockafellar, R. T. [1970]. *Convex Analysis*, Princeton Univ. Press, Princeton, NJ.

Rockafellar, R. T. [1981]. *The Theory of Subgradients and Its Applications to Problems of Optimization. Convex and Nonconvex Functions*, Research and Education in Mathematics 1, Heldermann Verlag, Berlin.

Schoenberg, I. J. [1935]. Remarks to M. Frechet's article "Sur la definition axiomatique d'une classe d'espace distances vectoriellement applicable sur l'espace de Hilbert", *Ann. of Math., 36, pp. 724–732.*

Shanno, D. F. [1970]. Conditioning of quasi–Newton methods for function minimization, *Maths. Comp., 24, pp. 647–656.*

Takane, Y. [1977]. On the relations among four methods of multidimensional scaling, *Behaviormetrika 4, pp. 29–43.*

Von Neumann, J. [1950]. *Functional Operators II, The geometry of orthogonal spaces,* Annals of Math. Studies No. 22, Princeton Univ. Press.

Woodhouse, B. [1976]. Lower bounds for the reliability of a test, M.Sc. Thesis, Dept of Statistics, University of Wales, Aberystwyth.

Woodhouse, B. and Jackson, P. H. [1977]. Lower bounds for the reliability of the total score on a test composed of non–homogeneous items: II. A search procedure to locate the greatest lower bound, *Psychometrika 42, pp. 579–591.*

Young, F. W. [1984]. Scaling, *Ann. Rev. Psychol. 35, pp. 55–81.*

Young, G. and Householder A. S. [1938]. Discussion of a set of points in terms of their mutual distances, *Psychometrika 3, pp. 19–22.*