Hybrid Methods for Finding the Nearest Euclidean Distance Matrix

Suliman Al-Homidan
Department of Mathematics, King Saud University, Riyadh 11451, PO Box 4511, Saudi Arabia

Roger Fletcher
Department of Mathematics and Computer Science, University of Dundee, Dundee DD1 4HN, Scotland, UK.

Abstract

A concise characterization is presented for a Euclidean distance matrix in terms of null-space matrices, and methods for the solution of the Euclidean distance matrix problem are considered. One approach (Glunt et al. [8]) is to formulate the problem as a constrained least distance problem in which the constraint is the intersection of two convex sets. The Dykstra-Han projection algorithm can then be used to solve the problem. This method is globally convergent but the rate of convergence is slow. However the method does have the capability of determining the correct rank of the solution matrix, and this can be done in relatively few iterations. If the correct rank of the solution matrix is known, it is shown how to formulate the problem as a smooth unconstrained minimization problem, for which rapid convergence can be obtained by for example the BFGS method. This paper studies hybrid methods that attempt to combine the best features of both types of method. An important feature concerns the interfacing of the component methods. 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 paper. Comparative numerical results are reported.

1 Introduction

Symmetric matrices that have non-negative offdiagonal elements and zero diagonal elements arise as data in many experimental sciences. This occurs when the values

are measurements of squared distances between points (e.g. atoms, stars, cities) 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 paper is to study methods for solving this problem.

An important application arises in the conformation of molecular structures from nuclear magnetic resonance data (see Havel et al. [10] and Crippen [4], [5]). Here a Euclidean distance matrix is used to represent the squares of distances between the atoms of a molecular structure. An attempt to determine such a structure by nuclear magnetic resonance experiments gives rise to a distance matrix F which, because of data errors, may not be Euclidean. There are many other applications in subjects as diverse as archeology, cartography, genetics, geography and multivariate analysis. Pertinent references are given by Al-Homidan [1].

Characterization theorems for the Euclidean distance matrix have been given in many forms over the years. In Section 2 we show that a very concise form of this result can be proved in terms of null-space matrices, that brings out the underlying structure and is readily applicable to the algorithms that follow.

Many advances have taken place in constrained optimization 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 paper addresses a rather more difficult non-smooth optimization problem 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 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.

One approach (Glunt et al. [8]) is to formulate the Euclidean distance matrix problem as a constrained least distance problem in which the constraint is the intersection of two convex sets. The Dykstra-Han alternating projection algorithm can then be used to solve the problem. This idea is outlined in Section 3. This method is globally convergent but the rate of convergence is linear or slower. It is this latter feature that has probably contributed to the relatively little interest that has been shown in such methods. However the method does have the capability of determin-

ing the correct rank of the solution matrix, and this can be done in relatively few iterations.

If the correct rank of the solution matrix is known, it is shown in Section 4 how to formulate the problem as a smooth unconstrained minimization problem, for which rapid convergence can be obtained by for example the BFGS method. We discuss how best to parametrize the problem, and give expressions for the objective function and its first derivatives. A trial and error approach to estimating the correct rank is possible, but is not very appealing.

Thus we are led to study hybrid methods in Section 5 of the paper. 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 a quasi-Newton method 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 paper. Numerical experiments are reported in Section 6.

Recently, and since the research in this paper was carried out, there has been much interest in interior point methods applied to problems with semi-definite matrix constraints (e.g. Alizadeh et al. [2]). It would certainly be of interest to compare this approach with the hybrid methods described in our paper.

Throughout this paper 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. We use the notation $\mathrm{Diag}(A)$ to denote $\mathrm{diag}(a_{ii})$, $i = 1, \ldots, n$. Superscript (k) generally denotes quantities related to the kth iterate, for example $\mathbf{f}^{(k)}$, $X^{(k)}$ etc,. Quantities relating to the solution are superscripted with an asterisk, e.g. r^*, D^*, etc.

2 The Euclidean Distance Matrix Problem

In this section the definition of the Euclidean distance matrix is given, and the relationship between points and distances is summarized. A characterization theorem for the Euclidean distance matrix is proved in a concise way that brings out the underlying structure and is readily applicable to the algorithms that follow. The theorem is essentially due to Schoenberg [12] in the case that $\mathbf{p} = \mathbf{x}_1$ (see below). Young and Householder [13] independently obtain a similar result.

It is necessary to distinguish between distance matrices that are obtained in practice and those that can be derived exactly from n vectors in an affine subspace.

Definition 2.1. *A matrix $D \in \mathbb{R}^{n\times n}$ is called a distance matrix iff it is symmetric, the diagonal elements are zero*

$$d_{ii} = 0 \qquad i = 1, ..., n,$$

and the off-diagonal entries are non-positive

$$d_{ij} \leq 0 \qquad \forall i \neq j.$$

Definition 2.2. *A matrix* $D \in \mathbb{R}^{n\times n}$ *is called a Euclidean distance matrix iff there exist* n *points* $\mathbf{x}_1, \ldots, \mathbf{x}_n$ *in an affine subspace of dimension* $\mathbb{R}^r$ $(r \leq n-1)$ *such that*

$$d_{ij} = -\|\mathbf{x}_i - \mathbf{x}_j\|_2^2 \qquad \forall i, j. \tag{2.1}$$

The negative sign in the definition of d_{ij} is not common, but is included to simplify the subsequent presentation.

The Euclidean distance problem can now be stated as follows. Given a distance matrix $F \in \mathbb{R}^{n\times n}$, find the Euclidean distance matrix $D \in \mathbb{R}^{n\times n}$ that minimizes

$$\|F - D\|_F \tag{2.2}$$

where $\|.\|_F$ denotes the Frobenius norm.

To determine the dimension of an affine subspace we need to make a translation into a subspace that contains the origin. Consider vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$ in $\mathbb{R}^m$. A vector $\mathbf{p}$ is said to be a *weighted combination* of these vectors iff

$$\mathbf{p} = \sum_{i=1}^{n} \mathbf{x}_i y_i, \qquad \sum_{i=1}^{n} y_i = 1.$$

We choose a fixed vector $\mathbf{p}$ in this way and examine the displacements from $\mathbf{p}$.

Definition 2.3. *The vectors* $\mathbf{x}_1, \ldots, \mathbf{x}_n$ *are in an affine subspace of dimension* r *iff the vectors* $\mathbf{x}_i - \mathbf{p}$, $i = 1, \ldots, n$ *have rank* r.

In matrix notation the vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$ are columns of an $m \times n$ matrix X and we can express $\mathbf{p} = X\mathbf{y}$ for some vector $\mathbf{y}$ such that $\mathbf{e}^T\mathbf{y} = 1$ where $\mathbf{e} = (1, \ldots, 1)^T$. The displacement vectors $\mathbf{x}_i - \mathbf{p}$, $i = 1, \ldots, n$ are columns of the matrix

$$X - \mathbf{p}\mathbf{e}^T = X - X\mathbf{y}\mathbf{e}^T = XP \tag{2.3}$$

where

$$P = I - \mathbf{y}\mathbf{e}^T \tag{2.4}$$

is a skew projection matrix. Clearly $\text{rank}(P) = n - 1$ and $P^T\mathbf{e} = \mathbf{0}$. A convenient choice for $\mathbf{p}$ is the vector $\mathbf{x}_1$, in which case $\mathbf{y}$ is the unit vector $\mathbf{e}_1$ and P is a matrix whose first column is the zero vector. Another possibility is to choose the centroid $\mathbf{p} = X\mathbf{e}/n$, in which case P becomes the symmetric projection matrix $P = I - \mathbf{e}\mathbf{e}^T/n$.

The vector $\mathbf{e}$ is seen to be significant and we denote

$$M = \{\mathbf{v} \in \mathbb{R}^n : \ \mathbf{v}^T\mathbf{e} = 0\}$$

as the null space of $\mathbf{e}$. Let columns of a matrix $Z \in \mathbb{R}^{n\times(n-1)}$ provide a basis for M. Z is called a *null-space matrix* and is characterized by $\text{rank}(Z) = n - 1$ and $Z^T\mathbf{e} = \mathbf{0}$.

A possible choice for Z is the matrix, Z_p say, obtained by selecting $n-1$ linearly independent columns from the matrix P in (2.4). Definition 2.3 is then equivalent to the statement that $\mathrm{rank}(XZ_p) = r$ and hence

$$\mathrm{rank}(XZ) = r \tag{2.5}$$

for any null-space matrix Z. This structure enables us to characterize a Euclidean distance matrix in a concise way.

Theorem 2.1. *Let $D \in \mathbb{R}^{n\times n}$ be a symmetric matrix with $\mathrm{Diag}(D) = 0$. Then D is a Euclidean distance matrix iff D is positive semi-definite on M (or equivalently Z^TDZ is positive semi-definite). Moreover if $\mathrm{rank}(Z^TDZ) = r$ then D can be derived from vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$ in an affine subspace of dimension r .*

Proof Let D be a Euclidean distance matrix which is derived from columns of the $m \times n$ matrix X. Then

$$[Z^TDZ]_{ij} = \sum_{kl} z_{ki} d_{kl} z_{lj} = \sum_{kl} z_{ki}(2\mathbf{x}_k^T\mathbf{x}_l - \mathbf{x}_k^T\mathbf{x}_k - \mathbf{x}_l^T\mathbf{x}_l)z_{lj}$$

from (2.1). The term involving $\sum_{kl} z_{ki}\mathbf{x}_k^T\mathbf{x}_k z_{lj}$ can be rearranged as $(\sum_k z_{ki}\mathbf{x}_k^T\mathbf{x}_k)\sum_l z_{lj}$, and is zero because $Z^T\mathbf{e} = \mathbf{0}$. Likewise the term derived from $\mathbf{x}_l^T\mathbf{x}_l$ is zero. Thus

$$[Z^TDZ]_{ij} = 2\sum_{kl} z_{ki}\mathbf{x}_k^T\mathbf{x}_l z_{lj} = 2\sum_{kql} z_{ki}x_{qk}x_{ql}z_{lj}$$

or in matrix notation

$$Z^TDZ = 2Z^TX^TXZ.$$

Hence Z^TDZ is positive semi-definite. If the columns of X are in an affine subspace of dimension r then from (2.5) $r = \mathrm{rank}(XZ) = \mathrm{rank}(Z^TDZ)$.

Conversely we let D be a symmetric matrix such that $\mathrm{Diag}(D) = 0$ and Z^TDZ is positive semi-definite of rank r, and we show how to construct a matrix X whose columns are in an affine subspace of dimension r such that D is derived from X. We define $A = \frac{1}{2}P^TDP$ where P is given by (2.4), and choose the matrix Z_p above as the null space matrix. It readily follows that

$$r = \mathrm{rank}(Z_p^TDZ_p) = \mathrm{rank}(A).$$

[*Proof: Since the extra column in P is a linear combination of the columns of Z_p it follows that* $\mathrm{rank}(Z_p^TDZ_p) = \mathrm{rank}(Z_p^TDP)$. *Likewise we deduce that* $\mathrm{rank}(Z_p^TDP) = \mathrm{rank}(P^TDP)$.] It also follows that A is positive semi-definite, so we can express

$$A = X^TX \tag{2.6}$$

where $X \in \mathbb{R}^{r\times n}$ and $\mathrm{rank}(X) = r$. Then

$$\begin{aligned} -\|\mathbf{x}_i - \mathbf{x}_j\|_2^2 &= 2\mathbf{x}_i^T\mathbf{x}_j - \mathbf{x}_i^T\mathbf{x}_i - \mathbf{x}_j^T\mathbf{x}_j = 2a_{ij} - a_{ii} - a_{jj} \\ &= \mathbf{p}_i^TD\mathbf{p}_j - \tfrac{1}{2}\mathbf{p}_i^TD\mathbf{p}_i - \tfrac{1}{2}\mathbf{p}_j^TD\mathbf{p}_j \end{aligned} \tag{2.7}$$

where $\mathbf{p}_i$ denotes column i of P. It follows from (2.4) that $\mathbf{p}_i = \mathbf{e}_i - \mathbf{y}$. Substituting into (2.7) and using $d_{ii} = 0$ yields

$$-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2 = d_{ij}.$$

Thus D is derived from X. Moreover $\mathbf{y}^T A\mathbf{y} = 0$ by definition of A and (2.4). It follows from (2.6) that $\|X\mathbf{y}\| = 0$ and hence $\mathbf{p} = X\mathbf{y} = \mathbf{0}$. Finally we can deduce from $\text{rank}(X) = r$ that $\text{rank}(XP) = 0$, and it follows from (2.3) and Definition 2.3 that the columns of X are in an affine subspace of dimension r. ■

The special case in which $\mathbf{p} = \mathbf{x}_1$, $\mathbf{y} = \mathbf{e}_1$ and $P = I - \mathbf{e}_1\mathbf{e}^T$ is particularly useful. The resulting matrix $A = \frac{1}{2}P^T DP$ has zeros in the first row and column, and a general expression for the remaining elements is

$$a_{ij} = \tfrac{1}{2}(d_{ij} - d_{1i} - d_{1j}) \qquad i \geq 2, \quad j \geq 2. \tag{2.8}$$

Then the vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$ that are constructed from (2.6) are such that $\mathbf{x}_1 = \mathbf{0}$.

Another useful application of Theorem 2.1 occurs in the projection algorithm of Section 3 for which an orthogonal basis for the null space is available.

Theorem 2.2. *Let $Q \in \mathbb{R}^{n\times n}$ be the Householder matrix given by*

$$Q = I - \frac{2}{\mathbf{w}^T\mathbf{w}}\mathbf{w}\mathbf{w}^T, \qquad \mathbf{w} = (1, \ldots, 1, 1 + \sqrt{n})^T. \tag{2.9}$$

Then the distance matrix $D \in \mathbb{R}^{n\times n}$ is a Euclidean distance matrix iff the $(n-1)\times(n-1)$ block D_1 in

$$QDQ = \begin{bmatrix} D_1 & \mathbf{d} \\ \mathbf{d}^T & \delta \end{bmatrix} \tag{2.10}$$

is positive semi-definite.

Proof Because Q is an othogonal matrix and $Q^T\mathbf{e} = \mathbf{e}_n$, it follows that the first $n-1$ columns of Q provide a null space matrix Z. Since $D_1 = Z^T DZ$ the result can be deduced from Theorem 2.1. ■

3 The Projection Algorithm

In this section we describe a projection algorithm due to Glunt et al. [8] for solving the Euclidean distance matrix problem (2.2). At the end of the section, a more simple and flexible rearrangement of the algorithm is also given. These algorithms are derived from an alternating projection algorithm due to Dykstra [6] for finding the least distance from a fixed point to an intersection of convex sets. This algorithm is given independently by Han [11]. An important feature is the generation of formulae for

certain projection maps that are needed. More background is given about projection methods for the Euclidean distance matrix problem in [1].

The Dykstra-Han algorithm solves the problem

$$\begin{array}{ll} \text{minimize} & \|\mathbf{f}-\mathbf{x}\|_2 \\ \text{subject to} & \mathbf{x} \in \bigcap_{i=1}^{m} K_i \end{array}$$

where the K_i are convex sets in $\mathbb{R}^n$ and $\mathbf{f}$ is given. The algorithm initializes $\mathbf{f}^0 = \mathbf{f}$ and generates a sequence $\{\mathbf{f}^{(k)}\}$ using the iteration formula

$$\mathbf{f}^{(k+1)} = \mathbf{f}^{(k)} + P_m(\ldots P_1(\mathbf{f}^{(k)})\ldots) - P_1(\mathbf{f}^{(k)}). \tag{3.1}$$

Here $P_i(\mathbf{f})$ denotes the l_2 projection of $\mathbf{f}$ on to K_i, that is the (unique) nearest vector to $\mathbf{f}$ in K_i. It is shown by Boyle and Dykstra [3] that $P_i(\ldots P_1(\mathbf{f}^{(k)})\ldots) \to \mathbf{x}^*$ for any $i \geq 1$. However the sequence $\{\mathbf{f}^{(k)}\}$ does not in general converge to $\mathbf{x}^*$ (see [1]).

In applying this method to the Euclidean distance matrix problem, it is appropriate to use the Frobenius matrix norm, and to express (2.2) as

$$\begin{array}{ll} \text{minimize} & \|F-D\|_F \\ \text{subject to} & D \in K_M \cap K_d \end{array} \tag{3.2}$$

where

$$K_M = \{A:\ A \in \mathbb{R}^{n\times n},\quad A^T = A,\quad \mathbf{x}^T A\mathbf{x} \geq 0 \quad \forall\, \mathbf{x} \in M\} \tag{3.3}$$

is a convex cone, and

$$K_d = \{A:\ A \in \mathbb{R}^{n\times n},\quad A^T = A,\quad a_{ii} = 0 \quad \forall\, i = 1,\ldots,n\} \tag{3.4}$$

is a subspace. Clearly from Theorem 2.1, $D \in K_M \cap K_d$ if and only if D is a Euclidean distance matrix.

To apply algorithm (3.1) we need formulae for the projection maps $P_M(.)$ and $P_d(.)$, corresponding respectively to $P_1(.)$ and $P_2(.)$ in (3.1). These are the maps from

$$K = \{A:\ A \in \mathbb{R}^{n\times n},\ A = A^T\}$$

on to K_M and K_d. Because these projections maintain symmetry, there is no need to impose the symmetry constraint explicitly. Since K_d is a subspace, P_d is straightforwardly defined by

$$P_d(F) = F - \mathrm{Diag}(F), \tag{3.5}$$

that is P_d maps F into the matrix obtained by zeroing the diagonal elements of F.

The projection map $P_M(F)$ is determined by finding the solution D of the problem

$$\begin{array}{ll} \text{minimize} & \|F-D\|_F \\ \text{subject to} & D \in K_M. \end{array} \tag{3.6}$$

It is convenient to use the orthogonal matrix Q in (2.9) to express

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

Then the constraint $D \in K_M$ is equivalent to the constraint that $D_1 \geq 0$. Since

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

it follows that $\|F - D\|_F$ is minimized when $\mathbf{d} = \mathbf{f}$, $\delta = \zeta$ and D_1 is the solution of the problem

$$\begin{array}{ll} \text{minimize} & \|F_1 - D_1\|_F \\ \text{subject to} & D_1 \geq 0. \end{array} \tag{3.7}$$

Using a theorem of Higham [11], the solution of (3.7) is given by

$$D_1 = U\Lambda^+ U^T, \tag{3.8}$$

where $U\Lambda U^T$ is the spectral decomposition of F_1 and the components of Λ^+ are defined by $\lambda_i^+ = \max(\lambda_i, 0), \quad i = 1, \ldots, n-1$. Together these results give

$$P_M(F) = Q \begin{bmatrix} U\Lambda^+ U^T & \mathbf{f} \\ \mathbf{f}^T & \zeta \end{bmatrix} Q \tag{3.9}$$

as the required solution of (3.6).

We can now use the projection maps $P_M(F)$ and $P_d(F)$ given by (3.9) and (3.5) to implement the Dykstra-Han algorithm (3.1). Given a distance matrix $F \in \mathbb{R}^{n \times n}$, the algorithm is initialized by $F^{(0)} = F$ and the iteration formula is

$$F^{(k+1)} = F^{(k)} + P_d(P_M(F^{(k)})) - P_M(F^{(k)}). \tag{3.10}$$

This is the form of the algorithm used by Glunt et al. [8]. The sequences $\{P_M(F^{(k)})\}$ and $\{P_d(P_M(F^{(k)}))\}$ both converge to the solution D^* of (3.2) and hence (2.2).

We have found it more convenient to use a different form of the algorithm. By virtue of (3.5), the iteration formula (3.10) can be rearraged as

$$F^{(k+1)} = F^{(k)} - \text{Diag}(P_M(F^{(k)})). \tag{3.11}$$

The effect of this formula is that it only changes the diagonal elements of $F^{(k)}$. This suggests that we iterate with the diagonal matrix

$$\Delta^{(k)} = F^{(k)} - F. \tag{3.12}$$

The iteration formula (3.11) then becomes

$$\Delta^{(k+1)} = \Delta^{(k)} - \text{Diag}(D^{(k)}), \tag{3.13}$$

where $D^{(k)} = P_M(F+\Delta^{(k)})$. These matrices $D^{(k)}$ converge to the solution of (2.2). An advantage of this formulation, which we make use of in Section 5, is that the iteration can be initialized with any diagonal matrix $\Delta^{(0)}$ and not just $\Delta^{(0)} = 0$. Moreover, given any $F^{(k)}$ (or $\Delta^{(k)}$), the test

$$\mathrm{Diag}(D^{(k)}) = 0 \tag{3.14}$$

determines whether $D^{(k)}$ is a Euclidean distance matrix or not.

4 Solution by Unconstrained Minimization

In this section we consider a different approach to the Euclidean distance matrix problem (2.2). The main idea is to replace (2.2) by a smooth unconstrained optimization problem in order to use superlinearly convergent quasi-Newton methods. To do this it is necessary to estimate the rank r of the underlying affine subspace, as this piece of information is not generally known. Once a value of r is chosen, the problem (2.2) is solved by the BFGS method. We give the relevant formulae for derivatives. At the end of the section we discuss details of initialization and implementation.

If the rank r is known, it is possible to express (2.2) as a smooth unconstrained optimization problem in the following way. The unknowns in the problem are chosen to be the elements of the matrix X introduced in (2.3). We take X to have r rows and choose the translation $\mathbf{p} = \mathbf{x}_1$ so that the vector $\mathbf{x}_1 = \mathbf{0}$. This gives us an unconstrained optimization problem in $r(n-1)$ unknowns. We therefore parametrize X by variables $x_i,\ i = 1, \ldots, r(n-1)$ in the following way

$$X = \begin{bmatrix} 0 & x_1 & x_2 & \cdots & x_{n-1} \\ 0 & x_n & x_{n+1} & \cdots & x_{2(n-1)} \\ \vdots & \vdots & & & \vdots \\ 0 & x_{(r-1)(n-1)+1} & \cdots & \cdots & x_{r(n-1)} \end{bmatrix}. \tag{4.1}$$

(In fact it is possible to parametrize the matrix with $\frac{1}{2}(r-1)(r-2)$ fewer unknowns by rotating X to be upper trapezoidal. However it is indicated in [1] that the resulting method tends to use more line searches on a selection of randomly generated problems.)

The objective function $\phi(X)$ is readily calculated by first forming D from X as indicated by (2.1), after which ϕ is given by $\phi(X) = \|D - F\|_F^2$. The elements of the matrix D take the form

$$d_{11} = 0, \qquad d_{i1} = d_{1i} = -\sum_{k=0}^{r-1} x_{i+km-1}^2 \quad i = 2, \ldots, n$$

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

where m denotes $n-1$. Hence

$$\begin{aligned}\phi(X) &= \sum_{i,j=1}^{n}(f_{ij} - d_{ij})^2 \\ &= 2\{\sum_{i=2}^{n}(f_{i1} - d_{i1})^2 + \sum_{\substack{i,j=2\\ i>j}}^{n}(f_{ij} - d_{ij})^2\} \\ &= 2\{\sum_{i=1}^{n}(\sum_{k=0}^{r-1} x_{i+km-1}^2 + f_{i1})^2 + \\ &\qquad \sum_{\substack{i,j=2\\ i>j}}^{n}(\sum_{k=0}^{r-1}(x_{i+km-1} - x_{j+km-1})^2 + f_{ij})^2\} \qquad (4.2)\end{aligned}$$

Our chosen method to minimize $\phi(X)$ is the BFGS quasi-Newton method (see for example [7]). This requires expressions for the first partial derivatives of ϕ, which are given from (4.2) by

$$\begin{aligned}\frac{\partial\phi}{\partial x_s} &= 8x_s\{\sum_{k=0}^{r-1} x_{l+km}^2 + f_{l+1,1}\} \\ &+ 8\{\sum_{j=1}^{m}[\sum_{k=0}^{r-1}(x_{l+km} - x_{j+km})^2 + f_{l+1,j+1}](x_s - x_{j+tm})\} \qquad (4.3)\end{aligned}$$

for all $s = 1,\ldots,r(n-1)$ where $t = (s-l)/m$ and $l = \mathrm{mod}(s,m)$ and if $l = 0$ then $l = m$. The BFGS method also requires the Hessian approximation to be initialized. Where necessary we do this using a unit matrix.

Some care has to be taken when choosing the initial value of the matrix X, in particular the rank of X must be r. If not the minimization method may not be able to increase the rank of X. An extreme case occurs when the initial matrix $X = 0$ is chosen, and $F \neq 0$. It can be seen from (4.3) that the components of the gradient vector are all zero, so that $X = 0$ is a stationary point, but not a minimizer. A gradient method will usually terminate in this situation, and so fail to find the solution.

A reliable method for initializing X is to use the construction suggested by (2.8) and (2.6). Thus we define the elements of A from those of F by

$$a_{ij} = \tfrac{1}{2}(f_{ij} - f_{1i} - f_{1j}) \qquad i \geq 2, \quad j \geq 2. \qquad (4.4)$$

The first row and column of A are zero and are ignored. We then find the spectral decomposition $U\Lambda U^T$ of the nontrivial part of A. Finally the nontrivial part of X in (4.1) is initialized to the matrix $\Lambda_r^{1/2}U_r^T$ where $\Lambda_r = \mathrm{diag}(\lambda_i)$, $i = 1,\ldots,r$ is composed of the r largest eigenvalues in Λ, and columns of U_r are the corresponding eigenvectors. When Λ_r is positive definite, this procedure ensures that X has rank r.

Otherwise the process must be modified in some way, for example by ensuring that the diagonal elements in Λ_r lie above a positive threshold.

An advantage of the unconstrained method is that it allows the spatial dimensions to be chosen by the user. This is useful when the rank is already known. For example if the entries in F are derived from distances between cities then the dimension will be no more than $r = 2$. Likewise, if the entries are derived from distances between atoms in a molecule or stars in space, then the maximum dimension is $r = 3$.

In general however the rank of the affine subspace is not known, for example the atoms in a molecule may turn out to be collinear or coplanar. We therefore must consider an algorithm in which we are prepared to revise our estimate of r. A simple strategy is to repeat the entire unconstrained method for different values of r. If r^* denotes the correct value of r which solves (2.2), then it is observed in [1] that the BFGS method converges rapidly if $r \leq r^*$, and exhibits superlinear convergence. On the other hand if $r > r^*$ then slow convergence is observed. One reason is that there are more variables in the problem. Also redundancy in the parameter space may have an effect. Thus it makes sense to start with a small value of r, and increase it by one until the solution is recognised. One way to recognise termination is when $D^{(r)}$ agrees sufficiently well with $D^{(r+1)}$, where $D^{(r)}$ denotes the Euclidean distance matrix obtained by minimizing ϕ when X in (4.1) has r rows. Numerical experience is reported in [1] for solving various test problems by this method.

An obvious alternative to using the BFGS method is to evaluate the Hessian matrix of second derivatives of $\phi(X)$ and use Newton's method. This would be likely to reduce the number of iterations required. However there is also the disadvantage of increased complexity, and increased housekeeping at each iteration. Moreover it is possible that the Hessian has some negative eigenvalues so a modified form of Newton's method would be required. A simple example serves to illustrate the possibility of a negative eigenvalue. Take $n = 2$, $r = 1$ and let $F = \left[\begin{smallmatrix} 0 & -1 \\ -1 & 0 \end{smallmatrix}\right]$ and $X = [0 \ x_1]$. Then $\phi = 2(1 - x_1^2)^2$. This has global minimizers at $x_1 = \pm 1$, a local maximizer at $x_1 = 0$, and the Hessian is negative for all x_1 such that $3x_1^2 < 1$.

5 Hybrid Methods

The algorithms of Sections 3 and 4 have entirely different features, some good, some bad, which suggests that a combination of both approaches might be successful. Projection methods are globally convergent and hence potentially reliable, but the rate of convergence is first order or slower, which can be very inefficient. Quasi-Newton methods are reliable and locally superlinearly convergent, but require that the correct rank r^* is known. We therefore consider hybrid methods in which the projection algorithm is used sparingly as a way of establishing the correct rank, whilst the BFGS method is used to provide rapid convergence.

In order to ensure that each component method is used to best effect, it is important to be able to transfer information from one method to the other. In particular a

mechanism must be established so that the result from one method is used to provide the initial data for the other, and vice versa. This mechanism must have a fixed point property, so that if one method finds the solution, then the other method is initialized with an iterate that also corresponds to a solution. We show in this section how this can be done.

We have already indicated at the end of Section 3 how the projection method can be initialized with any diagonal matrix Δ. However if $D^{(k)}$ is the Euclidean distance matrix derived from the result matrix $X^{(k)}$ of the BFGS method, it is not obvious how to calculate an initial matrix Δ for the projection method. To address this difficulty, we consider an iteration of the projection method. The current iterate $\Delta^{(k)}$ determines $F^{(k)} = F + \Delta^{(k)}$ and the product $QF^{(k)}Q$ yields $F_1^{(k)}$, $\mathbf{f}^{(k)}$ and $\zeta^{(k)}$, where

$$F^{(k)} = Q \begin{bmatrix} F_1^{(k)} & \mathbf{f}^{(k)} \\ \mathbf{f}^{(k)T} & \zeta \end{bmatrix} Q. \tag{5.1}$$

The spectral decomposition $F_1^{(k)} = U^{(k)}\Lambda^{(k)}U^{(k)T}$ is calculated and $D^{(k)}$ is determined by

$$D^{(k)} = P_M(F^{(k)}) = Q \begin{bmatrix} U^{(k)}\Lambda^{(k)+}U^{(k)T} & \mathbf{f}^{(k)} \\ \mathbf{f}^{(k)T} & \zeta^{(k)} \end{bmatrix} Q. \tag{5.2}$$

It follows from (5.1) and (5.2) that

$$(D^{(k)} - F^{(k)})\mathbf{e} = Q \begin{bmatrix} U^{(k)}(\Lambda^{(k)+} - \Lambda^{(k)})U^{(k)T} & \mathbf{0} \\ \mathbf{0}^T & 0 \end{bmatrix} Q\mathbf{e} = \mathbf{0},$$

since $Q\mathbf{e} = \mathbf{e}_n$. Setting $F^{(k)} = F + \Delta^{(k)}$ from (3.12) implies that

$$\Delta^{(k)}\mathbf{e} = (D^{(k)} - F)\mathbf{e}. \tag{5.3}$$

This expression is exact for the projection method.

Because $\Delta^{(k)}$ is diagonal, (5.3) can be used to compute a matrix $\Delta^{(k)}$ from any given matrix $D^{(k)}$. In our hybrid algorithm we use this as a way of initializing $\Delta^{(k)}$ for the projection method, from the $D^{(k)}$ matrix obtained from the BFGS method. If the BFGS method is using the correct rank $r = r^*$ and has found the global solution of ϕ, then $D^{(k)}$ is the solution D^* of (2.2). Hence (5.3) gives the correct solution Δ^* for the projection method. Even if the rank $r \neq r^*$ in the BFGS method, (5.3) enables some useful information to be extracted from $D^{(k)}$.

Conversely we let $D^{(k)}$ be the matrix obtained in (5.2) by the projection method, and consider how to initialize X for the BFGS method. If $D^{(k)}$ is a Euclidean distance matrix, then it solves (2.2), and by Theorems 2.1 and 2.2, the correct rank r^* is the number of positive eigenvalues in the matrix $\Lambda^{(k)}$. We denote this number by $\mathcal{N}(\Lambda^{(k)})$. In general, when $D^{(k)}$ is not a solution, we use $\mathcal{N}(\Lambda^{(k)})$ to determine the row dimension r of X in (4.1) for the BFGS method. To determine the elements of X we again use the construction suggested by (2.8) and (2.6). Thus we define the elements of A from those of $D^{(k)}$ by

$$a_{ij} = \tfrac{1}{2}(d_{ij} - d_{1i} - d_{1j}) \qquad i \geq 2, \quad j \geq 2. \tag{5.4}$$

The first row and column of A are zero and are ignored. We then find the spectral decomposition $U\Lambda U^T$ of the nontrivial part of A. Finally the nontrivial part of X in (4.1) is initialized to the matrix $\Lambda_r^{1/2} U_r^T$ where $\Lambda_r = \mathrm{diag}(\lambda_i),\ i = 1, \ldots, r$ contains the r positive eigenvalues of Λ, and columns of U_r are the corresponding eigenvectors.

We have found that it is sufficient to carry out only one iteration of the projection method between each call of the BFGS method. Thus we can express our hybrid algorithm in detail as

i. Initialize $k = 0, \quad r^{(0)}$ and $X^{(0)}$

ii. Minimize $\phi(X)$ using the BFGS method, giving $X^{(k)}$ and $D^{(k)}$

iii. Use (5.3) to calculate $\Delta^{(k)}$ from $D^{(k)}$

iv. Redefine $D^{(k)} = P_M(F + \Delta^{(k)})$ using (5.1) and (5.2)

v. Terminate if $\mathrm{Diag}(D^{(k)})$ is within tolerance

vi. Set $r^{(k+1)} = \mathcal{N}(\Lambda^{(k)})$ where $\Lambda^{(k)}$ is the eigenvector matrix of $F_1^{(k)}$.

vii. Initialize $X^{(k+1)}$ from $D^{(k)}$ using (5.4) ff.

viii. Set $k = k + 1$ and go to step ii.

An advantage of this approach is that if the rank is not correct, one iteration of the projection method can quickly give a better estimate. Also $r^{(k)}$ is nor restricted to being increased by one, as for the unconstrained algorithm, and can either increase or decrease. Moreover good approximations of $X^{(k)}$ can be made from the matrix $D^{(k)}$ obtained by the projection method.

We have evaluated two different versions of this algorithm which differ in respect of how $r^{(0)}$ and $X^{(0)}$ are initialized. In *Algorithm 1* we carry out iterations of the projection method starting with $\Delta^{(0)} = 0$ until

$$\mathcal{N}(\Lambda^{(k)}) = \mathcal{N}(\Lambda^{(k-j)}) \qquad j = 1, \ldots, s$$

where s is some pre-selected positive number. This value becomes $r^{(0)}$ for step i above, and $X^{(0)}$ is initialized as in (5.4). The choice of s is a compromise between two effects. If s is small then the rank may not be accurately estimated, but the number of (expensive) iterations taken in the projection method is small. On the other hand if s is large then a more accurate rank is obtained but the projection method needs more iterations. In practice we have found $s = 2$ to be adequate for problems in which r is small.

In *Algorithm 2*, $r^{(0)}$ is supplied by the user and $X^{(0)}$ is calculated from F as indicated in (4.4). This approach avoids the initial sequence of projection iterations, but works well if the user is able to make a good estimate of the rank, which is often the case.

6 Numerical Results

The algorithms have been tested on randomly generated distance matrices F with values distributed between 10^{-3} and 10^3. A Fortran 77 program has been written, using the NAG library to compute eigenvalues for the projection method. The computations are carried out in double precision on a SUN SPARCstation SLC. Table 1 summarizes the results for the four different approaches, the projection method, the unconstrained method, and the hybrid Algorithms 1 and 2. The termination criterion for the unconstrained method is $\|D^{(k)} - D^{(k-1)}\| < 10^{-5}$ and $\|\Delta^{(k)} - \Delta^{(k-1)}\| < 10^{-5}$ for the other methods. All four algorithms converge to essentially the same values. Table 1 shows the comparative results for all methods and Table 2 shows the progress of Algorithm 2 in more detail. An asterisk indicates where the correct rank has been identified. In some cases, with Algorithm 1, the final rank is $r^* + 1$ but the solution is within the required tolerance.

For the projection algorithm, each iteration involves the matrix product $QF^{(k)}Q$, followed by an eigensolution, which are relatively expensive $O(n^3)$ calculations. Thus the projection algorithm is not competitive. For the other algorithms, the housekeeping associated with each line search is $O(n^2)$. Also, if care is taken, it is possible to calculate $\phi(X)$ and $\nabla\phi(X)$ in $O(n^2)$ operations. Thus each line search is much less expensive than an iteration of the projection method. For the unconstrained algorithm the initial value $r^{(0)}$ is tabulated, and r is increased by one until the solution is found. The total number of line searches is tabulated, and within this figure, it is found that fewer line searches are required as r increases. It can be seen that the total number of line searches is much greater than is required by the hybrid methods. Also the initial value $r^{(0)} = 6$ is rather arbitrary: a smaller value of $r^{(0)}$ would have given an even larger number of line searches.

Both hybrid algorithms are seen to be effective. As n increases, Algorithm 1 takes an increasing number of projection iterations before the rank settles down. We find it better to increase the value of s as the value of r^* increases. Once the projection iteration has settled down, the BFGS method finds the solution rapidly and no further projection steps are needed. Algorithm 2 requires a relatively large number of line searches (see Table 2) in the first call of the BFGS method, after which one projection step finds the correct rank, and the next call of BFGS finds the solution in a few line searches. This is because of the good initial starting matrix X given by the projection method. Because the projection steps in Algorithm 1 are relatively expensive, the difference in computing time between these algorithms is not very significant.

	PA		UA			A1				A2
n	r^*	NPI	$r^{(0)}$	TNL	NV	s	NPI	$r^{(k)}$	NL	TNL
5	2	21	2*	12	8	2	2	2*	7	12
10	4	46	3	80	36	2	2	4*	15	44
15	5	64	4	140	70	3	4	6(5*)	22	76
20	7	101	5	176	133	3	4	7*	18	81
25	8	85	6	221	192	3	4	8*	14	106
30	9	129	6	144	261	3	4	10(9*)	19	52
35	9	115	6	382	306	4	8	9*	23	109
40	10	168	6	161	390	4	7	11(10*)	21	38
45	11	136	6	246	484	4	9	11*	17	64
50	13	171	6	288	637	4	7	13*	13	142

Table 1: Comparing four algorithms for the Euclidean distance matrix problem.

PA: The projection algorithm (Section 3).
UA: The unconstrained algorithm (Section 4).
A1: Hybrid Algorithm 1.
A2: Hybrid Algorithm 2.
NPI: Number of projection iterations.
NL: Number of line searches in the BFGS method.
TNL: Total number of line searches in the unconstrained algorithm.
NV: Maximum number of variables in the unconstrained algorithm.

	A2			
n	$r^{(0)}$	NL in BFGS	$r^{(k)}$ from OPA	NL in BFGS
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 2: Detailed progress of Algorithm 2.

OPA: One iteration of the projection algorithm.
NL: Number of line searches.

References

[1] Al-Homidan, S. [1993]. Hybrid methods for optimization problems with positive semi–definite matrix constraints, Ph.D. Thesis, Dept of Mathematics and Computer Science, University of Dundee, Dundee, Scotland.

[2] Alizadeh, F., Haeberly, J-P.A. and Overton, M.L. [1994]. Primal-Dual Interior-Point Methods for Semidefinite Programming, *working paper - XV Symposium on Mathematical Programming, Ann Arbor.*

[3] 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.L. Dykstra, T. Robertson, and F.T. Wright), Lecture Notes in Statistics 37, Springer–Verlag, Berlin, pp. 28–47.

[4] Crippen, G.M. [1977]. A novel approach to calculation of conformation: distance geometry. *J. Computational Physics 24, pp. 96–107.*

[5] Crippen, G.M. [1978]. Rapid calculation of coordinates from distance measures. *J. Computational Physics 26, pp. 449–452.*

[6] Dykstra, R.L. [1983]. An algorithm for restricted least squares regression, *J. Amer. Stat. Assoc. 78, pp. 839–842.*

[7] Fletcher, R. [1987]. *Practical Methods of Optimization, 2nd. Edition*, Wiley, Chichester.

[8] 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 Anal. Appl. , 4, pp. 589–600.*

[9] Han, S.P. [1988]. A successive projection method. *Math. Programming, 40, pp. 1–14.*

[10] Havel, T. Kuntz, I. and Crippen, G.M. [1983]. The theory and practice of distance geometry, *Bull. Math. Biol., 45, pp. 665–720.*

[11] Higham, N.J. [1988]. Computing a nearest symmetric positive semi–definite matrix, *Linear Alg. and Appl., 103, pp. 103–118.*

[12] 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.*

[13] Young, G. and Householder A.S. [1938]. Discussion of a set of points in terms of their mutual distances, *Psychometrika 3, pp. 19–22.*