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

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

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

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

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

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

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

$$No.\ \Lambda_r^{(k)} = No.\ \Lambda_r^{(k+j)} \quad j = 1, 2, \ldots, s \tag{4.3.2}$$

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 ϕ 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.