Fast algorithms for multidimensional scaling: A comparison of majorization and spectral gradient methods

P.J.F. Groenen * W. Glunt † T. L. Hayden ‡

August 2, 1999

Abstract

The majorization algorithm SMACOF and the spectral gradient algorithm are two methods used for minimizing the Stress function for multidimensional scaling. For large scale multidimensional scaling problems arising, e.g., in molecular conformation and distance-based multivariate analysis, the speed of the algorithm is crucial, since it determines the number of objects that can be handled by the algorithm. Here we investigate ways to increase the speed of these algorithms. In a simulation study, we compare the two algorithms. The spectral gradient algorithm turns out to be 7 to 10 times faster than SMACOF. The relaxed update for SMACOF (De Leeuw and Heiser 1980) needs a dilation factor and is about twice as fast as ordinary SMACOF. Acceleration of the spectral gradient algorithm by a dilation factor and a preconditioner are shown to increase the speed of the algorithm by about 15%.

Key Words: Distance Matrices, Majorization, Molecular Conformation, Spectral Gradient Method, Preconditioners.

*Department of Data Theory, Leiden University, P.O. Box 9555, 2300 RB Leiden, The Netherlands (e-mail: groenen@rulfsw.fsw.leidenuniv.nl). Supported by The Netherlands Organization for Scientific Research (NWO) by grant nr. 575-67-053 for the 'PIONEER' project 'Subject Oriented Multivariate Analysis'.

†Supported by NSF grant CHE-9005960

‡Supported by NSF grant CHE-9005960

1 Introduction

Multidimensional scaling (MDS) by minimizing Stress (Kruskal 1964a, 1964b) first arose in psychology and has been applied widely in the social sciences. The aim of MDS is to represent dissimilarities between objects as distances between points in a low dimensional space. More recently MDS by minimization of Stress has played a major role in chemistry for finding molecular conformations as a part of distance geometry algorithms Havel (1991), Glunt, Hayden, and Raydan (1993). The aim is to find a three dimensional representation of the atoms of a molecule. Another recent application of MDS occurs in distance-based multivariate analysis (Meulman 1986, 1992), where the data consist of an objects by variables matrix. The aim is to represent those objects as points in a low dimensional space such that their distances match the distances between the rows of the data matrix as well as possible. The number of objects in both applications of MDS can be considerable, e.g., in molecular conformation 500 to 5000 atoms in a single DNA molecule are not uncommon and in distance-based multivariate analysis the number of objects may correspond to the number of respondents in a sample ranging from 20 to several thousands. Since Stress cannot be minimized analytically, iterative algorithms have to be used. In order to to solve large MDS problems — generated by the two applications discussed above — fast algorithms are needed.

In this paper we compare the speed of two algorithms for minimizing Stress. The first algorithm (SMACOF) is based on majorization (De Leeuw 1977; De Leeuw and Heiser 1980; Meulman 1986; De Leeuw 1988). It has been standard in the social sciences for some time and was used by Havel (1991) to find molecular conformations. A second approach to minimization of Stress, the spectral gradient algorithm, has been recently

proposed (Glunt et al. 1993). The main problem with current algorithms for Stress is that they converge slowly, especially for large data sets. Therefore, the aim of this paper is to compare and improve the speed of these two algorithms. For this purpose, we explicitly describe the algorithms and propose some known and new acceleration tools for both algorithms.

For the majorization algorithm we propose to extend the relaxed update of De Leeuw and Heiser (1980) with an optimal dilation factor. To accelerate the spectral gradient algorithm we use preconditioners (Glunt, Hayden, and Raydan 1994) and apply an optimal dilation factor as well. We then present a numerical comparison of the performance of the majorization and the spectral gradient algorithms (with and without acceleration) to minimize Stress.

2 Minimizing STRESS

In this section we introduce the notation used in this paper. Suppose one is given non-negative dissimilarities δ_{ij} between pairs of objects $1 \leq i, j \leq n$. The multidimensional scaling problem can be described as seeking coordinates for n points in p dimensions, represented in an $n \times p$ matrix $\mathbf{X}$, whose Euclidean distances $d_{ij}(\mathbf{X})$ approximate δ_{ij} as closely as possible. The distance $d_{ij}(\mathbf{X})$ contains the Euclidean distances between the points, i.e.,

$$d_{ij}^2(\mathbf{X}) = \sum_{s=1}^{p}(x_{is} - x_{js})^2.$$

In the sequel we assume that $p \geq 2$ here because the unidimensional scaling problem turns out to be combinatorial in nature (see, e.g., Heiser and De Leeuw 1977; Defays 1978; Hubert and Arabie 1986; Pliner 1996; Groenen and Heiser 1996) and efficient

global minimization strategies exist for small n (Hubert and Arabie 1986).

Suppose that $\mathbf{W} = [w_{ij}]$ is a symmetric matrix of nonnegative weights with zero diagonal. The objective function to be minimized is

$$\sigma = \sigma(\mathbf{X}) = \sum_{i<j} w_{ij}(d_{ij}(\mathbf{X}) - \delta_{ij})^2, \tag{1}$$

called raw Stress in the psychometric literature but we shall refer to it as Stress. It is useful to decompose Stress into

$$\begin{aligned}
\sigma(\mathbf{X}) &= \sum_{i<j} w_{ij}(\delta_{ij} - d_{ij}(\mathbf{X}))^2 \\
&= \sum_{i<j} w_{ij}\delta_{ij}^2 + \sum_{i<j} w_{ij}d_{ij}^2(\mathbf{X}) - 2\sum_{i<j} w_{ij}\delta_{ij}d_{ij}(\mathbf{X}) \\
&= \eta_\delta^2 + \eta^2(\mathbf{X}) - 2\rho(\mathbf{X}).
\end{aligned}$$

Note that we assume that $\mathbf{W}$ is irreducible, i.e., there exists no partitioning of $\mathbf{W}$ such that $w_{ij} = 0$ if objects i and j are in different subsets. If $\mathbf{W}$ is reducible then the MDS problems can be solved by solving smaller irreducible MDS problems.

A different expression of the distance (Groenen, De Leeuw, and Mathar 1996; Borg and Groenen 1997) gives insight in the notation in matrix algebra of Stress proposed in earlier papers (De Leeuw 1977, De Leeuw and Heiser 1980, De Leeuw 1988). Let $\mathbf{x}_s$ be column s of $\mathbf{X}$. Then we can express $(x_{is} - x_{js})^2$ as

$$(\mathbf{x}_s'(\mathbf{e}_i - \mathbf{e}_j))^2 = \mathbf{x}_s'(\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)'\mathbf{x}_s,$$

where $\mathbf{e}_i$ is column i of the identity matrix. By defining the matrix

$$\mathbf{A}_{ij} = (\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)'$$

we can express the squared distances as

$$d_{ij}^2(\mathbf{X}) = \sum_{s=1}^{p} \mathbf{x}_s'\mathbf{A}_{ij}\mathbf{x}_s = \text{tr } \mathbf{X}'\mathbf{A}_{ij}\mathbf{X}. \tag{2}$$

Note that matrix $\mathbf{A}_{ij}$ is a symmetric rank one matrix with zero elements, except the diagonal elements $a_{ii} = a_{jj} = 1$ and off-diagonal elements $a_{ij} = a_{ji} = -1$.

Using (2) z rewrite $\eta^2(\mathbf{X})$ as

$$\begin{aligned}\eta^2(\mathbf{X}) &= \sum_{i<j} w_{ij} d_{ij}^2(\mathbf{X}) = \sum_{i<j} w_{ij} \text{tr } \mathbf{X}'\mathbf{A}_{ij}\mathbf{X} \\ &= \text{tr } \mathbf{X}'(\sum_{i<j} w_{ij}\mathbf{A}_{ij})\mathbf{X} = \text{tr } \mathbf{X}'\mathbf{V}\mathbf{X}. \end{aligned} \tag{3}$$

Thus, the elements of $\mathbf{V}$ are given by $v_{ij} = -w_{ij}$ if $i \neq j$ and $v_{ii} = \sum_{j \neq i} w_{ij}$. Since $\mathbf{V}$ is the sum of the matrices $\mathbf{A}_{ij}$ and each $\mathbf{A}_{ij}$ has row and column sum equal to zero, $\mathbf{V}$ must also be row and column centered. Moreover, $\mathbf{V}$ is positive semidefinite because it is the weighted sum of the positive semidefinite matrices $\mathbf{A}_{ij}$ with nonnegative weights w_{ij}. Since $\mathbf{W}$ is irreducible, $\mathbf{V}$ has rank $n - 1$.

In a similar way we can rewrite $\rho(\mathbf{X})$. For $d_{ij}(\mathbf{X}) \neq 0$, the Euclidean distance equals $d_{ij}(\mathbf{X}) = d_{ij}^{-1}(\mathbf{X})\text{tr } \mathbf{X}'\mathbf{A}_{ij}\mathbf{X}$, so that

$$\begin{aligned}\rho(\mathbf{X}) &= \sum_{i<j} w_{ij}\delta_{ij} d_{ij}(\mathbf{X}) = \sum_{i<j} w_{ij}\delta_{ij} d_{ij}^{-1}(\mathbf{X})\text{tr } \mathbf{X}'\mathbf{A}_{ij}\mathbf{X} \\ &= \text{tr } \mathbf{X}' \left(\sum_{i<j} w_{ij}\delta_{ij} d_{ij}^{-1}(\mathbf{X})\mathbf{A}_{ij} \right) \mathbf{X} = \text{tr } \mathbf{X}'\mathbf{B}(\mathbf{X})\mathbf{X}. \end{aligned} \tag{4}$$

The matrix function $\mathbf{B}(\mathbf{X}) = [b_{ij}(\mathbf{X})]$ is a function of $\mathbf{X}$ and is defined by

$$b_{ij}(\mathbf{X}) = \begin{cases} -w_{ij}\delta_{ij}/d_{ij}(\mathbf{X}) & \text{if } i \neq j \text{ and } d_{ij}(\mathbf{X}) > 0, \\ 0 & \text{if } i \neq j \text{ and } d_{ij}(\mathbf{X}) = 0, \\ -\sum_{k \neq i} b_{ik}(\mathbf{X}) & \text{if } i = j. \end{cases}$$

Inserting (3) and (4) in $\sigma(\mathbf{X})$ gives

$$\begin{aligned}\sigma(\mathbf{X}) &= \eta_\delta^2 + \eta^2(\mathbf{X}) - 2\rho(\mathbf{X}) \\ &= \eta_\delta^2 + \text{tr } \mathbf{X}'\mathbf{V}\mathbf{X} - 2\text{tr } \mathbf{X}'\mathbf{B}(\mathbf{X})\mathbf{X}. \end{aligned}$$

Instead of Stress (1) we report normalized STRESS, $\sigma_n = \sigma/\eta_\delta^2$ which has $0 \leq \sigma_n \leq 1$ regardless of the scale of the dissimilarities or n (see, e.g., Commandeur 1992). Since η_δ^2 does not depend on $\mathbf{X}$, the stationary $\mathbf{X}$ are equivalent for σ_n and σ. If $\eta_\delta^2 = 1$, as in De Leeuw (1988), σ_n and σ are equal. Moreover, at a local minimum σ_n is equal to the square of Kruskal's Stress-1 if $\mathbf{X}$ is optimally dillated (see Borg and Groenen 1997, p. 201).

In the sequel we need the gradient and the Hessian of Stress. The function $\sigma(\mathbf{X})$ is differentiable if and only if $d_{ij}(\mathbf{X}^*) > 0$ for all $i < j$. De Leeuw (1984) proved that $\sigma(\mathbf{X})$ is differentiable at a stationary $\mathbf{X}$ if $w_{ij}\delta_{ij} > 0$ for all $i < j$. In the applications discussed in the introduction this condition can be made to hold and, therefore, we will assume differentiability. A study of the gradient and Hessian for general MDS loss functions including STRESS, S-STRESS, and MULTISCALE was presented by Groenen, De Leeuw, and Mathar (1995). Their approach is based on finding the gradient and Hessian for $d_{ij}(\mathbf{X})$ and $d_{ij}^2(\mathbf{X})$, multiply the result by w_{ij} and sum over $i < j$. This procedure produces the gradient and Hessian of $\eta^2(\mathbf{X})$ and $\rho(\mathbf{X})$. This approach shows that the gradient of $\sigma(\mathbf{X})$ can be written as

$$\nabla\sigma(\mathbf{X}) = 2\sum_{i<j}^{n} w_{ij}(1 - \delta_{ij}d_{ij}^{-1}(\mathbf{X}))\mathbf{A}_{ij}\mathbf{X} = 2(\mathbf{V}\mathbf{X} - \mathbf{B}(\mathbf{X})\mathbf{X}). \tag{5}$$

The Hessian of $\sigma(\mathbf{X})$ has a block structure of $p \times p$ blocks $\mathbf{H}_{st}$ of size $n \times n$, where

$$\begin{aligned}\mathbf{H}_{st} &= 2\hat{\delta}_{st}\sum_{i<j}^{n} w_{ij}(1 - \delta_{ij}d_{ij}^{-1}(\mathbf{X}))\mathbf{A}_{ij} + \\ &\quad 2\sum_{i<j}^{n} w_{ij}\delta_{ij}d_{ij}^{-3}(\mathbf{X})(x_{is} - x_{js})(x_{it} - x_{jt})\mathbf{A}_{ij}\end{aligned}$$

and $\hat{\delta}$ is the Kronecker delta with $\hat{\delta}_{ss} = 1$ and $\hat{\delta}_{st} = 0$ if $t \neq s$. Among the properties of the Hessian they established was that the Hessian $\mathbf{H}$ of STRESS has p zero eigenvalues

due to the invariance of distances under translation. Furthermore, at a stationary point, $\mathbf{H}$ has at least $p(p-1)/2$ additional zero eigenvalues, due to the invariance of Euclidean distances under rotation. Note that Glunt et al. (1994) reported the Hessian incorrectly. They erroneously set the off-diagonal elements of $\mathbf{H}_{st}$ equal to zero for $s \neq t$.

2.1 The Majorizing Algorithm

The SMACOF algorithm (Scaling by MAximizing a COvex Function), for minimizing Stress was first proposed by De Leeuw (1977) applying theory from convex analysis. Later the algorithm was explained by iterative majorization and the acronym now stands for Scaling by MAjorizing a COmplicated Function). Iterative majorization has become increasingly popular recently (see, for example, De Leeuw 1988, Meulman 1992, Groenen 1993, Groenen et al. 1995). For a general overview of majorization, see De Leeuw (1994), Heiser (1995), or, for an introduction, see Borg and Groenen (1997). Algorithms based on iterative majorization have several attractive properties. First, the function value never increases, hence decreases or remains the same. Second, majorizing algorithms generally have global convergence of the function values, that is, from any admissible starting configuration the sequence of function values converges. Thirdly, constraints that have a least squares solution can be easily implemented without disturbing its convergence properties (De Leeuw and Heiser 1980).

The basic idea of iterative majorization is that in every iteration k we work on a auxiliary function, the majorizing function, that meets the following three requirements. It is simple, it agrees with the original function at the current estimate $\mathbf{X}_k$, and for all other $\mathbf{X}$ it is always larger than or equal to the original function. By minimizing the majorizing function, automatically a lower (or equal) value of Stress is obtained, since the

majorizing function is never smaller than Stress. Because Stress is bounded from below by zero, the majorization algorithm always gives a convergent sequence of nonincreasing function values, hence SMACOF has global convergence. Although the method is very simple it is not very fast. In fact, De Leeuw (1988) proved linear convergence for SMACOF.

The SMACOF algorithm is summarized as follows.

SMACOF Algorithm

1. Let $\mathbf{X}_0 \in R^{n\times p}$ be a centered starting configuration;
2. Compute $\mathbf{V}^+$, the Moore-Penrose inverse of $\mathbf{V}$;
3. Set $k = 0$;
4. Repeat
5. $\quad \overline{\mathbf{X}}_{k+1} = \mathbf{V}^+\mathbf{B}(\mathbf{X}_k)\mathbf{X}_k$;
6. $\quad \mathbf{X}_{k+1} = \overline{\mathbf{X}}_{k+1}$;
7. $\quad k = k + 1$;
8. Until $||\nabla\sigma(\mathbf{X}_k)|| \leq \varepsilon\,(1 + \sigma(\mathbf{X}_k))$.

Here, ε is a small stopping tolerance, and $\mathbf{V}^+$ is the Moore-Penrose inverse of $\mathbf{V}$ given by $\mathbf{V}^+ = (\mathbf{V} + \mathbf{1}\mathbf{1}')^{-1} - n^{-2}\mathbf{1}\mathbf{1}'$ with $\mathbf{1}\mathbf{1}'$ the matrix with all elements equal to one. Note that if all $w_{ij} = 1$ then $\mathbf{V}^+ = n^{-1}(\mathbf{I} - n^{-1}\mathbf{1}\mathbf{1}')$, the centering matrix.

2.2 The Spectral Gradient Algorithm

The spectral gradient (SPG) algorithm is a variation of the steepest descent algorithm. This method is due to Barzilai and Borwein (1988). In contrast to standard steepest

descent algorithms, the spectral gradient method does not require a line search. The stepsize is only dependent on the current gradient, the previous gradient, the current $\mathbf{X}$, and the previous $\mathbf{X}$, so that calculating the stepsize is a minor computational effort. The SPG algorithm has at most R-superlinear convergence. Although the SPG algorithm summarized below does not have global convergence, we have not encountered nonconvergence in practise. Under mild adaptations of the method, Raydan (1997) established global convergence for the SPG algorithm. The SPG method was applied first to multidimensional scaling by Glunt et al. (1993).

The spectral gradient algorithm for multidimensional scaling is summarized as follows.

Spectral Gradient Algorithm

1. Let $\mathbf{X}_0 \in R^{n\times p}$ be a centered starting configuration, $\alpha_0 = ||\nabla\sigma(\mathbf{X}_0)||$;

2. Let $\mathbf{X}_1 = \mathbf{X}_0 - \alpha_0^{-1}\nabla\sigma(\mathbf{X}_0)$;

3. Set $k = 1$;

4. Repeat;

5. $\mathbf{S}_{k-1} = \mathbf{X}_k - \mathbf{X}_{k-1}$;

6. $\mathbf{Y}_{k-1} = \nabla\sigma(\mathbf{X}_k) - \nabla\sigma(\mathbf{X}_{k-1})$;

7. $\alpha_k = \dfrac{\text{tr } \mathbf{S}'_{k-1}\mathbf{Y}_{k-1}}{\text{tr } \mathbf{S}'_{k-1}\mathbf{S}_{k-1}}$;

8. $\overline{\mathbf{X}}_k = \mathbf{X}_k - |\alpha_k^{-1}|\nabla\sigma(\mathbf{X}_k)$;

9. $\mathbf{X}_{k+1} = \overline{\mathbf{X}}_k$;

10. $k = k + 1$;

11. Until $||\nabla\sigma(\mathbf{X}_k)|| \leq \varepsilon\,(1 + \sigma(\mathbf{X}_k))$.

By taking the absolute value of α_k in Step 8 it is more likely to decrease Stress for negative $\alpha_k < 0$ than using α_k itself (as in Glunt et al., 1993). If α_k^{-1} were used instead of its absolute value, then a negative α_k almost certainly leads to an uphill step. However, $\alpha_k < 0$ seldomly occurred in our experiments. Alternatively, one could replace a negative α_k by $||\nabla\sigma(\mathbf{X}_k)||$ as in iteration 1.

3 Acceleration

In this section we discuss acceleration methods for the two algorithms described above. For the SMACOF algorithm we discuss the relaxed update, and for the spectral gradient method we discuss the use of a preconditioner. At the end of this section, a third method based on optimal dilation of the configuration is discussed that can be applied to both algorithms.

3.1 Acceleration of the Majorization Algorithm

To accelerate the SMACOF algorithm De Leeuw and Heiser (1980) proposed to use the linear combination

$$\mathbf{X}_{k+1} = \gamma\mathbf{X}_k + (1 - \gamma)\overline{\mathbf{X}}_{k+1}, \tag{6}$$

which preserves the convergence of the majorization algorithm for every $-1 < \gamma < 1$. De Leeuw and Heiser (1980) report that choosing $\gamma = -1$ approximately halves the number of iterations. They also found that it was not worth the effort to search for an optimal

γ. Inserting $\gamma = -1$ in (6) gives

$$\mathbf{X}_{k+1} = 2\overline{\mathbf{X}}_{k+1} - \mathbf{X}_k, \tag{7}$$

which they called the *relaxed update*. Thus, the relaxed update in the SMACOF algorithm is obtained by replacing Step 6 by (7).

3.2 The Preconditioned Spectral Gradient Method

An extension of the spectral gradient method, the preconditioned spectral gradient (P-SPG) algorithm, was given in Glunt, Hayden, and Raydan (1994). The aim is to accelerate the spectral gradient algorithm by decreasing the spread of the spectrum of the Hessian of $\sigma(\mathbf{X})$. The preconditioned spectral gradient algorithm is defined as follows. Let vec($\mathbf{Z}$) denote the $np \times 1$ vector consisting of the columns of $\mathbf{Z}$ stacked underneath each other.

Preconditioned Spectral Gradient Algorithm

1. Let $\mathbf{X}_0 \in R^{n\times p}$ be a centered starting configuration, $\alpha_0 = ||\nabla\sigma(\mathbf{X}_0)||$;

2. Set $q_{\text{prec}} = \text{false}$, $\varepsilon_{\text{prec}} = 10^{-3}$;

3. Let $\mathbf{X}_1 = \mathbf{X}_0 - \alpha_0^{-1}\nabla\sigma(\mathbf{X}_0)$;

4. Set $k = 1$;

5. Repeat

6. Let $\mathbf{S}_{k-1} = \mathbf{X}_k - \mathbf{X}_{k-1}$;

7. $\mathbf{Y}_{k-1} = \nabla\sigma(\mathbf{X}_k) - \nabla\sigma(\mathbf{X}_{k-1})$;

8. If $q_{\text{prec}} = \text{true}$

9. Solve $\mathbf{G}\text{vec}(\mathbf{Z}_k) = \text{vec}(\nabla\sigma(\mathbf{X}_k))$ for $\mathbf{Z}_k$ with $\mathbf{G} \in R^{np\times np}$ symmetric and positive definite matrix;

10. $\alpha_k = \dfrac{\text{tr } \mathbf{S}'_{k-1}\mathbf{Y}_{k-1}}{\text{vec}(\mathbf{S}_{k-1})'\mathbf{G}\text{vec}(\mathbf{S}_{k-1})}$;

11. $\overline{\mathbf{X}}_k = \mathbf{X}_k - |\alpha_k^{-1}|\mathbf{Z}_k$;

12. Center $\overline{\mathbf{X}}_k$ to obtain $\mathbf{X}_{k+1}$;

13. else

14. $\alpha_k = \dfrac{\text{tr } \mathbf{S}'_{k-1}\mathbf{Y}_{k-1}}{\text{tr } \mathbf{S}'_{k-1}\mathbf{S}_{k-1}}$;

15. $\mathbf{X}_{k+1} = \mathbf{X}_k - |\alpha_k^{-1}|\nabla\sigma(\mathbf{X}_k)$;

16. If $q_{\text{prec}} = \text{true}$

17. If $||\nabla\sigma(\mathbf{X}_{k+1})|| \geq 10\varepsilon_{\text{prec}}(1 + \sigma(\mathbf{X}_{k+1}))$

18. $q_{\text{prec}} = \text{false}$;

19. $\varepsilon_{\text{prec}} = \varepsilon_{\text{prec}}/2$;

20. If $q_{\text{prec}} = \text{false}$

21. If $||\nabla\sigma(\mathbf{X}_{k+1})|| \leq \varepsilon_{\text{prec}}(1 + \sigma(\mathbf{X}_k))$

22. $q_{\text{prec}} = \text{true}$;

23. $k = k + 1$,

24. Until $||\nabla\sigma(\mathbf{X}_k)|| \leq \varepsilon(1 + \sigma(\mathbf{X}_k))$.

The preconditioner must only be used in a close neighborhood of the minimum. Therefore, P-SPG is started by SPG steps (Steps 14-15, because $q_{\text{prec}} = \text{false}$). Glunt et al. (1994) found good results when the preconditioner is turned on once the configuration showed little change. If the preconditioner is turned on too early (i.e., too far away of the local minimum) one may observe excessively large increases in Stress. To ensure in a dynamical way that the preconditioner is not turned on too early, we turn it off (by the test in Step 18) whenever the norm of the gradient becomes too large. Moreover, the test for turning the preconditioner on again (by the test in Step 22) becomes more stringent by reducing the size of $\varepsilon_{\text{prec}}$ in Step 20.

Glunt et al. (1994) used a mathematically equivalent expression for Step 10, that included the factor α_{k-1}. It turned out that their definition is numerically less stable, due to the propagation of error. The definition of α_k in Step 10 does not contain this recursion and is, therefore, to be preferred. It can be verified that the SPG algorithm is a special case of the preconditioned one when $\mathbf{G} = \mathbf{I}$. To find a better $\mathbf{G}$ in the P-SPG algorithm we need the Hessian of $\sigma(\mathbf{X})$.

The preconditioner $\mathbf{G}$ is a partitioned block matrix with each block a $n \times n$ diagonal matrix. The blocks of $\mathbf{G}$ are equal to the diagonal matrix containing the diagonal elements of the blocks in the Hessian, i.e., $\mathbf{G}_{st} = \text{Diag}(\mathbf{H}_{st})$ where Diag is the operator that puts the diagonal elements in a diagonal matrix. Despite the error in the expression of the Hessian, the results of Glunt et al. (1994) are still valid, because the error only affects the off diagonal elements of $\mathbf{H}_{st}$ but not the diagonals that were used in the preconditioner $\mathbf{G}$.

To solve the system in Step 9 of the P-SPG algorithm we used Gaussian elimination (without pivoting) on the blocks, which uses the sparseness of $\mathbf{G}$. Gaussian elimination yields zero values below the main diagonal and retains exactly the same block structure (with different values) above the diagonal. Then backsolving involves no fill. The entire solution requires $np(p+1)$ multiplications/divisions and a similar number of additions or subtractions. The storage requirement for $\mathbf{G}$ is only $p(p+1)/2$ n-dimensional vectors.

Apart from the preconditioner defined above, a number of different preconditioners were also tested. For example, tridiagonal approximations in each block were used for the preconditioner. In addition a Gauss-Seidel approach was tried. Some improvement in iteration count was observed but the increased cost of computation always resulted in poorer results than the simple diagonal approximation. We found that if the full Hessian was used for a preconditioner (the best possible choice but too expensive to compute the inverse) then the number of iterations was about one half those required when the block diagonal approximation is used. This indicates that the block diagonal is not far from the best possible, and is very fast. Hence we continue to use the diagonal block approximation as the preconditioner.

3.3 Optimal Dilation

One property of the majorization algorithm (without relaxed update), is that in each step $\mathbf{X}_k$ is optimally dilated with respect to STRESS. However, this property is lost in the (preconditioned) spectral gradient algorithm as well as in the SMACOF algorithm with relaxed update. For these algorithms it makes sense to compute explicitly an optimal dilation factor in each iteration. To find an optimal dilation factor β for a fixed $\mathbf{X}$, we use the property that the distance function is positively homogeneous, i.e., $d_{ij}(\beta\mathbf{X}) = \beta d_{ij}(\mathbf{X})$

for any $\beta \geq 0$. Thus $\sigma(\beta \mathbf{X}) = \eta_\delta^2 + \beta^2 \eta^2(\mathbf{X}) - 2\beta\rho(\mathbf{X})$, which is minimized over β by $\beta^* = \rho(\mathbf{X})/\eta^2(\mathbf{X})$. With this choice of β we obtain

$$\sigma(\beta^* \mathbf{X}) = \eta_\delta^2 + \beta^{*2}\eta^2(\mathbf{X}) - 2\beta^*\rho(\mathbf{X}) = \eta_\delta^2 - \frac{\rho^2(\mathbf{X})}{\eta^2(\mathbf{X})},$$

a property already used by De Leeuw (1977) and Mathar and Groenen (1991) in a different context. Effectuating the dilation requires np operations for $\mathbf{X}$ and $n(n-1)/2$ operations for the distances.

As β^* differs more from 1, the gain in Stress by optimal dilation increases. Particularly for the (preconditioned) spectral gradient algorithm optimal dilation should increase efficiency if the steplength of α_k is too large.

To see why the optimal dilation factor is needed for the SMACOF algorithm with the relaxed update, consider the following. Start the algorithm with configuration with $a\mathbf{Z}$, where $\mathbf{Z}$ is a local minimum and a is a given multiplication factor with $0 < a < 2$ and $a \neq 1$. Using positive homogeneity of the distance function and that $\mathbf{Z}$ is a local minimum, we have the necessary condition that $\mathbf{Z} = \mathbf{V}^+\mathbf{B}(a\mathbf{Z})a\mathbf{Z}$, so that $\overline{\mathbf{X}} = \mathbf{Z}$. Then, the relaxed update at iteration 1 becomes $\mathbf{X}_1 = 2\overline{\mathbf{X}} - a\mathbf{Z} = 2\mathbf{Z} - a\mathbf{Z} = (2-a)\mathbf{Z}$. Iteration 2 becomes $\mathbf{X}_2 = 2\mathbf{Z} - (2-a)\mathbf{Z} = a\mathbf{Z}$. Therefore, the relaxed update oscillates between the configurations $(2-a)\mathbf{Z}$ and $a\mathbf{Z}$.

For $a \geq 2$ three situations can occur. First, if a is a multiple of 2, then in $a/2$ iterations the relaxed update yields the solution $\mathbf{X} = \mathbf{0}$. Second, if $a+1$ is a multiple of 2, then in $a/2$ iterations (and all subsequent iterations) the relaxed update yields $\mathbf{Z}$ as a solution, which is in this example the only case leading to the proper solution. Thirdly, if the first two cases do not hold, then after $\text{int}(a)/2$ iterations (where $\text{int}(a)$ takes the integer part of a) the situation is obtained as if had been a chosen as $a - \text{int}(a)$ (which falls in the

interval 0 to 2) and then the oscillation described above starts again. For $a < 0$ a similar situation occurs as for $a \geq 2$.

Of course, $a\mathbf{Z}$ is a special startconfiguration that is unlikely to occur in every day practise. However, we did encounter this type of behavior of the relaxed update without optimal dilation, which resulted in a nonzero norm of the gradient and large Stress values, which shows why an optimal dilation factor is needed when using the relaxed update. Therefore, we shall always use optimal dilation in conjunction with the relaxed update.

4 Comparing the Speed of the Algorithms

To compare the speed of the algorithms two experiments were done. First, the algorithms were compared on a perfect six dimensional Euclidean distance matrix, which has only one local minimum, that is also global. In this case we can genuinely compare the convergence speed. The second experiment uses random data sets, that have different local minima. For the random data sets one cannot guarantee that an algorithm started from the same configuration will yield the same local minimum. Therefore, we applied multiple random starts and analyze the algorithms over the random starts.

All computations were done in Fortran 77 on a PowerPC 603 processor under Macintosh OS. To measure the CPU time accurately there was no output (or input) during the experiments or other programs running in the background. CPU time was measured with a accuracy of 1/60-th of a second. The algorithms that we compare are (and their abbreviation used in the tables):

- the spectral gradient algorithm (SPG);
- the spectral gradient algorithm with optimal dilation (SPG+dil);

- the preconditioned spectral gradient algorithm (P-SPG);
- the preconditioned spectral gradient algorithm with optimal dilation (P-SPG+dil);
- the SMACOF algorithm (SMACOF);
- the SMACOF algorithm with relaxed update and optimal dilation (SMACOF+relax).

To compare the algorithms in a fair manner, we implemented the algorithms into a single program and optimized each algorithm to run as efficiently as possible. The CPU time was measured only over the algorithmic part and the initial and final input and output was excluded.

The tolerance ε for stopping the iterations was set to 10^{-6}. This rather stringent stopping criterion shows the convergence behavior more clearly. Moreover, it may avoid stopping the algorithm too early at a flat part of Stress. With weak tolerance ($\varepsilon = 10^{-2}$) we found occasionally that the SPG algorithm converged to saddle points in small examples (whereas SMACOF did not) but this never occured with stronger tolerance.

4.1 The Euclidean Distance Data

The first data set is a Euclidean distance matrix with $n = 50$ objects in 6 dimensions. We generated 10 different random start configurations and started each of the six algorithms. Table 1 summarizes the CPU seconds and number of iterations by their mean, the minimum, the maximum, and the 25%, 50%, and 75% percentiles.

Place Tables 1 about here.

These results show that using the preconditioner reduces the CPU time for the spectral

gradient algorithm by 15%. Optimal dilation helps to reduce the CPU time by another 15%. Comparing SMACOF and relaxed SMACOF confirms the findings of De Leeuw and Heiser (1980) that the relaxed update is about twice as fast as the plain SMACOF algorithm. SPG is about 7.5 times as fast as SMACOF. A similar result is found if we compare the accelerated versions of the algorithms, that is, compare P-SPG+dil with relaxed SMACOF, where the first is about 6 times faster than the latter.

For one starting configuration the Stress values versus the CPU time are given in Figure 1. The left hand panel is an enlargement of the right hand panel that shows the relation more clearly between SPG and P-SPG with and without dilation. Note that the vertical scale representing the Stress values is a logarithmic scale. The lines in the figure clearly show that the preconditioned spectral gradient with dilation is the fastest algorithm overall, followed by the spectral gradient algorithm with dilation. The minor increase in Stress of relaxed SMACOF near convergence is due to slight numerical instability occuring with very strong convergence criteria. In the first CPU second, the (relaxed) SMACOF algorithm gives somtimes lower Stress values. This means that in the initial stage or if only limited tolerance is required SMACOF can be prefered.

Place Figure 1 about here.

4.2 Random Dissimilarities

Our second set of test data sets is constructed in the same way has those used by Glunt et al. (1994). The dissimilarities are uniformly distributed on the interval [0, 1] and subsequently multiplied by 10. The weights w_{ij} are also uniformly distributed on the

interval [0, 1]. The factors that were varied in this experiment are:

- the six algorithms described above;
- the sample size $n = 25, 50, 100$;
- the dimensionality $p = 2, 3$.

Thus we generated three different data sets (for $n = 25$, for $n = 50$, and for $n = 100$) and applied the algorithms to find a solution in either two or three dimensions. Because these data sets have multiple local minima one cannot assume that the algorithms will end up in the same local minima even if they use the same starting configuration. Therefore, we applied two rational starting strategies. The first strategy uses a starting configuration obtained by classical MDS. The second strategy is multiple random start. This strategy amounts to generating many random starting configurations, apply the algorithm and see where the algorithms stop. We used 100 random starts for each combination of sample size and dimensionality. We report CPU time (Table 2), number of iterations (Table 3), and the proportion of preconditioner steps for the preconditioned spectral gradient algorithms (Table 4).

Place Tables 2, 3, and 4 about here.

First we look at the quality of the solutions, that is, how good or bad are the Stress values found for each of the methods. To establish this we subtracted the average Stress for each combination of n and p to control for these two factors. The average Stress over the 600 runs (with standard deviation within brackets) for each of the methods is for SPG $-.292 \cdot 10^{-3}$ $(3.100 \cdot 10^{-3})$, SPG+dil $-.030 \cdot 10^{-3}$ $(3.160 \cdot 10^{-3})$, P-SPG $-.189 \cdot 10^{-3}$

$(3.232 \cdot 10^{-3})$, P-SPG+dil $-.119 \cdot 10^{-3}$ $(3.116 \cdot 10^{-3})$, SMACOF $.316 \cdot 10^{-3}$ $(3.445 \cdot 10^{-3})$, and SMACOF+relax $.313 \cdot 10^{-3}$ $(3.543 \cdot 10^{-3})$. There is a slight indication that (relaxed) SMACOF yields somewhat higher Stress values, but considering the almost equal and high standard deviations, the effect is neglegible. Thus, in terms of the quality of the local minima found, all methods behave comparably.

In comparison with plain SMACOF, SPG on the average is about 5 to 7 times faster (Table 2). This difference in speed is less pronounced than for the perfect Euclidean data example. Here the relaxed SMACOF is about twice as fast as plain SMACOF. Optimal dilation for (P-)SPG saves another 15% CPU time.

5 Conclusions

We have extended the (preconditioned) spectral gradient algorithm with an optimal dilation factor and propose that such a factor should always be included for the SMACOF algorithm with relaxed update. From our numerical experiments we conclude that the preconditioned spectral gradient is about equally fast as the plain spectral gradient algorithm for non-Euclidean data. The (preconditioned) spectral gradient algorithms become about 10% faster if an optimal dilation factor is included. For the majorizing algorithm SMACOF the optimal dilation has to be included if the relaxed update of De Leeuw and Heiser (1980) is used. The relaxed update is about twice as fast as the plain SMACOF algorithm. Plain SMACOF is about 5 to 10 times slower than plain spectral gradient.

We conclude that the fastest algorithm overall is the spectral gradient algorithm with optimal dilation. For non-Euclidean data we found that preconditioning of the spectral gradient algorithm is not worthwhile. We recommend that optimal dilation always be

applied with the (preconditioned) spectral gradient algorithm. Next in speed comes the majorization algorithm SMACOF with relaxed update and optimal dilation. The slowest algorithm in our experiments was SMACOF.

If preconditioning could be invoked earlier by a different rule than we used without distorting convergence, then preconditioning could become faster than the plain spectral gradient method.

The convergence rates becomes important whenever strong convergence is required. Therefore, our results would have been less pronounced if less strict tolerance was used. Even though the majorization algorithm is slower in general, it still has attractive properties. In the early stages of the iterative process (relaxed) SMACOF sometimes yields lower Stress values than the spectral gradient methods. Also, whenever two points coincide (a situation that did not occur in our experiments) the gradient of Stress becomes undefined and the spectral gradient algorithm may fail but the majorization algorithm remains valid. A final advantage not shared by the spectral gradient method is that constraints can be easily imposed (De Leeuw and Heiser 1980) within the majorization framework.

References

BARZILAI, J., and BORWEIN, J. M. (1988), "Two point step size gradient methods," *IMA J. Numer. Anal.*, *8*, 141-148.

BORG, I., and GROENEN, P.J.F. (1997), *Modern multidimensional scaling: Theory and applications*, New York: Springer.

COMMANDEUR, J.J.F. (1992), *Missing data in the distance approach to Principal Component Analysis*, Research Report No. RR-92-07, Leiden: Department of Data Theory.

DE LEEUW, J. (1977), "Applications of convex analysis to multidimensional scaling," in *Recent developments in statistics*, Eds., J.R. Barra, F. Brodeau, G. Romier, and B. van Cutsem, Amsterdam: North-Holland, 133–145.

DE LEEUW, J., (1984),"Differentiability of Kruskal's Stress at a local minimum," *Psychometrika*, *49*, 111–113.

DE LEEUW, J. (1988), "Convergence of the majorization method for multidimensional scaling," *Journal of Classification*, *5*, 163-180.

DE LEEUW, J. (1994), "Block relaxation algorithms in statistics," in *Information systems and data analysis*, Eds., H.-H. Bock, W. Lenski, and M.M. Richter, Berlin: Springer, 308–324.

DE LEEUW, J., and HEISER, W.J. (1980), "Multidimensional scaling with restrictions on the configuration," in *Multivariate analysis V*, Ed., P.R. Krishnaiah, Amsterdam:

North-Holland, 501–522.

DEFAYS, D. (1978), "A short note on a method of seriation," *British Journal of Mathematical and Statistical Psychology*, *3*, 49–53.

GLUNT, W., HAYDEN, T.L., and RAYDAN, M., (1993), "Molecular conformation from distance matrices," *Journal of Computational Chemistry*, *14*, 114–120 .

GLUNT, W., HAYDEN, T.L., and RAYDAN, M., (1994), "Preconditioners for distance matrix algorithms," *Journal of Computational Chemistry*, *15*, 227–232.

GROENEN, P.J.F. (1993), *The majorization approach to multidimensional scaling: Some problems and extensions*, Leiden: DSWO Press.

GROENEN, P.J.F., DE LEEUW, J., and MATHAR, R. (1996), "Least squares multidimensional scaling with transformed distances," in *Studies in classification, data analysis, and knowledge organization*, Eds., W. Gaul and D. Pfeifer, Berlin, Springer, 177–185.

GROENEN, P.J.F., and HEISER, W.J. (1996), "The tunneling method for global optimization in multidimensional scaling," *Psychometrika*, *61*, 529-550.

GROENEN, P.J.F., MATHAR, R., and HEISER, W.J. (1995), "The majorization approach to multidimensional scaling for Minkowski distances" *Journal of Classification*, *12*, 3–19.

HAVEL, T.F. (1991), "An evaluation of computational strategies for use in the determination of protein structure from distance constraints obtained by Nuclear Magnetic Resonance," *Prog. Biophys. Mol. Biol.*, *46*, 43–78.

HEISER, W.J. (1995), "Convergent computation by iterative majorization: Theory and applications in multidimensional data analysis," in *Recent advances in descriptive multivariate analysis*, Ed., W. J. Krzanowski, Oxford: Oxford University Press, 157–189.

HEISER, W.J., and DE LEEUW, J. (1977) "How to use SMACOF-I," (3rd edition, 1986), Research Report UG-86-02, Leiden: Department of Data Theory.

HUBERT, L.J., and ARABIE, P. (1986), "Unidimensional scaling and combinatorial optimization," in *Multidimensional Data Analysis*, Eds., J. De Leeuw, W.J. Heiser, J. Meulman, and F. Critchley, Leiden: DSWO Press, 181-196.

KRUSKAL, J.B. (1964a), "Multidimensional scaling by optimizing goodness of fit to a non-metric hypothesis," *Psychometrika, 29*, 1-27.

KRUSKAL, J.B. (1964b), "Nonmetric multidimensional scaling: a numerical method," *Psychometrika, 29*, 115-129.

MATHAR, R., and GROENEN, P.J.F., (1991), "Algorithms in convex analysis applied to multidimensional scaling," in *Symbolic-numeric data analysis and learning*, Eds., E. Diday and Y. Lechevallier, Commack, NY: Nova Science Publishers, 45–56.

MEULMAN, J.J. (1986), "A Distance Approach to Nonlinear Multivariate Analysis," DSWO Press, Leiden: The Netherlands.

MEULMAN, J.J. (1992), "The integration of multidimensional scaling and multivariate analysis with optimal scaling," *Psychometrika*, *57*, 539–565.

PLINER, V. (1996), "Metric, unidimensional scaling and global optimization," *Journal of Classification*, *13*, 3–18.

RAYDAN, M. (1997), "The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem," *SIAM J. Opt.*, **7**, 26–33.

Table 1: CPU time and number of iterations needed for the algorithms to reach convergence using 10 random starts for a 6 dimensional Euclidean distance matrix. Reported are the mean, minimum, maximum, and the 25, 50, and 75 percentiles.

algorithm	mean	min	25%	50%	75%	max
CPU time						
SPG	4.0	3.3	3.6	3.8	4.2	5.2
SPG+dil	3.3	3.0	3.0	3.2	3.4	4.0
P-SPG	3.3	2.6	2.6	3.2	3.7	4.4
P-SPG+dil	2.8	2.6	2.6	2.7	2.9	3.6
smacof	30.9	25.4	28.7	30.1	34.7	35.3
smacof+relax	15.8	13.0	14.7	15.4	17.4	18.1
number of iterations						
SPG	126	106	115	120	134	166
SPG+dil	103	92	95	101	107	125
P-SPG	94	70	75	90	106	131
P-SPG+dil	77	69	70	74	80	102
smacof	1050	863	976	1026	1179	1196
smacof+relax	521	431	485	509	574	595

Table 2: CPU time needed for the algorithms to reach convergence using either a classical MDS start configuration or 100 random starts. Of the 100 random starts the mean, minimum, maximum, and the 25, 50, and 75 percentiles are reported.

	classical	100 random starts					
algorithm	scaling	mean	min	25%	50%	75%	max
random data, $n = 25, p = 2$							
SPG	.37	.43	.18	.32	.40	.52	.93
SPG+dil	.30	.41	.22	.32	.37	.50	.73
P-SPG	.34	.49	.20	.35	.42	.55	1.39
P-SPG+dil	.43	.44	.22	.33	.42	.53	1.05
smacof	1.90	2.52	.75	1.32	1.92	2.57	27.04
smacof+relax	.92	1.12	.40	.73	.96	1.23	5.17
random data, $n = 50, p = 2$							
SPG	1.30	3.06	1.47	2.16	2.67	3.63	8.03
SPG+dil	1.35	2.49	1.28	1.88	2.24	2.86	5.08
P-SPG	1.45	3.06	1.60	2.28	2.76	3.69	6.87
P-SPG+dil	1.49	2.75	1.43	2.13	2.55	3.26	6.15
smacof	6.61	15.80	6.83	10.46	13.48	16.95	62.16
smacof+relax	3.47	8.47	3.14	5.57	6.91	9.45	35.66
random data, $n = 100, p = 2$							
SPG	17.1	20.9	8.1	15.0	19.8	24.5	45.5
SPG+dil	12.3	17.0	7.2	13.0	15.6	20.2	44.2
P-SPG	15.1	21.2	10.2	15.4	20.0	24.1	45.1
P-SPG+dil	18.6	18.9	7.8	14.4	17.1	23.7	41.7
smacof	101.5	108.2	38.0	84.2	100.8	132.0	241.5
smacof+relax	60.6	59.0	22.2	42.2	56.3	72.5	141.5
random data, $n = 25, p = 3$							
SPG	.79	.78	.40	.57	.72	.92	1.74
SPG+dil	.39	.73	.37	.54	.66	.89	1.72
P-SPG	.73	.81	.38	.58	.74	.93	2.15
P-SPG+dil	.37	.78	.37	.55	.67	.96	2.09
smacof	3.20	5.93	2.30	4.18	4.97	6.53	26.08
smacof+relax	1.72	3.18	1.03	2.01	2.53	3.86	13.72
random data, $n = 50, p = 3$							
SPG	6.60	4.42	1.90	3.23	3.97	4.96	14.82
SPG+dil	2.63	3.89	1.78	2.89	3.56	4.56	8.25
P-SPG	3.42	4.37	1.83	3.22	4.05	5.21	15.82
P-SPG+dil	2.45	4.26	1.95	3.24	3.87	4.89	9.00
smacof	17.35	40.07	12.50	25.61	33.96	43.40	154.33
smacof+relax	9.46	19.96	6.48	12.59	17.07	22.90	72.91
random data, $n = 100, p = 3$							
SPG	27.1	32.8	9.6	23.5	31.2	40.0	88.7
SPG+dil	15.6	29.2	10.1	22.3	26.4	33.0	66.3
P-SPG	45.5	32.9	8.6	23.4	30.9	39.0	84.7
P-SPG+dil	20.5	29.5	10.0	21.0	26.7	35.5	69.7
smacof	288.0	281.0	69.8	192.6	257.9	345.5	838.4
smacof+relax	147.7	153.6	35.0	97.4	131.1	184.7	578.9

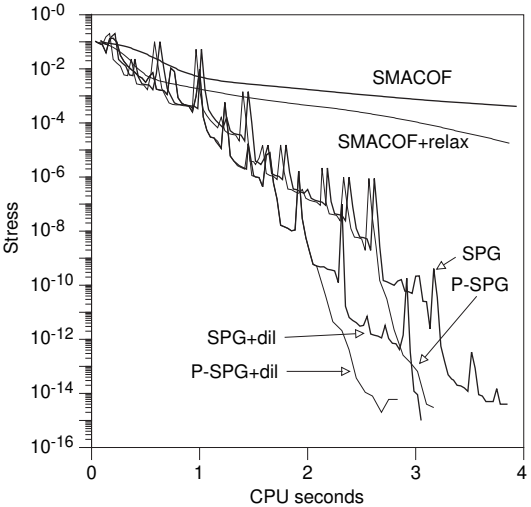
Table 3: Number of iterations needed for the algorithms to reach convergence using either a classical MDS start configuration or 100 random starts. Of the 100 random starts the mean, minimum, maximum, and the 25, 50, and 75 percentiles are reported.

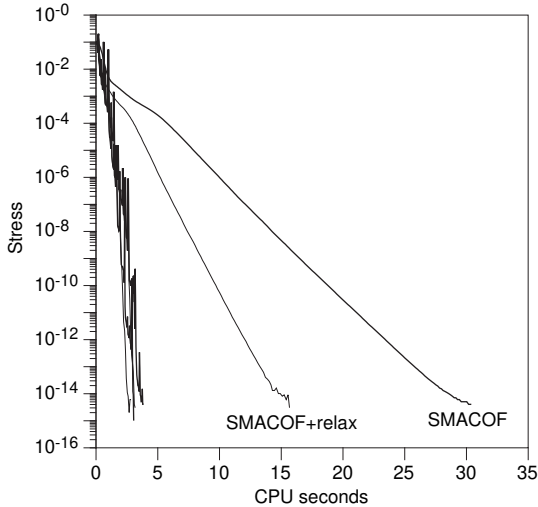
	classical	100 random starts					
algorithm	scaling	mean	min	25%	50%	75%	max
random data, $n = 25, p = 2$							
SPG	64	91	40	66	85	109	196
SPG+dil	61	84	44	63	76	102	154
P-SPG	61	87	33	59	78	103	259
P-SPG+dil	78	77	34	56	71	91	196
smacof	426	529	154	277	399	525	5675
smacof+relax	179	229	82	146	199	253	1073
random data, $n = 50, p = 2$							
SPG	63	153	73	109	134	182	403
SPG+dil	66	122	63	92	110	140	249
P-SPG	54	134	67	99	121	164	324
P-SPG+dil	51	116	58	87	109	141	261
smacof	330	770	333	511	652	833	3099
smacof+relax	166	398	146	261	322	444	1685
random data, $n = 100, p = 2$							
SPG	213	262	102	188	248	308	572
SPG+dil	149	208	88	159	190	246	539
P-SPG	162	238	103	165	225	273	529
P-SPG+dil	179	202	75	149	184	251	489
smacof	1224	1312	458	1022	1222	1601	2936
smacof+relax	708	692	258	494	661	852	1670
random data, $n = 25, p = 3$							
SPG	133	144	73	105	132	170	318
SPG+dil	70	132	66	97	119	160	308
P-SPG	110	124	57	85	112	142	329
P-SPG+dil	57	117	52	81	103	148	343
smacof	562	1063	384	734	889	1168	4721
smacof+relax	282	557	200	357	451	660	2382
random data, $n = 50, p = 3$							
SPG	287	193	83	141	174	214	650
SPG+dil	113	166	76	123	152	195	353
P-SPG	122	164	64	119	146	196	653
P-SPG+dil	85	153	65	116	142	178	344
smacof	718	1655	516	1056	1402	1797	6382
smacof+relax	379	803	262	507	686	924	2920
random data, $n = 100, p = 3$							
SPG	297	361	105	258	343	439	975
SPG+dil	167	312	108	238	283	354	710
P-SPG	474	325	74	219	299	396	895
P-SPG+dil	170	280	80	187	249	336	717
smacof	2963	2839	693	1937	2598	3495	8497
smacof+relax	1477	1534	347	971	1308	1845	5800

Table 4: Percentage of preconditioner steps using either a classical MDS start configuration or 100 random starts. Of the 100 random starts the mean, minimum, maximum, and the 25, 50, and 75 percentiles are reported

	classical	100 random starts					
algorithm	scaling	mean	min	25%	50%	75%	max
random data, $n = 25, p = 2$							
P-SPG	26.2	27.6	8.8	19.8	26.1	33.8	58.6
P-SPG+dil	20.5	27.9	9.8	19.7	25.2	37.0	54.8
random data, $n = 50, p = 2$							
P-SPG	51.9	23.6	6.4	17.9	22.7	27.6	50.6
P-SPG+dil	64.7	25.4	7.2	19.5	24.9	31.5	48.3
random data, $n = 100, p = 2$							
P-SPG	27.7	19.8	4.6	13.0	17.2	25.4	42.7
P-SPG+dil	40.8	23.8	6.3	15.9	21.6	30.1	65.4
random data, $n = 25, p = 3$							
P-SPG	24.5	25.5	6.7	19.5	24.3	30.1	63.5
P-SPG+dil	21.1	26.4	10.5	18.1	25.1	33.3	65.9
random data, $n = 50, p = 3$							
P-SPG	30.3	24.3	8.1	16.8	22.8	30.5	51.4
P-SPG+dil	34.1	27.2	11.5	21.2	26.8	32.3	53.4
random data, $n = 100, p = 3$							
P-SPG	7.4	17.6	4.8	10.7	16.0	22.8	40.0
P-SPG+dil	40.0	20.7	5.6	13.1	19.7	26.1	51.7

Figure 1: The Stress values versus the CPU time for the six algorithms on data that are starting 6 dimensional Euclidean distances. The same random start configuration was used for each algorithm. The left hand panel and right hand panel are the same up to a difference in scale of the CPU time.


Stress
CPU seconds
SMACOF
SMACOF+relax
SPG
P-SPG
SPG+dil
P-SPG+dil


Stress
CPU seconds
SMACOF+relax
SMACOF