RR-90-03

THE TUNNELING METHOD APPLIED TO METRIC MULTIDIMENSIONAL SCALING: PROGRESS REPORT 1

Patrick Groenen

Department of Data Theory

University of Leiden

1. Introduction.

In this progress report we will focus on one aspect of metric multidimensional scaling. One of the problems of this technique is that the solution given is not necessarily the best solution. Other configurations may exist that have a better fit. Therefore we intend to develop an algorithm within the SMACOF framework that finds the best solution within a reasonable amount of time.

Finding the best solution corresponds in mathematical terms to finding the *global minimum* of the stress function. Several *local* search procedures are well known, like the gradient method proposed by Kruskal (1964a, 1964b) and the SMACOF algorithm of De Leeuw (1977) and De Leeuw and Heiser (1980). Global optimization methods usually consist of a local search phase and a global search phase. Here we will use the SMACOF algorithm during the local search. For the global search we will develop the tunneling approach of Montalvo (1979), Gomez and Levy (1982), Levy and Gomez (1984), still remaining within the SMACOF theory.

A short summary of global optimization techniques will be given here. A recent overview of developments in this field is given by Törn and Žilinskas (1989). They distinguish six types of global optimization algorithms. The first type is formed by *covering* methods that exclude subregions that do not contain the global minimum. Klaassen (1989) has applied this method to the stress function of multidimensional scaling. Unfortunately his method was only successful for very small problems. Another covering method was used by Hubert and Arabie (1986). They used dynamic programming successfully for one-dimensional scaling. The second type is formed by *random search* methods like pure random search, singlestart and multistart. Random configurations are generated for starting a local search procedure that yield a local minimum and the lowest stress function value is considered to be the global minimum. The third type is formed by *clustering* methods. These methods refine the random search methods in that configurations leading to the same local minimum are evaluated only once (Rinnooy Kan and Timmer, 1987). An example of this method is the multi level single linkage algorithm of Timmer (1984). The fourth type is formed by methods *approximating the level sets*. The idea is to find a solution for which a volume measure between function surface and all solutions with the same function value equals zero. The fifth type is formed by methods *approximating the objective function*. Here a theory is developed which is based on a statistical model of the function. The unknown function values are treated as random variables. This class of methods seems very efficient for oscillating functions of one parameter only and is also applied to functions that are expensive to evaluate. For one-dimensional scaling this method was used by De Soete, Hubert and Arabie (1988). However, the implementation of simulated annealing was not satisfying. The last type is formed by *generalized descent* methods, where the function is modified to guarantee a lower minimum. Trajectory methods modify the differential equation describing the local descent trajectory. Penalty methods modify the function itself to prevent returning to local minima found in previous iterations.

The tunneling method, that we will use here to find the global minimum of the stress function, is a penalty method. It can be described by the following analogy. Suppose we wish to find the lowest spot in a selected area in the Alps. First we pour some water and see where is stops. This is the local search. From this point we dig tunnels horizontally until we come out of the mountain, the so called global search. Then we pour water again and dig tunnels again. If we stay underground for a long time while digging the tunnel, we conclude that the last spot was in fact the lowest place in the area. An important and attractive feature of the tunneling algorithm is that successive spots are always lower.

In section 2 we will reformulate the problem in mathematical terms and present the general methods we use to solve the problem. Section 3 applies these methods to our problem. Some basic aspects are investigated empirically in section 4 and a discussion, conclusions and directions for future research is given in section 5.

2. The MDS minimization problem.

In this section we will present the definition of the problem in mathematical terms and present the mathematical methods we need to solve our problem.

2.1. Definition of the problem.

Suppose we have the dissimilarities of all the pairs of the n stimuli. In metric multidimensional scaling we want to find p coordinates $\mathbf{x}_i$ for each stimulus in such a way that the distance between every pair of stimuli i, j are as equal as possible to the dissimilarity. In mathematical terms we want to minimize the stress function

$$\sigma(\mathbf{X}) = \sqrt{1/2\Sigma_i\Sigma_j(\delta_{ij} - \mathrm{d}_{ij}(\mathbf{X}))^2} \quad (1).$$

over $\mathbf{X}$, where δ_{ij} is the dissimilarity between stimuli i and j, $\mathbf{X}$ is an n by p matrix of coordinates and $\mathrm{d}_{ij}(\mathbf{X})$ the Euclidean distance between two stimuli. De Leeuw (1977) and De Leeuw and Heiser (1980) have given an convergent algorithm for minimizing (1). Unfortunately, the algorithm is convergent to a local minimum, not necessarily to a global minimum. Mathar (1989) notices that local minima are more likely to occur when the dissimilarities are far from Euclidean. It seems relevant to develop an algorithm for finding a global minimum.

Heiser and De Leeuw (1979) note that their algorithm is very likely to find a local minimum in one-dimensional scaling since it is a combinatorial problem. Defays (1978) proposed an algorithm for one dimensional scaling. The dynamic programming approach of Hubert and Arabie (1986) guarantees to find a global minimum for moderate size problems (up to 20 stimuli). Simulated annealing used by De Soete, Hubert and Arabie (1988) was rather disappointing. Contrary to theoretical expectations the method didn't perform better than the more traditional pairwise interchange strategy. In this report we will develop a method which can be used both in the one-dimensional case and the multidimensional case.

As stated above the tunneling algorithm tries to find a global minimum in two phases. First we find a local minimum using SMACOF theory (see section 2.2). Second we try to tunnel to another configuration with exactly the same STRESS. Figure 1 shows this situation. In the first phase a local minimum $\sigma(\mathbf{X}^*)$ is found at $\mathbf{X}^*$ and we wish to tunnel to configuration $\mathbf{X}^o$ with STRESS $\sigma(\mathbf{X}^*)$. Once arrived at $\mathbf{X}^o$ we go to phase one again to find the next local minimum.

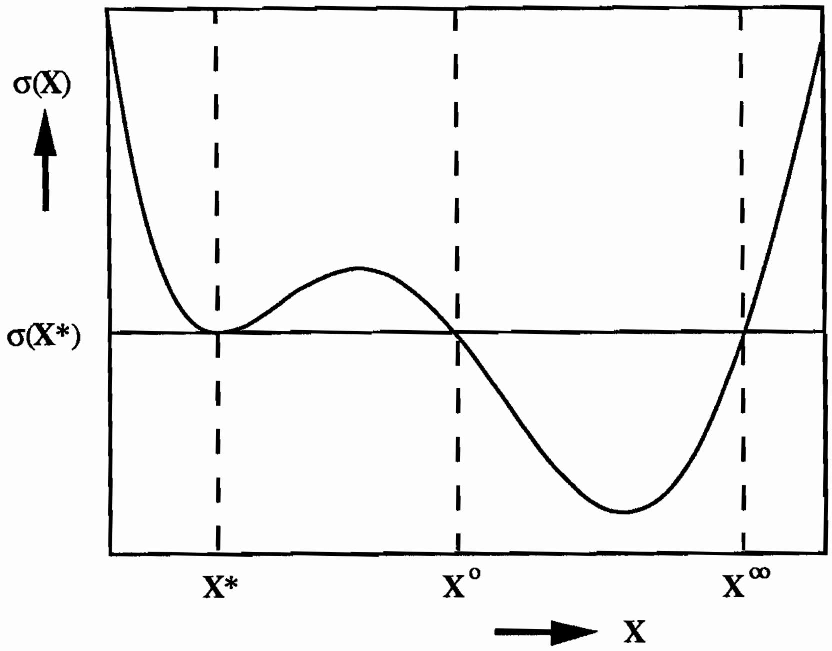

Figure 1. A simplified example of the stress function with a local minimum at **X*** and another configuration $\mathbf{X}^o$ or $\mathbf{X}^{oo}$ with the same STRESS.

In the second phase we want to find another configuration $\mathbf{X}^o$ with $\sigma(\mathbf{X}^o) - \sigma(\mathbf{X}^*) = 0$. This may be achieved by minimizing $(\sigma(\mathbf{X}) - \sigma(\mathbf{X}^*))^2$. However, we want to avoid the solution $\mathbf{X}^o = \mathbf{X}^*$. This can be achieved by dividing by $\text{tr}(\mathbf{X} - \mathbf{X}^*)'(\mathbf{X} - \mathbf{X}^*)$. In formula the tunneling phase comes to the minimization of the tunneling function

$$T(\mathbf{X}) = \frac{(\sigma(\mathbf{X}) - \sigma(\mathbf{X}^*))^2}{\text{tr}(\mathbf{X} - \mathbf{X}^*)'(\mathbf{X} - \mathbf{X}^*)} \,. \tag{2}$$

The graphical display of the tunneling function in Figure 2 can be derived from Figure 1. First we note that the tunneling function has two zero points, $\mathbf{X}^o$ and $\mathbf{X}^{oo}$, that have stress values equal to the stress value at local minimum **X***. The second feature of the tunneling function is the pole at **X*** to stay away from the solution **X***. Thirdly, we know that T(**X**) is always positive or zero and reaches zero only if the stress value of **X** equals $\sigma(\mathbf{X}^*)$.

Three problems arise when minimizing (2). The first is how to minimize a function that consists of a fraction. This will tackled in section 2.3 by making use of fractional programming (Dinkelbach, 1967). The second problem is that the numerator of T(**X**) is a rather complicated function. Therefore we will develop a more simpler form in section 3.1 by majorizing the numerator. The general idea of majorization is explained in section 2.2. The third problem is how to find a reasonable starting configuration, since the last local minimum is of no use; this matter will be discussed in section 3.3.

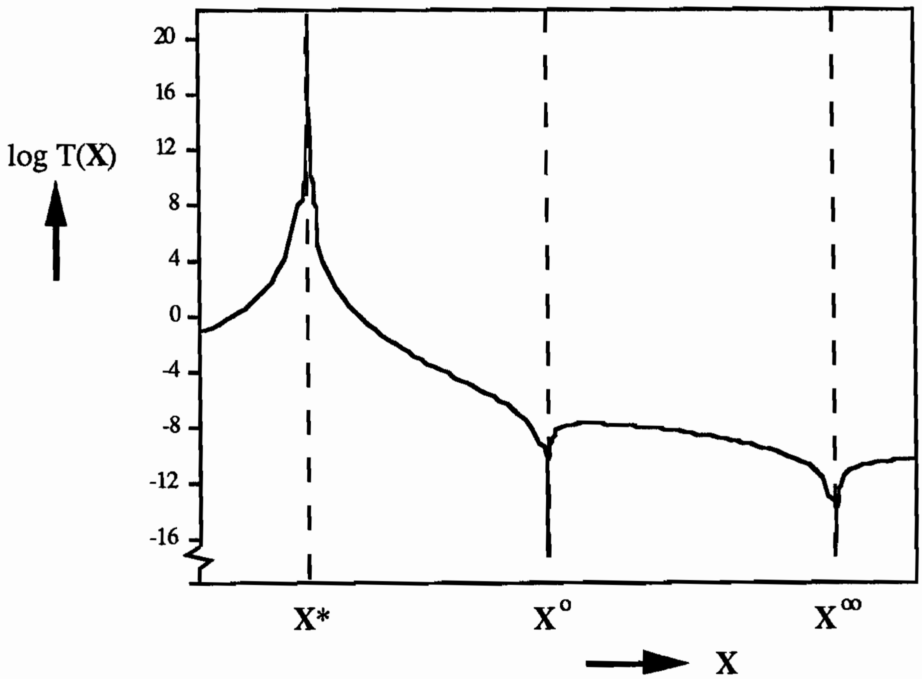

Figure 2. A simplified example of the tunneling function with a pole at **X*** and a minimum at $\mathbf{X}^{\mathbf{o}}$.

2.2. Majorization.

Majorization is a method used to simplify the minimization of complicated functions. It will be illustrated here by majorization of the stress function (1). The central idea is to replace the original function σ(X) by an auxiliary function μ(X,Y) at the supporting point Y. The majorizing function μ(Y,Y) is equal to σ(Y) at Y and at other places larger than or equal to the original function. Majorization in multidimensional scaling has been applied in a variety of settings by among others De Leeuw (1977), De Leeuw and Heiser (1980) and Meulman (1986). As an example we show how to the majorize the stress function σ(X) following the SMACOF theory.

An alternative way to write stress function (1) is

$$\sigma^2(\mathrm{X}) = 1/2\Sigma_i\Sigma_j\,\delta^2_{ij} + n/2\mathrm{trX'X} - \mathrm{trX'B(X)X}\ , \qquad (3)$$

$$\mathrm{B(X)} = \mathrm{B^*(X)} - \mathrm{B^o(X)}\ , \qquad (4)$$

with $\mathrm{B^*(X)}$ the diagonal matrix with elements $\mathbf{u}'\mathrm{B^o(X)}$ on the diagonal and $\mathrm{B^o(X)}$ a matrix with off diagonal elements

$$\mathrm{b}^{\mathrm{o}}_{ij}(\mathrm{X}) = \frac{\delta_{ij}}{\mathrm{d}_{ij}(\mathrm{X})} \qquad \text{if } i \neq j \text{ and } \mathrm{d}_{ij}(\mathrm{X}) \neq 0,$$

$$\mathrm{b}^{\mathrm{o}}_{ij}(\mathrm{X}) = 0 \qquad \text{otherwise.} \qquad (5)$$

Without loss of generality we assume in the following that the sum of squares of the dissimilarities is unity. The complicated part of 3 resides in the last term and is caused by the distance function $d_{ij}(\mathbf{X})$. This last term can be majorized by using the Cauchy-Schwartz inequality in the following form

$$\begin{aligned} d_{ij}(\mathbf{X})d_{ij}(\mathbf{Y}) &= \sqrt{(\mathbf{x}_i - \mathbf{x}_j)'(\mathbf{x}_i - \mathbf{x}_j)}\sqrt{(\mathbf{y}_i - \mathbf{y}_j)'(\mathbf{y}_i - \mathbf{y}_j)} \geq (\mathbf{x}_i - \mathbf{x}_j)'(\mathbf{y}_i - \mathbf{y}_j) \\ d_{ij}(\mathbf{X}) &\geq \frac{(\mathbf{x}_i - \mathbf{x}_j)'(\mathbf{y}_i - \mathbf{y}_j)}{d_{ij}(\mathbf{Y})} . \end{aligned} \tag{6}$$

Inserting the right side of (6) in (3) yields after appropriate manipulations

$$\sigma^2(\mathbf{X}) \leq \mu^2(\mathbf{X},\mathbf{Y}) = 1/2 + n/2\mathrm{tr}\mathbf{X}'\mathbf{X} - \mathrm{tr}\mathbf{X}'\mathrm{B}(\mathbf{Y})\mathbf{Y} \ . \tag{7}$$

The majorizing function $\mu^2(\mathbf{X},\mathbf{Y})$ is a quadratic function in $\mathbf{X}$ and has its minimum at

$$\mathbb{X} = n^{-1}\mathrm{B}(\mathbf{Y})\mathbf{Y} \ . \tag{8}$$

De Leeuw and Heiser (1980) call $\mathbb{X}$ the Guttman transform. The chain $\sigma^2(\mathbb{X}) \leq \mu^2(\mathbb{X},\mathbf{Y}) \leq \mu^2(\mathbf{Y},\mathbf{Y}) = \sigma^2(\mathbf{Y})$ shows that the stress function can always be decreased by minimizing the majorizing function with respect to the supporting point $\mathbf{Y}$. The SMACOF algorithm amounts to repeatedly computing the Guttman transform of the previous configuration.

2.3. **Fractional programming.**

The tunneling function (1) is a fraction of two functions. The fractional programming algorithm (Dinkelbach, 1967) can be used to minimize a ratio of two functions (see Heiser, 1981 p. 219). The only assumption made is that the denominator is greater than zero for each feasible $\mathbf{X}$. This means that $\mathbf{X}^*$ is not considered to be a feasible $\mathbf{X}$. We now introduce a concave function of the real valued parameter q

$$\mathrm{F}(\mathbf{X},q) = (\sigma(\mathbf{X}) - \sigma(\mathbf{X}^*))^2 - q\mathrm{tr}(\mathbf{X} - \mathbf{X}^*)'(\mathbf{X} - \mathbf{X}^*) \ . \tag{9}$$

The idea is that we want to find a q^* for which $\mathrm{F}(\mathbf{X},q^*) = 0$; this is a relatively simple problem because of the concavity of $\mathrm{F}(\mathbf{X},q)$. Furthermore, the minimum of $\mathrm{F}(\mathbf{X},q^*)$ over $\mathbf{X}$ is attained at the same point where the original function is minimized (Dinkelbach, 1967, p. 494).
The fractional programming algorithm is

1. $q^+ \leftarrow q^0$
2. $\mathbf{X}^+ \leftarrow \mathrm{argmin}\ \mathrm{F}(\mathbf{X},q^+)$ for fixed q^+
3. If $\mathrm{F}(\mathbf{X}^+,q^+) > \omega$ then *stop*
4. $q^+ \leftarrow \dfrac{(\sigma(\mathbf{X}^+) - \sigma(\mathbf{X}^*))^2}{\mathrm{tr}(\mathbf{X}^+ - \mathbf{X}^*)'(\mathbf{X}^+ - \mathbf{X}^*)}$
5. go to 2

where ω is a small negative value. The algorithm should lead us to a minimum of our tunneling function T(X). The sequence of F(X,q) is approaching zero from below.

We must be careful with the starting value of q. Initializing q with zero would cause $\mathbf{X}^+$ in step 2 to be the minimum of $(\sigma(\mathbf{X}) - \sigma(\mathbf{X}^*))^2$. Note that for large q the last term of (9) gains importance in step 2.

3. A tunneling method for minimizing STRESS.

Here we will develop a method for minimizing the tunneling function and propose an algorithm. In section 3.3 some possible starting configurations for the tunneling algorithm are given.

3.1. Majorization of the numerator.

In step 2 of the fractional programming algorithm we have to find the minimum of F(X,q) for a fixed q. The second part (derived from the denominator) is a quadratic function in X and does not give much problems. The first part (derived from the numerator), however, is rather complex. It can be written as

$$\begin{aligned}(\sigma(\mathbf{X}) - \sigma(\mathbf{X}^*))^2 &= [\sigma(\mathbf{X}^*)]^2 + [\sigma(\mathbf{X})]^2 - 2\sigma(\mathbf{X}^*)\sigma(\mathbf{X}) \\ &= [\sigma(\mathbf{X}^*)]^2 + [\sigma(\mathbf{X})]^2 - 2\sigma(\mathbf{X}^*)\sqrt{1/2\Sigma_i\Sigma_j(\delta_{ij} - d_{ij}(\mathbf{X}))^2}. \qquad (10)\end{aligned}$$

The first term on the right hand side $[\sigma(\mathbf{X}^*)]^2$ is a constant and $[\sigma(\mathbf{X})]^2$ can be majorized using the SMACOF theory. The extra complication of (10) is the last term $-2\sigma(\mathbf{X}^*)\sigma(\mathbf{X})$. One important step in majorizing this term is to realize that $\sigma(\mathbf{X})$ is the distance between the vector with dissimilarities δ_{ij} and the vector with elements $d_{ij}(\mathbf{X})$. Applying the Cauchy-Schwartz inequality on the vector of residuals gives

$$\begin{aligned}\sigma(\mathbf{X})\sigma(\mathbf{Y}) &= \sqrt{1/2\Sigma_i\Sigma_j(\delta_{ij} - d_{ij}(\mathbf{X}))^2}\sqrt{1/2\Sigma_i\Sigma_j(\delta_{ij} - d_{ij}(\mathbf{Y}))^2} \\ \sigma(\mathbf{X})\sigma(\mathbf{Y}) &\geq 1/2\Sigma_i\Sigma_j(\delta_{ij} - d_{ij}(\mathbf{X}))(\delta_{ij} - d_{ij}(\mathbf{Y}))\,. \qquad (11)\end{aligned}$$

Multiplying both sides with $-2\sigma(\mathbf{X}^*)$ and dividing by $\sigma(\mathbf{Y})$ yields

$$-2\sigma(\mathbf{X}^*)\sigma(\mathbf{X}) \leq -\frac{\sigma(\mathbf{X}^*)}{\sigma(\mathbf{Y})}\Sigma_i\Sigma_j[\delta^2_{ij} - \delta_{ij}d_{ij}(\mathbf{Y})] - \frac{\sigma(\mathbf{X}^*)}{\sigma(\mathbf{Y})}\Sigma_i\Sigma_j[d_{ij}(\mathbf{Y})d_{ij}(\mathbf{X}) - \delta_{ij}d_{ij}(\mathbf{X})]. \qquad (12)$$

Let C contain each term that is not a function of X. Then (11) can be majorized as follows:

$$\begin{aligned}(\sigma(\mathbf{X}) - \sigma(\mathbf{X}^*))^2 &\leq C + 1/2\Sigma_i\Sigma_j d^2_{ij}(\mathbf{X}) - \Sigma_i\Sigma_j\left[(1 - \frac{\sigma(\mathbf{X}^*)}{\sigma(\mathbf{Y})})\delta_{ij} + \frac{\sigma(\mathbf{X}^*)}{\sigma(\mathbf{Y})}d_{ij}(\mathbf{Y})\right]d_{ij}(\mathbf{X}) \\ (\sigma(\mathbf{X}) - \sigma(\mathbf{X}^*))^2 &\leq C + n/2\mathrm{tr}\mathbf{X}'\mathbf{X} - \mathrm{tr}\mathbf{X}'\tilde{\mathbf{B}}(\mathbf{Y},\mathbf{X})\mathbf{X} \qquad (13)\end{aligned}$$

where

$$\mathrm{C} \quad = [\sigma(\mathbf{X}^*)]^2 + 1/2 - \frac{\sigma(\mathbf{X}^*)}{2\sigma(\mathbf{Y})} + \frac{\sigma(\mathbf{X}^*)}{\sigma(\mathbf{Y})}\Sigma_i\Sigma_j \mathrm{d}_{ij}(\mathbf{Y})\delta_{ij}\,, \tag{14}$$

$$\tilde{\mathrm{B}}(\mathbf{Y},\mathbf{X}) = \tilde{\mathrm{B}}^*(\mathbf{Y},\mathbf{X}) - \tilde{\mathrm{B}}^{\mathrm{o}}(\mathbf{Y},\mathbf{X}) \tag{15}$$

with $\tilde{\mathrm{B}}^*(\mathbf{Y},\mathbf{X})$ the diagonal matrix with elements $\mathbf{u}'\tilde{\mathrm{B}}^{\mathrm{o}}(\mathbf{Y},\mathbf{X})$ and $\tilde{\mathrm{B}}^{\mathrm{o}}(\mathbf{Y},\mathbf{X})$ with off diagonal elements

$$\tilde{\mathrm{b}}^{\mathrm{o}}_{ij}(\mathbf{Y},\mathbf{X}) = \frac{(1 - \frac{\sigma(\mathbf{X}^*)}{\sigma(\mathbf{Y})})\delta_{ij} + \frac{\sigma(\mathbf{X}^*)}{\sigma(\mathbf{Y})}\mathrm{d}_{ij}(\mathbf{Y})}{\mathrm{d}_{ij}(\mathbf{X})}\,, \qquad \text{if } i \neq j \text{ and } \mathrm{d}_{ij}(\mathbf{X}) \neq 0,$$

$$\tilde{\mathrm{b}}^{\mathrm{o}}_{ij}(\mathbf{Y},\mathbf{X}) = 0 \qquad \text{otherwise.} \tag{16}$$

We can simplify (13) to a quadratic function by majorizing it again according to the SMACOF theory. However, this result is only valid if the numerator of $\tilde{\mathrm{b}}^{\mathrm{o}}_{ij}(\mathbf{Y},\mathbf{X})$ is non-negative for all combinations ij. This factor becomes negative only if $\sigma(\mathbf{X}^*)/\sigma(\mathbf{Y})>1$, or $\sigma(\mathbf{Y})<\sigma(\mathbf{X}^*)$. Suppose that **Y** equals the previous **X** in the iterative process. Now it is easy to see that the factor can be negative only if the previous configuration has a lower STRESS than **X***. In that case we immediately stop since we have found another configuration with a lower STRESS than **X*** and thus have reached our primary goal.

Majorizing the right side of inequality (13) using the Cauchy-Schwartz inequality yields a term $-\mathrm{tr}\mathbf{X}'\tilde{\mathrm{B}}(\mathbf{Y},\mathbf{Z})\mathbf{Z}$. This result is valid for every **Z**, including **Z** = **Y**. Therefore it is true that

$$(\sigma(\mathbf{X}) - \sigma(\mathbf{X}^*))^2 \leq \mathrm{C} + n/2\mathrm{tr}\mathbf{X}'\mathbf{X} - \mathrm{tr}\mathbf{X}'\tilde{\mathrm{B}}(\mathbf{Y},\mathbf{Y})\mathbf{Y}\;. \tag{17}$$

From (16) we can derive

$$\tilde{\mathrm{b}}^{\mathrm{o}}_{ij}(\mathbf{Y},\mathbf{Y}) = \left[1 - \frac{\sigma(\mathbf{X}^*)}{\sigma(\mathbf{Y})}\right]\frac{\delta_{ij}}{\mathrm{d}_{ij}(\mathbf{Y})} + \frac{\sigma(\mathbf{X}^*)}{\sigma(\mathbf{Y})} = \left[1 - \frac{\sigma(\mathbf{X}^*)}{\sigma(\mathbf{Y})}\right]\mathrm{b}^{\mathrm{o}}_{ij}(\mathbf{Y}) + \frac{\sigma(\mathbf{X}^*)}{\sigma(\mathbf{Y})} \tag{18}$$

$$(\sigma(\mathbf{X}) - \sigma(\mathbf{X}^*))^2 \leq \mathrm{C} + n/2\mathrm{tr}\mathbf{X}'\mathbf{X} - \left[1 - \frac{\sigma(\mathbf{X}^*)}{\sigma(\mathbf{Y})}\right]\mathrm{tr}\mathbf{X}'\mathrm{B}(\mathbf{Y})\mathbf{Y} - \frac{n\,\sigma(\mathbf{X}^*)}{\sigma(\mathbf{Y})}\mathrm{tr}\mathbf{X}'\mathbf{Y}. \tag{19}$$

We have achieved a simple function quadratic in **X** by doubly majorizing the numerator.

3.2. Merging the majorization result and fractional programming.

In order to apply the fractional programming algorithm we have to be certain that all assumptions are fulfilled. The positivity assumption of the denominator is violated only if **X** = **X***. Thus we must either define **X*** not to be a feasible **X** or to add a small positive constant to the denominator. We choose here for the first option.

The second step of the algorithm requires that a $\mathbf{X}^+$ is found that minimizes $\mathrm{F}(\mathbf{X},q^+)$ for q^+ fixed. Now we can use the majorization result of the previous section.

$$F(\mathbf{X},q^+) \le C + n/2\mathrm{tr}\mathbf{X}'\mathbf{X} - (1 - \frac{\sigma(\mathbf{X}^*)}{\sigma(\mathbf{Y})})\mathrm{tr}\mathbf{X}'\mathbf{B}(\mathbf{Y})\mathbf{Y} - \frac{n\,\sigma(\mathbf{X}^*)}{\sigma(\mathbf{Y})}\mathrm{tr}\mathbf{X}'\mathbf{Y}$$

$$- \; q^+[\mathrm{tr}\mathbf{X}'\mathbf{X} + \mathrm{tr}\mathbf{X}^{*\prime}\mathbf{X}^* - 2\mathrm{tr}\mathbf{X}'\mathbf{X}^*]$$

$$F(\mathbf{X},q^+) \le C - q^+\mathrm{tr}\mathbf{X}^{*\prime}\mathbf{X}^* + (n/2 - q^+)\mathrm{tr}\mathbf{X}'\mathbf{X}$$

$$- \; \mathrm{tr}\mathbf{X}'\left[(1 - \frac{\sigma(\mathbf{X}^*)}{\sigma(\mathbf{Y})})\mathbf{B}(\mathbf{Y})\mathbf{Y} + \frac{n\,\sigma(\mathbf{X}^*)}{\sigma(\mathbf{Y})}\mathbf{Y} - q^+\mathbf{X}^*\right] \qquad (20)$$

Since (20) is a quadratic function in **X** the following algorithm can be used.

2a. $\mathbb{X} \leftarrow \frac{1}{n - 2q^+}\left[(1 - \frac{\sigma(\mathbf{X}^*)}{\sigma(\mathbf{X}^+)})\mathbf{B}(\mathbf{X}^+)\mathbf{X}^+ + \frac{n\,\sigma(\mathbf{X}^*)}{\sigma(\mathbf{X}^+)}\mathbf{X}^+ - q^+\mathbf{X}^*\right]$

2b. if $\sigma(\mathbf{Y}) < \sigma(\mathbf{X}^*)$ then *stop*

2c. if tr $(\mathbf{X}^+ - \mathbb{X})'(\mathbf{X}^+ - \mathbb{X}) < \omega$ then *go to* 3

2d. $\mathbf{X}^+ \leftarrow \mathbb{X}$

2e. *go to* 2a .

3.3. Choosing a start configuration of the tunneling phase.

Since **X*** is not a feasible **X** of the tunneling function we have to choose a start configuration different from **X***. The most straightforward option is to use a small pertubation of **X***, like $\mathbf{X}^* + \varepsilon$ with ε normally distributed with mean zero and a small variance. Another option is using one of the previous configurations before reaching **X***, for example the last one or the configuration 5 iterations before the last one. The third option is to select the $p + 1$ worst fitting points, apply the SMACOF algorithm (that will lead to a perfect solution of the selected points) and replace their coordinates in **X*** to achieve the start configuration. These configurations will have to be tested empirically on their merits.

4. Simulation results.

In this section we will look at the strength of the pole. The pole of the tunneling function (2) should be strong enough in order to cancel out the local minimum at **X***. If it is too weak the tunneling function will asymptotically go to zero as **X** approaches **X***. Then minimization of the tunneling function might yield **X*** as minimum, being the solution we wanted to avoid. In other words, we have to check whether T(**X**) actually increases when we force **X** to approach **X***.

It can be seen quite easily from our algorithm that a weak pole will lead to **X***. Suppose that **X** is near **X*** and that the pole is too weak. In step 4 of the algorithm in section 2.3 q^+ is defined as T(**X**). Because of the pole being too weak q^+ approaches zero. The importance of q^+ can be seen in section 3.2. There step 2a shows that when q^+ approaches zero this step amounts to minimizing $(\sigma(\mathbf{X}) - \sigma(\mathbf{X}^*))^2$. This will lead to the minimum of **X***. Thus we must make sure that when **X** approaches **X*** T(**X**) gets large.

In order to check this we used data about the Mani collection of archaeological deposits reported by Hubert and Arabie (1986) who took it from Robinson (1951). We performed a two dimensional scaling analysis using the metric SMACOF algorithm. We set the convergence criterion strong (10^{-10}) so that we have an accurate estimate of X^*. After 44 iterations convergence was reached with STRESS 0.0809031525. Then SMACOF was performed again using the same start configuration. Since X gets nearer to X^* every iteration we computed T(X) for every iteration. We also computed two other fractions. Let $\nu(X) = (\sigma(X) - \sigma(X^*))^2$ and $\tau(X) = \text{tr}(X - X^*)'(X - X^*)$. Furthermore, the values of $\nu(X)/\tau(X)^2$ and $\nu(X)^{.5}/\tau(X)^2$ are reported in Table 1. A graphical display of the fractions on a log scale is shown in Figure 3. Note that Table 1 only reports 43 iterations since $\tau(X)$ equals zero in iteration 44 and the fractions are undefined.

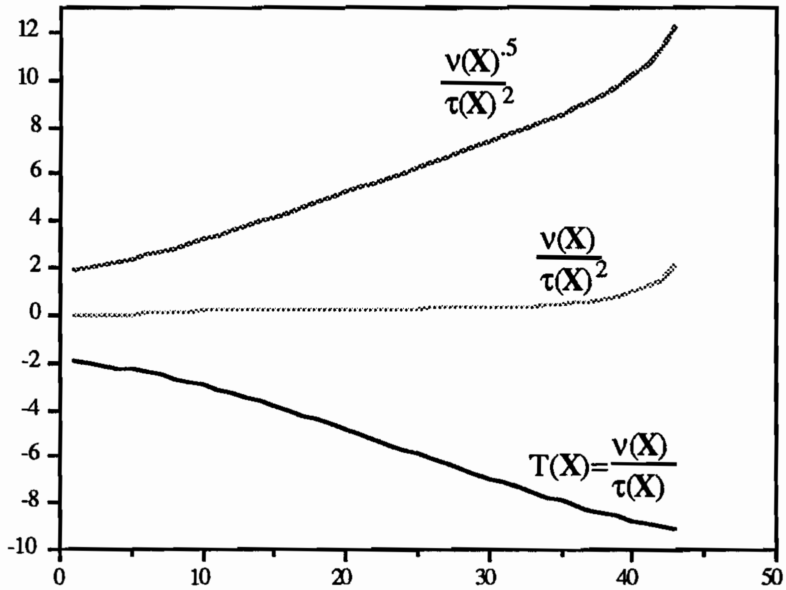

Figure 3. Behavior of T(X) and some fractions of $\nu(X)$ and $\tau(X)$ as X approaches X^*. The fractions are given vertically on a log scale and the iteration numbers horizontally. The data stem from the Mani collection.

In the plot we clearly see that the pole of the tunneling function is not strong enough. Two alternative ratios do result in a pole that is strong enough. Whereas the ratio $\nu(X)/\tau(X)^2$ has only towards the last iterations high values, the ratio $\nu(X)^{.5}/\tau(X)^2$ increases much faster. Therefore the latter seems most promising.

In this section we have seen that the strength of the pole in the tunneling function is not strong enough.

Table 1. Behavior of some fractions of the numerator and denominator as **X** approaches **X***. The data used are the Mani collection data.

Iteration	STRESS	difference	$\nu(X)$	$\tau(X)$	$T(X) = \frac{\nu(X)}{\tau(X)}$	$\frac{\nu(X)}{\tau(X)^2}$	$\frac{\nu(X)^5}{\tau(X)^2}$
0	.3537639261						
1	.0943974495	.2593664766	1.821E-04	1.365E-02	.0133409122	.977	7.243E01
2	.0907912176	.0036062319	9.777E-05	1.047E-02	.0093346847	.891	9.013E01
3	.0886526417	.0021385759	6.005E-05	8.052E-03	.0074581063	.926	1.195E02
4	.0869873466	.0016652951	3.702E-05	6.106E-03	.0060624406	.993	1.632E02
5	.0856132582	.0013740884	2.219E-05	4.545E-03	.0048811152	1.074	2.280E02
6	.0844736300	.0011396282	1.275E-05	3.313E-03	.0038482014	1.162	3.253E02
7	.0835444994	.0009291306	6.977E-06	2.362E-03	.0029534877	1.250	4.734E02
8	.0828080853	.0007364141	3.629E-06	1.648E-03	.0022015067	1.336	7.011E02
9	.0822433637	.0005647216	1.796E-06	1.127E-03	.0015938295	1.414	1.055E03
10	.0818246423	.0004187214	8.491E-07	7.563E-04	.0011226886	1.484	1.611E03
11	.0815238915	.0003007508	3.853E-07	4.994E-04	.0007715629	1.545	2.489E03
12	.0813139508	.0002099407	1.688E-07	3.251E-04	.0005190146	1.596	3.886E03
13	.0811709747	.0001429761	7.173E-08	2.092E-04	.0003428577	1.639	6.119E03
14	.0810756085	.0000953662	2.974E-08	1.333E-04	.0002231113	1.674	9.705E03
15	.0810130823	.0000625262	1.208E-08	8.426E-05	.0001434194	1.702	1.548E04
16	.0809726568	.0000404255	4.831E-09	5.292E-05	.0000912882	1.725	2.482E04
17	.0809468127	.0000258441	1.906E-09	3.306E-05	.0000576531	1.744	3.994E04
18	.0809304381	.0000163746	7.445E-10	2.057E-05	.0000361881	1.759	6.447E04
19	.0809201367	.0000103015	2.885E-10	1.276E-05	.0000226075	1.772	1.043E05
20	.0809136919	.0000064448	1.111E-10	7.893E-06	.0000140729	1.783	1.692E05
21	.0809096773	.0000040146	4.257E-11	4.872E-06	.0000087374	1.793	2.748E05
22	.0809071849	.0000024924	1.626E-11	3.003E-06	.0000054150	1.803	4.472E05
23	.0809056415	.0000015434	6.195E-12	1.848E-06	.0000033523	1.814	7.288E05
24	.0809046875	.0000009540	2.356E-12	1.136E-06	.0000020744	1.826	1.190E06
25	.0809040987	.0000005888	8.953E-13	6.973E-07	.0000012838	1.841	1.946E06
26	.0809037356	.0000003631	3.400E-13	4.276E-07	.0000007951	1.859	3.189E06
27	.0809035118	.0000002238	1.291E-13	2.618E-07	.0000004931	1.883	5.241E06
28	.0809033740	.0000001379	4.904E-14	1.601E-07	.0000003064	1.914	8.643E06
29	.0809032891	.0000000849	1.864E-14	9.766E-08	.0000001909	1.955	1.432E07
30	.0809032367	.0000000523	7.091E-15	5.941E-08	.0000001193	2.009	2.385E07
31	.0809032045	.0000000322	2.700E-15	3.602E-08	.0000000750	2.081	4.005E07
32	.0809031846	.0000000199	1.029E-15	2.173E-08	.0000000473	2.178	6.793E07
33	.0809031723	.0000000123	3.921E-16	1.302E-08	.0000000301	2.311	1.167E08
34	.0809031648	.0000000076	1.494E-16	7.739E-09	.0000000193	2.495	2.041E08
35	.0809031601	.0000000047	5.684E-17	4.544E-09	.0000000125	2.752	3.651E08
36	.0809031572	.0000000029	2.153E-17	2.625E-09	.0000000082	3.125	6.733E08
37	.0809031554	.0000000018	8.092E-18	1.483E-09	.0000000055	3.681	1.294E09
38	.0809031543	.0000000011	2.995E-18	8.110E-10	.0000000037	4.553	2.631E09
39	.0809031536	.0000000007	1.078E-18	4.233E-10	.0000000025	6.017	5.796E09
40	.0809031531	.0000000004	3.689E-19	2.055E-10	.0000000018	8.734	1.438E10
41	.0809031529	.0000000003	1.148E-19	8.858E-11	.0000000013	14.628	4.318E10
42	.0809031527	.0000000002	2.923E-20	3.046E-11	.0000000010	31.506	1.843E11
43	.0809031526	.0000000001	4.345E-21	5.950E-12	.0000000007	122.735	1.862E12

5. Discussion and conclusions.

In this paper convergent algorithms were derived that are necessary for applying the tunneling method to metric MDS. We have used majorization to prove convergence and fractional programming to deal with the denominator of the tunneling function.

However, the simple tunneling method needs to be refined; provisions for a stronger pole must be incorporated. The solution for creating a stronger pole was given first by Levy and

Gomez (1985). A stronger pole is achieved by raising the denominator to the power κ. Since the minimum does not change under any monotone transformation of the tunneling function we might as well leave the denominator as it is and raise the numerator to power $1/\kappa$ instead. The tunneling function would change to

$$T(X) = \frac{|\sigma(X) - \sigma(X^*)|^{1/\kappa}}{\mathrm{tr}(X - X^*)'(X - X^*)} \,. \tag{21}$$

The simulation results showed that we need at least $\kappa = 2$. For majorizing the numerator of (21) we can use a generalization of the inequality used by Heiser (1987) to majorize absolute residuals. This will be described in a forthcoming paper.

However, Törn and Žilinskas (1989) temper the optimism of the success of the tunneling method with a strong pole. They expect difficulties when κ becomes to large. The hypersurface of the tunneling function might become too flat and the zero point of the tunneling function could be missed. In their opinion it is impossible to guarantee numerical stability.

When the method stops it cannot find a solution with the same stress, so the last minimum is candidate global minimum. Timmer (1989, personal communication) has a theoretical objection against the method. First the tunneling method can not be proven *always* to converge to the global minimum. Moreover he claims that since the minimization of the tunneling function is highly related to the minimization of the objective function itself not much is gained by tunneling in terms of efficiency.

Therefore further research needs to be done on the effectiveness and numerical stability of the algorithm. The use of majorization has to be evaluated. It is particularly effective when the majorizing function is close to the original function. If not, only small improvements can be expected each iteration.

The tunneling algorithm of Gomez and Levy (1982) also introduces a moving pole with a similar function as the pole at X^*. It is used to avoid getting stuck at a stationary point that is not a minimum of the tunneling function. We do not know if our algorithm needs such a moving pole. If so, a generalization of the fractional programming algorithm should be developed allowing for multiple factors in the denominator of the tunneling function.

References.

De Leeuw, J. (1977). Applications of convex analysis to multidimensional scaling. J.R. Barra et al. (Eds.), *Recent developments in statistics* . pp.133-145. Amsterdam: North-Holland.

De Leeuw, J., Heiser, W.J. (1980). Multidimensional scaling with restrictions on the configuration. P.R. Krishnaiah (Ed.), *Multivariate analysis,* Vol. V, pp. 501-522. Amsterdam: North-Holland.

De Soete, G. Hubert, L., Arabie, P. (1988). On the use of simulated annealing for combinatorial data analysis. In: W. Gaul and M. Schader (Eds.), *Data, expert, knowledge and decisions*. Berlin: Springer-Verlag.

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

Dinkelbach, W. (1967). On nonlinear fractional programming. *Management Science,* Vol 13, no. 7, pp. 492-498.

Gomez, S., Levy, A.V. (1982). The tunneling method for solving the constrained global optimization problem with non-connected feasible regions. *Lecture notes in mathematics,* 909, pp.34-47. Springer-Verlag.

Heiser, W.J. (1981). Unfolding analysis of proximity data. Doctoral dissertation. Leiden: University of Leiden.

Heiser, W.J. (1987). Correspondence analysis with least absolute residuals. *Computational statistics & Data Analysis*, 5, pp. 337-356.

Heiser, W.J., De Leeuw, J. (1977). How to use SMACOF-I. (3nd ed. 1986). *Research report* UG-86-02. Leiden: Department of Datatheory.

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

Klaassen, P. (1989). Globale optimalisatie & multidimensional scaling. Unpublished master's thesis. Rotterdam: Faculty of Economics, Erasmus University.

Kruskal, J.B. (1964a). Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika,* 29, pp. 1-28.

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

Levy, A.V., Gomez, S. (1985). The tunneling method applied to global optimization. P.T. Boggs, R.H. Byrd, R.B. Schnabel (Ed.), *Numerical optimization 1984 (SIAM Philadelphia)*, pp.213-244.

Mathar, R. (1989). Convex analysis and algorithms in multidimensional scaling. Report no. 120. Augsburg: Institut für Mathematik.

Meulman, J. (1986). A distance approach to nonlinear multivariate analysis. Leiden: DSWO press.

Montalvo, A. (1979). Development of a new algorithm for the global minimization of functions. Ph.D. thesis in Theoretical and applied Mechanics. Universidad Nacional Autonoma de Mexico.

Rinnooy Kan, A.H.G., Timmer, G.T. (1987a). Stochastic global optimization methods part I: clustering methods.. *Mathematical Programming* 39, pp.27-56.

Rinnooy Kan, A.H.G., Timmer, G.T. (1987b). Stochastic global optimization methods part II: multi level methods.. *Mathematical Programming* 39, pp.57-78.

Robinson, W.S. (1951). A method for chronologically ordering archaeological deposits. *American Antiquity*, 16, pp.293-301.

Timmer, G.T. (1984). Global optimization: a stochastic approach. Ph.D. thesis. Rotterdam: Erasmus University.

Timmer, G.T. (1989). Personal communications.

Törn, A., Žilinskas, A. (1989). Global optimization. G. Goos and J. Hartmanis (Eds.), *Lecture notes in computer science,* Vol. 350. Berlin: Springer-Verlag.