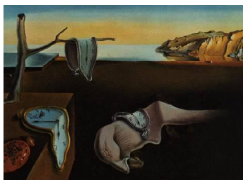

Journal of Statistical Software

September 2016, Volume 73, Issue 8. *doi: 10.18637/jss.v073.i08*

Multidimensional Scaling by Majorization: A Review

Patrick J. F. Groenen
Erasmus University Rotterdam

Michel van de Velden
Erasmus University Rotterdam

Abstract

A major breakthrough in the visualization of dissimilarities between pairs of objects was the formulation of the least-squares multidimensional scaling (MDS) model as defined by the Stress function. This function is quite flexible in that it allows possibly nonlinear transformations of the dissimilarities to be represented by distances between points in a low dimensional space. To obtain the visualization, the Stress function should be minimized over the coordinates of the points and the over the transformation. In a series of papers, Jan de Leeuw has made a significant contribution to majorization methods for the minimization of Stress in least-squares MDS. In this paper, we present a review of the majorization algorithm for MDS as implemented in the **smacof** package and related approaches. We present several illustrative examples and special cases.

Keywords: multidimensional scaling, majorization, MM, **smacof**.

1. Introduction

The aim of multidimensional scaling (MDS) is to visualize objects as points in a low, often two-dimensional space such that the Euclidean distance of pairs of points optimally represent the observed dissimilarities between pairs of objects. Although any distance measure can be used, we limit ourselves here to the Euclidean distance as it is the most popular one given its ease of interpretation and calculation, and De Leeuw has made contributions using this measure.

Amongst the earliest approaches to MDS is so-called "classical scaling", also referred to as Torgerson-Gower scaling after Torgerson (1958) and Gower (1966). They showed that a low dimensional solution can be obtained through an eigendecomposition. Classical scaling is closely related to principal component analysis and also referred to as principal coordinate analysis by Gower (1966). A major breakthrough in calculating the parameters for the MDS model was achieved by Kruskal (1964a,b). His least-squares version of the MDS model allows a direct fitting of Euclidean distances to possibly transformed dissimilarities. This approach has become the most widely used version of MDS and will be the one we consider here.

The mathematical optimization problem underlying least-squares MDS is not trivial. Jan de Leeuw has made many different theoretical contributions to the numerical algorithm used for MDS. His work has led to one of the best algorithms for MDS: The SMACOF algorithm. In doing so, De Leeuw laid the foundation of a whole class of majorization-based algorithms, nowadays better known under the name MM algorithms. The purpose of this paper is to present an overview of MDS, the SMACOF algorithm, and the contributions of De Leeuw, and to provide practical illustrations through the **smacof** package in R (De Leeuw and Mair 2009; De Leeuw, Mair, and Groenen 2016d).

We start with an example of MDS followed by an explanation of the SMACOF algorithm. The next section discusses another important contribution of De Leeuw, that is, extending the SMACOF algorithm to impose constraints on the configuration thereby allowing a variety of different models to be fit. We then discuss the occurrence of local minima in least-squares MDS. Finally, we extensively discuss different usages of weights that follow as a direct consequence of methods De Leeuw introduced for the addition of weights to the MDS loss function.

2. Least-squares MDS

To illustrate MDS, we use data from Bell and Lattin (1998) who followed 448 US households over a period of 104 weeks from June 1991 to June 1993 with the aim to study the loyalty of these households with respect to different brands of cola. Our dataset (available in `cola.switch.RData`) consists of switching behavior among fifteen different cola brands. The rows indicate from which product the change is made and the columns correspond to the product to which individuals switched. The interest lies in how easily households switch between these brands and how this can be visualized through MDS.

The values in `cola.switch` are counts. They reflect how often households switched from one brand to another. We can interpret these counts as follows: high counts indicate that there is more switching, indicating that the brands were considered to be more similar in some respect. For visualization through MDS, one usually starts from dissimilarities rather than similarities. Large dissimilarities will be represented by large distances and small dissimilarities by small distances. Therefore, we transform the switching counts s_{ij} (similarities) to dissimilarities by using the function `sim2diss()` from the **smacof** package, using `method = "counts"`. In this way, the dissimilarity between any pair of brands i and j is calculated as

$$\delta_{ij} = -\log\left(\frac{s_{ii}s_{jj}}{s_{ij}s_{ji}}\right),$$

which is symmetric and contains no negative numbers as the diagonal elements are necessarily larger than the corresponding off diagonal elements. Seven entries in the original data have $s_{ij} = 0$ which causes the logarithm to be undefined. The standard approach would be to replace them by missing values (`NA`). However, for illustrational purposes, we replace the `NA` by $\delta_{ij} = 1$. In this particular case, this procedure hardly affects the solution (i.e., approximately the same solution is obtained if we treat the values as missing).

A straightforward formalization of least-squares MDS was given by Kruskal (1964a) who defined the *Stress-1* formula as

$$\sigma_1(\mathbf{X}, \hat{\mathbf{d}}) = \left(\frac{\sum_{i<j}(\hat{d}_{ij} - d_{ij}(\mathbf{X}))^2}{\sum_{i<j} d_{ij}^2(\mathbf{X})} \right)^{1/2}, \tag{1}$$

where $\mathbf{X}$ is $n \times p$ matrix with coordinates of n objects in p dimensions and the squared $d_{ij}(\mathbf{X}) = \sqrt{\sum_{s=1}^{p}(x_{is} - x_{js})^2}$ is the Euclidean distance between rows i and j of $\mathbf{X}$. The $\hat{d}_{ij}$s are called pseudo-distances (or disparities) and they are a function of the given dissimilarities δ_{ij}. We consider this explicit formulation of the least-squares error function (1) as one of the major contributions of Kruskal to the development of least-squares MDS. Another major contribution was his proposal for an ordinal version of MDS that allowed using only the rankorder information of the dissimilarities. He did so by constraining the $\hat{d}_{ij}$s to have the same order as the dissimilarities. Other *transformations* are possible such as the linear $\hat{d}_{ij} = a + b\delta_{ij}$, the polynomial $\hat{d}_{ij} = b_0 + \sum_q b_q \delta_{ij}^q$, and monotone spline transformations. In this paper, the emphasis is on the ratio transformation $\hat{d}_{ij} = b\delta_{ij}$, as the main contributions of De Leeuw for MDS concern fitting of the distances. Therefore, we shall assume that $\hat{d}_{ij} = \delta_{ij}$. We also require $\delta_{ij} \geq 0$ and, thus, $\hat{d}_{ij} \geq 0$, as negative values can never be modeled by nonnegative distances.

In all of his work, De Leeuw takes great care in employing effective and consistent mathematical notation. Here, we will use his notation for MDS. In one of his first contributions to least-squares MDS, De Leeuw (1977) focused on *raw Stress*, essentially the numerator of the Stress-1 loss function, that is,

$$\sigma^2_{\text{raw}}(\mathbf{X}, \hat{\mathbf{d}}) = \sum_{i<j} w_{ij}(\hat{d}_{ij} - d_{ij}(\mathbf{X}))^2. \tag{2}$$

A noticeable extension to earlier formulations of raw Stress is the addition of the nonnegative weights w_{ij} that indicate the importance of the residual $\hat{d}_{ij} - d_{ij}(\mathbf{X})$ of object pair ij. These weights can be used for handling missing data, that is, $w_{ij} = 0$ if δ_{ij} is missing and $w_{ij} = 1$ otherwise. In Section 6, we review other usages as well. During a visit at the Department of Statistics at UCLA in 1993, one of the coauthors asked De Leeuw why he had added the w_{ij}s to Stress. The characteristic answer by Jan de Leeuw was that weights make the problem more complex and thus more interesting.

As distances in a plot are interpreted relative to each other, their overall scaling does not matter. Thus, if all $\hat{d}_{ij}$ are multiplied by some scalar a, it suffices to multiply all coordinates in $\mathbf{X}$ by the same constant a so that the distances become $d_{ij}(a\mathbf{X}) = ad_{ij}(\mathbf{X})$ due to the Euclidean distance being homogeneous. Using this property, De Leeuw standardized the disparities to some fixed constant, for example, $\sum_{i<j} w_{ij}\hat{d}_{ij}^2 = n(n-1)/2$. This standardization has the advantage that, irrespective of the number of objects n, the disparities are on average scattered around 1. By explicitly standardizing the disparities, De Leeuw changed the minimization problem from Stress-1 to $\sigma^2_{\text{raw}}(\mathbf{X}, \hat{\mathbf{d}})$ subject to the length constraint $\sum_{i<j} w_{ij}\hat{d}_{ij}^2 = n(n-1)/2$.

Consider the cola switching data again. The following code produces an MDS solution and a Shepard plot showing the residuals in Figure 1.

```
R> load("cola.switch.RData")
R> library("smacof")
R> dis <- sim2diss(as.matrix(cola.switch), method = "counts")
```

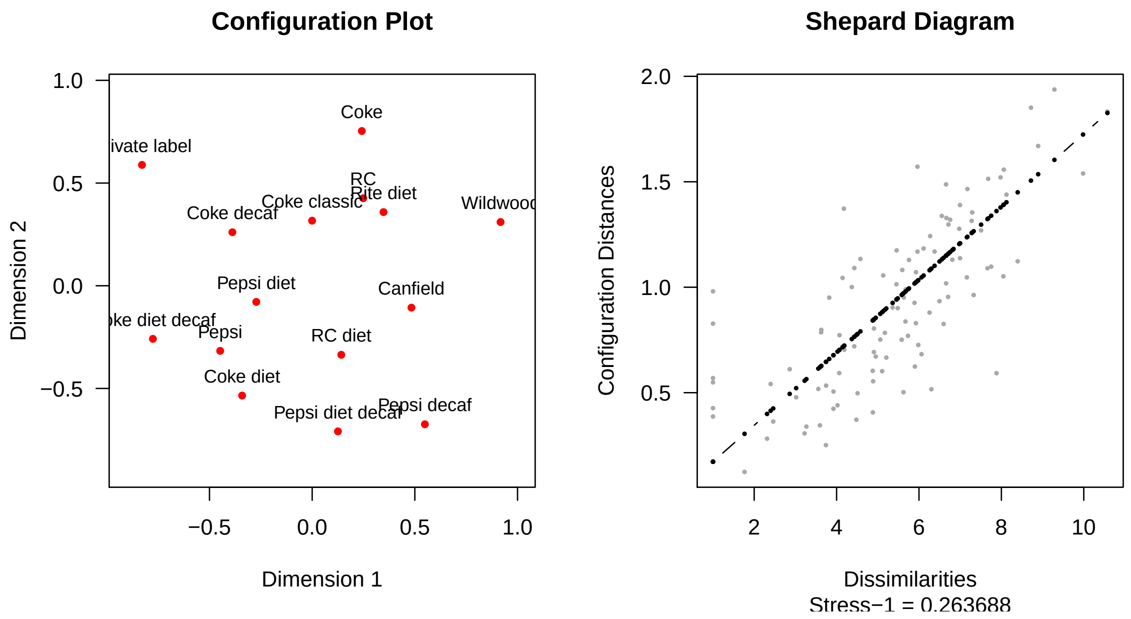

Figure 1: Configuration plot of cola brand switching data (left-hand panel) and a Shepard plot (right-hand panel).

```
R> dis[is.infinite(dis)] <- 1
R> res2 <- mds(dis)
R> par(mfrow = c(1, 2))
R> plot(res2, las = 1, col = "red")
R> plot(res2, plot.type = "Shepard", las = 1,
+    sub = paste("Stress-1 =", formatC(res2$stress, digits = 6)))
```

The overall fit has Stress-1 equal to 0.263688 which is not too bad for ratio MDS. There seems to be quite some switching between diet and decaf versions of Coke and Pepsi in the lower left corner because these two points are close together. In contrast, there is little switching going on between these diet and decaf versions and Private Label, Coke, and Wildwood as their distances are large. Many more relations can be found by simply interpreting the distance between pairs of points.

It is useful to decompose the raw Stress function as

$$\begin{aligned}
\sigma^2_{\text{raw}}(\mathbf{X}, \hat{\mathbf{d}}) &= \sum_{i<j} w_{ij}\hat{d}^2_{ij} + \sum_{i<j} w_{ij} d^2_{ij}(\mathbf{X}) - 2\sum_{i<j} w_{ij}\hat{d}_{ij} d_{ij}(\mathbf{X}) && (3)\\
&= \eta^2_{\hat{d}} + \eta^2(\mathbf{X}) - 2\rho(\mathbf{X}) && (4)
\end{aligned}$$

with $\eta^2_{\hat{d}}$ the *total dispersion*, $\eta^2(\mathbf{X})$ the *reconstructed dispersion*, and $\rho(\mathbf{X})$ the *codispersion*. It turns out that normalized Stress, $\sigma^2_{\text{norm}}(\mathbf{X}^*) = \sigma^2_{\text{raw}}(\mathbf{X}^*)/\eta^2_{\hat{d}}$, has the convenient property that it is between 0 and 1 at any stationary $\mathbf{X}$ and that its square root is equal to Stress-1. These properties can be seen as follows. Suppose that $\mathbf{X}^*$ is a local minimum of raw Stress and assume that $\hat{\mathbf{d}}$ is fixed. As a consequence, the first derivative of $\sigma^2_{\text{raw}}(\alpha\mathbf{X}^*)$ with respect to α must be zero, that is,

$$\frac{\partial \sigma^2_{\text{raw}}(\alpha \mathbf{X}^*)}{\partial \alpha} = \frac{\partial \eta^2_{\hat{d}}}{\partial \alpha} + \frac{\partial \alpha^2 \eta^2(\mathbf{X}^*)}{\partial \alpha} - 2\frac{\partial \alpha \rho(\mathbf{X}^*)}{\partial \alpha} = 2\alpha\eta^2(\mathbf{X}^*) - 2\rho(\mathbf{X}^*) = 0 \tag{5}$$

so that $\alpha^* = \rho(\mathbf{X}^*)/\eta^2(\mathbf{X}^*)$. Then,

$$\sigma^2_{\text{raw}}(\alpha^* \mathbf{X}^*) = \eta^2_{\hat{d}} + \frac{\rho^2(\mathbf{X}^*)}{\eta^4(\mathbf{X}^*)}\eta^2(\mathbf{X}^*) - 2\frac{\rho(\mathbf{X}^*)}{\eta^2(\mathbf{X}^*)}\rho(\mathbf{X}^*) = \eta^2_{\hat{d}} - \frac{\rho^2(\mathbf{X}^*)}{\eta^2(\mathbf{X}^*)}.$$

Now, normalized Stress is defined as

$$\sigma^2_{\text{norm}}(\mathbf{X}^*) = \frac{\sigma^2_{\text{raw}}(\mathbf{X}^*)}{\eta^2_{\hat{d}}} = 1 - \frac{\rho^2(\mathbf{X}^*)}{\eta^2_{\hat{d}}\eta^2(\mathbf{X}^*)} = 1 - \lambda^2(\mathbf{X}^*)$$

where $\lambda(\mathbf{X}^*)$ may also be written as

$$\lambda(\mathbf{X}^*) = \frac{\sum_{i<j} w_{ij}\hat{d}_{ij} d_{ij}(\mathbf{X}^*)}{\left(\sum_{i<j} w_{ij}\hat{d}^2_{ij}\right)^{1/2} \left(\sum_{i<j} w_{ij} d^2_{ij}(\mathbf{X}^*)\right)^{1/2}}$$

which is *Tucker's congruence coefficient*, or, equivalently, the cosine of the vectors of the disparities and the distances. Because both vectors have nonnegative values only, their angle is at most 90°, so that the cosine is between 0 and 1. Therefore, at a local minimum, normalized Stress is necessarily between 0 and 1, irrespective of the normalization of the disparities. Note that $\lambda^2(\mathbf{X}^*)$ is also referred to as *dispersion accounted for* (DAF) as an analogy to variance accounted for in multiple regression.

In our example, it can be verified that this is indeed the case by the following code.

```
R> d <- dist(res2$conf)
R> eta.dhat <- sum(res2$dhat^2)
R> eta.X <- sum(d^2)
R> rho.X <- sum(res2$dhat * d)
R> lambda <- rho.X/(eta.dhat * eta.X)^0.5
R> lambda

[1] 0.964608

R> lambda^2

[1] 0.930468

R> 1 - lambda^2

[1] 0.0695316

R> sum((res2$dhat - d)^2)/eta.dhat

[1] 0.0695316
```

In the MDS literature, the usage of reporting Stress-1 had been dominant. As a consequence, the stress reported by `mds()` in the **smacof** package is Stress-1.

3. The SMACOF algorithm for MDS

One of the most important contributions of De Leeuw to MDS is the invention of the SMACOF algorithm (implemented in the `mds()` function of the **smacof** package). With the introduction of the SMACOF algorithm, De Leeuw in fact established a general optimization approach that he termed *majorization*. In the context of a line search, the same principle was brought forward by Ortega and Rheinboldt (1970) who also used the term majorization. This approach was independently brought forward by Voss and Eckhardt (1980) who called it the *generalized Weiszfeld's method*. Currently, the method is better known as an MM algorithm, which stands for *minimization by majorisation* and *maximization by minorisation* (see, for example, Hunter and Lange 2004). In the machine learning literature, the method is better known as the *convex-concave procedure* (CCCP, Yuille and Rangarajan 2003).

The basic idea of majorization is to replace the original function $f(\mathbf{x})$ by a simpler function, the majorizing function, $g(\mathbf{x}, \mathbf{y})$. Here, $\mathbf{x} \in \mathbb{R}^n$ are the parameters over which one needs to minimize $f(\mathbf{x})$ and $\mathbf{y} \in \mathbb{R}^n$ is a vector with known values, the *supporting point*, for example, obtained from the previous iteration. Then, the majorizing function $g(\mathbf{x}, \mathbf{y})$ needs to satisfy the following requirements:

1. $f(\mathbf{y}) = g(\mathbf{y}, \mathbf{y})$, that is, equality at the supporting point $\mathbf{y}$,
2. $f(\mathbf{x}) \leq g(\mathbf{x}, \mathbf{y})$, and
3. $g(\mathbf{x}, \mathbf{y})$ must be simple (usually linear or quadratic).

Let $\mathbf{x}^+$ be chosen such that $g(\mathbf{x}^+, \mathbf{y}) \leq g(\mathbf{y}, \mathbf{y})$, for example, by choosing $\mathbf{x}^+$ as argmin $g(\mathbf{x}, \mathbf{y})$. From these requirements, the following chain of inequalities can be derived:

$$f(\mathbf{x}^+) \leq g(\mathbf{x}^+, \mathbf{y}) \leq g(\mathbf{y}, \mathbf{y}) = f(\mathbf{y}). \tag{6}$$

De Leeuw referred to this chain as the *sandwich inequality* because the value of the minimum of the majorizing function is sandwiched in-between the old and the new function value. The important consequence of (6) is that repeatedly computing the update $\mathbf{x}^+$ and setting $\mathbf{y} \leftarrow \mathbf{x}^+$ yields an algorithm that has guaranteed descent. If $f(\mathbf{x})$ is bounded from below or $\mathbf{x}$ is sufficiently constrained, then the sequence of $f(\mathbf{x}^+)$ values converges. In addition, there is global convergence, which means that a start from any $\mathbf{y} \in \mathbb{R}^n$ causes the sequence to converge.

Standard nonlinear optimization methods such as steepest descent and the Newton method require elaborate stepsize procedures to establish convergence properties. The beauty of majorization is that the idea is quite simple, powerful, and has guaranteed descent. In practice, $g(\mathbf{x}, \mathbf{y})$ is often linear or quadratic so that argmin $g(\mathbf{x}, \mathbf{y})$ is easy to compute and no complicated stepsize procedures are needed to guarantee descent.

Before deriving the update for raw-Stress through majorization, we introduce some convenient notation for the squared Euclidean distance

$$\begin{aligned} d_{ij}^2(\mathbf{X}) &= \sum_{s=1}^{p}(x_{is}-x_{js})^2 = \sum_{s=1}^{p}(\mathbf{x}_s^\top(\mathbf{e}_i-\mathbf{e}_j))^2 = \sum_{s=1}^{p}\mathbf{x}_s^\top(\mathbf{e}_i-\mathbf{e}_j)(\mathbf{e}_i-\mathbf{e}_j)^\top\mathbf{x}_s \\ &= \mathrm{tr}\mathbf{X}^\top(\mathbf{e}_i-\mathbf{e}_j)(\mathbf{e}_i-\mathbf{e}_j)^\top\mathbf{X} = \mathrm{tr}\mathbf{X}^\top\mathbf{A}_{ij}\mathbf{X} \end{aligned} \tag{7}$$

with $\mathbf{e}_i$ the i-th column of the identity matrix $\mathbf{I}$ and $\mathbf{x}_s$ column s of $\mathbf{X}$. The matrix $\mathbf{A}_{ij}$ has zeros everywhere except $a_{ii} = a_{jj} = 1$ and $a_{ij} = a_{ji} = -1$. Then, $\eta^2(\mathbf{X})$ as defined in (4)

$$\eta^2(\mathbf{X}) = \sum_{i<j} w_{ij}d_{ij}^2(\mathbf{X}) = \sum_{i<j} w_{ij}\mathrm{tr}\mathbf{X}^\top\mathbf{A}_{ij}\mathbf{X} = \mathrm{tr}\mathbf{X}^\top\left(\sum_{i<j} w_{ij}\mathbf{A}_{ij}\right)\mathbf{X} = \mathrm{tr}\mathbf{X}^\top\mathbf{V}\mathbf{X}.$$

which shows that the sum of squared Euclidean distances is quadratic in $\mathbf{X}$. When all $w_{ij} = 1$, the matrix $\mathbf{V}$ simplifies into $\mathbf{V} = n\mathbf{I} - \mathbf{1}\mathbf{1}^\top$. Assuming that all distances are positive, that is, $d_{ij}(\mathbf{X}) > 0$, (7) can also be used to express the Euclidean distance between rows i and j of the matrix $\mathbf{X}$ as

$$d_{ij}(\mathbf{X}) \quad = \quad \mathrm{tr}\mathbf{X}^\top(d_{ij}^{-1}(\mathbf{X})\mathbf{A}_{ij})\mathbf{X}$$

so that

$$\rho(\mathbf{X}) = \sum_{i<j} w_{ij}\hat{d}_{ij}d_{ij}(\mathbf{X}) = \mathrm{tr}\mathbf{X}^\top\left(\sum_{i<j} b_{ij}\mathbf{A}_{ij}\right)\mathbf{X} = \mathrm{tr}\mathbf{X}^\top\mathbf{B}(\mathbf{X})\mathbf{X}.$$

with $b_{ij} = w_{ij}\hat{d}_{ij}/d_{ij}(\mathbf{X})$.

To minimize $\sigma^2_{\text{raw}}(\mathbf{X}, \hat{\mathbf{d}})$ over $\mathbf{X}$, we use the SMACOF approach. Initially, the acronym SMACOF stood for 'scaling by majorizing a convex function' following the convex analysis approach of De Leeuw (1977). Here we follow De Leeuw and Heiser (1980) and De Leeuw (1988) who redefine the acronym as *scaling by majorizing a complicated function.*

In the expression of $\sigma^2_{\text{raw}}(\mathbf{X}, \hat{\mathbf{d}})$, the term $\eta^2(\mathbf{X})$ is quadratic in $\mathbf{X}$ and thus does not need any further elaboration. The difficult part lies in $-2\rho(\mathbf{X})$ which can be interpreted as a a weighted sum of $-d_{ij}(\mathbf{X})$. Through the Cauchy-Schwartz inequality we have

$$\begin{aligned} d_{ij}(\mathbf{X})d_{ij}(\mathbf{Y}) &\geq \mathrm{tr}\mathbf{X}^\top\mathbf{A}_{ij}\mathbf{Y} \\ -d_{ij}(\mathbf{X})d_{ij}(\mathbf{Y}) &\leq -\mathrm{tr}\mathbf{X}^\top\mathbf{A}_{ij}\mathbf{Y} \\ -d_{ij}(\mathbf{X}) &\leq -\mathrm{tr}\mathbf{X}^\top(d_{ij}^{-1}(\mathbf{Y})\mathbf{A}_{ij})\mathbf{Y} \text{ assuming } d_{ij}(\mathbf{Y}) > 0. \end{aligned}$$

This inequality shows that $-d_{ij}(\mathbf{X})$ is linearly majorized by $-\mathrm{tr}\mathbf{X}^\top(d_{ij}^{-1}(\mathbf{Y})\mathbf{A}_{ij})\mathbf{Y}$ which is linear in $\mathbf{X}$. Note that the inequality assumes that $d_{ij}(\mathbf{Y}) > 0$. Figure 2 shows the function $-d_{ij}(\mathbf{X})$ and a linear majorizing function for a supporting point at coordinates $(-1, -1)$. In the special case of $d_{ij}(\mathbf{Y}) = 0$, a simple constant majorizing function is obtained by $-d_{ij}(\mathbf{X}) \leq 0$, so that the constant function 0 linearly majorizes $-d_{ij}(\mathbf{X})$. Define

$$b_{ij} = \begin{cases} w_{ij}\hat{d}_{ij}/d_{ij}(\mathbf{Y}) & \text{if } d_{ij}(\mathbf{Y}) > 0 \\ 0 & \text{if } d_{ij}(\mathbf{Y}) = 0 \end{cases}. \tag{8}$$

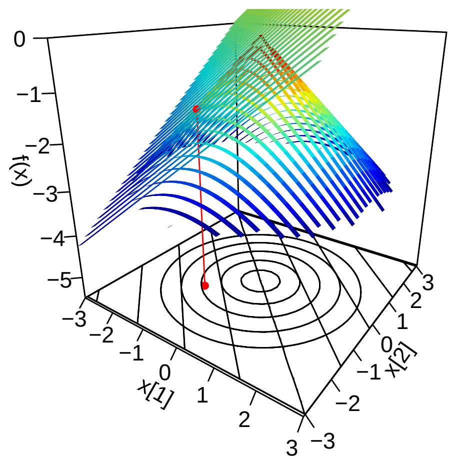

Figure 2: The function $-d_{ij}(\mathbf{X})$ and a linear majorizing function at a supporting point with coordinates $(-1, -1)$.

Now,

$$-\rho(\mathbf{X}) = -\sum_{i<j} w_{ij}\hat{d}_{ij}d_{ij}(\mathbf{X}) \leq -\text{tr}\mathbf{X}^\top\left(\sum_{i<j} b_{ij}\mathbf{A}_{ij}\right)\mathbf{X} = -\text{tr}\mathbf{X}^\top\mathbf{B}(\mathbf{Y})\mathbf{Y}. \tag{9}$$

with $\mathbf{B}(\mathbf{Y}) = \sum_{i<j} b_{ij}\mathbf{A}_{ij}$ and b_{ij} defined in (8).

Combining the results above gives the majorizing inequality

$$\begin{aligned}\sigma^2_{\text{raw}}(\mathbf{X}, \hat{\mathbf{d}}) &= \eta^2_{\hat{d}} + \eta^2(\mathbf{X}) - 2\rho(\mathbf{X}) \\ &\leq \eta^2_{\hat{d}} + \text{tr}\mathbf{X}^\top\mathbf{V}\mathbf{X} - 2\text{tr}\mathbf{X}^\top\mathbf{B}(\mathbf{Y})\mathbf{Y} = g(\mathbf{X}, \mathbf{Y}).\end{aligned} \tag{10}$$

It is not difficult to see that $g(\mathbf{X}, \mathbf{Y})$ is quadratic in $\mathbf{X}$. The update $\mathbf{X}^+$ is obtained by equating the first derivative to zero, that is,

$$\begin{aligned}2\mathbf{V}\mathbf{X} - 2\mathbf{B}(\mathbf{Y})\mathbf{Y} &= \mathbf{0} \\ \mathbf{V}\mathbf{X} &= \mathbf{B}(\mathbf{Y})\mathbf{Y} \\ \mathbf{X}^+ &= \mathbf{V}^-\mathbf{B}(\mathbf{Y})\mathbf{Y},\end{aligned} \tag{11}$$

where $\mathbf{V}^-$ is the Moore-Penrose inverse of $\mathbf{V}$. In case of all $w_{ij} = 1$, (11) simplifies to $\mathbf{X}^+ = n^{-1}\mathbf{B}(\mathbf{Y})\mathbf{Y}$. In honour of the contributions of Guttman to MDS algorithms, De Leeuw named update (11) the *Guttman transform.*

Apart from the guaranteed descent properties, De Leeuw (1988) proved several other properties. We mention four of them here.

1. After convergence, $\sigma^2_{\text{raw}} \approx \eta^2_{\hat{d}} - \eta^2(\mathbf{X})$ which means that the more the points spread in the space, the better the fit and the lower the Stress.

2. $\eta^2(\mathbf{X} - \mathbf{X}^+)$ converges to zero. This property of SMACOF is very useful as it indicates that the difference between subsequent updates decreases. Therefore, the SMACOF algorithm is well suited for a dynamic visualization of the updates as the points will not make large jumps from one iteration to another.

3. The convergence rate, that is, $\eta(\mathbf{X}_{k+1} - \mathbf{X}^+_{k+1})/\eta(\mathbf{X}_k - \mathbf{X}^+_k)$ is linear with k being the iteration counter here. This property implies that with a strong convergence criterion, the SMACOF algorithm may need many iterations. Fortunately, the first iterations of the algorithm have a much better convergence rate. In practise, often less than 500 or even 100 iterations are needed to get sufficient convergence.

4. De Leeuw and Stoop (1984) showed that at a local minimum, if all $w_{ij} > 0$, and $\hat{d}_{ij} > 0$, then necessarily all $d_{ij}(\mathbf{X}) > 0$. Consequently, for such data no two points will coincide at a local minimum and Stress is differentiable at this local minimum.

4. Constrained MDS

An important paper on MDS with constraints was written by De Leeuw and Heiser (1980). For the first time within the framework of majorization, theory was developed that enabled imposition of constraints on the configuration. As a consequence, a wide variety of models became available in a single, coherent optimization algorithm.

The basic idea brought forward in De Leeuw and Heiser (1980) is quite simple. Let $\mathbf{X}$ be constrained to be an element of a set C satisfying the constraints. One could for example restrict $\mathbf{X} = \mathbf{HC}$ with $\mathbf{H}$ a given $n \times q$ matrix of q external variables with information on the objects and the $q \times q$ matrix $\mathbf{C}$ has unknown weights.

Two assumptions are needed to make the majorizing algorithm work: (a) the previous estimate $\mathbf{Y}$ must satisfy the constraints and (b) a solution must exist for the least-squares problem. Using (10) and (11), we can rewrite $g(\mathbf{X}, \mathbf{Y})$ as

$$g(\mathbf{X}, \mathbf{Y}) = \eta^2_{\hat{d}} + \mathrm{tr}(\mathbf{X} - \mathbf{X}^+)^\top \mathbf{V}(\mathbf{X} - \mathbf{X}^+) - \mathrm{tr}\mathbf{X}^{+\top}\mathbf{V}\mathbf{X}^+ \tag{12}$$

so that finding an update for $\mathbf{X}$ that minimizes the least-squares problem $\mathrm{tr}(\mathbf{X}-\mathbf{X}^+)^\top\mathbf{V}(\mathbf{X}-\mathbf{X}^+)$ subject to the constraints also reduces raw Stress. For example, if $\mathbf{X}$ is linearly constrained as $\mathbf{X} = \mathbf{HC}$, then the update for $\mathbf{C}$ (and thus $\mathbf{X}$) is obtained by

$$\mathbf{C}_{\text{upd}} = (\mathbf{H}^\top\mathbf{V}\mathbf{H})^{-1}\mathbf{H}^\top\mathbf{V}\mathbf{X}^+.$$

Constrained MDS can also be used for three-way MDS models where K replications of the dissimilarity matrices are available. A well known model for three-way MDS is the dimension weighting model (also called individual differences scaling or INDSCAL) that assumes a an $n \times p$ matrix $\mathbf{Z}$ with the common space and individual spaces $\mathbf{X}_k = \mathbf{ZD}_k$ with $\mathbf{D}_k$ a diagonal $p \times p$ matrix with dimension weights (Horan 1969; Carroll and Chang 1970). This model has as its loss function

$$\sigma^2_{\text{3-way}}(\mathbf{X}, \hat{\mathbf{d}}) \quad = \quad \sum_{k=1}^{K}\sum_{i<j} w_{ijk}(\hat{d}_{ijk} - d_{ij}(\mathbf{ZD}_k))^2. \tag{13}$$

However, this model perfectly fits within the constrained majorization approach as follows. Consider the block matrices

$$\mathbf{W} = \begin{bmatrix} \mathbf{W}_1 & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{W}_2 & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{W}_K \end{bmatrix}, \boldsymbol{\Delta} = \begin{bmatrix} \boldsymbol{\Delta}_1 & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Delta}_2 & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \dots & \boldsymbol{\Delta}_K \end{bmatrix}, \text{ and } \mathbf{X} = \begin{bmatrix} \mathbf{ZD}_1 \\ \mathbf{ZD}_2 \\ \vdots \\ \mathbf{ZD}_K \end{bmatrix}. \tag{14}$$

Then, the dimension weighting model can also be seen as a constrained MDS of the $nK \times nK$ matrix of dissimilarities $\mathbf{\Delta}$ using weights $\mathbf{W}$ under the constraint $\mathbf{X}_k = \mathbf{Z}\mathbf{D}_k$. The functions `smacofIndDiff()` and `indscal()` in the **smacof** implement this algorithm.

Some constraints do not have an easy solution when the least-squares metric is $\mathbf{V}$. One such example is a circular constraint, that is, a constraint requiring the coordinates to lie on a circle of radius 1. For such cases, an additional majorization step is is needed for $\eta^2(\mathbf{X})$. Let λ be the largest eigenvalue of $\mathbf{V}$ so that $\mathbf{V} - \lambda\mathbf{I}$ is negative semidefinite. As a consequence, we have

$$\begin{aligned}
\mathrm{tr}(\mathbf{X}-\mathbf{Y})^\top(\mathbf{V}-\lambda\mathbf{I})(\mathbf{X}-\mathbf{Y}) &\leq 0,\\
\mathrm{tr}(\mathbf{X}-\mathbf{Y})^\top\mathbf{V}(\mathbf{X}-\mathbf{Y}) &\leq \lambda\mathrm{tr}(\mathbf{X}-\mathbf{Y})^\top(\mathbf{X}-\mathbf{Y}),\\
\mathrm{tr}\mathbf{X}^\top\mathbf{V}\mathbf{X} + \mathrm{tr}\mathbf{Y}^\top\mathbf{V}\mathbf{Y} - 2\mathrm{tr}\mathbf{X}^\top\mathbf{V}\mathbf{Y} &\leq \lambda\mathrm{tr}\mathbf{X}^\top\mathbf{X} - 2\lambda\mathrm{tr}\mathbf{X}^\top\mathbf{Y} + \lambda\mathrm{tr}\mathbf{Y}^\top\mathbf{Y},\\
\mathrm{tr}\mathbf{X}^\top\mathbf{V}\mathbf{X} &\leq \lambda\mathrm{tr}\mathbf{X}^\top\mathbf{X} - 2\lambda\mathrm{tr}\mathbf{X}^\top(\mathbf{Y}-\lambda^{-1}\mathbf{V}\mathbf{Y})\\
&\quad +\lambda\mathrm{tr}\mathbf{Y}^\top(\mathbf{I}-\lambda^{-1}\mathbf{V})\mathbf{Y}. \qquad (15)
\end{aligned}$$

Combining (15) and (10) yields for the majorizing inequality

$$\begin{aligned}
\sigma^2_{\mathrm{raw}}(\mathbf{X},\hat{\mathbf{d}}) &\leq \lambda\mathrm{tr}\mathbf{X}^\top\mathbf{X} - 2\lambda\mathrm{tr}\mathbf{X}^\top(\mathbf{Y}-\lambda^{-1}\mathbf{V}\mathbf{Y}+\lambda^{-1}\mathbf{B}(\mathbf{Y})\mathbf{Y})\\
&\quad \eta^2_{\hat{d}} + \lambda\mathrm{tr}\mathbf{Y}^\top(\mathbf{I}-\lambda^{-1}\mathbf{V})\mathbf{Y}\\
&= \lambda\mathrm{tr}(\mathbf{X}-\mathbf{Z})^\top(\mathbf{X}-\mathbf{Z}) + \mathrm{const} = g(\mathbf{X},\mathbf{Y}). \qquad (16)
\end{aligned}$$

with $\mathbf{Z} = \mathbf{Y} - \lambda^{-1}\mathbf{V}\mathbf{Y} + \lambda^{-1}\mathbf{B}(\mathbf{Y})\mathbf{Y}$ being the unconstrained update and "const" a term with only constants. It may be verified that the least-squares update that minimizes $\mathrm{tr}(\mathbf{X} - \mathbf{Z})^\top(\mathbf{X}-\mathbf{Z})$ subject to $\mathbf{x}_i^\top\mathbf{x}_i = 1$ (with $\mathbf{x}_i^\top$ row i of $\mathbf{X}$) is obtained by $\mathbf{x}_i = \mathbf{z}_i/(\mathbf{z}_i^\top\mathbf{z}_i)^{1/2}$. Consequently, this update ensures that the points are on a circle with radius 1 and that raw Stress is reduced in each iteration. This additional majorization step has been implemented in the `smacofConstraint()` function of **smacof** when allowing optimal scaling of the external variables which can be used for imposing regional constraints, (see Borg, Groenen, Jehn, Bilsky, and Schwartz 2011).

5. Local minima

While writing his Ph.D. thesis, which was for a large part devoted to local mimima in least-squares MDS (Groenen 1993), the first author met Jan de Leeuw during the Distancia 1992 conference in Rennes, France. Casually, Jan mentioned having solved the local minimum problem for least-squares MDS. Obviously, this remark rather shocked the first author who feared that the work he had done so far had been a waste of time. Fortunately, this was not the case. In fact, De Leeuw showed in Rennes for the first time his discovery that full dimensional scaling does not suffer from the local minimum problem. Rather than the total collapse of his Ph.D. thesis, the encounter in Rennes led to a research visit of the first author to the University of California, Los Angeles (UCLA) in 1993 and a paper on *inverse multidimensional scaling*[1] (De Leeuw and Groenen 1997)

Below we discuss the local mininimum problem in unidimensional scaling first, before switching to full dimensional scaling and a brief discussion on local minima for other dimensionalities.

[1]Inverse MDS deals with finding the set S of dissimilarity data $\mathbf{\Delta}$ for a given given solution $\mathbf{X}$ such that for each $\mathbf{\Delta} \in S$ the matrix of coordinates $\mathbf{X}$ is a local minimum (or a stationary point).

The local minimum problem for MDS through Stress *unidimensional scaling* ($p = 1$) turns out to be particularly severe. One of the reasons is that the effective part of the Guttman update (11) can be written as

$$\mathbf{B}(\mathbf{x})\mathbf{x} = \sum_{i<j} w_{ij}\delta_{ij}\frac{x_i - x_j}{|x_i - x_j|} = \sum_{i<j} w_{ij}\delta_{ij}\text{sign}(x_i - x_j) \tag{17}$$

with $\text{sign}(x_i - x_j)$ is -1 if $x_i - x_j < 0$, 0 if $x_i - x_j = 0$, and 1 if $x_i - x_j > 0$. Consequently, $\mathbf{B}(\mathbf{x})\mathbf{x}$ is only dependent on the order of the elements $\mathbf{x}$, not on the values themselves. Therefore, the space $\mathbf{x} \in \mathbb{R}^n$ can be subdivided into polyhedrons such that all $\mathbf{x}$ within a polyhedron have the same ordering of the values in $\mathbf{x}$ and thus the same values of $\text{sign}(x_i - x_j)$. Then, within the polydron $\rho(\mathbf{x}) = \mathbf{x}^\top\mathbf{B}(\mathbf{x})\mathbf{x}$ is linear in $\mathbf{x}$ so that the Stress function is quadratic in $\mathbf{x}$. Therefore, if the minimum of this quadratic function yields an update with the same ordering (and thus in the same polyhedron), then the update is a local minimum and the next iteration shows an improvement in Stress of zero. This dependency of the update on the rank order of the coordinates was first observed in De Leeuw and Heiser (1977). We can illustrate this effect by applying unidimensional scaling to the cola data by running the code below.

```
R> res1 <- mds(dis, ndim = 1, verbose = TRUE)

Iteration:    1  Stress (raw):   0.21115017  Difference:    0.02842097
Iteration:    2  Stress (raw):   0.21099866  Difference:    0.00015151
Iteration:    3  Stress (raw):   0.21099866  Difference:    0.00000000
```

In this example, only three iterations are performed. The final iteration has zero improvement. In general, the number of iterations may vary, but generally it is low (smaller than 10).

Defays (1978) showed that a consequence of the exclusive dependency on the rank order of the coordinates is that the unidimensional scaling problem is combinatorial in nature. Therefore, a single run of the MDS algorithm is simply not good enough. A crude solution is to simply try out all possible $n!$ rank orders of the coordinates, and retain the best. Obviously, this crude approach is only feasible for small n. Other successful solutions that aim at finding a global minimum for unidimensional scaling includes the unidimensional smoothing approach of Pliner (1996) and its multidimensional extension by Groenen, Heiser, and Meulman (1999). Using dynamic programming in a combinatorial approach, Hubert and Golledge (1981) were able to guarantee a global optimum while at the same time reducing the complexity from $O(n!)$ to $O(2^n)$, which should be feasible on current computers for $n < 35$.

Alternatively, one could use multiple random starts and retain the best solution as a candidate global minimum. The function `multistart()` can be used to achieve this. For example,

```
R> multistart <- function(dis, n.starts = 1000, seed = NULL, ...) {
+    stress <- rep(0, n.starts + 1)
+    res <- mds(dis, init = "torgerson", ...)
+    stress[n.starts + 1] <- res$stress
+    stress.best <- res$stress
+    x <- res$conf
+    if (!is.null(seed)) set.seed(seed)
+    for (i in 1:n.starts) {
+      res <- mds(dis, init = "random", ...)
+      stress[i] <- res$stress
+      if (stress[i] < stress.best){
+        stress.best <- stress[i]
+        x <- res$conf
+      }
+    }
+    res <- mds(dis, init = x, ...)
+    return(list(res = res, stress = stress))
+  }
```

The default number of random starts (`n.starts = 1000`) will generally be too low to guarantee a globally optimal solution. However, this default usually gives a reasonable candidate global minimum. The code below computes multiple random starts for unidimensional scaling of the cola switching data.

```
R> res.multi <- multistart(dis, ndim = 1, seed = 1234)
R> x <- res.multi$res$conf
R> par(mfrow = c(2, 2))
R> plot(rep(0, length(x)), x, type = "p", pch = 20, xaxt = "n", xlab = "",
+    las = 1, ylab = "Dimension 1", xlim = c(-1, 1), col = "red")
R> text(rep(0, length(x)), x, labels = rownames(x), pos = 4)
R> plot(res.multi$res, plot.type = "Shepard", las = 1,
+    sub = paste("Stress-1 =", formatC(res.multi$res$stress, digits = 6)))
R> hist(res.multi$stress, las = 1, xlab = "Stress-1")
R> abline(v = res.multi$res$stress, col = "red")
R> abline(v = res.multi$stress[1], col = "blue")
```

The situation for *full dimensional scaling* ($p = n-1$) can be seen as the complete opposite from unidimensional scaling. De Leeuw (1993) showed that full dimensional scaling necessarily has a global minimum. This can be seen as follows. First of all, note that the squared Euclidean distance can also be written as

$$d_{ij}^2(\mathbf{X}) \quad = \quad = \mathrm{tr}\mathbf{X}^\top\mathbf{A}_{ij}\mathbf{X} = \mathrm{tr}\mathbf{A}_{ij}\mathbf{X}\mathbf{X}^\top = \mathrm{tr}\mathbf{A}_{ij}\mathbf{C} \tag{18}$$

with $\mathbf{C}$ an $n \times n$ matrix that is constrained to be in the convex cone of positive definite matrices. Then, raw Stress can be written as

$$\sigma^2_{\mathrm{raw}}(\mathbf{C}) \quad = \quad \sum_{i<j} w_{ij}\delta_{ij}^2 + \sum_{i<j} w_{ij}\mathrm{tr}\mathbf{A}_{ij}\mathbf{C} - 2\sum_{i<j} w_{ij}\delta_{ij}(\mathrm{tr}\mathbf{A}_{ij}\mathbf{C})^{1/2}. \tag{19}$$

As a function of $\mathbf{C}$, the quadratic Euclidean distance is linear and $-d_{ij}(\mathbf{C}) = -(\mathrm{tr}\mathbf{A}_{ij}\mathbf{C})^{1/2}$ is convex. Therefore, $\sigma^2_{\mathrm{raw}}(\mathbf{C})$ is a convex function in $\mathbf{C}$ over the convex constraint set of positive semi-definite matrices. It is well known that minimizing a convex function over a convex set yields a global minimum, and, thus, full dimensional scaling has a global minimum.

Even though in full dimensional scaling, the initial rank of $\mathbf{C}$ may be $n-1$, it often happens that at convergence the rank of $\mathbf{C}$ is smaller than $n-1$. In fact, numerical experiments show

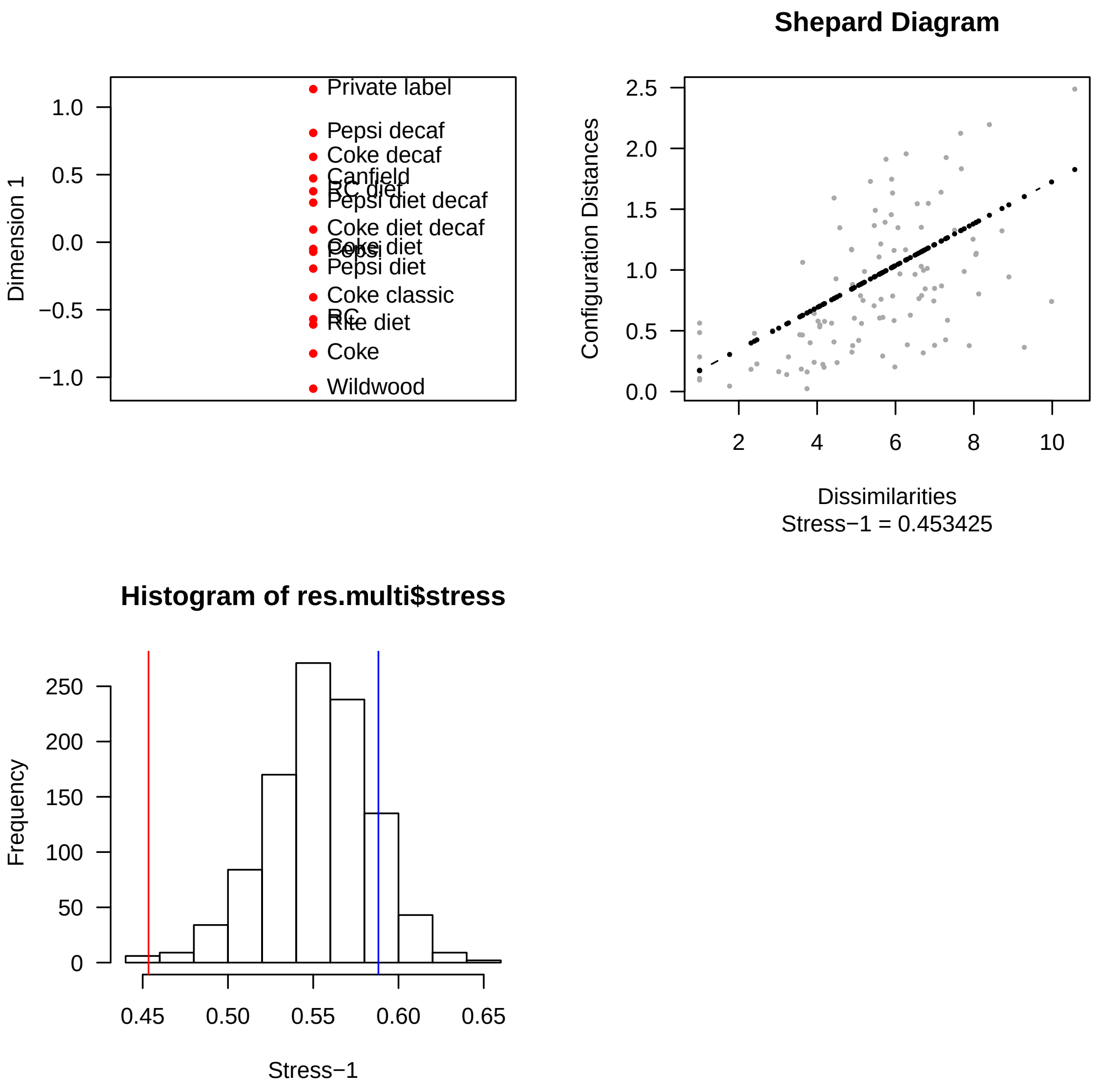

Figure 3: Unidimensional scaling plot of cola brand switching data (top left-hand panel), the Shepard plot (top right-hand panel), and a histogram of Stress-1 values of 1000 random starts.

that the maximum rank is never larger than the number of positive eigenvalues in classical scaling. The Gower conjecture (De Leeuw, Groenen, and Mair 2016a) states that this is always the case, but so far no proof has been found. Recently, De Leeuw confessed to the first author that he gave up on finding a proof for this conjecture.

More technical details on full dimensional scaling and its properties is given by De Leeuw *et al.* (2016a). One of the properties they derive is that $\mathbf{V} - \mathbf{B}(\mathbf{C})$ must be positive semi-definite for a full dimensional scaling solution. Thus, if this matrix is positive semi-definite, then the solution is a full dimensional scaling solution and hence a global minimum.

For other dimensionalities, the local minimum problem seems to be more severe whenever p is small. In addition, the problem appears to be data dependent, (see Groenen and Heiser 1996, for more details).

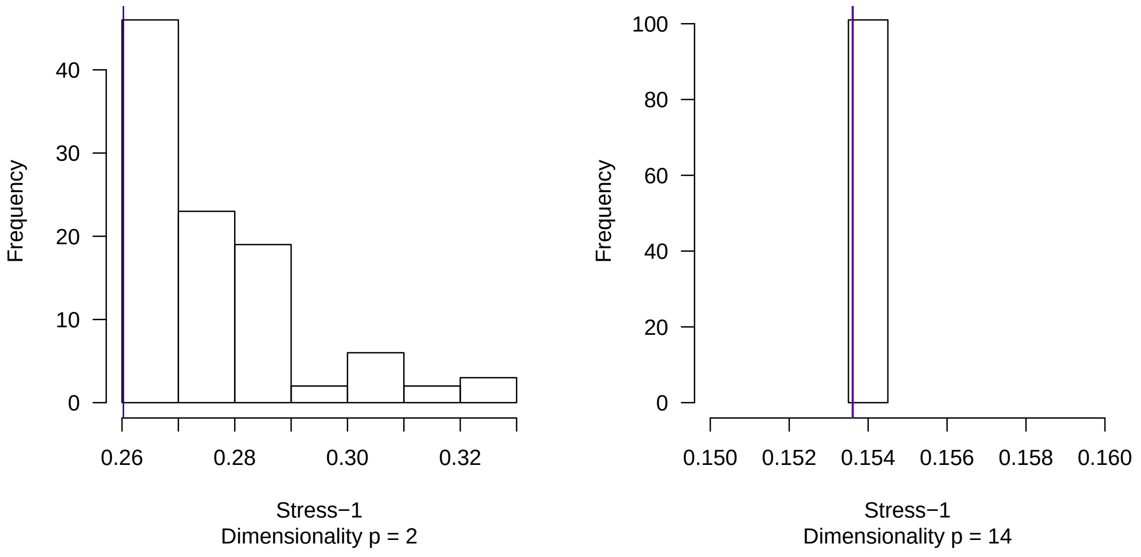

Figure 4: Histograms of Stress-1 values of 100 random starts for $p = 2$ (left-hand panel) and full dimensional scaling with $p = 14$ (right-hand panel). The vertical blue lines gives Stress-1 obtained by a classical MDS start and the red vertical lines give Stress-1 value of the candidate global minimum.

For the cola switching data, the code below performs multistart for $p = 2$ and full dimensional scaling with $p = 14$. The respective histograms of the Stress-1 values for 100 random starts are presented in Figure 4.

```
R> res.multi.2  <- multistart(dis, n.starts = 100, ndim =  2, seed = 5678)
R> res.multi.14 <- multistart(dis, n.starts = 100, ndim = 14, seed = 9012)
R> par(mfrow = c(1, 2))
R> hist(res.multi.2$stress, las = 1, xlab = "Stress-1",
+    main = "", sub = "Dimensionality p = 2")
R> abline(v = res.multi.2$res$stress, col = "red")
R> abline(v = res.multi.2$stress[1], col = "blue")
R> tt <- hist(res.multi.14$stress, las = 1, xlab = "Stress-1",
+    main = "", sub = "Dimensionality p = 14",
+    xlim = c(0.15, 0.16), breaks = c(0.1535, 0.1545))
R> abline(v = res.multi.14$res$stress, col = "red")
R> abline(v = res.multi.14$stress[1], col = "blue")
```

One can clearly see that for $p = 2$ the algorithm stops at different Stress-1 values, whereas, in accordance with the theory, full dimensional scaling with $p = 14$ yields a unique solution irrespective of the type of start. For $p = 2$, the classical scaling start in Figure 1 yielded a Stress-1 value of 0.263688 whereas the best out of 100 multiple random starts yields a slightly lower Stress-1 value of 0.260253. The difference between the two configurations can be seen by applying a *generalized Procrustes transformation* through the `Procrustes()` function in **smacof**. This Procrustes transformation transforms the best multistart solution to the classical scaling start using translation, scaling, and rotation. The code below produces Figure 5 with the classical scaling start (light blue points) and the best random start (light red points).

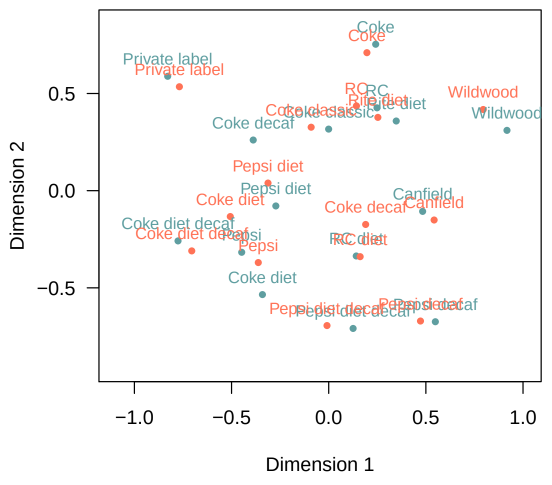

Figure 5: The two-dimensional solution obtained by a classical scaling start (light-blue points, see Figure 1) and the best of 100 random starts (light red points).

The main differences concern the locations of Coke diet and Coke decaf. For interpretation, it would be better to interpret the light red points as their Stress-1 is lower. However, the visible relations are hardly affected.

```
R> res.procr <- Procrustes(res2$conf, res.multi.2$res$conf)
R> plot(res.procr, las = 1, main = "")
```

6. The use of weights

The introduction of weights w_{ij} by De Leeuw did not only complicate the problem, but facilitated the following practices.

1. Handling missing data is done by specifying $w_{ij} = 0$ for missings and 1 otherwise thereby ignoring the error corresponding to the missing dissimilarities.
2. Correcting for nonuniform distributions of the dissimilarities to avoid dominance of the most frequently occurring dissimilarities.
3. Mimicking alternative fit functions for MDS by minimizing Stress with w_{ij} being a function of the dissimilarities.
4. Using a power of the dissimilarities to emphasize the fitting of either large or small dissimilarities.
5. Special patterns of weights for specific models.
6. Using a specific choice of weights to avoid nonuniqueness.

In the following, we assume that the weight matrix $\mathbf{W}$ is irreducible, that is, it is not possible to partition the objects in sets such that all between set $w_{ij} = 0$. Should $\mathbf{W}$ be reducible, then one should do separate MDS analyses for each of the sets of objects.

6.1. Weighting for uniform distribution of dissimilarities

An important use of weights has to do with the distribution of the δ_{ij}'s. Consider the histogram of the dissimilarities of the cola data (second row, left-hand panel of Figure 6). Here, the majority of the δ_{ijs}s are between 4 and 7. Consequently, these values will on average be better fit by Stress. The corresponding Shepard diagram shows that these values are indeed quite well-fit, whereas the few small and large dissimilarities have relatively large errors.

The **smacof** package has the function `dissWeights()` that with the option `type = "unif"` ensures all values to be equally important. Consequently, dissimilarities that are in a dense area are down weighted, and those in less dense areas receive larger weights. The corresponding weighted empirical distribution function and weighted histograms are in the right column of Figure 6. The corresponding Shepard diagram shows indeed less error for the extremes (that had few observations) and more error for the more densely middle values of the dissimilarities. The code below can be used to create Figure 6.

```
R> w <- dissWeights(dis, type = "unif")
R> ind <- order(as.vector(dis))
R> res.unw <- mds(dis)
R> res.wgt <- mds(dis, weightmat = w)
R> par(mfrow = c(4, 2))
R> plot(dis[ind], 1:length(dis)/length(dis), type = "l", las = 1,
+    col = "blue", main = "EDF", xlab = "Dissimilarities",
+    ylab = "Proportion")
R> plot(dis[ind], cumsum(w[ind])/sum(w), type = "l", las = 1,
+    col = "blue", main = "Weighted EDF", xlab = "Dissimilarities",
+    ylab = "Proportion")
R> plot(res.unw, plot.type = "histogram", las = 1,
+    main = "Unweighted histogram", border = "blue")
R> plot(res.wgt, plot.type = "histogram", las = 1,
+    main = "Weighted histogram", border = "blue")
R> plot(res.unw, las = 1, col = "red")
R> plot(res.wgt, las = 1, col = "red")
R> plot(res.unw, plot.type = "Shepard", ylim = c(0, 2))
R> plot(res.wgt, plot.type = "Shepard", ylim = c(0, 2))
```

We may conclude that if the dissimilarity matrix is large, and the distribution of dissimilarities is not uniform, it may be wise to correct the MDS solution by supplying weights in such a way that the weighted distribution of the dissimilarities becomes approximately uniform. In general, we believe that the larger n, the more important it becomes to apply this correction.

6.2. Mimicking other MDS loss functions

S-Stress

Another idea for using the weights stems from Heiser (1988). There, it was proposed to use the weights to mimick other loss functions. Consider the *S-Stress* loss function

$$\sigma^2_{\text{S-Stress}}(\mathbf{X}) = \sum_{i<j}(\delta^2_{ij} - d^2_{ij}(\mathbf{X}))^2. \tag{20}$$

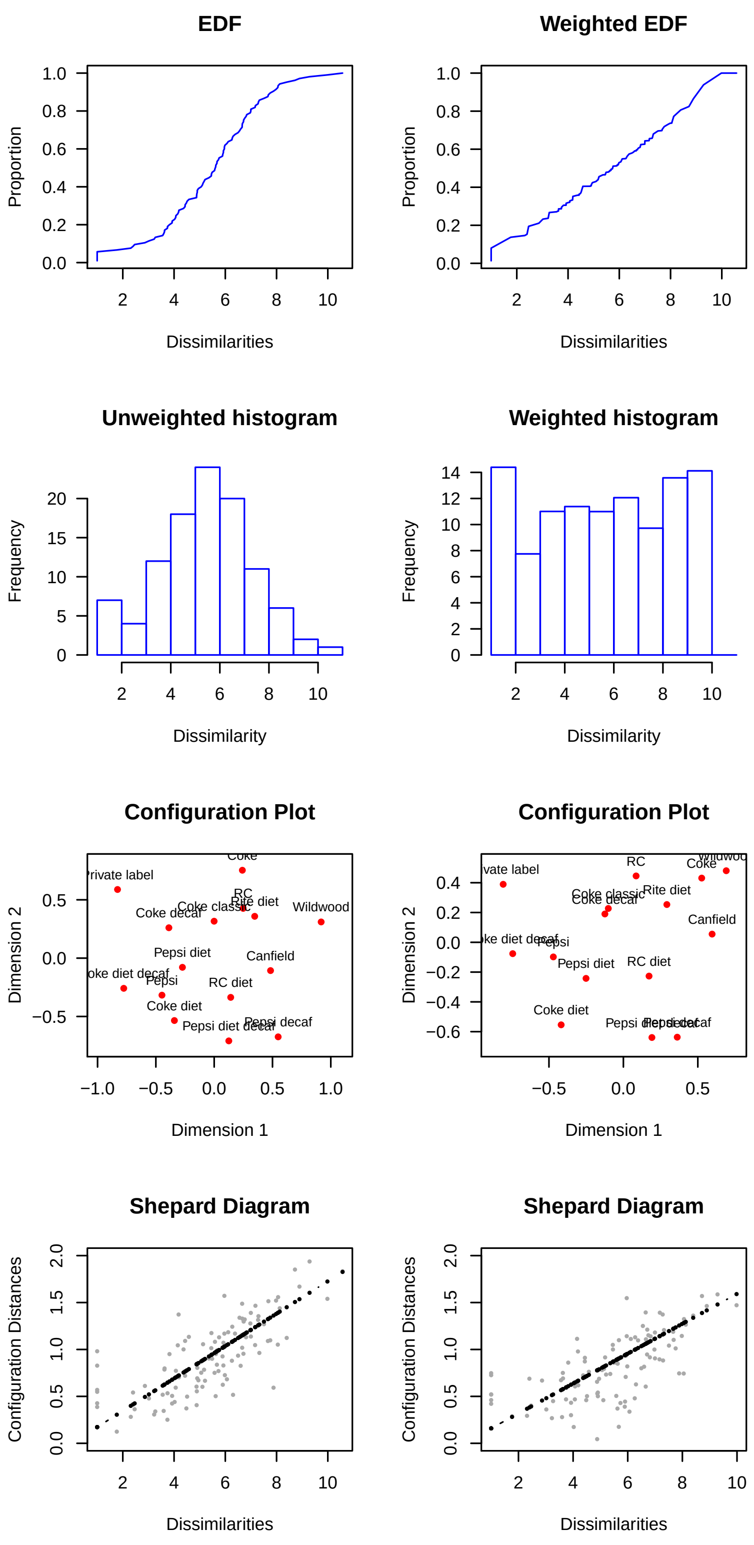

Figure 6: Empirical distribution, histogram, configuration, and Shepard diagram of the cola dissimilarities with equal weights (left column) and weights that make the weighted density of the cola dissimilarities uniform (right column).

It is not hard to see that

$$(\delta_{ij}^2 - d_{ij}^2(\mathbf{X}))^2 = (\delta_{ij} - d_{ij}(\mathbf{X}))^2(\delta_{ij} + d_{ij}(\mathbf{X}))^2 \approx (\delta_{ij} - d_{ij}(\mathbf{X}))^2 4\delta_{ij}^2 \tag{21}$$

where the approximation holds if $d_{ij}(\mathbf{X})$ is close to δ_{ij}. Therefore, choosing $w_{ij} = 4\delta_{ij}^2$ causes the minimization of raw Stress to approximate S-Stress.

Polynomial stress

A similar approximation can be done for *polynomial Stress*, that is,

$$\sigma_{\text{poly}}^2(\mathbf{X}) = \sum_{i<j} w_{ij}(\delta_{ij}^r - d_{ij}^r(\mathbf{X}))^2. \tag{22}$$

A first order Taylor approximation of $d_{ij}^r(\mathbf{X})$ can be given by $d_{ij}^r(\mathbf{Y}) + r(d_{ij}(\mathbf{X}) - d_{ij}^r(\mathbf{Y}))d_{ij}^{r-1}(\mathbf{Y})$. Assuming again that $d_{ij}(\mathbf{X})$ is close to δ_{ij}, the Taylor approximation reduces to $\delta_{ij}^r + r(d_{ij}(\mathbf{X}) - \delta_{ij})\delta_{ij}^{r-1}$ so that a residual becomes

$$\delta_{ij}^r - \delta_{ij}^r - r(d_{ij}(\mathbf{X}) - \delta_{ij})\delta_{ij}^{r-1} = -r(d_{ij}(\mathbf{X}) - \delta_{ij})\delta_{ij}^{r-1}. \tag{23}$$

Hence, choosing the weights as $w_{ij} = r^2\delta_{ij}^{2r-2}$ causes raw Stress to resemble $\sigma_{\text{poly}}^2(\mathbf{X})$. It is not so difficult to see that for $r > 1$, the raw-Stress function puts more emphasis on estimating large δ_{ij}, whereas for $r < 1$ the smaller δ_{ij}s will be better fitted by raw Stress.

Multiscale

The *multiscale* loss function of Ramsay (1977) is given by

$$\sigma_{\text{mult}}^2(\mathbf{X}) = \sum_{i<j}(\log(\delta_{ij}) - \log(d_{ij}(\mathbf{X})))^2 = \sum_{i<j} \log^2\left(\frac{d_{ij}(\mathbf{X})}{\delta_{ij}}\right). \tag{24}$$

The $\log(x)$ can be approximated by $x - 1$ so that

$$\log\left(\frac{d_{ij}(\mathbf{X})}{\delta_{ij}}\right) \approx 1 - \frac{d_{ij}(\mathbf{X})}{\delta_{ij}}. \tag{25}$$

Therefore, choosing $w_{ij} = \delta_{ij}^{-2}$ ensures that raw Stress becomes

$$\sigma_{\text{raw}}^2(\mathbf{X}) = \sum_{i<j} \delta_{ij}^{-2}(\delta_{ij} - d_{ij}(\mathbf{X}))^2 = \sum_{i<j}\left(1 - \frac{d_{ij}(\mathbf{X})}{\delta_{ij}}\right)^2$$

and thus approximates $\sigma_{\text{mult}}^2(\mathbf{X})$.

Sammon mapping

In the machine learning literature, MDS is often referred to as *Sammon mapping* after Sammon (1969). The loss function used by Sammon is raw Stress with weights $w_{ij} = \delta_{ij}^{-1}$.

6.3. Power weights

The previous subsection showed that power Stress can be mimicked by setting $w_{ij} = \delta_{ij}^q$ for some $q > 0$. Buja and Swayne (2002) do exactly this. For large q, the fitting of the larger

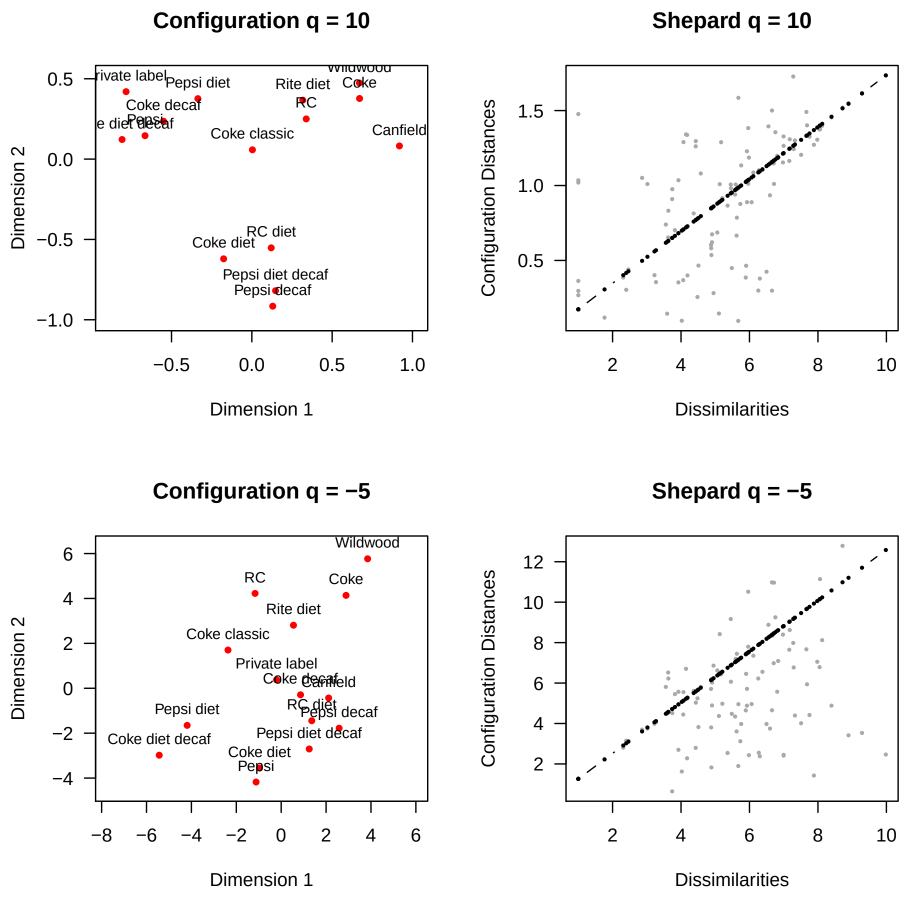

Figure 7: MDS solutions of cola switching data with power weight $q = 10$ (top row) and $q = -5$ (bottom row)

dissimilarities is being emphasized, whereas for q close to 0, the fitting of smaller dissimilarities is emphasized. The code below shows the effect of the power weights for $q = 10$ (emphasizing the larger dissimilarities) and $q = -5$ (emphasizing the smaller dissimilarities). Note that we first removed the effect of a nonuniform distribution of the dissimilarities by using `type = "unifpower"` of the function `dissWeights()`. The Shepard diagrams for these two cases in Figure 7 show almost zero error for the large dissimilarities when using $w_{ij} = \delta_{ij}^{10}$ and almost zero errors for the smaller dissimilarities for $w_{ij} = \delta_{ij}^{-5}$. The configuration for $q = 10$, shows as the main distinction the clustering into two groups: one consisting of Coke Diet, Riet Diet, Pepsi Diet, and Pepsi Decaf against a group of the remaining soft drinks. The difference in overall scaling of between the two sets of points is due to the automatic scaling of the $\hat{d}_{ij}$s in `mds()` to $\eta_{\delta}^2 = \sum_{i<j} w_{ij}\hat{d}_{ij}^2 = n(n-1)/2$. Therefore, one should interpret the relative differences in the Shepard diagram.

```
R> par(mfrow = c(2, 2))
R> w <- dissWeights(dis, type = "unifpower", power = 10)
R> w <- w * (n * (n - 1) * 0.5/sum(w^2))^0.5
R> res.5 <- mds(dis, weightmat = w,  ndim = 2)
R> plot(res.5, las = 1, main = "Configuration q = 10", col = "red")
```

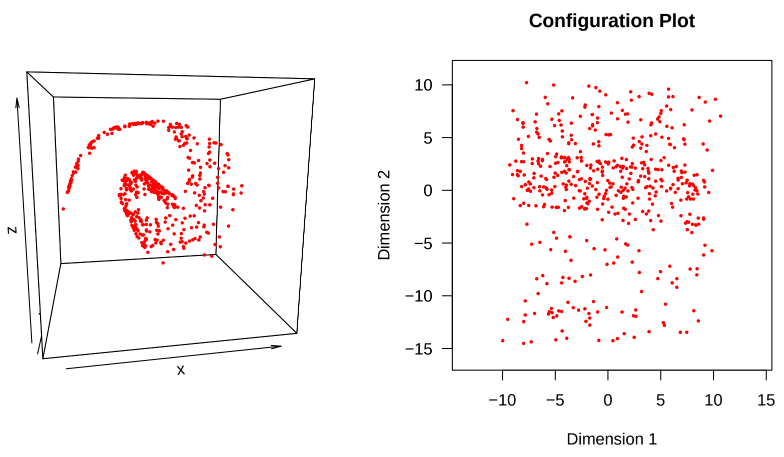

Figure 8: Points on a Swiss roll (left panel) and 2D MDS solution (right-hand panel).

```
R> plot(res.5, plot.type = "Shepard", las = 1, main = "Shepard q = 10")
R> w <- dissWeights(dis, type = "unifpower", power = -5)
R> w <- w * (n * (n - 1) * 0.5/sum(w^2))^0.5
R> res.minus.5 <- mds(dis, weightmat = w,  ndim = 2)
R> plot(res.minus.5, las = 1, main = "Configuration q = -5", col = "red")
R> plot(res.minus.5, plot.type = "Shepard", las = 1, main = "Shepard q = -5")
```

Another application of weights concerns focusing on fitting dissimilarities that are nearby. This has been the goal of what has been called parametric mapping. To illustrate this, consider 500 points randomly distributed on a Swiss roll, which may be visualized using the **plot3D** package (Soetaert 2016) as shown in the left-hand panel of Figure 8. The option `type = "knn"` of the function `dissWeights()` computes a weight matrix such that each object i is connected to the k other objects j with the smallest δ_{ij}. The code below makes the Swiss roll, computes the Euclidean distances in 3 dimensions, calculates the weights w_{ij} corresponding to the $k = 15$ nearest neighbors, and does an MDS.

```
R> par(mfrow = c(1, 2))
R> n <- 500
R> y <- runif(n, min = -1, max = 1)
R> theta <- runif(n)
R> x <- theta * cos(3 * pi * theta)
R> z <- theta * sin(3 * pi * theta)
R> lims <- c(-1.2, 1.2)
R> points3D(x, y, z, col = "red", pch = 20, cex = 0.5, phi = 10,
+    theta = -10, xlim = lims, ylim = lims, zlim = lims)
R> X <- cbind(x, y, z)
R> dis <- dist(X)
R> w <- dissWeights(dis, type = "knn", k = 15)
R> w <- w * (n * (n - 1) * 0.5/sum(w^2))^0.5
R> res <- mds(dis, weightmat = w, init = "torgerson")
R> plot(res, col = "red", pch = 20, cex = 0.5, las = 1,
+    label.conf = list(label = FALSE))
```

While using only the smallest 3.5% of the distances in 3 dimensions, MDS is able to unroll the Swiss roll in 2D.

6.4. Solving nonuniqueness for sparse data

A powerful application of weights for sparse data has been proposed by Buja, Swayne, Littman, Dean, Hofmann, and Chen (2008). Consider a social network of weddings between 16 Florentine families in `flo` from the **network** package (Butts 2008). In these data, the $\delta_{ij} = 1$ and $w_{ij} = 1$ if there is a marriage between families i and j, and otherwise the $\delta_{ij} = 0$ and $w_{ij} = 0$.

Consider a family that is connected to only one other family, for example, the Pazzi family that is connected only to the Salviati family. Then, the point for the Pazzi family could be located anywhere on a circle with distance 1 from the the Salviati family without changing Stress. This form of nonuniqueness is unpleasant as it may produce a clutter of points. The idea of Buja *et al.* (2008) is to introduce adapted weights w^*_{ij} and dissimilarities δ^*_{ij} such that a small force is created that is outward oriented. Let c be a small constant, say $c = 0.1$ or $c = 0.01$. Then, choosing

$$w^*_{ij} = \begin{cases} 1 & \text{if } \delta_{ij} \text{ is not missing,} \\ c & \text{if } \delta_{ij} \text{ is missing,} \end{cases} \qquad \delta^*_{ij} = \begin{cases} 1 & \text{if } \delta_{ij} \text{ is not missing,} \\ 1/c & \text{if } \delta_{ij} \text{ is missing} \end{cases}$$

introduces these small outward forces. Then, raw Stress can be decomposed as

$$\sigma^2_{\text{raw}}(\mathbf{X}) = \sum_{i<j} w^*_{ij} \left(\delta^*_{ij} - d_{ij}(\mathbf{X})\right)^2 = \sum_{i<j} w_{ij} \left(\delta_{ij} - d_{ij}(\mathbf{X})\right)^2 + c \sum_{i<j} (1 - w_{ij}) \left(1/c - d_{ij}(\mathbf{X})\right)^2 .$$

The last term of this decomposition indicates that there should be a large distance of preferably $1/c$ between pairs of points that have missing dissimilarities although its contribution to

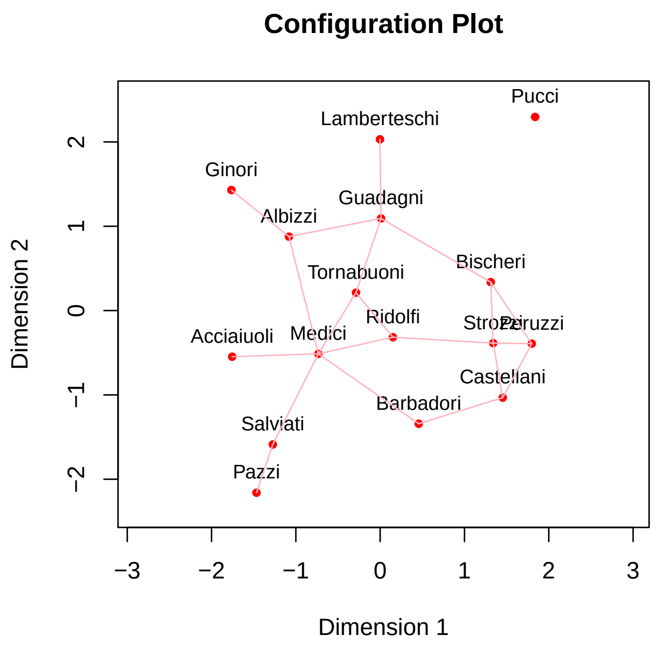

Figure 9: Marriages amongst Florentine families.

the Stress is minor due to the multiplication by the small value of c. So, the smaller c, the smaller the contribution to the fit.

The following code computes the adapted dissimilarities δ^*_{ij} and weights w^*_{ij}, does the MDS, produces Figure 9 of the MDS solution, and adds lines between families that are connected.

```
R> library("network")
R> data("flo")
R> c <- 0.1
R> dis <- flo
R> w <- flo
R> dis.star <- w + (1 - w)/c
R> w.star <- w + (1 - w) * c
R> res <- mds(dis.star, weightmat = w.star, init = "torgerson", ndim = 2)
R> plot(res, col = "red")
R> n <- nrow(res$conf)
R> k <- 0
R> w.mat <- as.matrix(w)
R> for (j in 1:(n - 1)) {
+    for (i in (j + 1):n) {
+      if (w.mat[i, j] == 1) {
+        lines(res$conf[c(i, j), 1], res$conf[c(i, j), 2], col = "lightpink")
+      }
+    }
+ }
```

The resulting MDS configuration is shown in Figure 9. We see that the Medici family was the most central one, with five connections to other families, followed by the Strozzi and Guadagni families with four connections. The families with a single connection are nicely pointing outwards. The Pucci family has no connection to other families. Note that the standard majorization algorithm would not be able to handle this case as the weight matrix $\mathbf{W}$ is reducible. The adapted weights w^*_{ij} and dissimilarities δ^*_{ij} make $\mathbf{W}^*$ irreducible, allowing the standard SMACOF algorithm to handle these data.

7. Concluding remarks

In this paper, we presented a review of several important contributions that Jan de Leeuw has made to MDS. An important contribution has been the development of the SMACOF algorithm and its algorithmic properties. The algorithm is based on the principle of majorization and as such has been one of the first instances of majorizing (or MM algorithms) under this name. Arguably, the development of majorization is one of Jan's most important scientific contributions. This optimization principle has become standard methodology at many (statistical) departments and is expanding to computer science and engineering. Although its linear convergence rate is necessarily slower than the golden standard of the Newton method, it has many applications in large scale optimization and is useful when high precision is not needed. Besides the MDS applications reviewed in this paper, majorization has been used in many other statistical techniques, such as, for example, support vector machines (Groenen, Nalbantov, Bioch, and C 2008) and logistic regression (De Leeuw 2006). It is our expectation that a decade from now majorization will be included in many nonlinear optimization textbooks.

Another key contribution of Jan de Leeuw has been the inclusion of weights for the residuals in least-squares MDS models. This inclusion has had several important practical consequences. We illustrated, for example, how using such weights to emphasize or deemphasize certain dissimilarities has led to important usages and insights in the area of MDS.

The present review has been limited to Euclidean distances and raw Stress. It seems that this form of MDS has become the most popular form of MDS in practice. A Bayesian approach for MDS was proposed by Oh and Raftery (2001) but it has not yet gained the popularity of least-squares MDS.

It is expected that De Leeuw stays active in the area of algorithmic and theoretical properties of MDS with recent contributions on his RPubs page (De Leeuw 2016) a majorization algorithm for MDS with power Stress (De Leeuw, Groenen, and Mair 2016c) and a small power (De Leeuw, Groenen, and Mair 2016b). Consequently, we foresee several complex, original, and, most importantly, interesting contributions to the field in the near and distant future.

References

Bell DR, Lattin JM (1998). "Shopping Behavior and Consumer Preference for Store Price Format: Why 'Large Basket' Shoppers Prefer EDLP." *Marketing Science*, **17**, 66–88. `doi:10.1287/mksc.17.1.66`.

Borg I, Groenen PJF, Jehn KA, Bilsky W, Schwartz SH (2011). "Embedding the Organizational Culture Profile into Schwartz's Theory of Universals in Values." *Journal of Personnel Psychology*, **10**, 1–12. `doi:10.1027/1866-5888/a000028`.

Buja A, Swayne DF (2002). "Visualization Methodology for Multidimensional Scaling." *Journal of Classification*, **19**, 7–44. `doi:10.1007/s00357-001-0031-0`.

Buja A, Swayne DF, Littman ML, Dean N, Hofmann H, Chen L (2008). "Data Visualization with Multidimensional Scaling." *Journal of Computational and Graphical Statistics*, **17**(2), 444–472. `doi:10.1198/106186008x318440`.

Butts CT (2008). "**network**: A Package for Managing Relational Data in R." *Journal of Statistical Software*, **24**(2), 1–36. `doi:10.18637/jss.v024.i02`.

Carroll JD, Chang JJ (1970). "Analysis of Individual Differences in Multidimensional Scaling via an n-Way Generalization of 'Eckart-Young' Decomposition." *Psychometrika*, **35**, 283–320. `doi:10.1007/bf02310791`.

De Leeuw J (1977). "Applications of Convex Analysis to Multidimensional Scaling." In JR Barra, F Brodeau, G Romier, B van Cutsem (eds.), *Recent Developments in Statistics*, pp. 133–145. North-Holland, Amsterdam, The Netherlands.

De Leeuw J (1988). "Convergence of the Majorization Method for Multidimensional Scaling." *Journal of Classification*, **5**, 163–180. `doi:10.1007/bf01897162`.

De Leeuw J (1993). "Fitting Distances by Least Squares." *Technical Report 130*, Interdivisional Program in Statistics, UCLA, Los Angeles.

De Leeuw J (2006). "Principal Component Analysis of Binary Data by Iterated Singular Value Decomposition." *Computational Statistics & Data Analysis*, **50**(1), 21–39. doi:10.1016/j.csda.2004.07.010.

De Leeuw J (2016). Date accessed 06-06-16, URL http://rpubs.com/deleeuw.

De Leeuw J, Groenen PJF (1997). "Inverse Multidimensional Scaling." *Journal of Classification*, **14**(1), 3–21. doi:10.1007/s003579900001.

De Leeuw J, Groenen PJF, Mair P (2016a). "Full Dimensional Scaling." doi:10.13140/rg.2.1.1038.4407.

De Leeuw J, Groenen PJF, Mair P (2016b). "Minimizing qStress for Small q." doi:10.13140/rg.2.1.4843.1764.

De Leeuw J, Groenen PJF, Mair P (2016c). "Minimizing rStress Using Majorization." doi:10.13140/rg.2.1.3871.3366.

De Leeuw J, Heiser WJ (1977). "Convergence of Correction-Matrix Algorithms for Multidimensional Scaling." In JC Lingoes, EE Roskam, I Borg (eds.), *Geometric Representations of Relational Data*, pp. 735–752. Mathesis Press, Ann Arbor, MI.

De Leeuw J, Heiser WJ (1980). "Multidimensional Scaling with Restrictions on the Configuration." In PR Krishnaiah (ed.), *Multivariate Analysis*, volume V, pp. 501–522. North-Holland, Amsterdam, The Netherlands.

De Leeuw J, Mair P (2009). "Multidimensional Scaling Using Majorization: SMACOF in R." *Journal of Statistical Software*, **31**(3), 1–30. doi:10.18637/jss.v031.i03.

De Leeuw J, Mair P, Groenen PJF (2016d). *Multidimensional Scaling in R: SMACOF*. URL https://cran.r-project.org/web/packages/smacof/vignettes/smacof.pdf.

De Leeuw J, Stoop I (1984). "Upper Bounds of Kruskal's Stress." *Psychometrika*, **49**, 391–402. doi:10.1007/bf02306028.

Defays D (1978). "A Short Note on a Method of Seriation." *British Journal of Mathematical and Statistical Psychology*, **31**, 49–53. doi:10.1111/j.2044-8317.1978.tb00571.x.

Gower JC (1966). "Some Distance Properties of Latent Root and Vector Methods Used in Multivariate Analysis." *Biometrika*, **53**, 325–338. doi:10.1093/biomet/53.3-4.325.

Groenen PJ, Nalbantov G, Bioch, C J (2008). "SVM-Maj: a Majorization Approach to Linear Support Vector Machines with Different Hinge Errors." *Advances in Data Analysis and Classification*, **2**(1), 17–43. doi:10.1007/s11634-008-0020-9.

Groenen PJF (1993). *The Majorization Approach to Multidimensional Scaling: Some Problems and Extensions*. DSWO Press, Leiden, The Netherlands.

Groenen PJF, Heiser WJ (1996). "The Tunneling Method for Global Optimization in Multidimensional Scaling." *Psychometrika*, **61**, 529–550. doi:10.1007/bf02294553.

Groenen PJF, Heiser WJ, Meulman JJ (1999). "Global Optimization in Least-Squares Multidimensional Scaling by Distance Smoothing." *Journal of Classification*, **16**, 225–254. doi:10.1007/s003579900055.

Heiser WJ (1988). "Multidimensional Scaling with Least Absolute Residuals." In HH Bock (ed.), *Classification and Related Methods*, pp. 455–462. North-Holland, Amsterdam.

Horan CB (1969). "Multidimensional Scaling: Combining Observations When Individuals Have Different Perceptual Structures." *Psychometrika*, **34**, 139–165. doi:10.1007/bf02289341.

Hubert LJ, Golledge RG (1981). "Matrix Reorganisation and Dynamic Programming: Applications to Paired Comparison and Unidimensional Seriation." *Psychometrika*, **46**, 429–441. doi:10.1007/bf02293800.

Hunter DR, Lange K (2004). "A Tutorial on MM Algorithms." *The American Statistician*, **39**, 30–37. doi:10.1198/0003130042836.

Kruskal JB (1964a). "Multidimensional Scaling by Optimizing Goodness of Fit to a Nonmetric Hypothesis." *Psychometrika*, **29**, 1–27. doi:10.1007/bf02289565.

Kruskal JB (1964b). "Nonmetric Multidimensional Scaling: A Numerical Method." *Psychometrika*, **29**, 115–129. doi:10.1007/bf02289694.

Oh MS, Raftery AE (2001). "Bayesian Multidimensional Scaling and Choice of Dimension." *Journal of the American Statistical Association*, **96**(455), 1031–1044. doi:10.1198/016214501753208690.

Ortega JM, Rheinboldt WC (1970). *Iterative Solutions of Nonlinear Equations in Several Variables*. Academic Press, New York.

Pliner V (1996). "Metric, Unidimensional Scaling and Global Optimization." *Journal of Classification*, **13**, 3–18. doi:10.1007/bf01202579.

Ramsay JO (1977). "Maximum Likelihood Estimation in Multidimensional Scaling." *Psychometrika*, **42**, 241–266. doi:10.1007/bf02294052.

Sammon JW (1969). "A Non-Linear Mapping for Data Structure Analysis." *IEEE Transactions on Computers*, **18**, 401–409. doi:10.1109/t-c.1969.222678.

Soetaert K (2016). ***plot3D**: Plotting Multi-Dimensional Data*. R package version 1.1, URL https://CRAN.R-project.org/package=plot3D.

Torgerson WS (1958). *Theory and Methods of Scaling*. John Wiley & Sons, New York.

Voss H, Eckhardt U (1980). "Linear Convergence of Generalized Weiszfeld's Method." *Computing*, **25**(3), 243–251. doi:10.1007/bf02242002.

Yuille AL, Rangarajan A (2003). "The Concave-Convex Procedure." *Neural Computation*, **15**(4), 915–936. doi:10.1162/08997660360581958.

Affiliation:

Patrick J. F. Groenen
Econometric Institute
Erasmus School of Economics
Erasmus University Rotterdam
P.O. Box 1738
3000 DR Rotterdam, The Netherlands
E-mail: groenen@ese.eur.nl
URL: http://people.few.eur.nl/groenen/

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
September 2016, Volume 73, Issue 8 *Submitted:* 2016-03-24
doi:10.18637/jss.v073.i08 *Accepted:* 2016-06-01