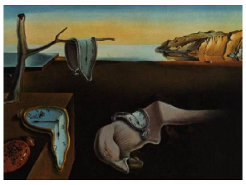


Journal of Statistical Software
April 2022, Volume 102, Issue 10. doi: 10.18637/jss.v102.i10

More on Multidimensional Scaling and Unfolding in R: smacof Version 2

Patrick Mair
Harvard University

Patrick J. F. Groenen
Erasmus University Rotterdam

Jan de Leeuw
University of California, Los Angeles

Abstract

The **smacof** package offers a comprehensive implementation of multidimensional scaling (MDS) techniques in R. Since its first publication (De Leeuw and Mair 2009b) the functionality of the package has been enhanced, and several additional methods, features and utilities were added. Major updates include a complete re-implementation of multidimensional unfolding allowing for monotone dissimilarity transformations, including row-conditional, circular, and external unfolding. Additionally, the constrained MDS implementation was extended in terms of optimal scaling of the external variables. Further package additions include various tools and functions for goodness-of-fit assessment, unidimensional scaling, gravity MDS, asymmetric MDS, Procrustes, and MDS biplots. All these new package functionalities are illustrated using a variety of real-life applications.

Keywords: multidimensional scaling, constrained multidimensional scaling, multidimensional unfolding, SMACOF, R.

1. Introduction

Multidimensional scaling (MDS; Torgerson 1952; Kruskal 1964; Borg and Groenen 2005) is a technique that represents proximities among objects as distances among points in a low-dimensional space. Multidimensional unfolding (Coombs 1964; Busing, Groenen, and Heiser 2005; Borg and Groenen 2005) is a related technique that represents input preference data as distances (among individuals and objects) in a low-dimensional space. Nowadays, MDS as well as unfolding problems are typically solved through numeric optimization. The state-of-the-art approach is called SMACOF (Stress Majorization of a Complicated Function; De Leeuw 1977)[1] and provides the user with a great amount of flexibility for specifying

[1]Originally, the "C" in SMACOF stood for "convex" which was later changed to "complicated" as the stress function is not convex.

Purpose	Function names
Goodness-of-fit	`icExplore`, `randomstress`, `permtest`
Stability	`jackmds`, `bootmds`, `confEllipse`
Constrained MDS	`smacofConstraint` (arguments `type`, `constraint.type`)
Unfolding	`unfolding` (arguments `type`, `conditionality`, `circle`, `fixed`, `fixed.coord`), `vmu`
MDS variants	`uniscale`, `gravity`, `driftVectors`, `Procrustes`
Plots	`biplotmds`
Utilities	`sim2diss`, `stress0`

Table 1: Overview of newly implemented **smacof** functions (and key arguments), grouped by their purpose.

MDS and unfolding variants. Since the first publication of the **smacof** package in R by De Leeuw and Mair (2009b), several additional MDS and unfolding approaches as well as various extensions and utility functions have been implemented, as presented in this article. We keep our elaborations fairly applied since the core technical details were already provided in the original publication.

The first part of this paper gives the reader the key ingredients of MDS, with a special focus on newly implemented dissimilarity transformation functions. This is followed by a section on MDS goodness-of-fit assessment, including various ways of assessing the stability of a solution, and a section on MDS biplots. The incorporation of optimal scaling on the external variables, as presented in a subsequent section, makes MDS an attractive tool for confirmatory research. What follows next is a detailed presentation of the recently implemented unfolding function, which adds great amounts of flexibility in model specification as compared to the original implementation. Finally, several smaller additions such as Procrustes transformation, asymmetric MDS, gravity MDS, and unidimensional scaling are presented. Table 1 gives an overview of these developments. Related R packages are mentioned in the respective sections.

2. SMACOF in a nutshell

MDS takes a symmetric dissimilarity matrix Δ of dimension $n \times n$ with non-negative elements δ_{ij} as input. These dissimilarities can be either directly observed (e.g., in an experimental setting a participant has to rate similarities between pairs of stimuli) or derived (e.g., by applying a proximity measure on a multivariate data frame). If the data are collected or derived as similarities s_{ij}, the `sim2diss` function supports users to convert them into dissimilarities δ_{ij}. Corresponding conversion formulas are given in Table 2. Additional technical details on various conversions can be found in Shepard (1957), Gower and Legendre (1986), Ramsay (1997), Esposito, Malerba, Tamma, and Bock (2000), Fleiss, Levin, and Paik (2003), Heiser and Busing (2004), and Keshavarzi, Dehghan, and Mashinchi (2009). The resulting matrix Δ can then be passed to the respective MDS functions.

SMACOF uses majorization (see De Leeuw and Mair 2009b, for details) to solve Kruskal's *stress* target function (Kruskal 1964)

$$\sigma^2(\hat{\mathbf{D}}, \mathbf{X}) = \sum_{i<j} w_{ij}(\hat{d}_{ij} - d_{ij}(\mathbf{X}))^2 \rightarrow \text{min!} \tag{1}$$

Method (argument)	Conversion formula
Correlation (`"corr"`)	$\delta_{ij} = \sqrt{1 - r_{ij}}$
Reverse (`"reverse"`)	$\delta_{ij} = \min(s_{ij}) + \max(s_{ij}) - s_{ij}$
Reciprocal (`"reciprocal"`)	$\delta_{ij} = 1/s_{ij}$
Membership (`"membership"`)	$\delta_{ij} = 1 - s_{ij}$
Rank orders (`"ranks"`)	$\delta_{ij} = \text{rank}(-s_{ij})$
Exponential (`"exp"`)	$\delta_{ij} = -\log(s_{ij}/\max(s_{ij}))$
Gaussian (`"Gaussian"`)	$\delta_{ij} = \sqrt{-\log(s_{ij}/\max(s_{ij}))}$
Transition frequencies (`"transition"`)	$\delta_{ij} = 1/\sqrt{f_{ij}}$
Co-occurrences (`"cooccurrence"`)	$\delta_{ij} = \left(1 + \frac{f_{ij}\sum_{i,j} f_{ij}}{\sum_i f_{ij} \sum_j f_{ij}}\right)^{-1}$
Gravity (`"gravity"`)	$\delta_{ij} = \sqrt{\frac{\sum_i f_{ij} \sum_j f_{ij}}{f_{ij}\sum_{i,j} f_{ij}}}$
Confusion proportions (`"confusion"`)	$\delta_{ij} = 1 - p_{ij}$
Probabilities (`"probability"`)	$\delta_{ij} = 1/\sqrt{\arcsin(p_{ij})}$
Integer value z	$\delta_{ij} = z - s_{ij}$

Table 2: Conversions of similarities into dissimilarities: similarities s_{ij}, correlations r_{ij}, frequencies f_{ij}, proportions/probabilities p_{ij}.

with $\sum_{i<j} w_{ij}\hat{d}_{ij}^2 = n(n-1)/2$ as constraint. Let us explain the components involved in this expression (i.e., w_{ij}, $\hat{d}_{ij}$, $d_{ij}(\mathbf{X})$) in more detail.

We begin with w_{ij} which denotes a non-negative a priori weight for δ_{ij}. By default, $w_{ij} = 1$. If a δ_{ij} is missing, all functions in **smacof** set the corresponding $w_{ij} = 0$ such that these entries are blanked out from optimization. Solving the stress function results in an $n \times p$ matrix $\mathbf{X}$ of point coordinates located in a p-dimensional space (p fixed a priori) with Euclidean distances

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

The $\hat{d}_{ij}$'s are the *disparities* (also called *d-hats*), collected in the $n \times n$ matrix $\hat{\mathbf{D}}$. Disparities are optimally scaled dissimilarities. That is, a transformation admissible on the assumed scale level ("measurement levels as functions"; see, e.g., Jacoby 1999) is applied. The first **smacof** package incarnation offered only two specification options: metric or non-metric. The new package version implements the following bundle of transformation functions (ordered from most restrictive to least restrictive):

- Ratio MDS: $\hat{d}_{ij} = b\delta_{ij}$.
- Interval MDS: $\hat{d}_{ij} = a + b\delta_{ij}$.
- Monotone spline MDS: $\hat{d}_{ij} = f(\delta_{ij})$ where f is an I-spline (integrated spline) transformation (Ramsay 1988) with fixed number of knots and spline degree.
- Ordinal MDS: $\hat{d}_{ij} = f(\delta_{ij})$ where f is a monotone step function. Approaches for tie handling (i.e., in case of $\delta_{ij} = \delta_{i'j'}$) are the following:

- Primary approach ("break ties"): does not require that $\hat{d}_{ij} = \hat{d}_{i'j'}$.
- Secondary approach ("keep ties tied"): requires that $\hat{d}_{ij} = \hat{d}_{i'j'}$.
- Tertiary approach: requires that the means of the tie blocks are in the correct order.

Since dissimilarities are non-negative, these monotone transformations impose non-negativity on the disparities as well.

In order to make stress scale-free, it needs to be normalized either implicitly or explicitly. SMACOF uses an *explicit* normalization using the constraint $\sum_{i<j} w_{ij}\hat{d}_{ij}^2 = n(n-1)/2$. This results in the normalized stress expression

$$\sigma_n(\hat{\mathbf{D}}, \mathbf{X}) = \frac{\sum_{i<j} w_{ij}(\hat{d}_{ij} - d_{ij}(\mathbf{X}))^2}{\sum_{i<j} w_{ij}\hat{d}_{ij}^2} = \frac{\sum_{i<j} w_{ij}(\hat{d}_{ij} - d_{ij}(\mathbf{X}))^2}{n(n-1)/2}. \tag{2}$$

Kruskal (1964) proposed an *implicit* stress normalization called "stress-1":

$$\sigma_1(\hat{\mathbf{D}}, \mathbf{X}) = \sqrt{\frac{\sum_{i<j} w_{ij}(\hat{d}_{ij} - d_{ij}(\mathbf{X}))^2}{\sum_{i<j} w_{ij}d_{ij}^2(\mathbf{X})}}. \tag{3}$$

In the MDS literature, many experiments and other MDS software in mainstream statistical packages have been using stress-1. Fortunately, there exists a simple relation between σ_n and σ_1, as shown in detail in Borg and Groenen (2005, Chapter 11). They prove that at a local minimum $\mathbf{X}^*$

$$\sigma_1(\hat{\mathbf{D}}, \mathbf{X}^*) = \sqrt{\sigma_n(\hat{\mathbf{D}}, \mathbf{X}^*)}. \tag{4}$$

Therefore, without loss of generality, we report stress-1 in all MDS functions implemented in **smacof**[2].

To illustrate MDS with different types of transformation functions we use a simple dataset from Guttman (1965). The data consist of an 8×8 matrix containing correlations of eight items in an intelligence test. First, we need to convert these similarities into dissimilarities, as all **smacof** functions operate on dissimilarities. Second, we fit four MDS versions and report the corresponding stress values.

```
R> library("smacof")
R> idiss <- sim2diss(intelligence[,paste0("T", 1:8)])
R> fitrat <- mds(idiss)
R> fitint <- mds(idiss, type = "interval")
R> fitord <- mds(idiss, type = "ordinal")
R> fitspl <- mds(idiss, type = "mspline")
R> round(c(fitrat$stress, fitint$stress, fitord$stress, fitspl$stress), 3)

[1] 0.227 0.080 0.015 0.070
```

[2]From now on, whenever we say "stress", we refer to "stress-1".

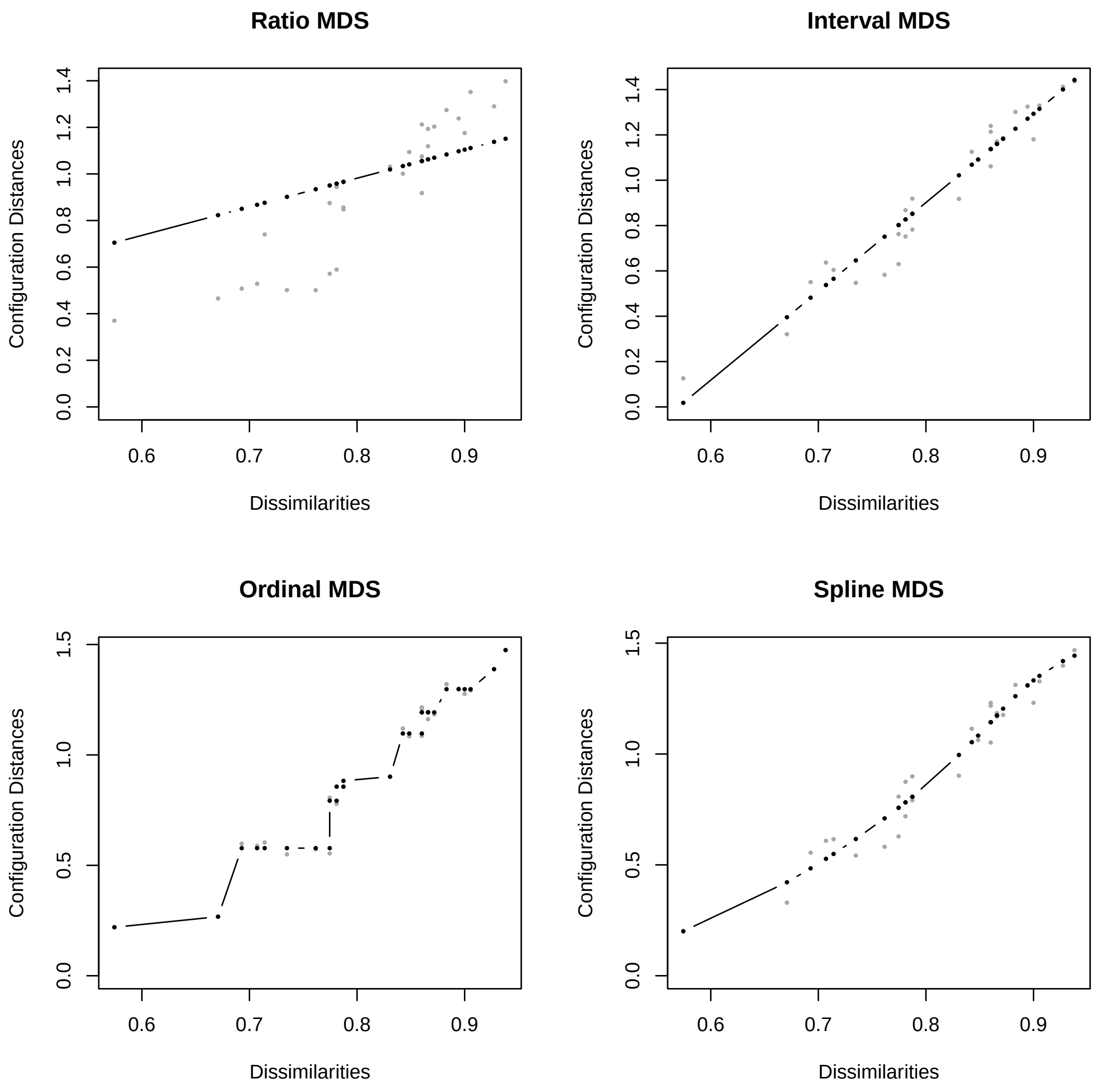


Figure 1: Shepard diagrams for four different dissimilarity transformations.

The variability in the stress values across the different transformations is due to the differing amounts of flexibility provided by each of the transformations. Figure 1 shows the Shepard diagrams involving four different transformation functions. These diagrams plot the observed dissimilarities δ_{ij} against the fitted distances $d_{ij}(\mathbf{X})$, and map the disparities $\hat{d}_{ij}$ into the point cloud (De Leeuw and Mair 2015).

The option to apply various dissimilarity transformations is one of the advantages of the SMACOF framework compared to classical scaling (Torgerson 1952) as implemented in **stats**' `cmdscale`. In **smacof**, these transformation functions are now also available for all kinds of three-way MDS models (`indscal` and `idioscal` functions), as well as for confirmatory MDS and unfolding, as described further below.

3. Tools for goodness-of-fit assessment

Mair, Borg, and Rusch (2016) give an extensive treatment of how to assess goodness-of-fit in MDS. Here we present some recently implemented utility functions that support users with this task.

3.1. Configuration starting values

Optimizing the stress function in Equation 1 through majorization leads to local minima problems since the stress surface is generally bumpy. By default, **smacof** uses a classical scaling solution to support the algorithm with a reasonable starting configuration. This is not necessarily the best choice because it does not always lead to the lowest stress value. A common heuristic strategy is to try out several random starting configurations, and pick the fit with the lowest stress value.

To illustrate this approach, we use one of the classical MDS textbook datasets from Wish (1971), containing similarity ratings for 12 countries, for which we fit a ratio MDS. The first line converts the input similarities into dissimilarities by subtracting each s_{ij} from 7 (cf. last row in Table 2).

```
R> WishD <- sim2diss(wish, method = 7)
R> fitWish <- mds(WishD)
```

This leads to a stress value of 0.2185. Now we fit 100 additional ratio MDS models based on different random starts, and report the lowest stress value.

```
R> set.seed(123)
R> stressvec <- rep(NA, 100)
R> fitbest <- mds(WishD, init = "random")
R> stressvec[1] <- fitbest$stress
R> for(i in 2:100) {
+    fitran <- mds(WishD, init = "random")
+    stressvec[i] <- fitran$stress
+    if (fitran$stress < fitbest$stress) fitbest <- fitran
+ }
R> round(fitbest$stress, 4)

[1] 0.2178
```

This solution leads to a slightly lower stress value than the one obtained with a classical scaling start. From a purely statistical point of view the user would normally decide to go with this solution. However, from a more substantive perspective, interpretability plays an important role. For instance, there might be a solution with a reasonably low stress value (but not the lowest) which leads to better interpretability. This issue is studied in detail in Borg and Mair (2017) who propose the following strategy (p. 21–22):

1. Run an MDS analysis with a set of different initial configurations (e.g., using many random configurations).

2. Save all resulting MDS solutions and their stress values.
3. Use Procrustean fitting (see Section 7.4) to eliminate all meaningless differences (i.e., differences not driven by the data) among the MDS solutions.
4. Compute the similarity of each pair of MDS configurations.
5. Analyze the similarity structure of the MDS configurations with two-dimensional MDS (to visualize the similarity structure) or cluster analysis (to identify types of MDS configurations).
6. For each type of MDS configuration with a reasonably low stress value, plot one prototypical MDS solution and check its interpretability.
7. Pick the MDS solution that is acceptable in terms of stress value and gives the best interpretation.

These steps to explore initial configurations are implemented in the `icExplore` function. Again, we fit 100 ratio MDS models with random starts and save all fitted MDS objects (`returnfit` argument).

```
R> set.seed(123)
R> icWish <- icExplore(WishD, nrep = 100, returnfit = TRUE)
R> plot(icWish, main = "IC Plot Wish")
```

Figure 2 shows the configuration plot of the 100 MDS solutions based on random starts (cf. Step 5).[3] The larger the size of the label, the larger the stress value and, therefore, the worse the fit of the solution. Based on this plot the user can extract various solutions that fit satisfactorily, plot the configurations, and interpret the solutions.

3.2. Stress norms and permutation tests

Regarding stress-1 values (on a percentage scale), Kruskal (1964, p. 3) says that "our experience with experimental and synthetic data suggests the following verbal evaluation: 20% poor, 10% fair, 5% good, 2.5% excellent, 0% perfect". In subsequent years, these rules of thumb have been applied in a somewhat mechanical manner. This is problematic for various reasons (see Mair *et al.* 2016; Borg, Groenen, and Mair 2018); one of which is that the stress value depends on n, as is obvious in Equation 1: the larger n, the larger the stress value[4].

This issue was recognized in the early days of MDS. Throughout the 1970s various researchers have studied this phenomenon by means of Monte Carlo simulations within the context of ordinal MDS (see Spence and Young 1978, for an overview). These studies lead to the concept of *stress norms*. The idea is to create random dissimilarities (e.g., by drawing from a uniform $U(0, 1)$ distribution) for a given n and p. For each random draw an MDS solution is fitted. Subsequently, the average stress value and the standard deviation can be computed.

A corresponding implementation is provided by the function `randomstress` which allows users to not only derive ordinal MDS norms, but also to obtain stress norms for other types

[3]Despite setting random seeds some of the figures in this paper may have swapped labels or axis orientations compared to results obtained on other systems.

[4]In modern MDS applications researchers often have to scale a large number of objects.

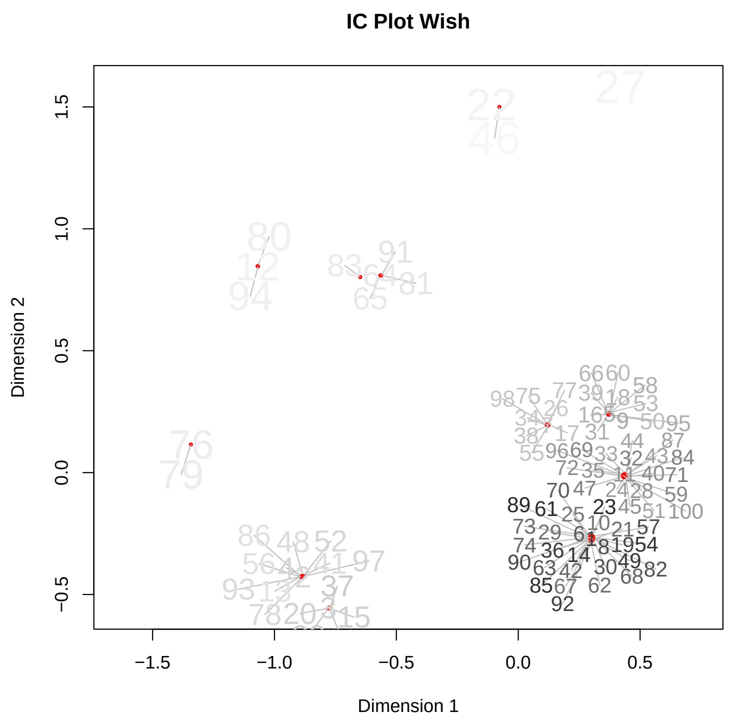


Figure 2: Similarity structure of 100 MDS solutions. Each label corresponds to an MDS solution. The size of the labels (and their color shading) is proportional to the stress value.

of MDS from Section 2. As an example, we use a dataset from Lawler (1967) who studied the performance of managers. There are three traits (T1 = quality of output, T2 = ability to generate output, T3 = demonstrated effort to perform), and three methods (M1 = rating by superior, M2 = peer rating, M3 = self-rating). We start the stress norm analysis by fitting a 2D ratio MDS model:

```
R> LawlerD <- sim2diss(Lawler, to.dist = TRUE)
R> fitLaw <- mds(LawlerD)
```

This leads to a stress value of 0.241. Let us explore the random stress values for this example ($n = 9$, $p = 2$; 500 replications):

```
R> set.seed(123)
R> rstress <- randomstress(n = 9, ndim = 2, nrep = 500, type = "ratio")
```

This function call returns a vector of 500 stress values. Let $\bar{x}_r$ denote the average random stress value and σ_r the standard deviation. The default in the random stress literature (see, e.g., Spence and Ogilvie 1973) is to use $\bar{x}_r - 2\sigma_r$ as upper bound: if the observed stress value is smaller than this cutoff, the stress can be considered as "significant".

```
R> bound <- mean(rstress) - 2 * sd(rstress)
R> round(bound, 3)
```

```
[1] 0.22
```

In our example the stress value of 0.241 from the original MDS fit is above this cutoff. This suggests a "non-significant" result which implies that the 2D ratio MDS solution does not fit satisfactorily.

There are several issues associated with such random stress norms. First, as Spence and Ogilvie (1973) point out, the dispersion of the random stress norms is in general very small. In most practical applications the strategy applied above leads to "significant" results; our example is somewhat of a rare exception. Second, apart from n and p, other circumstances such as the error in the data, missing values, as well as ties affect the stress (Mair *et al.* 2016; Borg and Groenen 2005). Third, the benchmark is based on completely random configurations. Real-life data almost always have some sort of structure in it such that the random stress strategy leads to "significant" results in most cases.

Instead of generating random dissimilarities, permutation tests can be used, as formalized in Mair *et al.* (2016). They lead to "sharper" tests than random null configurations. There are two scenarios for setting up a permutation scheme. First, in the case of directly observed dissimilarities the elements in Δ can be permuted. For each permutation sample an MDS model of choice is fitted. By doing this many times it results in a null distribution of stress values. Second, for derived dissimilarities, Mair *et al.* (2016) propose a strategy for systematic column-wise permutations (one variable at a time). This permutation scheme gives a more informative null distribution compared to full column-wise permutations. For each permutation sample a dissimilarity matrix is computed, and an MDS fitted. Again, this gives a stress distribution under the H_0 of little departure from complete exchangeability of dissimilarities in the data-generating process.

Let us illustrate both permutation scenarios. For directly observed dissimilarities we continue with the Lawler example from above (500 permutations):

```
R> set.seed(123)
R> permLaw <- permtest(fitLaw, nrep = 500, verbose = FALSE)
R> permLaw


Call: permtest.smacof(object = fitLaw, nrep = 500, verbose = FALSE)

SMACOF Permutation Test
Number of objects: 9
Number of replications (permutations): 500

Observed stress value: 0.241
p-value: 0.294
```

We cannot reject the H_0 of "stress/configuration are obtained from a random permutation of dissimilarities". For the derived dissimilarity situation we use a dataset from McNally, Robinaugh, Wu, Wang, Deserno, and Borsboom (2015) which is included in the **MPsychoR** package (Mair 2020). It includes 17 posttraumatic stress disorder (PTSD) symptoms reported by survivors of the Wenchuan earthquake in 2008, scaled on a 5-point rating scale. We use the Euclidean distance as (derived) dissimilarity measure and compute an interval MDS. This leads to the following stress value:

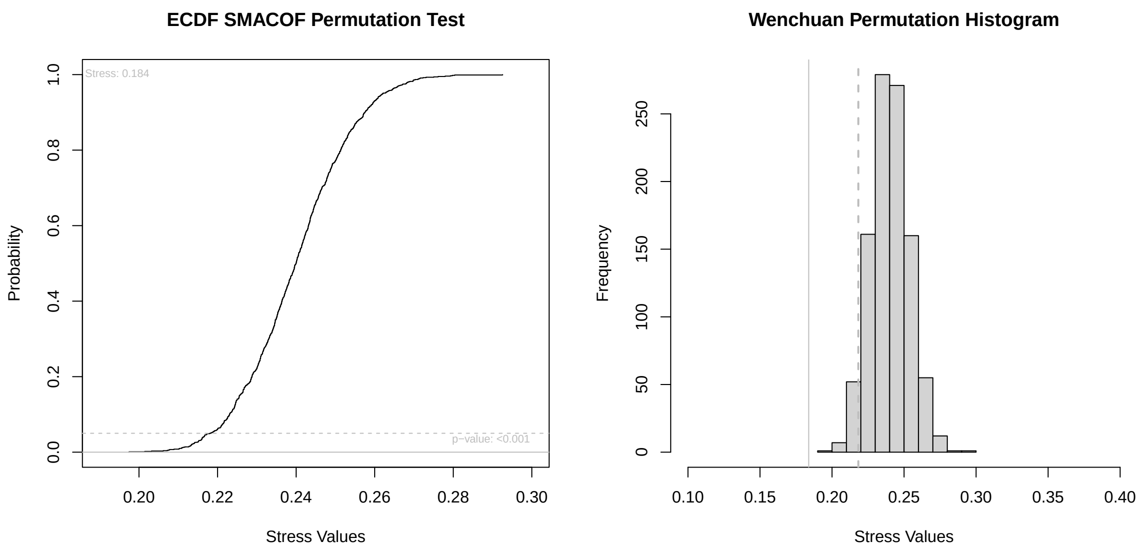


Figure 3: Left panel: ECDF of the permuted stress values (dashed gray line at $\alpha = 0.05$, solid gray line at the p-value). Right panel: Permutation stress histogram (red dashed line at critical value, solid black line at observed stress value).

```
R> library("MPsychoR")
R> data("Wenchuan", package = "MPsychoR")
R> Wdelta <- dist(t(Wenchuan))
R> fitWen <- mds(Wdelta, type = "interval")
R> round(fitWen$stress, 3)

[1] 0.184
```

In the subsequent `permtest` call we provide the raw input data through the `data` argument. This way the function knows that the permutations should be performed on the raw data rather than on Δ. We also need to tell the function which dissimilarity measure we used above before fitting the MDS. We perform 1000 replications.

```
R> set.seed(123)
R> permWen <- permtest(fitWen, data = Wenchuan, method.dat = "euclidean",
+    nrep = 1000, verbose = FALSE)
R> permWen

Call: permtest.smacof(object = fitWen, data = Wenchuan,
    method.dat = "euclidean", nrep = 1000, verbose = FALSE)

SMACOF Permutation Test
Number of objects: 17
Number of replications (permutations): 1000

Observed stress value: 0.184
p-value: <0.001
```

This time we reject H_0. Figure 3, obtained by calling `plot(permWen)`, visualizes the results in two ways: the left panel shows the *empirical cumulative distribution function* (ECDF) of the permutation stress values, whereas the right panel shows the permutation stress histogram including the critical value (lower 5% quantile) and the observed stress value.

Note that such permutation strategies can be applied to unfolding models (see Section 6) as well (see Mair *et al.* 2016, for details).

3.3. Stability of a solution I: Jackknife

De Leeuw and Meulman (1986) developed a jackknife strategy for MDS in order to examine the stability of a solution. Their approach, implemented in the `jackknife` function, computes $i = 1, \ldots, n$ additional solutions with configurations $\mathbf{X}_{-i}$ (object i being left out). Note that each $\mathbf{X}_{-i}$ has row i missing and has therefore $n-1$ rows in total. To make the $\mathbf{X}_{-i}$'s comparable, the location of the missing point is estimated by minimizing a least squares problem, and subsequently transformed using Procrustes (see Section 7.4) with $\mathbf{X}$ as target. Let us denote the resulting configurations by $\mathbf{X}^*_{-i}$, each of them of dimension $n \times p$. From these configurations the average (centroid) jackknife solution $\bar{\mathbf{X}}^*$ can be computed. Thus, we have $n+2$ comparable configurations in total which can be represented in a single plot, as shown below.

De Leeuw and Meulman (1986) also introduced various measures related to the jackknife solution. The first one is a stability measure and is computed as follows:

$$ST = 1 - \frac{\sum_{i=1}^n \|\mathbf{X}^*_{-i} - \bar{\mathbf{X}}^*\|^2}{\sum_{i=1}^n \|\mathbf{X}^*_{-i}\|^2}. \tag{5}$$

ST can be interpreted as the ratio of between and total variance. To measure the cross-validity, that is, comparing the "predicted" configuration of object i as the i-th row in $\bar{\mathbf{X}}^*$ with the actual configuration (i-th row in $\mathbf{X}$),

$$CV = 1 - \frac{n\|\mathbf{X} - \bar{\mathbf{X}}^*\|^2}{\sum_{i=1}^n \|\mathbf{X}^*_{-i}\|^2} \tag{6}$$

can be used. Using these two normalized measures the dispersion around the original solution $\mathbf{X}$ can be simply expressed as

$$DI = 2 - (ST + CV). \tag{7}$$

The dataset we use to illustrate the jackknife MDS is from McNally, Mair, Mugno, and Riemann (2017), included in the **MPsychoR** package. Below we scale 16 depression symptoms reported by patients using the Quick Inventory of Depressive Symptomatology (QIDS-SR). We fit a 2D ordinal MDS on the Euclidean distance input matrix, subject to an MDS jackknife.

```
R> data("Rogers", package = "MPsychoR")
R> RogersSub <- Rogers[,1:16]
R> RogersD <- dist(t(RogersSub))
R> fitRogers <- mds(RogersD, type = "ordinal")
R> jackRogers <- jackmds(fitRogers)
R> jackRogers
```

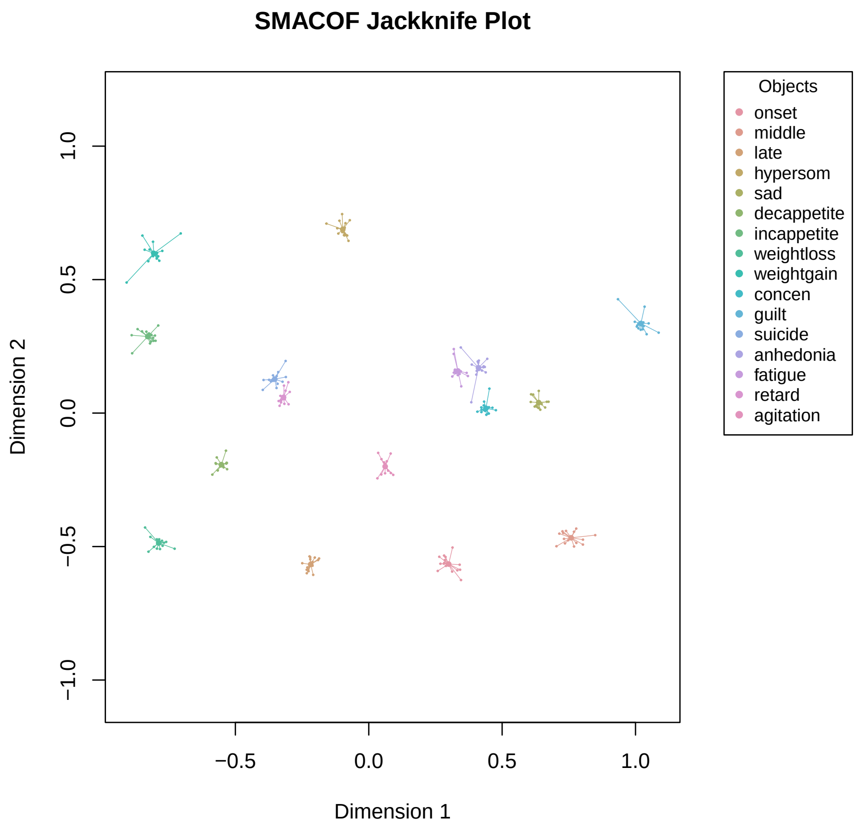


Figure 4: Jackknife MDS plot. The labels are positioned at the original point coordinates, and the stars represent the resampled solutions with the jackknife centroid at the center.

```
Call: jackmds.smacofB(object = fitRogers)

SMACOF Jackknife
Number of objects: 16
Value loss function: 0.3444
Number of iterations: 12

Stability measure: 0.998
Cross validity: 1
Dispersion: 0.002
```

```
R> plot(jackRogers, legend = TRUE, cex.legend = 0.8, inset = c(-0.3, 0))
```

The print output shows the jackknife measures reported above. Figure 4 shows the jackknife MDS plot. The points are placed at $\mathbf{X}$ (MDS configuration). The centers of the stars denote the jackknife centroids, the rays the $n - 1$ jackknife solutions. This result suggests that the solution is very stable.

Further options for using jackknife in MDS are presented in Vera (2017) where the distances are subject to stability analysis.

3.4. Stability of a solution II: Bootstrap

Bootstrap approaches for stability assessment in MDS were proposed by Meulman and Heiser (1983), Heiser and Meulman (1983), Weinberg, Carroll, and Cohen (1984), and further refined by Jacoby and Armstrong (2014). The **smacof** implementation resamples the original data and therefore works for derived dissimilarities only. The confidence ellipsoids are computed as follows. Let $\mathbf{x}_i$ denote the row coordinates of object i from the original configuration $\mathbf{X}$. Let $\mathbf{S}_i$ be the $p \times p$ covariance matrix of the bootstrapped solutions of object i. The $100(1-\alpha)\%$ confidence ellipsoid for object i is determined by the points $\mathbf{z}_j$ for which

$$(\mathbf{z}_j - \mathbf{x}_i)\mathbf{S}_i^{-1}(\mathbf{z}_j - \mathbf{x}_i)^\top = \chi^2(\alpha; p), \tag{8}$$

where $\chi^2(\alpha; p)$ is the α-quantile of the χ^2-distribution with $df = p$. In R, this computation can be easily achieved using the **ellipse** package (Murdoch and Chow 2020). As a stability measure, we can use a slight modification of Equation 5:

$$ST = 1 - \frac{\sum_{l=1}^{N} \|\mathbf{X}_l^* - \bar{\mathbf{X}}^*\|^2}{\sum_{l=1}^{N} \|\mathbf{X}_l^*\|^2}. \tag{9}$$

N denotes the number of bootstrap replications, $\mathbf{X}_l^*$ the configuration of the l-th replication, $\bar{\mathbf{X}}^*$ the bootstrap centroid configuration. Again, ST reflects a between/total variance ratio and can be used to compare various MDS solutions against each other (Heiser and Meulman 1983). For instance, one could compare an unrestricted solution with a restricted solution (see Section 5). The larger ST, the more stable the solution.

Let us apply the corresponding `bootmds` function on the depression data from above. We use $N = 500$ bootstrap replications.

```
R> set.seed(123)
R> bootRogers <- bootmds(fitRogers, RogersSub, method.dat = "euclidean",
+    nrep = 500)
R> bootRogers

Call: bootmds.smacofB(object = fitRogers, data = RogersSub,
    method.dat = "euclidean", nrep = 500)

SMACOF Bootstrap:
Number of objects: 16
Number of replications: 500

Mean bootstrap stress:  0.1179
Stress percentile CI:
  2.5%  97.5%
0.0974 0.1433

Stability coefficient: 0.9631
```

In addition to the stability coefficient, the function also reports the stress averaged across bootstrap samples, including the 95% confidence interval (bootstrap percentile).

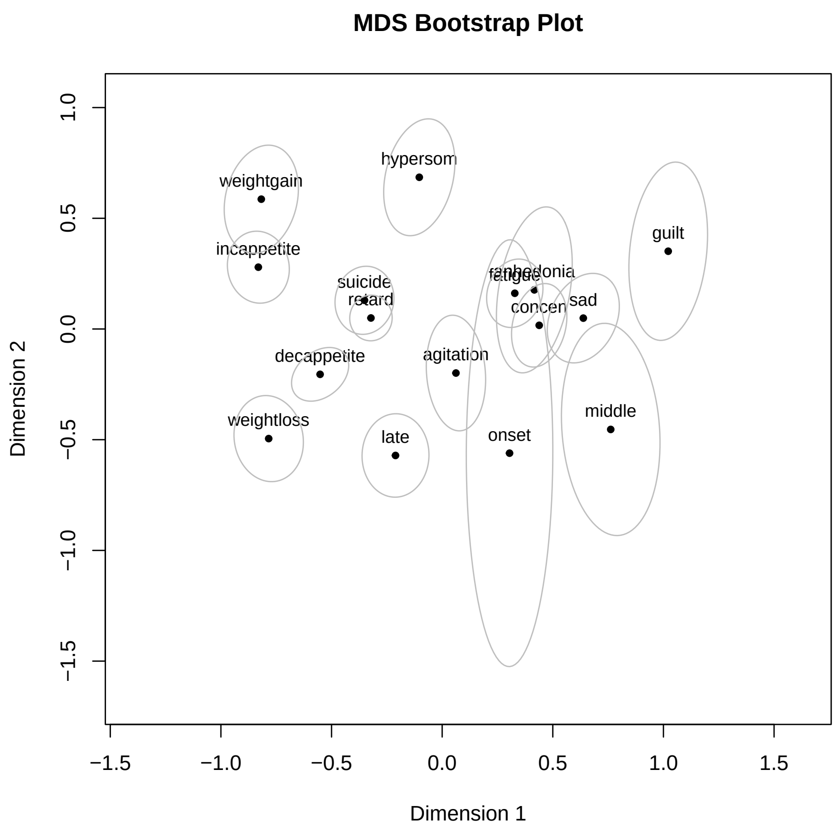


Figure 5: Bootstrap MDS plot with 95% confidence ellipsoids.

```
R> plot(bootRogers)
```

Figure 5 shows the resulting bootstrap configuration with the confidence ellipsoids. There is a fair amount of instability associated with the sleep-onset insomnia item (labeled "onset").

3.5. Stability of a solution III: Pseudo-confidence ellipses

Ramsay (1977, 1982) incorporated MDS into a parametric (log-)normal framework with maximum-likelihood estimation. This idea makes it easy to derive confidence ellipsoids around the points in the configuration. De Leeuw (2019) achieved similar ellipsoids without any distributional assumptions, based on taking the second derivatives of the stress. This approach works for symmetric MDS solutions as well as for individual difference models (INDSCAL, IDIOSCAL) of arbitrary dimensions. In its current form, its implementation is limited to ratio transformations. Expressions for the stress derivatives can be found in the corresponding paper.

Let us use the same dataset as above and fit a ratio MDS. The `confEllipse` function computes the stress derivatives and subsequently the confidence ellipsoids.

```
R> fitRogers2 <- mds(RogersD)
R> confRogers <- confEllipse(fitRogers2)
```

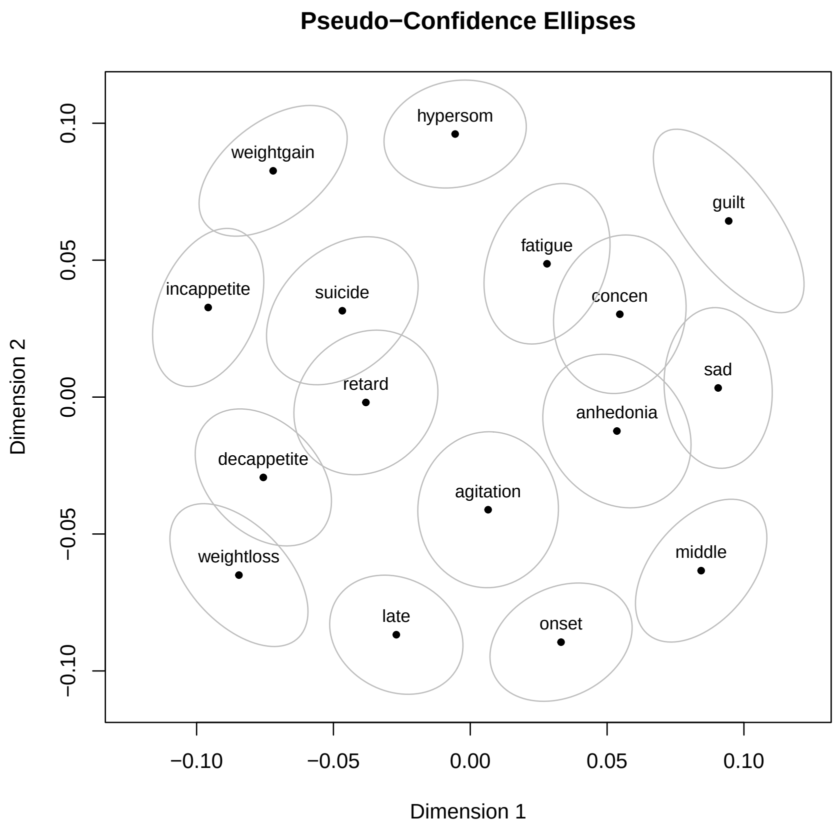


Figure 6: Pseudo-confidence ellipses for ratio MDS solution.

The following plot function takes this object and produces the configuration plot with the ellipsoids. Of importance is the `eps` argument which we set to 0.01 below. This value implies that we look at a perturbation region where the stress value is at most 1% larger than the local minimum we have found. Figure 6 shows the corresponding configuration plot.

```
R> plot(confRogers, eps = 0.01, ylim = c(-0.11, 0.11),
+    ell = list(lty = 1, col = "gray"))
```

Note that the scales along the axes differ from the ones in Figures 5 and 6 (apart from the fact that ratio MDS is used). This is because the SMACOF engine for estimating pseudo-confidence ellipsoids normalizes the coordinates differently (see De Leeuw 2019, for details). Also, the shape differences in the confidence ellipsoids are due to different methods used to construct the ellipsoids.

4. MDS biplots

Biplots were developed within the context of principal component analysis (PCA; Gabriel 1971). In a PCA biplot the loading vectors are mapped on top of the scatterplot of the principal component scores. However, the concept of biplots can be applied to other multivariate techniques as well, as elaborated in Greenacre (2010), Gower, Lubbe, and Le Roux (2011), and Mair (2018). In MDS, biplots are often used to map external variables onto the MDS configuration. Such covariates allow users to explore meaningful directions in the MDS space rather than trying to interpret the dimensions directly. Note that Rabinowitz (1975) was one of the first to suggest embedding axes representing external variables into MDS solutions in order to facilitate substantive interpretations.

Let $\mathbf{Y}$ be a $n \times q$ matrix with q external variables in the columns, each of them centered and optionally standardized (the latter simply changes the length of the biplot vector, not its direction). To produce an MDS biplot, the following multivariate regression problem needs to be solved:

$$\mathbf{Y} = \mathbf{X}\mathbf{B} + \mathbf{E}, \qquad (10)$$

where $\mathbf{B}$ is a $p \times q$ containing p regression coefficients for each of the q variables, and $\mathbf{E}$ is the $n \times q$ matrix of errors. The corresponding OLS estimates $\hat{\mathbf{B}} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{Y}$ give the coordinates of the external variables in the MDS space. The **smacof** package provides the `biplotmds` function which performs the regression fit. By default, the external variables are standardized internally (default `scale = TRUE`; `scale = FALSE` does centering only).

Let us start with a simple example where we map a single metric variable onto a configuration. We use a dataset taken from Engen, Levy, and Schlosberg (1958) on facial expressions (see also Heiser and Meulman 1983). Participants had to rate proximities of 13 facial expressions, resulting in the dissimilarity matrix Δ. Rating scale values were collected by Abelson and Sermat (1962) for the dimensions "pleasant-unpleasant" (PU), "attention-rejection" (AR), and "tension-sleep" (TS).

We fit an ordinal MDS solution, and map the pleasant-unpleasant (PU) variable on top of the configuration. We present two biplot versions. First, we focus on the vector representation.

```
R> fitFace <- mds(FaceExp, type = "ordinal")
R> ext <- data.frame(PU = FaceScale[,1])
R> biFace <- biplotmds(fitFace, extvar = ext)
R> coef(biFace)

           PU
D1 -1.6214189
D2 -0.6295513
```

These regression coefficients determine the direction and length of the biplot vector.

Second, we use the axis representation for which the **calibrate** package (Graffelman 2020) turns out to be helpful. We start by computing the regression coefficients based on the centered external variable. In order to make sure that the ticks on the biplot axis correspond to the original scale, some additional preliminary lines are needed.

```
R> library("calibrate")
R> biFace2 <- biplotmds(fitFace, extvar = ext, scale = FALSE)
R> coef(biFace2)

          PU
D1 -3.865508
D2 -1.500868
```

```
R> PUc <- scale(ext, scale = FALSE)
R> tm <- seq(floor(min(ext)), ceiling(max(ext)), by = 1)
R> tmc_PU <- tm - mean(tm)
R> X <- fitFace$conf
```

The plots from Figure 7 can be produced as follows:

```
R> plot(biFace, main = "Biplot Vector Representation", vecscale = 0.8,
+    xlim = c(-1.5, 1.5), vec.conf = list(col = "brown"), pch = 20, cex = 0.5)
R> plot(fitFace, main = "Biplot Axis Representation", xlim = c(-1.5, 1.5))
R> abline(h = 0, v = 0, lty = 2, col = "gray")
R> calPU <- calibrate(coef(biFace2), PUc, tm = tmc_PU, tmlab = tm, Fr = X,
+    dp = TRUE, axiscol = "brown", axislab = "PU", labpos = 3, verb = FALSE)
```

The top panel uses the vector representation as advocated in Greenacre (2010). Using the `vecscale` argument the biplot vector can be scaled by its length. The bottom panel uses the axis representation as preferred by Gower *et al.* (2011). For the axis representation we can do an orthogonal projection of the points on the axis, which gives the fitted values.

Let us move on with a second, more complex example involving multiple external variables which reproduces part of the analysis presented in Mair (2018). We use the mental states dataset from Tamir, Thornton, Contreras, and Mitchell (2016) who, for each individual, collected a dissimilarity matrix involving 60 mental states, derived from functional magnetic resonance imaging (fMRI) scans. The data are included in the **MPsychoR** package. We average across the individuals, which leads to a single 60×60 dissimilarity matrix, subject to a 2D monotone spline MDS. After the biplot computations, we print out the R^2 values from the individual regression fits.

```
R> data("NeuralActivity")
R> data("NeuralScales")
R> NeuralD <- Reduce("+", NeuralActivity)/length(NeuralActivity)
R> fitNeural <- mds(NeuralD, type = "mspline")
R> biNeural <- biplotmds(fitNeural, NeuralScales[,1:8])
R> round(biNeural$R2vec, 3)
```

```
      Agency    Experience High.Arousal  Low.Arousal         Body
       0.242         0.299        0.134        0.093        0.232
        Mind       Emotion       Reason
       0.085         0.441        0.357
```

The vector version of the MDS biplot is given in Figure 8. The longer a covariate vector, the larger the corresponding R^2. That is, the more accurate the corresponding axis projections are in relation to the raw data. The orientation of the vectors reflects the correlation patterns among the external variables, assuming the plot gives an accurate representation of the data (of course, we lose information here due to projecting into a low-dimensional space). Other options such as nonlinear MDS biplots are presented in Gower *et al.* (2011, Chapter 5), including corresponding R code.

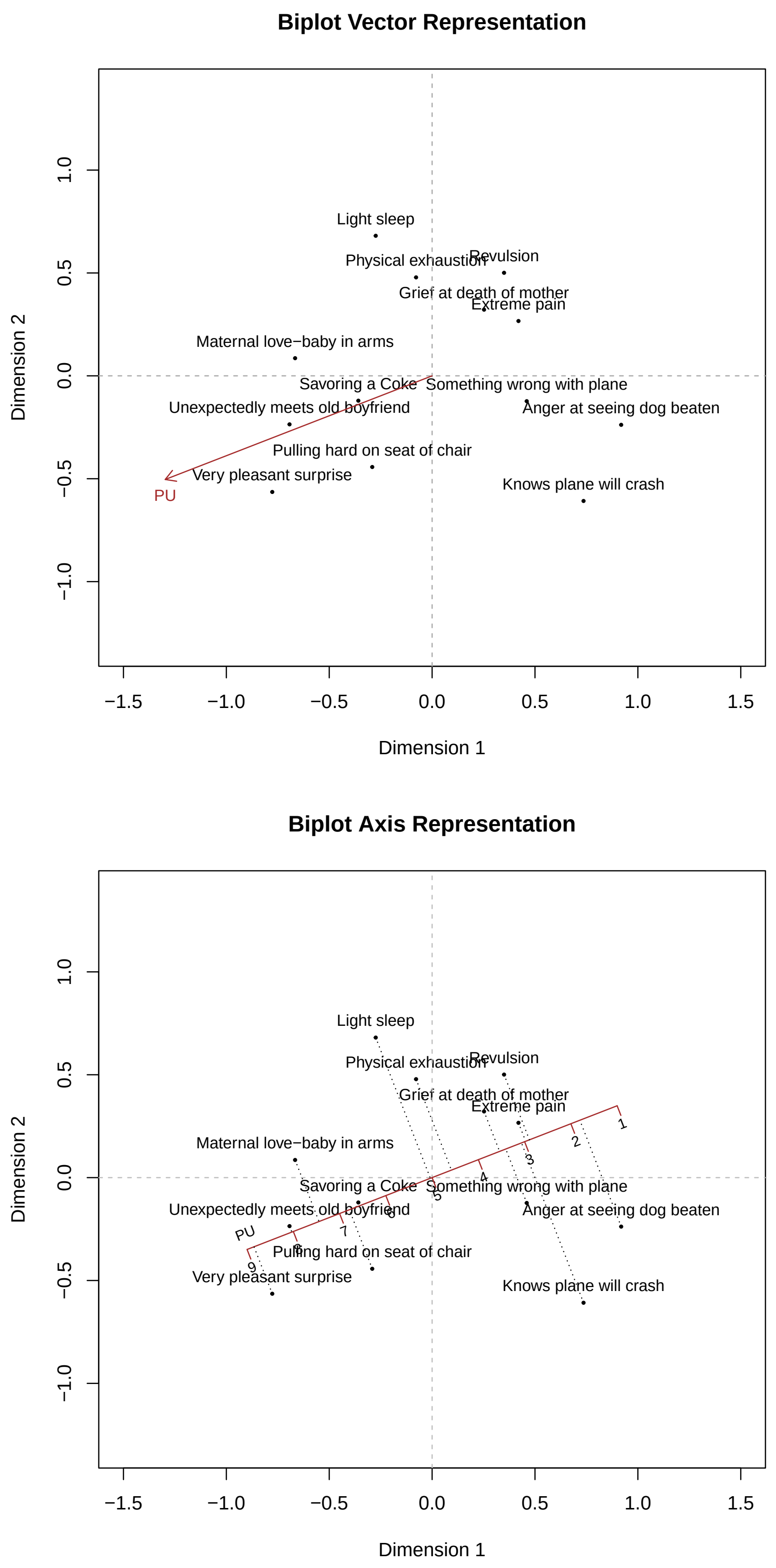


Figure 7: Top panel: Vector representation of external variable. Bottom panel: Axis representation of external variable.

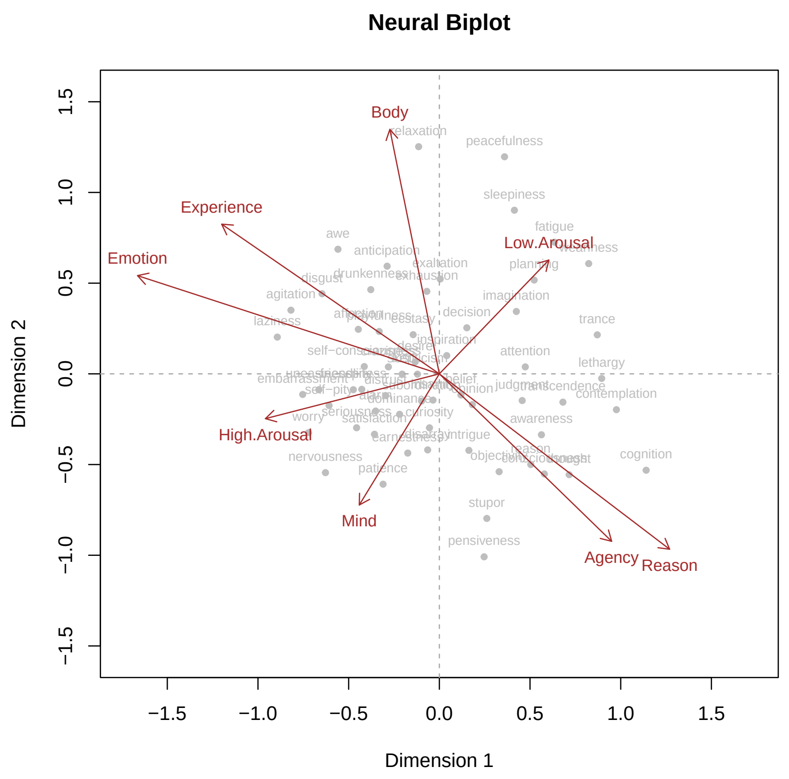


Figure 8: Biplot for mental state MDS configuration. External variables are represented as vectors.

5. MDS with optimal scaling on external predictors

Another advantage of the SMACOF framework compared to classical MDS is the option to fit restricted MDS variants. There are two basic strategies to constrain an MDS configuration. The first option involves internal constraints where the points are forced to be located on a geometric shape. For the particular case of a sphere this can be achieved using `smacofSphere`. The corresponding theory was described in De Leeuw and Mair (2009b). The only spherical update since the original publication has been the incorporation of various types of dissimilarity transformations in `smacofSphere`.

Here we focus on a second strategy, that is, imposing external constraints on the configuration in the tradition of De Leeuw and Heiser (1980), Borg and Lingoes (1980), and Heiser and Meulman (1983). The simplest form of such a restriction is a linear restriction

$$\mathbf{X} = \mathbf{Z}\mathbf{C}, \tag{11}$$

directly incorporated into the stress formula given in (1). $\mathbf{Z}$ is a known covariate matrix of dimension $n \times q$ with number of covariates $q \geq p$. $\mathbf{C}$ is a $q \times p$ matrix of regression weights to be estimated, subject to potential additional restrictions, as outlined below.

For practical purposes, however, this basic implementation is of limited use. For instance, specifying a 2×2 ANOVA design in $\mathbf{Z}$ collapses point coordinates to only four points in a 2D configuration. What makes the external restriction concept attractive in practice is to apply an additional optimal scaling step on the external scales within each majorization iteration.

Equation 11 changes to

$$\mathbf{X} = \hat{\mathbf{Z}}\mathbf{C}. \tag{12}$$

Each predictor variable $\mathbf{z}_1, \ldots, \mathbf{z}_q$ is subject to an optimal scaling transformation. A popular option is to scale these vectors in an ordinal way (i.e., using monotone regression). Other transformations such as interval or splines (with or without monotonicity constraints) are implemented in **smacof** as well. Note that, from a specification point of view, these external variable transformations are unrelated to the dissimilarity transformations introduced in Section 2.

Let us illustrate such a constrained MDS using the face expression data from Section 4. We include the two external variables "pleasant-unpleasant" (PU) and "tension-sleep" (TS). They constitute the matrix $\mathbf{Z}$. We restrict $\mathbf{C}$ to be diagonal, which performs dimensional weighting. Note that for this diagonal restriction the number of dimensions is determined by the number of covariates (i.e., $q = p$), since each covariate defines an axis (dimension). We also use the configuration from an unrestricted ordinal fit as initial configuration. It is important that the user provides a reasonable starting configuration for the constrained MDS computation; using one from an unrestricted fit is in general a good option.

Let us start with the first constrained MDS model: ordinal dissimilarity transformation of Δ, interval transformed external variables in $\mathbf{Z}$, diagonal regression weights restriction in $\mathbf{C}$.

```
R> fitFace <- mds(FaceExp, type = "ordinal")
R> Z <- FaceScale[, c(1,3)]
R> fitFaceC1 <- smacofConstraint(FaceExp, type = "ordinal",
+    constraint = "diagonal", external = Z, constraint.type = "interval",
+    init = fitFace$conf)
R> round(fitFaceC1$C, 3)
```

```
      [,1]  [,2]
[1,] 1.068 0.000
[2,] 0.000 1.211
```

The last line shows the implied diagonal restriction in $\mathbf{C}$. We obtain a stress value of 0.183 which, of course, is larger than the one from the unconstrained fit (0.106).

The resulting MDS configuration is given in Figure 9. Using the **calibrate** package the axes of the external variables (original scales) can be added (see supplemental code materials). These axes are a simple form of biplot axes, resulting from the diagonal restriction in $\mathbf{C}$. For this interval transformed solution the observed values in $\mathbf{Z}$ can be directly read from the PU and TS axes; the configuration coordinates reproduce these values exactly.

In a second fit we relax the interval transformation of $\mathbf{Z}$ in terms of an ordinal transformation. $\mathbf{C}$ is still kept diagonal.

```
R> fitFaceC2 <- smacofConstraint(FaceExp, type = "ordinal",
+    constraint = "diagonal", external = Z, constraint.type = "ordinal",
+    init = fitFace$conf)
R> round(fitFaceC2$C, 3)
```

```
       [,1]  [,2]
[1,] -1.034  0.00
[2,]  0.000 -1.08
```

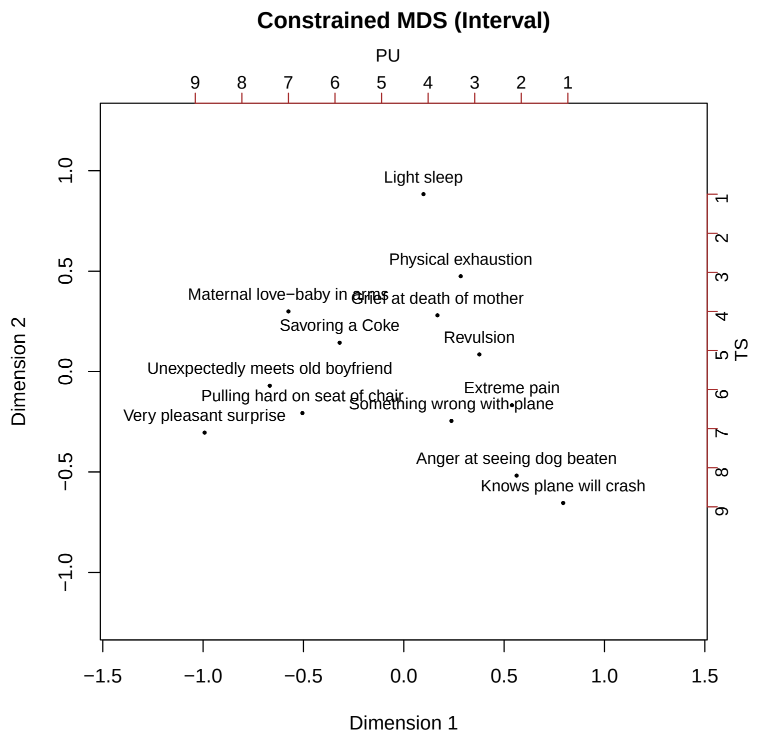
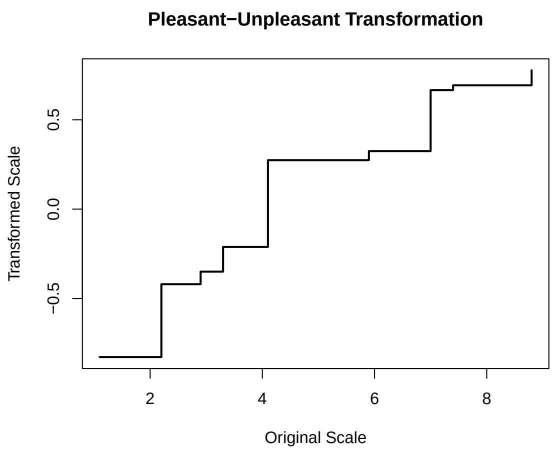


Figure 9: Constrained MDS configurations (**C** diagonal) of face expression data: interval transformed external variables.



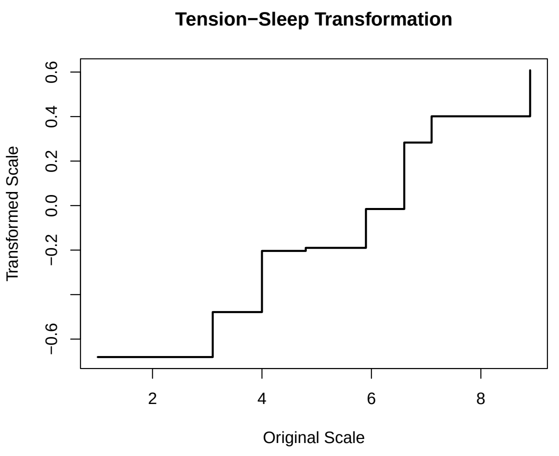


Figure 10: Transformation plots for external variables (original scores from **Z** on the x-axis, transformed scores from $\hat{\mathbf{Z}}$ on the y-axis).

Due to the less restrictive nature of this specification this solution has a lower stress value (0.159) than the interval transformed solution from above. Figure 10 gives some insight into the ordinal transformations performed internally on each column of **Z**.

Figure 11 shows the configuration with the transformed axes on top and to the right. Again, the points can be projected onto these axes. The corresponding values match the ones in $\hat{\mathbf{Z}}$.

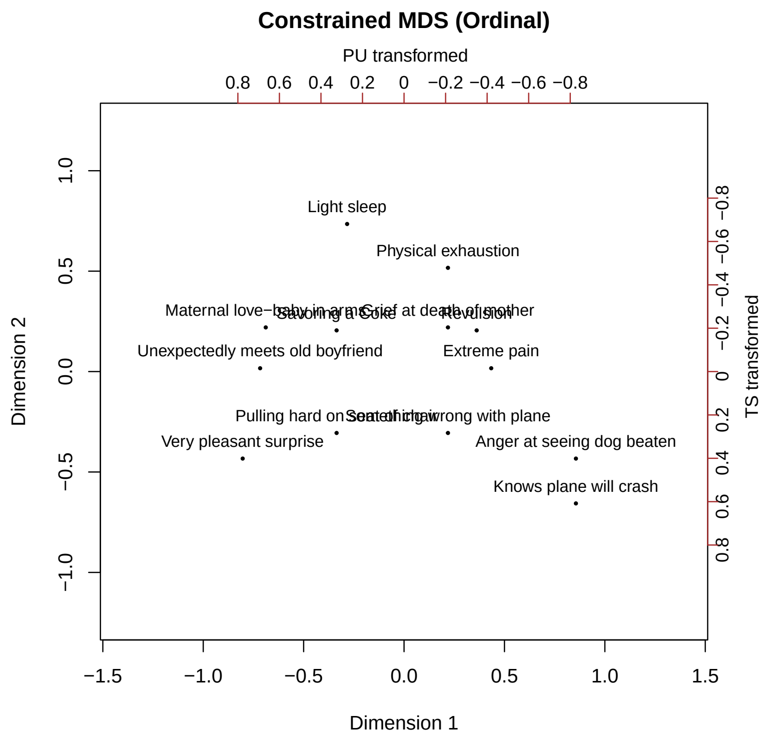


Figure 11: Constrained MDS configuration (**C** diagonal) of face expression data: ordinal transformed external variables.

For the next constrained MDS variant we use all three external variables in the dataset (i.e., PU, AR, and TS). As $q > p$ we need to relax the diagonal restriction in **C**: we keep **C** unrestricted and use once more an interval transformation of **Z**.

```
R> Z <- FaceScale
R> fitFaceC3 <- smacofConstraint(FaceExp, type = "ordinal",
+    constraint = "unrestricted", external = Z, constraint.type = "interval",
+    init = fitFace$conf)
R> round(fitFaceC3$C, 3)
```

```
         D1     D2
[1,] -0.887 -0.231
[2,]  0.087 -0.413
[3,] -2.571  4.344
```

Again, the three biplot axes can be mapped onto the configuration using the **calibrate** package, after computing the regressions $\mathbf{Z} = \mathbf{XB}$ with **Z** column-centered (see supplemental materials for the entire code chunk).

Figure 12 displays the corresponding constrained MDS configuration with the biplot axes on top. Each point can be projected on each axis. The projections are stored in each of the calibrate objects (value: `yt`). Generally, the projected values do not correspond to the observed values in **Z** as these calibrated biplot axes do not reproduce **Z** perfectly. As far as

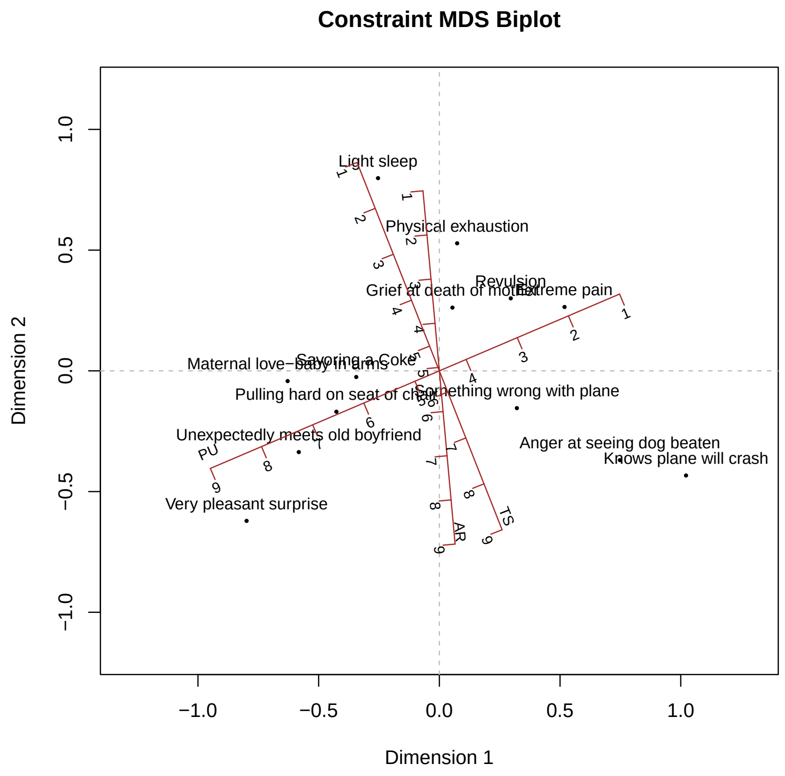


Figure 12: Constraint MDS configuration with **C** unrestricted. Three external covariates are added as biplot axes.

the axes are concerned, the biplot suggests that PS and TU are almost orthogonal, whereas the predictions TS and AR are highly correlated in this 2D space. Dimension 2 lines up with the AR axis which is useful for the interpretation of the configuration.

A general dimensional interpretation on the basis of the external variables no longer holds since **C** is not diagonal: the solution is rotated/reflected followed by dimensional stretching. By applying an SVD on **C** the user can get the rotation matrices and the dimension stretching values (see Borg and Groenen 2005, p. 232, for details).

6. Unfolding

As mentioned in the introduction, one of the major updates since the first publication of the package was a complete re-implementation of the `unfolding` function. This update gives the user the possibility to apply the usual transformations on the dissimilarities, to incorporate circular restrictions, and to fit row-conditional and external unfolding models.

6.1. Unfolding theory

Unfolding takes a rectangular dissimilarity matrix $\boldsymbol{\Delta}$ of dimension $n \times m$ with elements δ_{ij} ($i = 1, \ldots, n$ and $j = 1, \ldots, m$) as input. Such data are most often rankings or ratings. The stress function from Equation 1 changes to

$$\sigma^2(\hat{\mathbf{D}}, \mathbf{X}_1, \mathbf{X}_2) = \sum_{i=1}^{n} \sum_{j=1}^{m} w_{ij} (\hat{d}_{ij} - d_{ij}(\mathbf{X}_1, \mathbf{X}_2))^2, \tag{13}$$

with the fitted Euclidean distances (p-dimensional space) expressed as

$$d_{ij}(\mathbf{X}_1, \mathbf{X}_2) = \sqrt{\sum_{s=1}^{p} (x_{1is} - x_{2js})^2}. \tag{14}$$

$\mathbf{X}_1$ is an $n \times p$ matrix (row configuration), $\mathbf{X}_2$ an $m \times p$ matrix (column configuration), and $\hat{\mathbf{D}}$ the $n \times m$ matrix of disparities. Again, the weights w_{ij} and the dissimilarities δ_{ij} must be non-negative.

In terms of stress normalization, Borg and Groenen (2005, Section 11.1) argue that one could find an optimal dilation factor that multiplies both the row and column coordinates by the same constant, leading to

$$\sigma_1(\hat{\mathbf{D}}, \mathbf{X}_1, \mathbf{X}_2) = \sqrt{1 - \frac{\sum_{i,j} (w_{ij} \hat{d}_{ij} d_{ij}(\mathbf{X}_1, \mathbf{X}_2))^2}{\sum_{i,j} w_{ij} \hat{d}_{ij}^2 \sum_{i,j} w_{ij} d_{ij}^2(\mathbf{X}_1, \mathbf{X}_2)}}. \tag{15}$$

This expression provides a short cut to compute the stress-1 value, given that we allow for an optimal dilation constant. At the same time it is a trick for interpretation in terms of the well known stress-1 value after all the majorization computations are done. Details on the majorization approach in the case of ratio transformations are given in (De Leeuw and Mair 2009b). Below we elaborate on a modification that is able to handle general monotone dissimilarity transformations from Section 2.

6.2. Transformation functions

Early versions of non-metric multidimensional unfolding (i.e., ordinal transformation) are described in Coombs (1964). Busing *et al.* (2005) elaborate in detail on the challenges of including transformation functions in unfolding with respect to optimization. One major problem is degeneracy of the solution due to equal disparities. They suggest to penalize the stress function by the coefficient of variation, which moves the algorithm away from solutions with small variation in the fitted distances. The corresponding badness-of-fit target is called *p-stress*:

$$\sigma_p^2(\hat{\mathbf{D}}, \mathbf{X}_1, \mathbf{X}_2) = \sigma^{2\lambda}(\hat{\mathbf{D}}, \mathbf{X}_1, \mathbf{X}_2) \mu(\hat{\mathbf{D}}). \tag{16}$$

The coefficient of variation $\nu(\hat{\mathbf{D}})$ is calculated on the basis of the disparities and enters the penalty term as follows:

$$\mu(\hat{\mathbf{D}}) = 1 + \frac{\omega}{\nu^2(\hat{\mathbf{D}})}. \tag{17}$$

Obviously this penalty term acts as a multiplicative factor in Equation 16. As $\nu(\hat{\mathbf{D}})$ decreases, the p-stress penalization increases. There are two tuning parameters involved in this p-stress setup:

- $\lambda \in (0; 1]$ is a lack-of-penalty parameter that controls the influence of penalty term: the larger λ, the smaller the penalty influence.

- ω acts as range parameter in the penalty term: for a small ω the penalty is especially effective if $\nu(\hat{\mathbf{D}})$ is small.

Busing *et al.* (2005) did an extensive simulation study in order to provide suggestions on how to fix the tuning parameters. For conditional unfolding, it is suggested to set $\lambda = 0.5$, and $\omega = 1$ (default settings in `unfolding`)[5]. For unconditional unfolding, they suggest that one uses $\lambda = 0.5$ and $\omega > 0.1$. Further details can be found in the corresponding publication.

The p-stress target can be minimized using majorization, for which the details are again given in Busing *et al.* (2005). From a practical point of view, after obtaining a p-stress optimized solution, users can consider the stress-1 from Equation 15 as goodness-of-fit index[6]. Note that all the dissimilarity transformation functions from MDS (i.e., ratio, interval, ordinal, spline; cf. Section 2) are implemented for unfolding as well.

Let us illustrate an example of an ordinal unfolding solution. We use a dataset from Dabic and Hatzinger (2009), available in the **prefmod** package (Hatzinger and Dittrich 2012), where individuals were asked to configure a car according to their preferences. They could choose freely from several modules such as exterior and interior design, technical equipment, brand, price, and producing country. We use only the first 100 individuals in this analysis.

```
R> library("prefmod")
R> carconf1 <- carconf[1:100, 1:6]
R> head(carconf1)

  price exterior brand tech.equip country interior
1     3        2     5          6       4        1
2     4        1     5          2       6        3
3     6        3     2          5       4        1
4     1        4     2          3       6        5
5    NA        2     4         NA       3        1
6    NA        2     4          3      NA        1
```

Since not all individuals ranked all objects, we have the situation of "partial rankings". The `unfolding` function specifies a proper weight matrix $\mathbf{W}$ automatically: $w_{ij} = 0$ if δ_{ij} is missing; $w_{ij} = 1$ otherwise. This way, the corresponding missing dissimilarities are blanked out from the optimization. For the ordinal unfolding model we are going to fit, this weight matrix can be extracted using `unf_ord$weightmat`.

```
R> unf_ord <- unfolding(carconf1, type = "ordinal")
R> unf_ord

Call: unfolding(delta = carconf1, type = "ordinal")

Model:                Rectangular smacof
Number of subjects:   100
Number of objects:    6
```

[5]Personal communication with Frank Busing.

[6]For details on how to assess the goodness-of-fit of an unfolding solution see Mair *et al.* (2016).

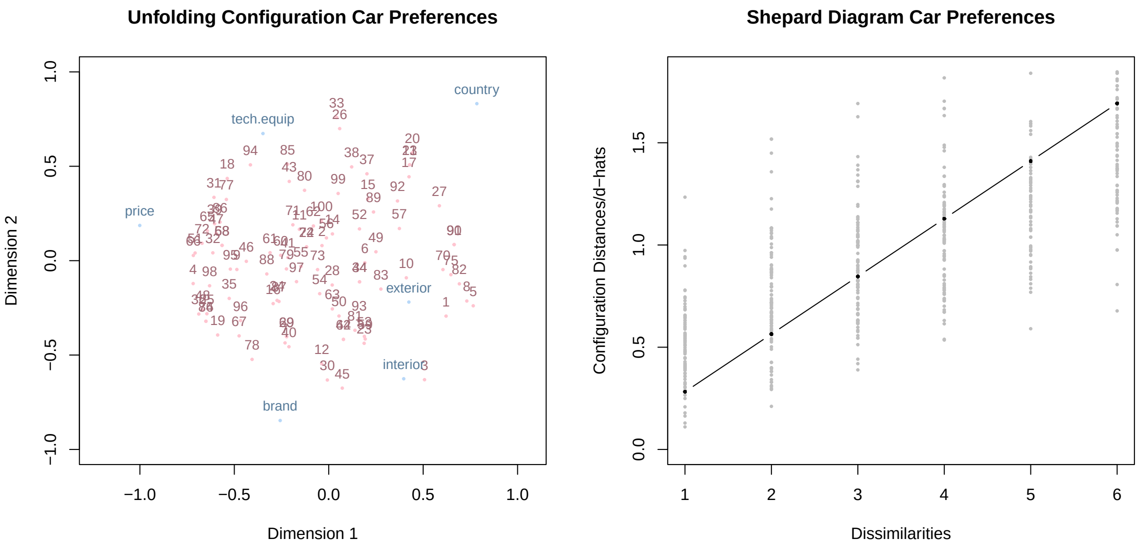


Figure 13: Left panel: Unfolding configuration of car preference data. Right panel: Shepard diagram car preference data (ordinal transformation).

```
Transformation:      ordinalp
Conditionality:      matrix

Stress-1 value:    0.297371
Penalized Stress:  2.330348
Number of iterations: 135
```

This call prints out the stress-1 value as well as the final p-stress value. The configuration plot and Shepard diagram shown in Figure 13 can be produced as follows:

```
R> plot(unf_ord, main = "Unfolding Configuration Car Preferences",
+    ylim = c(-1, 1))
R> plot(unf_ord, plot.type = "Shepard",
+    main = "Shepard Diagram Car Preferences")
```

The Shepard diagram shows the ordinal transformation of the input dissimilarities, whereas the configuration plot maps the row and column coordinates into a joint space, which makes distances between any pair of points interpretable.

6.3. Row-conditional unfolding

The solution above is an *unconditional* (also called *matrix-conditional*) unfolding solution since a single transformation function is estimated that applies to all individuals. The ranks are compared unconditionally which, in some situations, is a fairly restrictive assumption. For instance, in our example it might be the case that individual i is quite indifferent to all car characteristics, but still ranks them. Individual i', however, could have strong preferences but might end up with the same ranking as individual i. Treating these ranks as equal is a strong assumption. Similarly, for rating data in $\mathbf{\Delta}$, individual i''s rating of, for instance, 2 is assumed to be equal to any other individual's rating of 2.

Row-conditional unfolding relaxes this single transformation assumption by estimating separate transformation functions for each individual (i.e., for each row in $\mathbf{\Delta}$). Technically, what changes with respect to the p-stress expression in Equation 16 is the penalty term. Busing *et al.* (2005) suggest using the harmonic mean for row-wise aggregation of the penalty components which modifies Equation 17 to

$$\mu_c(\hat{\mathbf{D}}) = 1 + \frac{\omega}{\left(\frac{1}{n}\sum_{i=1}^{n} \nu^{-2}(\hat{\boldsymbol{d}}_i)\right)^{-1}}. \tag{18}$$

The $\hat{\boldsymbol{d}}_i$'s are the row vectors in $\hat{\mathbf{D}}$. The raw stress term in Equation 16 remains unadjusted since it is additive over the rows.

Let us fit a row-conditional version of the ordinal unfolding on the car characteristics data. We use the final configuration obtained above as starting configuration. Note that for running time purposes we set a slightly more generous convergence boundary ε than the default[7]. In general, we recommend to increase the number of iterations using the `itmax` argument, if needed. For a reasonably large sample size it can take a while for the algorithm to converge. A parallelized fit can be evoked through the `parallelize` argument.

```
R> startconf <- list(unf_ord$conf.row, unf_ord$conf.col)
R> unf_cond <- unfolding(carconf1, type = "ordinal", conditionality = "row",
+    eps = 6e-5, init = startconf)
R> unf_cond


Call: unfolding(delta = carconf1, type = "ordinal", conditionality = "row",
    init = startconf, eps = 6e-05)

Model:                 Rectangular smacof
Number of subjects:    100
Number of objects:     6
Transformation:        ordinalp
Conditionality:        row

Stress-1 value:    0.202789
Penalized Stress:  29.40951
Number of iterations: 744
```

Compared to the unconditional fit, the row-conditional version clearly reduced the stress-1. Figure 14 shows the resulting configuration plot in the left panel. The Shepard diagram in the right panel nicely illustrates the difference between unconditional and row-conditional unfolding. While in unconditional unfolding we fitted only a single transformation function (see right panel of Figure 13), in row-conditional unfolding each individual gets its own transformation function. Since we have missing values in our data, not all individuals have the full six-ranking monotone trajectories.

[7]In the t-th iteration the convergence criterion used in `unfolding` is
$2(\sigma_p(\mathbf{X}_1,\mathbf{X}_2)^{(t-1)} - \sigma_p(\mathbf{X}_1,\mathbf{X}_2)^{(t)}) \leq \varepsilon(\sigma_p(\mathbf{X}_1,\mathbf{X}_2)^{(t-1)} + \sigma_p(\mathbf{X}_1,\mathbf{X}_2)^{(t)} + 10^{-15})$.

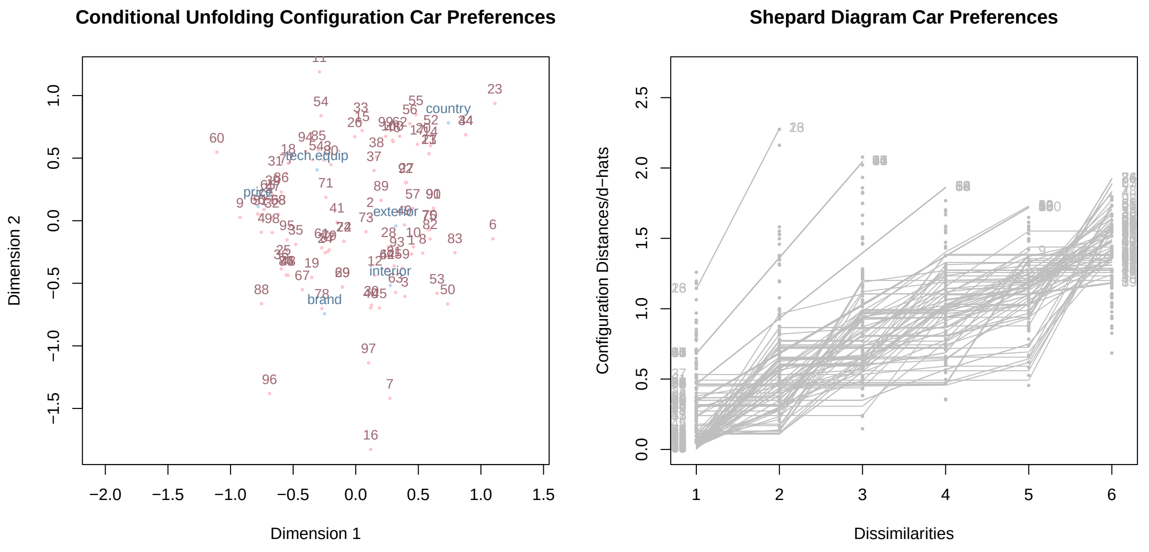


Figure 14: Left panel: Row-conditional unfolding configuration of car preference data. Right panel: Shepard diagram (ordinal transformation) row-conditional unfolding.

```
R> plot(unf_cond,
+    main = "Conditional Unfolding Configuration Car Preferences")
R> plot(unf_cond, plot.type = "Shepard",
+    main = "Shepard Diagram Car Preferences", col.dhat = "gray",
+    xlim = c(0.9, 6.1))
```

Some practical suggestions on when to use row-conditional unfolding as opposed to unconditional unfolding are given in Borg *et al.* (2018, p. 100).

6.4. External unfolding

External unfolding uses fixed coordinates for either the rows or the columns. Such fixed coordinates might come from a previous analysis, or might be based on an underlying theory. Early external unfolding references are Carroll (1972), Srinivasan and Shocker (1973), Rodgers and Young (1981), and DeSarbo and Rao (1984). Within each iteration either the row coordinates in $\mathbf{X}_1$ (in case of fixed coordinates denoted as $\mathbf{F}_1$), or the column coordinates in $\mathbf{X}_2$ (in case of fixed coordinates denoted as $\mathbf{F}_2$) need to be constrained and scaled. The scaling factors for fixed rows and fixed columns, respectively, are

$$s_1 = \frac{\text{trace}(\mathbf{F}_1^\top \mathbf{X}_1)}{\|\mathbf{F}_1\|},$$
$$s_2 = \frac{\text{trace}(\mathbf{F}_2^\top \mathbf{X}_2)}{\|\mathbf{F}_2\|}.$$

Based on these scaling factors the updated coordinates are $\mathbf{X}_1 := s_1\mathbf{F}_1$ in the case of row restrictions, or $\mathbf{X}_2 := s_2\mathbf{F}_2$ in the case of column restrictions. Using this adjustment the new coordinates are properly scaled with respect to the unconstrained column/row coordinates, while maintaining the specified shape constraints.

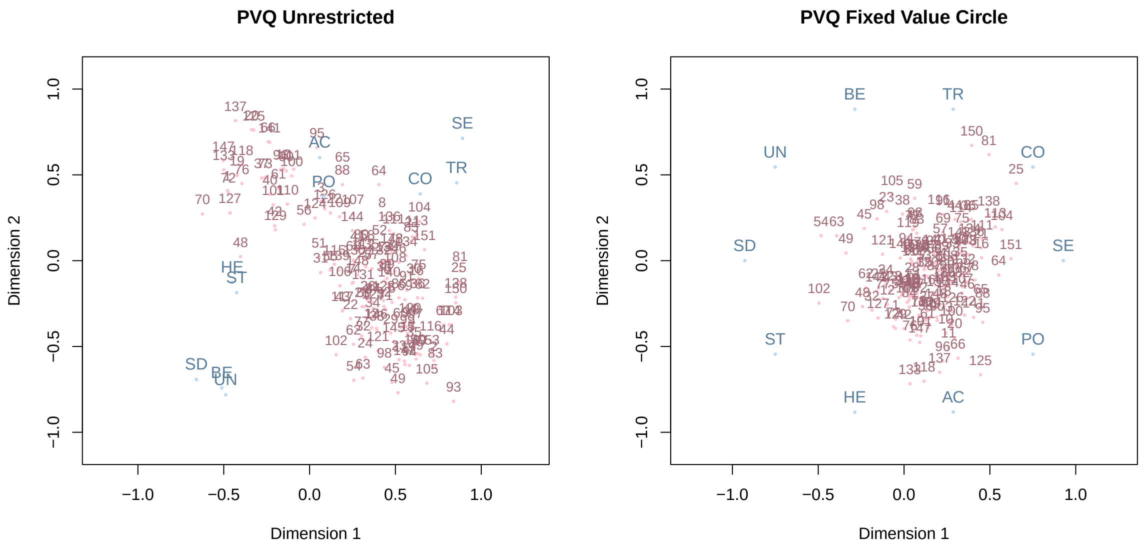


Figure 15: Left panel: Unrestricted unfolding solution. Right panel: Externally restricted unfolding solution (fixed circular coordinates for personal values).

To illustrate, we use a dataset from Borg, Bardi, and Schwartz (2017). We focus on the Portrait Value Questionnaire (PVQ) portion of the data which result from a questionnaire of 40 items assessing how persons rate the personal importance of ten basic values: power (PO), achievement (AC), hedonism (HE), stimulation (ST), self-direction (SD), universalism (UN), benevolence (BE), tradition (TR), conformity (CO), security (SE) on a scale from 0 to 6. We use an aggregated version where the item scores belonging to the same psychological value are averaged. As fixed coordinates we use the following value circle coordinates:

```
R> tuv <- matrix(NA, nrow = ncol(PVQ40agg), ncol = 2)
R> alpha <- -360/10
R> for (i in 1:10){
+    alpha <- alpha+360/10
+    tuv[i,1] <- cos(alpha*pi/180)
+    tuv[i,2] <- sin(alpha*pi/180)
+  }
```

This specification is different from spherical unfolding introduced below, as we fix the value coordinates on the circle (equidistant) instead of just forcing them to be aligned on a circle. Of course, in external unfolding we can specify any arbitrarily fixed configuration; it does not have to be a circle.

Below we fit two solutions: an unconstrained ordinal unfolding solution, and a constrained ordinal unfolding solution with fixed circular column coordinates. Since smaller responses in the PVQ data reflect larger dissimilarities, we reverse the category scores.

```
R> delta <- (max(PVQ40agg) + 1) - PVQ40agg
R> unf_pvq <- unfolding(delta, type = "ordinal")
R> unf_pvqext <- unfolding(delta, type = "ordinal", fixed = "column",
+    fixed.coord = tuv)
```

The stress value of the unconstrained solution is 0.208, whereas that of the external solution is 0.274, which is clearly larger. The plots given in Figure 15 reflect the corresponding differences in the configurations. The unrestricted solution clearly deviates from the theoretical circle.

6.5. Spherical unfolding

Sometimes it is of interest to restrict either row coordinates or column coordinates to be on a geometric shape such as a sphere. Technically, this implies that within each iteration the row (or column) coordinates have to be spherically restricted. Let us elaborate on this restriction for the row coordinates in $\mathbf{X} := \mathbf{X}_1$ for $p = 2$ (for the column coordinates in $\mathbf{X}_2$ it works in an analogous fashion). Each row vector $\mathbf{x}_i$ can be expressed in polar coordinates (see Cox and Cox 1991):

$$\mathbf{x}_i = (r_i \cos(\theta_i), r_i \sin(\theta_i))^\top,$$

with θ_i as the corresponding angle and r_i as the radius. We aim to find a circular restricted configuration $\mathbf{X}_c$ for which the row vectors have polar coordinates

$$\mathbf{x}_{c,i} = (r \cos(\theta_{c,i}), r \sin(\theta_{c,i}))^\top.$$

We have a single radius r and the corresponding angle $\theta_{c,i}$. To compute $\mathbf{X}_c$, we want to minimize the quadratic part of the majorizing function:

$$\begin{aligned}
\|\mathbf{X} - \mathbf{X}_c\|^2 &= \|\mathbf{X}\|^2 + \|\mathbf{X}_c\|^2 - 2 \times \text{trace}(\mathbf{X}_c^\top \mathbf{X}) \\
&= \|\mathbf{X}\|^2 + nr^2 - 2 \times \text{trace}(\mathbf{X}_c^\top \mathbf{X}) \\
&= \|\mathbf{X}\|^2 + nr^2 - 2\sum_{i=1}^{n} r \times r_i(\cos(\theta_{c,i})\cos(\theta_i) + \sin(\theta_{c,i})\sin(\theta_i)).
\end{aligned}$$

In the last term, the best θ_{c_i} that can be chosen is the one that maximizes the following expression: $\cos(\theta_{c,i})\cos(\theta_i) + \sin(\theta_{c,i})\sin(\theta_i)$. This implies choosing $\theta_{c,i} = \theta_i$ so that

$$\cos(\theta_{c,i})\cos(\theta_i) + \sin(\theta_{c,i})\sin(\theta_i) = \cos^2(\theta_{c,i}) + \sin^2(\theta_i) = 1.$$

Substituting the optimal $\theta_{c,i} = \theta_i$ gives

$$\|\mathbf{X} - \mathbf{X}_c\|^2 = \|\mathbf{X}\|^2 + nr^2 - 2r\sum_{i=1}^{n} r_i.$$

Setting the first derivative equal to zero yields the update

$$r = \frac{1}{n}\sum_{i=1}^{n} r_i.$$

This simple expression gives us the optimal circular projection of the row coordinates in $\mathbf{X}$. As mentioned above, the same steps can be carried out for the column coordinates ($\mathbf{X} := \mathbf{X}_2$; replace i by j, and n by m in these equations).

To illustrate an unfolding solution where we restrict the column coordinates to be on a circle, we use once more a dataset from Borg *et al.* (2017) which builds on the Schwartz (1992) value circle theory. The data are derived from the Schwartz Value Survey (SVS). They were centered (row-wise) and converted from preferences into dissimilarities, hence representing a

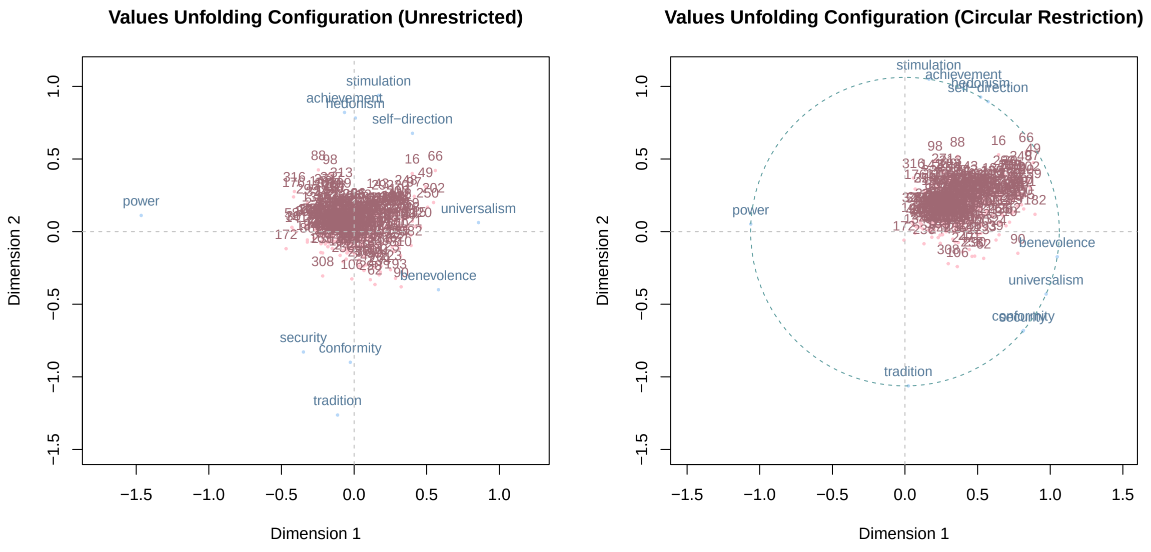


Figure 16: Left panel: Unrestricted unfolding configuration of personal values. Right panel: Circular unfolding solution personal values (circle superimposed).

rectangular dissimilarity matrix $\mathbf{\Delta}$ with 327 persons and 10 variables referring to Schwartz' psychological values: power, achievement, hedonism, stimulation, self-direction, universalism, benevolence, tradition, conformity, and security. We fit two (ratio) unfolding solutions: an unrestricted one as well as one with circular restrictions on the column coordinates (values):

```
R> unf_vals <- unfolding(indvalues)
R> unf_valsc <- unfolding(indvalues, circle = "column")
```

Comparing the stress-1 values we get 0.171 for the unrestricted solution, and 0.179 for the restricted solution. This suggests that the circular solution is basically as good as the unrestricted one.

The reason for this becomes obvious when looking at the configuration plots in Figure 16. The unrestricted solution in the left panel suggests that the personal values approximate a circle, as suggested by Schwartz' value theory. The configuration in the right panel results from forcing the psychological values to be arranged on a circle.

6.6. Vector model of unfolding

Tucker (1960) propsed the *vector model of unfolding* (VMU) which Carroll (1972) later called MDPREF. It is basically a principal component analysis (PCA) on the transposed input similarity matrix $\mathbf{P}$. After a singular value decomposition $\mathbf{P}^\top \approx \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ for given dimensionality p, the row coordinates are obtained by $\mathbf{X}_1 = \sqrt{m-1}\mathbf{U}\mathbf{\Sigma}$, and the column coordinates by $\mathbf{X}_2 = (m-1)^{-1/2}\mathbf{V}$.

We apply the VMU on the unreversed PVQ data from above, since the inputs have to be similarities. Note that, by default, the `vmu` function does an internal row-wise centering of the data.

```
R> fit_vmu <- vmu(PVQ40agg)
R> fit_vmu
```

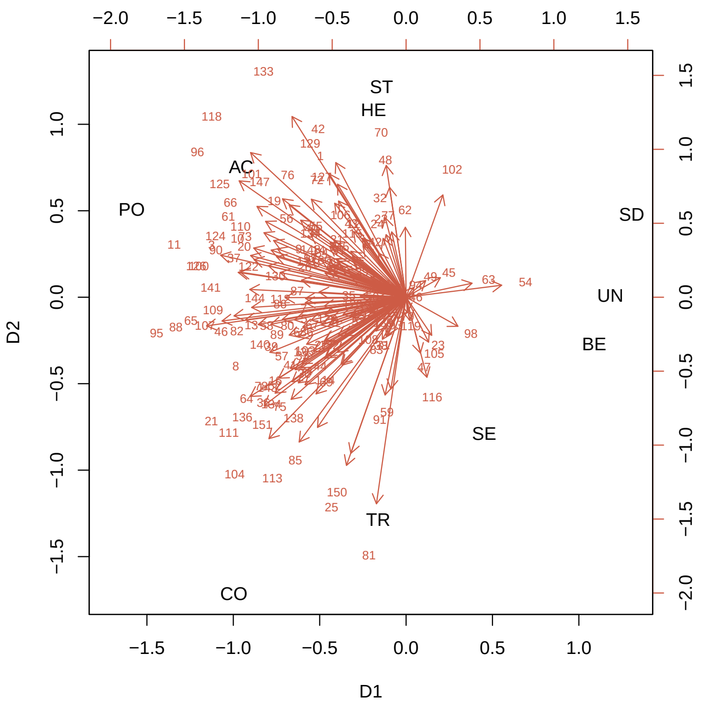


Figure 17: VMU biplot for PVQ data.

```
Call: vmu(delta = PVQ40agg)

Number of subjects: 151
Number of objects: 10
Number of dimensions: 2
Variance accounted for: 66.21%
```

The results can be visualized using a biplot, where the row scores are represented as preference vectors (see Figure 17).

```
R> plot(fit_vmu, col = c("black", "coral3"), cex = c(1, 0.7))
```

Note that ordinal versions of VMU amount to fitting an ordinal PCA (Princals; Gifi 1990; De Leeuw and Mair 2009a).

7. Other MDS variants and utilities

7.1. Unidimensional scaling

Unidimensional scaling can be applied in situations where one has a strong reason to believe that there is only one underlying dimension, such as time, ability, or preference. Even though unidimensional scaling can be considered as a special case of MDS, it is generally discussed separately (Mair and De Leeuw 2015) since the local minimum problem becomes serious if `mds` is used with `ndim = 1`. The **smacof** package provides a simple implementation where all possible $n!$ dissimilarity permutations are considered for scaling, and the one which leads to a minimal stress value is returned. Obviously, this strategy is applicable only to a small number of objects, say less than 10 objects[8].

In the following example we examine seven works by Plato, map them on a single dimension, and explore to which degree the mapping reflects the chronological order by which they are written. The input dissimilarities are derived according to the following strategy: Cox and Brandwood (1959) extracted the last five syllables of each sentence; each syllable is classified as long or short which gives 32 types; and based on this classification a percentage distribution across the 32 scenarios for each of the seven works can be computed, subject to a Euclidean distance computation.

```
R> PlatoD <- dist(t(Plato7))
R> fitPlato <- uniscale(PlatoD, verbose = FALSE)
R> round(sort(fitPlato$conf), 3)
```

```
  Critias  Republic   Timaeus   Sophist Politicus  Philebus      Laws
   -0.903    -0.599    -0.335    -0.060     0.373     0.656     0.869
```

The last line prints the 1D "time" configuration of Plato's works that lead to the lowest stress value. Note that the exact chronological order of Plato's works is unknown; scholars only know that "Republic" was the first work, and "Laws" his last one. Copleston (1949, p. 140) suggests the following temporal order of these selected seven works: Republic, Sophist, Politicus, Philebus, Timaeus, Critias, Laws. Obviously, our unidimensional scaling model advocates a different chronological order.

7.2. Gravity model

The idea of the *gravity model* goes back to Newton's law of universal gravitation where he states that force is proportional to the product of the two masses, and inversely proportional to the square of the distance between them. Haynes and Fotheringham (1984) present a wide range of statistical applications of this gravity concept with special emphasis on spatial interactions. Within an MDS context, the gravity model turns out to be especially useful for text co-occurrence data (Mair, Rusch, and Hornik 2014; Borg *et al.* 2018) in order to avoid that words with high co-occurrences dominate the solution. Note that the `sim2diss` function already includes a basic gravity computation (see Table 2). Here we present the `gravity` function, which does a very similar transformation, but is designed to take a document-term matrix (DTM) as input.

The first step is to binarize the DTM: if a word (columns) is mentioned at least once in a particular document (rows), the corresponding cell entry is 1, and 0 otherwise. Let $\mathbf{Y}$ denote this binarized DTM. From this simplified structure the word co-occurrence matrix $\mathbf{C}$ can be computed by $\mathbf{C} = \mathbf{Y}^\top\mathbf{Y}$. Thus, only the information on whether words occur together or not is considered. $\mathbf{C}$ is a symmetric matrix (with elements c_{ij}) with co-occurrence frequencies in the off-diagonals. Let $c_{i+} = \sum_{j\neq i} c_{ij}$ and $c_{+j} = \sum_{i\neq j} c_{ij}$ be the respective margins of $\mathbf{C}$

[8]Approximate timings: 1s for $n = 7$; 8s for $n = 8$; 75s for $n = 9$; for $n = 10$ the running time is already exceedingly long.

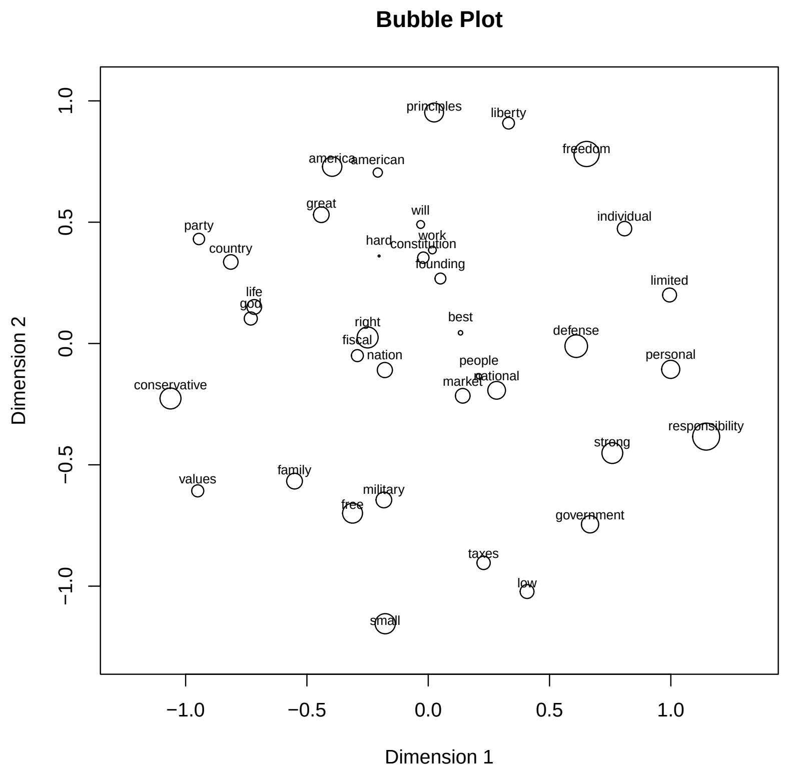


Figure 18: Bubble plot for gravity MDS solution on GOP data. The larger the bubbles, the larger the SPP.

(diagonal blanked out). The gravity model defines the following dissimilarities (for $i \neq j$):

$$\delta_{ij} = \sqrt{\frac{c_{i+}c_{+j}}{c_{ij}}}. \tag{19}$$

To illustrate a gravity application on text data we use a DTM similar to the one presented in Mair *et al.* (2014). This DTM was created on the basis of statements of 254 Republican voters who had to complete the sentence "I am a Republican because...". First, let us create the gravity dissimilarities according to the strategy outlined above.

```
R> gopD <- gravity(GOPdtm)$gravdiss
```

Note that using text data, $\mathbf{C}$ is typically sparse (i.e., many elements $c_{ij} = 0$). For these elements we cannot compute Equation (19) since we divide by 0. The function sets the corresponding entries to `NA`. In the subsequent MDS call, these elements are automatically blanked out by setting the corresponding weight w_{ij} to 0 in the basic stress equation.

```
R> fitGOP <- mds(gopD, type = "ordinal")
R> plot(fitGOP, plot.type = "bubbleplot", bubscale = 20)
```

Figure 18 shows the bubble plot which incorporates the stress-per-point (SPP) information. The larger a bubble, the larger the contribution of a particular object (here, word) to the total stress. Objects with large SPP values are responsible for misfit. The closer two words are in the configuration, the more frequently they have been mentioned together in a single statement.

An extension of this model is presented in Mair *et al.* (2014), who introduce the exponent λ in order to emphasize larger dissimilarities. The reason for this is that in text data we often end up with little variance in the input dissimilarities which leads to a concentric, circular representation of the configuration. Equation 19 changes to

$$\delta_{ij} = \left(\frac{c_{i+}c_{+j}}{c_{ij}}\right)^{\frac{\lambda}{2}}. \tag{20}$$

This extension is called *power gravity model*. The parameter λ needs to be chosen *ad hoc* and can take values from $[-\infty, \infty]$. For $\lambda < 1$ we shrink large dissimilarities, for $\lambda = 1$ we end up with the ordinary gravity model, and for $\lambda > 1$ we stretch large dissimilarities. Note that there is a trade-off between the choice of λ and the stress value: the more structure we create, the higher the stress value. This extension is relevant for metric MDS strategies such as ratio, interval, or spline MDS. The λ parameter can be specified in the `gravity` function.

A recent work by Rusch, Mair, and Hornik (2021) embeds the gravity formulation into a more general loss function which, among other things, finds an optimal λ during the optimization process.

7.3. Asymmetric MDS

So far, except in unfolding, Δ has been a symmetric matrix of input dissimilarities. In this section we aim to scale square asymmetric dissimilarity matrices. Young (1975), Collins (1987), Zielman and Heiser (1996), Borg and Groenen (2005, Chapter 23), and Bove and Okada (2018) present various asymmetric MDS variants of which **smacof** implements the *drift vector model* (Borg 1979). The starting point of this approach is to decompose the asymmetric dissimilarity matrix Δ into a symmetric part $\mathbf{M}$, and a skew-symmetric part $\mathbf{N}$:

$$\Delta = \mathbf{M} + \mathbf{N}, \tag{21}$$

with $\mathbf{M} = (\Delta + \Delta^\top)/2$, and $\mathbf{N} = (\Delta - \Delta^\top)/2$. In **smacof**, the `symdecomp` function can be used for this decomposition. Drift vector models display simultaneously the symmetric and the skew-symmetric part of Δ. They first fit an MDS (of any type) on the symmetric matrix $\mathbf{M}$, resulting in the configuration $\mathbf{X}$. The asymmetric information is then incorporated as follows (Borg and Groenen 2005, p. 503):

- For each object pair i, j compute $\mathbf{a}_{ij} = \mathbf{x}_i - \mathbf{x}_j$ which results in a vector of length p.
- Norm $\mathbf{a}_{ij}$ to unit length, resulting in $\mathbf{b}_{ij} = \mathbf{a}_{ij}/\sqrt{\mathbf{a}_{ij}^\top \mathbf{a}_{ij}}$.
- Incorporate the skew-symmetric part: $\mathbf{c}_{ij} = n_{ij}\mathbf{b}_{ij}$ with n_{ij} as the corresponding element in $\mathbf{N}$ (drift vectors).
- For a given point i, average the elements in $\mathbf{c}_{ij}$: $\mathbf{d}_i = n^{-1}\sum_j \mathbf{c}_{ij}$ (average drift vectors).
- For plotting, compute the vector lengths of $\mathbf{d}_i$ (root mean square of its elements, scaled by a factor of $\sqrt{n}/\mathrm{mean}(\mathbf{M})$), and the direction angle (relative to the y-axis) of $\alpha_i = \arccos(\mathbf{d}_i^\top \mathbf{u}/\sqrt{\mathbf{d}_i^\top \mathbf{d}_i})$ with $\mathbf{u}^\top = (0, 1)$.

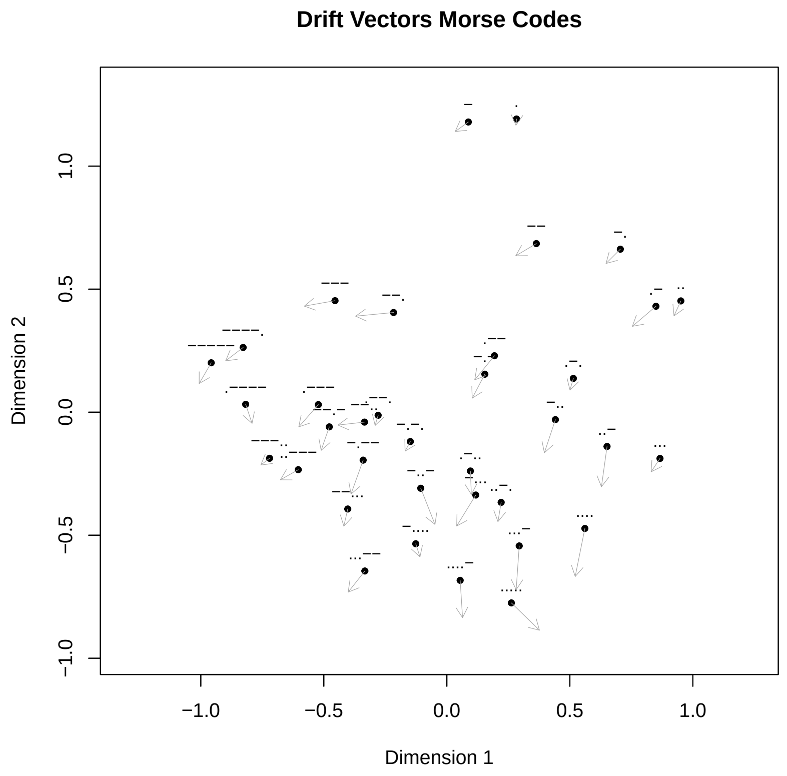


Figure 19: MDS solution for asymmetric Morse code data including drift vectors.

To illustrate the drift vector model, we use the classical Morse code data by Rothkopf (1957). Rothkopf asked 598 subjects to judge whether two signals, presented acoustically one after another, were the same or not. The values are the average percentages for which the answer "Same!" was given in each combination of row stimulus i and column stimulus j, where either i or j was the first signal presented. The responses were aggregated to confusion rates and subsequently subtracted from 1, such that the values represent dissimilarities. The `driftVector` function performs the decomposition from Equation 21, fits an MDS of choice on $\mathbf{M}$, and applies the drift vector computation steps outlined above. For the Morse code data, the resulting drift configuration plot based on a 2D ordinal MDS fit, is given in Figure 19.

```
R> morseDrift <- driftVectors(morse2, type = "ordinal")
R> plot(morseDrift, main = "Drift Vectors Morse Codes",
+    col.drift = "darkgray")
```

We see that the vectors tend to point in the bottom left direction; they are certainly not random. In the bottom left quadrant we mostly have longer signals suggesting that shorter signals are more often confused with longer ones than vice versa. Note that the plot function has a `vecscale` argument by which the user can modify the length of the drift vectors by a scaling factor.

Other approaches to scale asymmetric data implemented in R are the following. Vera and Rivera (2014) embed MDS into a structural equation modeling framework. Their approach is implemented in the **semds** package (Vera and Mair 2019). Zielman and Heiser (1993) developed a slide-vector approach which is implemented in **asymmetry** (Zielman 2021). Unfolding strategies from Section 6 can also be used for asymmetric dissimilarity matrices. An application can be found in Sagarra, Busing, Mar-Molinero, and Rialp (2018).

7.4. Procrustes

Sometimes it is of interest to compare multiple MDS configurations based on, for instance, different experimental conditions (the objects need to be the same within each condition). The idea of Procrustes (Hurley and Cattell 1962) is to remove "meaningless" configuration differences such as rotation, translation, and dilation (see Commandeur 1991, for an overview of Procrustean models). Note that Procrustes transformations do not change the fit (stress value) of an MDS.

In brief, Procrustes works as follows. Let $\mathbf{X}$ and $\mathbf{Y}$ be two MDS configuration matrices. $\mathbf{X}$ is the target configuration, and $\mathbf{Y}$ the configuration subject to Procrustes transformation leading to the transformed configuration matrix $\hat{\mathbf{Y}}$. Further, let $\mathbf{Z}$ be a centering matrix ($\mathbf{Z} = \mathbf{I} - n^{-1}\mathbf{1}\mathbf{1}^\top$). Procrustes involves the following steps:

1. Compute $\mathbf{C} = \mathbf{X}^\top\mathbf{Z}\mathbf{Y}$.
2. Perform an SVD on $\mathbf{C}$, i.e., $\mathbf{C} = \mathbf{P}\Phi\mathbf{Q}^\top$. Compute:
    - rotation matrix: $\mathbf{T} = \mathbf{Q}\mathbf{P}^\top$,
    - dilation factor: $s = tr(\mathbf{X}^\top\mathbf{Z}\mathbf{Y}\mathbf{T})/tr(\mathbf{Y}^\top\mathbf{Z}\mathbf{Y})$,
    - translation vector $\mathbf{t} = n^{-1}(\mathbf{X} - s\mathbf{Y}\mathbf{T}^\top)\mathbf{1}$.
3. Final solution: $\hat{\mathbf{Y}} = s\mathbf{Y}\mathbf{T} + \mathbf{1}\mathbf{t}^\top$.

The matrix $\hat{\mathbf{Y}}$ contains the Procrustes transformed configuration and replaces $\mathbf{Y}$. The target configuration $\mathbf{X}$ and $\hat{\mathbf{Y}}$ can be plotted jointly and allows researchers to explore potential differences between the configurations.

The dataset we use to illustrate Procrustes is taken from Vaziri-Pashkam and Xu (2019). In their fMRI experiment on visual object representations they used both natural and artificial shape categories to study the activation of various brain regions (each object represents a particular brain region). We start with fitting two ordinal MDS solutions, one for each condition

```
R> artD <- sim2diss(VaziriXu$artificialR)
R> fitart <- mds(artD, type = "ordinal")
R> realD <- sim2diss(VaziriXu$realR)
R> fitnat <- mds(realD, type = "ordinal")
```

By plotting the two configurations, Figure 20 suggests that these configurations are different. Let us apply a Procrustes transformation with the artificial condition as target configuration $\mathbf{X}$, and the natural condition solution as testee configuration $\mathbf{Y}$, subject to Procrustes transformation.

```
R> fitproc <- Procrustes(fitart$conf, fitnat$conf)
R> fitproc
```

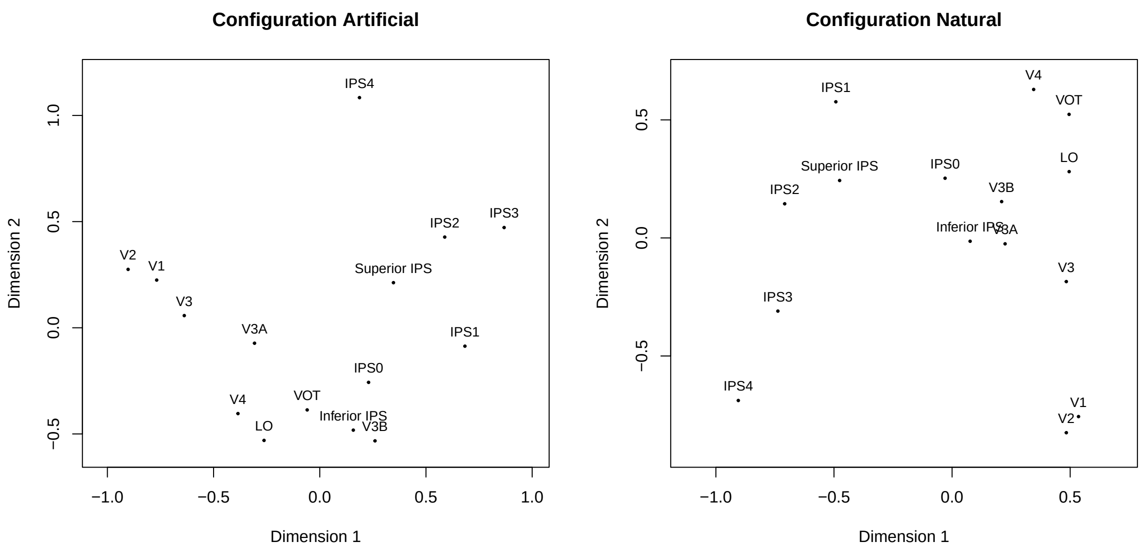


Figure 20: Left panel: MDS configuration natural condition. Right panel: MDS configuration artificial condition.

```
Call: Procrustes(X = fitart$conf, Y = fitnat$conf)

Congruence coefficient: 0.971
Alienation coefficient: 0.241

Rotation matrix:
       D1     D2
D1 -0.826 -0.564
D2  0.564 -0.826

Translation vector: 0 0
Dilation factor: 0.906
```

The print output shows the rotation matrix, the dilation factor, and the translation vector (which is always 0 if two MDS configurations are involved, due to normalization constraints). In addition, it reports Tucker's *congruence coefficient* for judging the similarity of two configurations. This coefficient is derived from factor analysis and can be computed as follows:

$$c(\mathbf{X}, \mathbf{Y}) = \frac{\sum_{i<j} d_{ij}(\mathbf{X}) d_{ij}(\mathbf{Y})}{\sqrt{\sum_{i<j} d_{ij}^2(\mathbf{X})}\sqrt{\sum_{i<j} d_{ij}^2(\mathbf{Y})}} \tag{22}$$

A few remarks regarding the congruence coefficient. First, it is generally recommended that one uses the congruence coefficient to judge configurational similarity and not the correlation coefficient, since correlating distances does not properly assess the similarity of configurations (see Borg and Groenen 2005, p. 439–440 for details). Second, there is actually no Procrustes transformation needed to compute the congruence coefficient, since $c(\mathbf{X}, \mathbf{Y}) = c(\mathbf{X}, \hat{\mathbf{Y}})$. Third, when applying (22) within an MDS context, the resulting value of $c(\mathbf{X}, \mathbf{Y})$ is generally high. In factor analysis, values in the range of 0.85–0.94 are considered to be "fairly

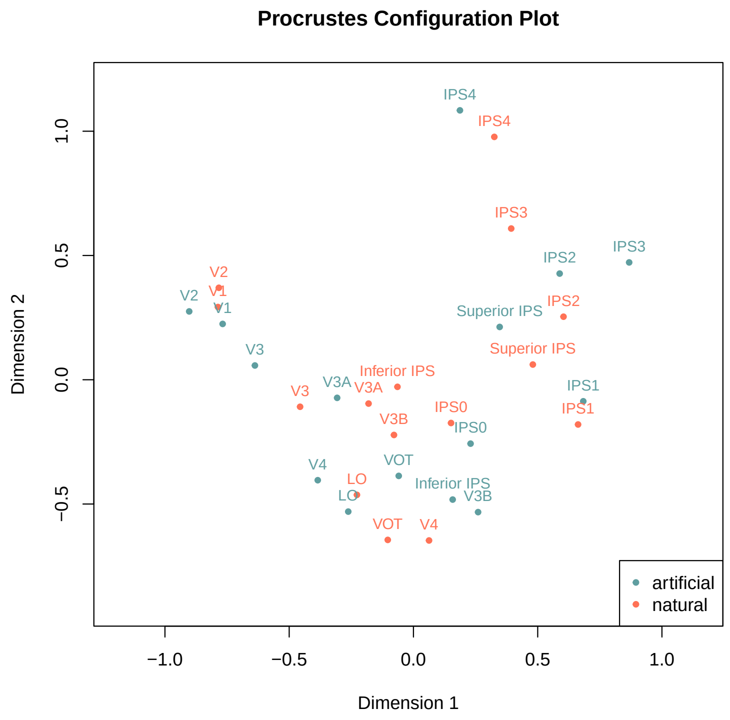


Figure 21: Procrustes on two MDS configurations: the configuration from the artificial condition acts as target; the one from the natural condition is Procrustes transformed.

similar", and values higher than 0.95 suggest that the two factors are considered equal (see Lorenzo-Seva and Ten Berge 2006, for details).

As an alternative we can consider Guttman's *alienation coefficient*, which is simply

$$K(\mathbf{X}, \mathbf{Y}) = \sqrt{1 - c(\mathbf{X}, \mathbf{Y})^2}. \tag{23}$$

This measure differentiates better between two solutions than the congruence coefficient.

Figure 21 shows the Procrustes transformed solution (i.e., plotting $\mathbf{X}$ and $\hat{\mathbf{Y}}$ jointly) and suggests that the two configurations are actually very similar, apart from a few points. The user can request the (sorted) distances between each pair of testee and target points via `fitproc$pairdist`. V4, inferior IPS, IPS3, and V3B show the biggest differences across the two conditions.

```
R> plot(fitproc, legend = list(labels = c("artificial", "natural")))
```

Another option to apply Procrustes is to use a theoretical configuration as target. For instance, Borg and Leutner (1983) constructed rectangles on the basis of a grid design (as contained in `rect_constr`) which we use as target configuration. Participants had to rate similarity among rectangles within this grid. Based on these ratings a dissimilarity matrix was constructed, here subject to a 2D ordinal MDS solution. Within the context of theoretical solutions it is sometimes interesting to determine the stress value based on the dissimilarity matrix and an initial configuration (with 0 iterations). The `stress0` function does the job.

```
R> stress0(rectangles, init = rect_constr)
```

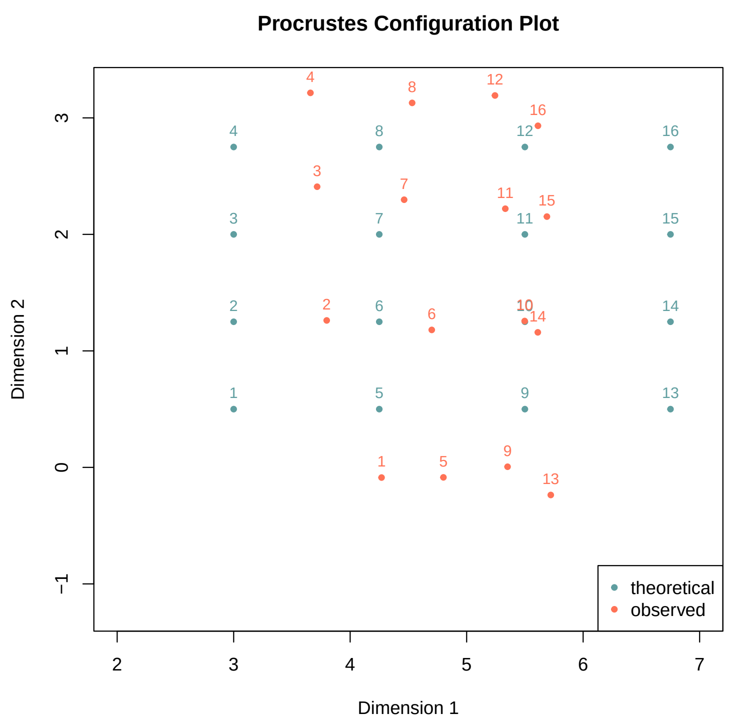


Figure 22: Procrustes with theoretical grid as target configuration (blue dots).

```
Call:
stress0(delta = rectangles, init = rect_constr)

Model: Symmetric SMACOF
Number of objects: 16
Stress-1 value: 0.323
Number of iterations: 0
```

Now we fit an ordinal MDS model, using the theoretical rectangle alignment as starting value. The resulting MDS configuration is used as testee configuration $\mathbf{Y}$, subject to Procrustes.

```
R> fitrect <- mds(rectangles, type = "ordinal", init = rect_constr)
R> procRect <- Procrustes(rect_constr, fitrect$conf)
R> plot(procRect, legend = list(labels = c("theoretical", "observed")),
+    xlim = c(2, 7))
```

Figure 22 plots the rectangle grid and the Procrustes transformed configuration jointly. We see clear differences, especially in the right end of the rectangle grid. These differences are also reflect in the considerably high alienation coefficient of 0.35.

Note that Procrustes is not limited to MDS applications. It can be applied to any configuration matrices $\mathbf{X}$ and $\mathbf{Y}$ as long as the objects involved are the same and the matrices have the same dimensions. Other forms of generalized Procustes analysis are given in Borg and Groenen (2005, Section 20.9).

8. Conclusion

In this follow-up paper to De Leeuw and Mair (2009b), who introduced the **smacof** package, we presented numerous updates that have been implemented over the years. It is safe to say that these developments establish **smacof** as the most comprehensive implementation of MDS and unfolding techniques in R. Still, there are several tasks on our to-do list. First, we plan to implement a `fastMDS` routine entirely written in C to speed up computations for large data settings. Second, we will work on an implementation of inverse MDS (De Leeuw and Groenen 1997). Third, we aim to extend spherical MDS and unfolding to more general geometric shapes such as a pre-specified polygon mesh.

Acknowledgments

We would like to thank Ingwer Borg for his contributions to the **smacof** package and his critical applied remarks which enhanced the usability of the package, and Frank Busing for his contributions to the unfolding implementation and the `sim2diss` function. In addition, we would like to thank three anonymous reviewers and the associate editor for their helpful comments.

References

Abelson RP, Sermat V (1962). "Multidimensional Scaling of Facial Expressions." *Journal of Experimental Psychology*, **63**, 546–554. doi:10.1037/h0042280.

Borg I (1979). "Ein Verfahren zur Analyse metrischer asymmetrischer Proximitätsmatrizen [A Method to Analyze Asymmetric Proximity Matrices]." *Archiv für Psychologie*, **131**, 183–194.

Borg I, Bardi A, Schwartz SH (2017). "Does the Value Circle Exist within Persons or Only across Persons?" *Journal of Personality*, **85**, 151–162. doi:10.1111/jopy.12228.

Borg I, Groenen PJF (2005). *Modern Multidimensional Scaling: Theory and Applications*. 2nd edition. Springer-Verlag, New York.

Borg I, Groenen PJF, Mair P (2018). *Applied Multidimensional Scaling and Unfolding*. 2nd edition. Springer-Verlag, New York.

Borg I, Leutner D (1983). "Dimensional Models for the Perception of Rectangles." *Perception and Psychophysics*, **34**, 257–269. doi:10.3758/BF03202954.

Borg I, Lingoes JC (1980). "A Model and Algorithm for Multidimensional Scaling with External Constraints on the Distances." *Psychometrika*, **45**, 25–38. doi:10.1007/BF02293597.

Borg I, Mair P (2017). "The Choice of Initial Configurations in Multidimensional Scaling: Local Minima, Fit, and Interpretability." *Austrian Journal of Statistics*, **46**, 19–32. doi:10.17713/ajs.v46i2.561.

Bove G, Okada A (2018). "Methods for the Analysis of Asymmetric Pairwise Relationships." *Advances in Data Analysis and Classification*, **12**, 5–31. doi:10.1007/s11634-017-0307-9.

Busing FMTA, Groenen PJF, Heiser WJ (2005). "Avoiding Degeneracy in Multidimensional Unfolding by Penalizing on the Coefficient of Variation." *Psychometrika*, **70**, 71–98. doi:10.1007/s11336-001-0908-1.

Carroll JD (1972). "Multidimensional Scaling: Theory and Applications in the Behavioral Sciences." In RN Shepard, AK Romney, JB Nerlove (eds.), *Individual Differences in Multidimensional Scaling*, pp. 105–155. Seminar Press, New York.

Collins LM (1987). "Sociometry: Deriving Sociograms via Asymmetric Multidimensional Scaling." In FW Young, RM Hamer (eds.), *Multidimensional Scaling: History, Theory, and Applications*, pp. 179–198. Lawrence Erlbaum Associates, New York.

Commandeur JJF (1991). *Matching Configurations.* DSWO Press, Leiden.

Coombs CH (1964). *A Theory of Data.* John Wiley & Sons, New York.

Copleston FC (1949). *A History of Philosophy. Volume I: Greece and Rome.* Doubleday, New York.

Cox DR, Brandwood L (1959). "On a Discriminatory Problem Connected with the Work of Plato." *Journal of the Royal Statistical Society B*, **21**, 195–200. doi:10.1111/j.2517-6161.1959.tb00329.x.

Cox TF, Cox MAA (1991). "Multidimensional Scaling on a Sphere." *Communications in Statistics*, **20**, 2943–2953. doi:10.1080/03610929108830679.

Dabic M, Hatzinger R (2009). "Zielgruppenadäquate Abläufe in Konfigurationssystemen: Eine empirische Studie im Automobilmarkt – Partial Rankings [Targeted Processes in Configuration Systems: An Empirical Study on the Car Market – Partial Rankings]." In R Hatzinger, R Dittrich, T Salzberger (eds.), *Präferenzanalyse mit R: Anwendungen aus Marketing, Behavioural Finance und Human Resource Management [Analysis of Preferences with R: Applications in Marketing, Behavioral Finance and Human Resource Management]*, pp. 119–150. Facultas, Vienna.

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 Publishing Company, Amsterdam.

De Leeuw J (2019). *Pseudo Confidence Regions for MDS.* URL http://deleeuwpdx.net/pubfolders/confidence/confidence.pdf.

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

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 Publishing Company, Amsterdam.

De Leeuw J, Mair P (2009a). "Gifi Methods for Optimal Scaling in R: The package **homals**." *Journal of Statistical Software*, **31**(4), 1–20. doi:10.18637/jss.v031.i04.

De Leeuw J, Mair P (2009b). "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 (2015). "Shepard Diagram." In *Wiley StatsRef: Statistics Reference Online*. John Wiley & Sons, New York. doi:10.1002/9781118445112.stat06268.pub2.

De Leeuw J, Meulman J (1986). "A Special Jackknife for Multidimensional Scaling." *Journal of Classification*, **3**, 97–112. doi:10.1007/BF01896814.

DeSarbo WS, Rao VR (1984). "GENFOLD2: A Set of Models and Algorithms for the GENeral UnFOLDing Analysis of Preference/Dominance Data." *Journal of Classification*, **1**, 147–186. doi:10.1007/BF01890122.

Engen T, Levy N, Schlosberg H (1958). "The Dimensional Analysis of a New Series of Facial Expressions." *Journal of Experimental Psychology*, **55**, 454–458. doi:10.1037/h0047240.

Esposito F, Malerba D, Tamma V, Bock HH (2000). "Similarity and Dissimilarity." In E Diday, HH Bock (eds.), *Analysis of Symbolic Data*, pp. 139–197. Springer-Verlag, New York.

Fleiss JL, Levin BA, Paik MC (2003). *Statistical Methods for Rates and Proportions*. 3rd edition. John Wiley & Sons, Hoboken.

Gabriel KR (1971). "The Biplot Graphical Display of Matrices with Application to Principal Component Analysis." *Biometrika*, **58**, 453–457. doi:10.1093/biomet/58.3.453.

Gifi A (1990). *Nonlinear Multivariate Analysis*. John Wiley & Sons, Chichester.

Gower JC, Legendre P (1986). "Metric and Euclidean Properties of Dissimilarity Coefficients." *Journal of Classification*, **3**, 5–48. doi:10.1007/BF01896809.

Gower JC, Lubbe S, Le Roux N (2011). *Understanding Biplots*. John Wiley & Sons, Chichester.

Graffelman J (2020). ***calibrate**: Calibration of Scatterplot and Biplot Axes*. R package version 1.7.7, URL http://CRAN.R-project.org/package=calibrate.

Greenacre M (2010). *Biplots in Practice*. Fundación BBVA, Bilbao.

Guttman L (1965). "A Faceted Definition of Intelligence." *Scripta Hierosolymitana*, **14**, 166–181.

Hatzinger R, Dittrich R (2012). "**prefmod**: An R Package for Modeling Preferences Based on Paired Comparisons, Rankings, or Ratings." *Journal of Statistical Software*, **48**(10), 1–31. doi:10.18637/jss.v048.i10.

Haynes KE, Fotheringham AS (1984). *Gravity and Spatial Interaction Models*. Sage, Beverly Hills.

Heiser WJ, Busing FMTA (2004). "Multidimensional Scaling and Unfolding of Symmetric and Asymmetric Proximity Relations." In D Kaplan (ed.), *The Sage Handbook of Quantitative Methodology for the Social Sciences*, pp. 25–48. Sage Publications, Thousand Oaks.

Heiser WJ, Meulman J (1983). "Constrained Multidimensional Scaling, Including Confirmation." *Applied Psychological Measurement*, **7**, 381–404. doi:10.1177/014662168300700402.

Hurley JR, Cattell RB (1962). "The Procrustes Program: Producing Direct Rotation to Test a Hypothesized Factor Structure." *Behavioral Science*, **7**, 258–262. doi:10.1002/bs.3830070216.

Jacoby WG (1999). "Levels of Measurement and Political Research: An Optimistic View." *American Journal of Political Science*, **43**, 271–301. doi:10.2307/2991794.

Jacoby WG, Armstrong DA (2014). "Bootstrap Confidence Regions for Multidimensional Scaling Solutions." *American Journal of Political Science*, **58**, 264–278. doi:10.1111/ajps.12056.

Keshavarzi M, Dehghan MA, Mashinchi M (2009). "Classification Based on Similarity and Dissimilarity through Equivalence Classes." *Applied and Computational Mathematics*, **8**, 203–215.

Kruskal JB (1964). "Multidimensional Scaling by Optimizing Goodness of Fit to a Nonmetric Hypothesis." *Psychometrika*, **29**, 1–27. doi:10.1007/BF02289565.

Lawler EE (1967). "The Multitrait-Multirater Approach to Measuring Job Managerial Performance." *Journal of Applied Psychology*, **51**, 369–381. doi:10.1037/h0025095.

Lorenzo-Seva U, Ten Berge JMF (2006). "Tucker's Congruence Coefficient as a Meaningful Index of Factor Similarity." *Methodology*, **2**, 57–64. doi:10.1027/1614-2241.2.2.57.

Mair P (2018). *Modern Psychometrics with R*. Springer-Verlag, New York. doi:10.1007/978-3-319-93177-7.

Mair P (2020). ***MPsychoR**: Modern Psychometrics with R*. R package version 0.10-8, URL https://CRAN.R-project.org/package=MPsychoR.

Mair P, Borg I, Rusch T (2016). "Goodness-of-Fit Assessment in Multidimensional Scaling and Unfolding." *Multivariate Behavioral Research*, **51**, 772–789. doi:10.1080/00273171.2016.1235966.

Mair P, De Leeuw J (2015). "Unidimensional Scaling." In *Wiley StatsRef: Statistics Reference Online*. John Wiley & Sons, New York. doi:10.1002/9781118445112.stat06462.pub2.

Mair P, Rusch T, Hornik K (2014). "The Grand Old Party: A Party of Values?" *SpringerPlus*, **3**(697), 1–10. doi:10.1186/2193-1801-3-697.

McNally RJ, Mair P, Mugno BL, Riemann BC (2017). "Comorbid Obsessive-Compulsive Disorder and Depression: A Bayesian Network Approach." *Psychological Medicine*, **47**, 1204–1214. doi:10.1017/S0033291716003287.

McNally RJ, Robinaugh DJ, Wu GWY, Wang L, Deserno MK, Borsboom D (2015). "Mental Disorders as Causal Systems: A Network Approach to Posttraumatic Stress Disorder." *Clinical Psychological Science*, **3**, 836–849. doi:10.1177/2167702614553230.

Meulman J, Heiser WJ (1983). "The Display of Bootstrap Solutions in MDS." *Technical report*, Bell Laboratories, Murray Hill.

Murdoch D, Chow ED (2020). ***ellipse**: Functions for Drawing Ellipses and Ellipse-Like Confidence Regions.* R package version 0.4.2, URL https://CRAN.R-project.org/package=ellipse.

Rabinowitz GB (1975). "Introduction to Nonmetric Multidimensional Scaling." *American Journal of Political Science*, **19**, 343–390.

Ramsay JO (1977). "Maximum Likelihood Estimation in Multidimensional Scaling." *Psychometrika*, **42**, 241–266. doi:10.1007/BF02294052.

Ramsay JO (1982). "Some Statistical Approaches to Multidimensional Scaling Data." *Journal of the Royal Statistical Society A*, **145**, 285–303. doi:10.2307/2981865.

Ramsay JO (1988). "Monotone Regression Splines in Action." *Statistical Science*, **3**, 425–461. doi:10.1214/ss/1177012761.

Ramsay JO (1997). *Multiscale Manual (Extended Version).* McGill University, Montreal.

Rodgers JL, Young FW (1981). "Successive Unfolding of Family Preferences." *Applied Psychological Measurement*, **5**, 51–62. doi:10.1177/014662168100500108.

Rothkopf EZ (1957). "A Measure of Stimulus Similarity and Errors in some Paired-Associate Learning." *Journal of Experimental Psychology*, **53**, 94–101. doi:10.1037/h0041867.

Rusch T, Mair P, Hornik K (2021). "Cluster Optimized Proximity Scaling." *Journal of Computational and Graphical Statistics*, **30**(4), 1156–1167. doi:10.1080/10618600.2020.1869027.

Sagarra M, Busing FMTA, Mar-Molinero C, Rialp J (2018). "Assessing the Asymmetric Effects on Branch Rivalry of Spanish Financial Sector Restructuring." *Advances in Data Analysis and Classification*, **12**, 131–153. doi:10.1007/s11634-014-0186-2.

Schwartz SH (1992). "Universals in the Content and Structure of Values: Theoretical Advances and Empirical Tests in 20 Countries." In M Zanna (ed.), *Advances in Experimental Social Psychology*, pp. 1–65. Academic Press, New York.

Shepard RN (1957). "Stimulus and Response Generalization: A Stochastic Model Relating Generalization to Distance in Psychological Space." *Psychometrika*, **22**, 325–345. doi:10.1007/BF02288967.

Spence I, Ogilvie JC (1973). "A Table of Expected Stress Values for Random Rankings in Nonmetric Multidimensional Scaling." *Multivariate Behavioral Research*, **8**, 511–517. doi:10.1207/s15327906mbr0804_8.

Spence I, Young FW (1978). "Monte Carlo Studies in Nonmetric Scaling." *Psychometrika*, **43**, 115–117. doi:10.1007/BF02294095.

Srinivasan V, Shocker AD (1973). "Linear Programming Techniques for Multidimensional Analysis." *Psychometrika*, **38**, 337–369. doi:10.1007/BF02291658.

Tamir DI, Thornton MA, Contreras JM, Mitchell JP (2016). "Neural Evidence that Three Dimensions Organize Mental State Representation: Rationality, Social Impact, and Valence." *Proceedings of the National Academy of Sciences of the United States of America*, **113**, 194–199. doi:10.1073/pnas.1511905112.

Torgerson WS (1952). "Multidimensional Scaling: I. Theory and Method." *Psychometrika*, **17**, 401–419. doi:10.1007/BF02288916.

Tucker LR (1960). "Intra-Individual and Inter-Individual Multidimensionality." In H Gulliksen, S Messick (eds.), *Psychological Scaling: Theory and Applications*, pp. 155–167. John Wiley & Sons, New York.

Vaziri-Pashkam M, Xu Y (2019). "An Information-Driven Two-Pathway Characterization of Occipito-Temporal and Posterior Parietal Visual Object Representations." *Cerebral Cortex*, **29**, 2034–2050. doi:10.1093/cercor/bhy080.

Vera JF (2017). "Distance Stability Analysis in Multidimensional Scaling Using the Jackknife Method." *British Journal of Mathematical and Statistical Psychology*, **70**, 25–41. doi:10.1111/bmsp.12079.

Vera JF, Mair P (2019). "**semds**: An R Package for Structural Equation Multidimensional Scaling." *Structural Equation Modeling: A Multidisciplinary Journal*, **26**(5), 803–818. doi:10.1080/10705511.2018.1561292.

Vera JF, Rivera CD (2014). "A Structural Equation Multidimensional Scaling Model for One-Mode Asymmetric Dissimilarity Data." *Structural Equation Modeling: A Multidisciplinary Journal*, **21**, 54–62. doi:10.1080/10705511.2014.856696.

Weinberg SL, Carroll JD, Cohen HS (1984). "Confidence Regions for INDSCAL Using the Jackknife and Bootstrap Techniques." *Psychometrika*, **49**, 475–491. doi:10.1007/BF02302586.

Wish M (1971). "Individual Differences in Perceptions and Preferences Among Nations." In CW King, D Tigert (eds.), *Attitude Research Reaches New Heights*, pp. 312–328. American Marketing Association, Chicago.

Young FW (1975). "An Asymmetirc Euclidean Model for Multi-Process Asymmetric Data." Paper presented at US-Japan Seminar on MDS, San Diego.

Zielman B (2021). ***asymmetry**: Multidimensional Scaling of Asymmetric Data.* R package version 2.0.3, URL https://CRAN.R-project.org/package=asymmetry.

Zielman B, Heiser WJ (1993). "The Analysis of Asymmetry by a Slide-Vector." *Psychometrika*, **58**, 101–114. doi:10.1007/BF02294474.

Zielman B, Heiser WJ (1996). "Models for Asymmetric Proximities." *British Journal of Mathematical and Statistical Psychology*, **49**, 127–146. doi:10.1111/j.2044-8317.1996.tb01078.x.

Affiliation:

Patrick Mair
Department of Psychology
Harvard University
33 Kirkland Street
Cambridge, MA 02138, United States of America
E-mail: `mair@fas.harvard.edu`
URL: `http://scholar.harvard.edu/mair`

Journal of Statistical Software `https://www.jstatsoft.org/`
published by the Foundation for Open Access Statistics `https://www.foastat.org/`
April 2022, Volume 102, Issue 10 *Submitted:* 2019-06-06
`doi:10.18637/jss.v102.i10` *Accepted:* 2021-09-27