

Global Minima by Penalized Full-dimensional Scaling

Jan de Leeuw

First created on May 06, 2019. Last update on July 21, 2023

Abstract

The full-dimensional (metric, Euclidean, least squares) multidimensional scaling stress loss function is combined with a quadratic external penalty function term. The trajectory of minimizers of stress for increasing values of the penalty parameter is then used to find (tentative) global minima for low-dimensional multidimensional scaling. This is illustrated with several one-dimensional and two-dimensional examples.

Contents

Introduction	2
Convex FDS	3
FDS using SMACOF	4
Penalizing Dimensions	5
Local Minima	6
Algorithm	7
Examples	7
Chi Squares	8
Regular Simplex	10
Intelligence	12
Countries	14
Dutch Political Parties	16
Ekman	19
Morse in Two	23
Vegetables	26
Plato	29
Morse in One	38
Discussion	41

Appendix A: Exterior Penalty Methods 42

Appendix B: Code 44

penalty.R	44
runPenalty.R	46
matchMe.R	49
plotMe.R	50
checkUni.R	51

References 51

Note: This is a working paper which will be expanded/updated frequently. All suggestions for improvement are welcome. The directory deleeuwpx.net/pubfolders/penalty has a pdf version, a html version, the bib files, the complete Rmd file with the code chunks, and the R source code.

Introduction

Full-dimensional Scaling (FDS) was introduced by De Leeuw (1993). De Leeuw, Groenen, and Mair (2016) discuss it in some detail. In FDS we minimize the usual Multidimensional Scaling (MDS) least squares loss function first used by Kruskal (1964a) and Kruskal (1964b).

$$\sigma(Z) = \frac{1}{2} \sum_{1 \leq i < j \leq n} \sum w_{ij} (\delta_{ij} - d_{ij}(Z))^2 \quad (1)$$

over all $n \times n$ *configuration matrices* Z . The loss at Z is often called the *stress* of configuration Z . More generally we define pMDS as the problem of minimizing (1) over all $n \times p$ matrices. Thus FDS is the same as nMDS. If a configuration Z has n columns (i.e. is square) it is called a *full configuration*.

In (1) the matrices $W = \{w_{ij}\}$ and $\Delta = \{\delta_{ij}\}$ of *weights* and *dissimilarities* are non-negative, symmetric, and hollow. To simplify matters we suppose both W and Δ have positive off-diagonal elements. The matrix $D(X) = \{d_{ij}(Z)\}$ has the *Euclidean distances* between the rows of the configuration Z . Thus

$$d_{ij}(Z) = \sqrt{(z_i - z_j)'(z_i - z_j)}.$$

We now introduce some standard MDS notation, following De Leeuw (1977). Define the matrix $V = \{v_{ij}\}$ by

$$v_{ij} = \begin{cases} -w_{ij} & \text{if } i \neq j, \\ \sum_{j=1}^n w_{ij} & \text{if } i = j, \end{cases} \quad (2)$$

and the matrix valued function $B(Z) = \{b_{ij}(Z)\}$ by

$$b_{ij}(Z) = \begin{cases} -w_{ij}e_{ij}(Z) & \text{if } i \neq j, \\ \sum_{j=1}^n w_{ij}e_{ij}(Z) & \text{if } i = j, \end{cases} \quad (3)$$

where $E(Z) = \{e_{ij}(Z)\}$ is defined as

$$e_{ij}(Z) = \begin{cases} \frac{\delta_{ij}}{d_{ij}(Z)} & \text{if } d_{ij}(Z) > 0, \\ 0 & \text{otherwise.} \end{cases} \quad (4)$$

Note that V and $B(Z)$ are both positive semi-definite and doubly-centered. Matrix V has rank $n - 1$. If all off-diagonal $d_{ij}(Z)$ are positive then $B(Z)$ has rank $n - 1$ for all Z . Note that De Leeuw (1984) established that near a local minimum of stress all off-diagonal distances are indeed positive. The only vectors in the null-space of both V and $B(Z)$ are the vectors proportional to the vector with all elements equal to one.

We assume in addition, without loss of generality, that

$$\frac{1}{2} \sum_{1 \leq i < j \leq n} w_{ij} \delta_{ij}^2 = 1.$$

With these definitions we can rewrite the stress (1) as

$$\sigma(Z) = 1 - \mathbf{tr} \, Z' B(Z) Z + \frac{1}{2} \mathbf{tr} \, Z' V Z, \quad (5)$$

and we can write the stationary equations as

$$(V - B(Z))Z = 0, \quad (6)$$

or, in fixed point form, $Z = V^+ B(Z) Z$.

Equation (5) shows, by the way, something which is already obvious from (1). Distances are invariant under translation. This is reflected in $B(Z)$ and V being doubly-centered. As a consequence we usually require, again without loss of generality, that Z is column-centered. And that implies that Z has rank at most $n - 1$, which means that FDS is equivalent to minimizing stress over all $n \times (n - 1)$ matrices, which we can assume to be column-centered as well. Configurations with $n - 1$ columns can be called full configurations as well. In addition, distances are invariant under rotation, and consequently if Z solves the stationary equations with value $\sigma(Z)$ then ZK solves the stationary equations for all rotation matrices K , and $\sigma(ZK) = \sigma(Z)$. This means there are no isolated local minima in configuration space, each local minimum is actually a continuum of rotated matrices in $\mathbb{R}^{n \times n}$. This is a nuisance in the analysis of FDS and pMDS that is best dealt with by switching to the parametrization outlined in De Leeuw (1993).

Convex FDS

Instead of defining the loss function (1) on the space of all $n \times n$ configuration matrices Z we can also define it over the space of all positive semidefinite matrices C of order n . This gives

$$\sigma(C) = 1 - \sum_{1 \leq i < j \leq n} w_{ij} \delta_{ij} \sqrt{c_{ii} + c_{jj} - 2c_{ij}} + \frac{1}{2} \mathbf{tr} \, VC. \quad (7)$$

The Convex Full-dimensional Scaling (CFDS) problem is to minimize loss function (7) over all $C \succeq 0$. Obviously if Z minimizes (1) then $C = ZZ'$ minimizes (7). And, conversely, if C minimizes (7) then any Z such that $C = ZZ'$ minimizes (1).

The definition (7) shows that the CFDS loss function is a convex function on the cone of positive semi-definite matrices, because the square root of a non-negative linear function of the elements of C is concave. Positivity of the weights and dissimilarities implies that loss is actually strictly convex. The necessary and sufficient conditions for C to be the unique solution of the CFDS problem are simply the conditions for a proper convex function to attain its minimum at C on a closed convex cone (Rockafellar (1970), theorem 31.4).

$$\begin{aligned} V - B(C) &\succeq 0, \\ C &\succeq 0, \\ \text{tr } C(V - B(C)) &= 0. \end{aligned}$$

The conditions say that C and $V - B(C)$ must be positive semi-definite and have complementary null spaces.

By the same reasoning as in the full configuration case, we also see that CFDS is equivalent to maximizing (7) over all doubly-centered positive semi-definite matrices.

If C is the solution of the CFDS problem then $\mathbf{rank}(C)$ is called the *Gower rank* of the MDS problem defined by W and Δ (De Leeuw (2016)). Although there is a unique Gower rank associated with each CFDS problem, we can also talk about the *approximate Gower rank* by ignoring the small eigenvalues of C .

FDS using SMACOF

The usual SMACOF algorithm can be applied to FDS as well. The iterations start with $Z^{(0)}$ and use the update rule

$$Z^{(k+1)} = V^+ B(Z^{(k)}) Z^{(k)}, \quad (8)$$

where V^+ is the Moore-Penrose inverse of V , and is consequently also doubly-centered. This means that all $Z^{(k)}$ in the SMACOF sequence, except possibly $Z^{(0)}$, are column-centered and of rank at most $n - 1$. Equation (8) also shows that if $Z^{(0)}$ is of rank $p < n - 1$ then all $Z^{(k)}$ are of rank p as well.

De Leeuw (1977) shows global convergence of the SMACOF sequence for pMDS, generated by (8), to a stationary point, i.e. a point satisfying $(V - B(Z))Z = 0$. This result also applies, of course, to nMDS, i.e. FDS. If Z is a solution of the stationary equations then with $C = ZZ'$ we have both $(V - B(C))C = 0$ and $C \succeq 0$, but since we generally do not have $V - B(Z) \succeq 0$, this does not mean that C solves the CFDS problem.

In fact, suppose the unique CMDS solution has Gower rank $r \geq 2$. Start the SMACOF FDS iterations (8) with $Z^{(0)}$ of the form $Z^{(0)} = \begin{bmatrix} X^{(0)} & | & 0 \end{bmatrix}$, where $X^{(0)}$ is an $n \times p$ matrix of rank $p < r$. All $Z^{(k)}$ will be of this form and will also be of rank p , and all accumulation points Z of the SMACOF sequence will have this form and $\mathbf{rank}(Z) \leq p$. Thus $C = ZZ'$ cannot be the solution of the CMDS problem.

The next result shows that things are alright, after all. Although stress in FDS is certainly not a convex function of Z , it remains true that all local minima are global.

Lemma 1: [Expand] If FDS stress has a local minimum at $\begin{bmatrix} X & | & 0 \end{bmatrix}$, where X is $n \times p$ and the zero block is $n \times q$ with $q > 1$, then

$$1: \mathcal{D}\sigma(X) = (V - B(X))X = 0.$$

$$2: \mathcal{D}^2\sigma(X) \gtrsim 0.$$

$$3: V - B(X) \gtrsim 0.$$

Proof: We use the fact that stress is differentiable at a local minimum (De Leeuw (1984)). If $Z = \begin{bmatrix} X & | & 0 \end{bmatrix} + \epsilon \begin{bmatrix} P & | & Q \end{bmatrix}$ then we must have $\sigma(Z) \geq \sigma(X)$ for all P and Q . Now

$$\begin{aligned} \sigma(Z) = \sigma(X) + \epsilon \operatorname{tr} P' \mathcal{D}\sigma(X) + \\ + \frac{1}{2} \epsilon^2 \mathcal{D}^2\sigma(X)(P, P) + \frac{1}{2} \epsilon^2 \operatorname{tr} Q'(V - B(X))Q + o(\epsilon^2). \end{aligned} \quad (9)$$

The lemma follows from this expansion. ■

Theorem 1: [FDS Local Minima] If stationary point Z of FDS is a local minimum, then it also is the global minimum, and $C = ZZ'$ solves the CFDS problem.

Proof: We start with a special case. Suppose Z is a doubly-centered solution of the FDS stationary equations with $\mathbf{rank}(Z) = n - 1$. Then $(V - B(Z))Z = 0$ implies $V = B(Z)$, which implies $\delta_{ij} = d_{ij}(Z)$ for all i, j . Thus $\sigma(Z) = 0$, which obviously is the global minimum.

Now suppose Z is a doubly-centered local minimum solution of the FDS stationary equations with $\mathbf{rank}(Z) = r < n - 1$. Without loss of generality we assume Z is of the form $Z = \begin{bmatrix} X & | & 0 \end{bmatrix}$, with X an $n \times r$ matrix of rank r . For $C = ZZ'$ to be a solution of the CFDS problem it is necessary and sufficient that $V - B(Z) \gtrsim 0$. Lemma 1 shows that this is indeed the case at a local minimum. ■

Corollary 1: [Saddle] A pMDS solution of the stationary equations with Z singular is a saddle point.

Corollary 2: [Nested] Solutions of the stationary equations of pMDS are saddle points of qMDS with $q > p$.

The proof of lemma 1 shows that for any $n \times p$ configuration Z , not just for solutions of the FDS stationary equations, if $V - B(Z)$ is indefinite we can decrease loss by adding another dimension. If Z is a stationary point and $V - B(Z)$ is positive semi-definite then we actually have found the CFDS solution, the Gower rank, and the global minimum (De Leeuw (2014)).

Penalizing Dimensions

In Shepard (1962a) and Shepard (1962b) a nonmetric multidimensional scaling technique is developed which minimizes a loss function over configurations in full dimensionality $n - 1$. In that sense the technique is similar to FDS. Shepard's iterative process aims to maintain

monotonicity between distances and dissimilarities and at the same time concentrate as much of the variation as possible in a small number of dimensions (De Leeuw (2017)).

Let us explore the idea of concentrating variation in $p < n - 1$ dimensions, but use an approach which is quite different from the one used by Shepard. We remain in the FDS framework, but we aim for solutions in $p < n - 1$ dimensions by penalizing $n - p$ dimensions of the full configuration, using the classical Courant quadratic penalty function.

Partition a full configuration $Z = \begin{bmatrix} X & | & Y \end{bmatrix}$, with X of dimension $n \times p$ and Y of dimension $n \times (n - p)$. Then

$$\sigma(Z) = 1 - \text{tr } X' B(Z) X - \text{tr } Y' B(Z) Y + \frac{1}{2} \text{tr } X' V X + \frac{1}{2} \text{tr } Y' V Y. \quad (10)$$

Also define the *penalty term*

$$\tau(Y) = \frac{1}{2} \text{tr } Y' V Y, \quad (11)$$

and *penalized stress*

$$\pi(Z, \lambda) = \sigma(Z) + \lambda \tau(Y). \quad (12)$$

Our proposed method is to minimize penalized stress over Z for a sequence of values $0 = \lambda_1 < \lambda_2 < \dots < \lambda_m$. For $\lambda = 0$ this is simply the FDS problem, for which we know we can compute the global minimum. For fixed $0 < \lambda < +\infty$ this is a Penalized FDS or PFDS problem. PFDS problems with increasing values of λ generate a *trajectory* $Z(\lambda)$ in configuration space.

The general theory of exterior penalty functions, which we review in appendix A of this paper, shows that increasing λ leads to an increasing sequence of stress values σ and a decreasing sequence of penalty terms τ . If $\lambda \rightarrow +\infty$ we approximate the global minimum of the FDS problem with Z of the form $Z = \begin{bmatrix} X & | & 0 \end{bmatrix}$, i.e. of the pMDS problem. This assumes we do actually compute the global minimum for each value of λ , which we hope we can do because we start at the FDS global minimum, and we slowly increase λ . There is also a local version of the exterior penalty result, which implies that $\lambda \rightarrow \infty$ takes us to a local minimum of pMDS, so there is always the possibility of taking the wrong trajectory to a local minimum of pMDS.

Local Minima

The stationary equations of the PFDS problem are solutions to the equations

$$(V - B(Z))X = 0, \quad (13)$$

$$((1 + \lambda)V - B(Z))Y = 0. \quad (14)$$

We can easily related stationary points and local minima of the FDS and PFDS problem.

Theorem 2: [PFDS Local Minima]

1: If X is a stationary point of the pMDS problem then $Z = \begin{bmatrix} X & | & 0 \end{bmatrix}$ is a stationary point of the PFDS problem, no matter what λ is.

2: If $Z = [X \mid 0]$ is a local minimum of the PFDS problem then X is a local minimum of pMDS and $(1 + \lambda)V - B(X) \gtrsim 0$, or $\lambda \geq \|V^+ B(X)\|_\infty - 1$, with $\|\bullet\|_\infty$ the spectral radius (largest eigenvalue).

Proof:

Part 1 follows by simple substitution in the stationary equations.

Part 2 follows from the expansion for $Z = [X + \epsilon P \mid \epsilon Q]$.

$$\begin{aligned} \pi(Z) = \pi(X) + \epsilon \operatorname{tr} P' \mathcal{D}\sigma(X) + \\ + \frac{1}{2} \epsilon^2 \mathcal{D}^2 \sigma(X)(P, P) + \frac{1}{2} \epsilon^2 \operatorname{tr} Q'((1 + \lambda)V - B(X))Q + o(\epsilon^2). \end{aligned} \quad (15)$$

At a local minimum we must have $\mathcal{D}\sigma(X) = 0$ and $\mathcal{D}^2 \sigma(X)(P, P) \gtrsim 0$, which are the necessary conditions for a local minimum of pMDS. We also must have $((1 + \lambda)V - B(X)) \gtrsim 0$. ■

Note that the conditions in part 2 of theorem 2 are also sufficient for PFDS to have a local minimum at $[X \mid 0]$, provided we eliminate translational and rotational indeterminacy by a suitable reparametrization, as in De Leeuw (1993).

Algorithm

The SMACOF algorithm for penalized stress is a small modification of the unpenalized FDS algorithm (8). We start our iterations for λ_j with the solution for λ_{j-1} (the starting solution for $\lambda_1 = 0$ can be completely arbitrary). The update rules for fixed λ are

$$X^{(k+1)} = V^+ B(Z^{(k)}) X^{(k)}, \quad (16)$$

$$Y^{(k+1)} = \frac{1}{1 + \lambda} V^+ B(Z^{(k)}) Y^{(k)}. \quad (17)$$

Thus we compute the FDS update $Z^{(k+1)} = V^+ B(Z^{(k)}) Z^{(k)}$ and then divide the last $n - p$ columns by $1 + \lambda$.

Code is in the appendix. Let us analyze a number of examples.

Examples

This section has a number of two-dimensional and a number of one-dimensional examples. The one-dimensional examples are of interest, because of the documented large number of local minima of stress in the one-dimensional case, and the fact that for small and medium n exact solutions are available (for example, De Leeuw (2005)). By default we use `seq(0, 1, length = 101)` for λ in most examples, but for some of them we dig a bit deeper and use longer sequences with smaller increments.

If for some value of λ the penalty term drops below the small cutoff γ , for example 10^{-10} , then there is not need to try larger values of λ , because they will just repeat the same result. We hope that result is the global minimum of the 2MDS problem.

The output for each example is a table in which we give, the minimum value of stress, the value of the penalty term at the minimum, the value of λ , and the number of iterations needed for convergence. Typically we print for the first three, the last three, and some regularly spaced intermediate values of λ . Remember that the stress values increase with increasing λ , and the penalty values decrease.

For two-dimensional examples we plot all two-dimensional configurations, after rotating to optimum match (using the function `matchMe()` from the appendix). We connect corresponding points for different values of λ . Points corresponding to the highest value of λ are labeled and have a different plot symbol. For one-dimensional examples we put `1:n` on the horizontal axes and plot the single dimension on the vertical axis, again connecting corresponding points. We label the points corresponding with the highest value of λ , and draw horizontal lines through them to more clearly show their order on the dimension.

The appendix also has code for the function `checkUni()`, which we have used to check the solutions in the one dimensional case are indeed local minima. The function checks the necessary condition for a local minimum $x = V^+u$, with

$$u_i = \sum_{j=1}^n w_{ij} \delta_{ij} \text{sign}(x_i - x_j).$$

It should be emphasized that all examples are just meant to study performance and convergence of penalized FDS. There is no interpretation of the MDS results

Chi Squares

In this example, of order 10, the δ_{ij} are square roots of independent draws from a central chi-square distribution with two degrees of freedom. We use a fixed seed for the random number generator.

If we analyze the data with `smacof` in two dimensions, using the Torgerson initial configuration, we find a stress of 0.0862287021 after 65 iterations. All `smacof` runs in this paper have a maximum number of iterations of 10,000 and stop if the difference between successive stress values is less than 1e-10. A full-dimensional scaling with $p = 9$ gives stress 0.0730261617 after 198 iterations. The singular values of the full dimensional solution are

```
## [1] +0.322688 +0.207891 +0.163634 +0.096146 +0.045038 +0.000001 +0.000000
## [8] +0.000000 +0.000000
```

and thus the Gower rank of these data is five.

We can also start our `smacof` iterations with the first two dimensions of the full dimensional solution. Stress is 0.0862287021 after 62 iterations, the same solution as with the Torgerson start.

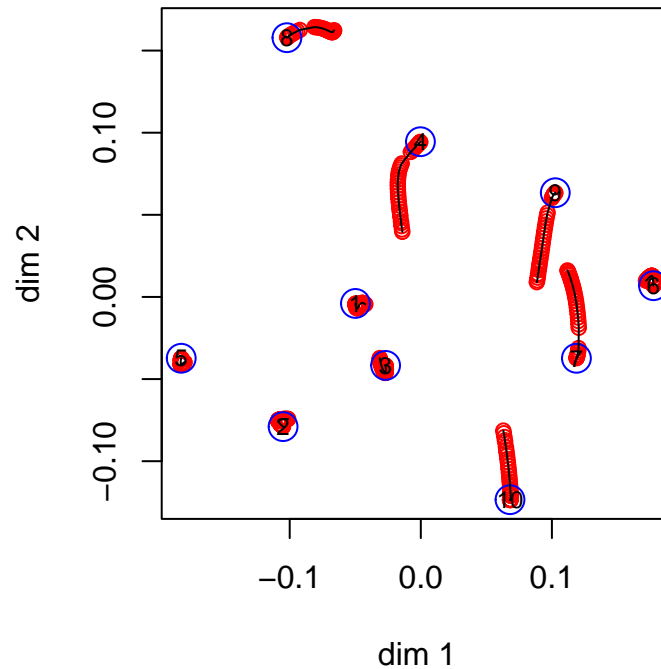
If we run smacof 1000 times with a random start we find five different local minima, and the one with the smallest stress is found in about 40% of the cases. Again, this is the Torgerson solution.

```
##
## 0.0862287 0.0955797 0.0990518 0.0995697 0.100125
##      394      246      174      69      117
```

We now apply our global optimization method, using 101 values of λ , equally spaced between zero and one. Again we converge on the Torgerson solution, which by all indications is the global minimum of stress for these data.

```
## itel 203 lambda 0.000000 stress 0.073026 penalty 0.422435
## itel 4 lambda 0.010000 stress 0.073054 penalty 0.091005
## itel 3 lambda 0.020000 stress 0.073141 penalty 0.086529
## itel 2 lambda 0.030000 stress 0.073269 penalty 0.082545
## itel 1 lambda 0.040000 stress 0.073390 penalty 0.079778
## itel 2 lambda 0.050000 stress 0.073688 penalty 0.074219
## itel 1 lambda 0.060000 stress 0.073899 penalty 0.071053
## itel 1 lambda 0.070000 stress 0.074170 penalty 0.067522
## itel 1 lambda 0.080000 stress 0.074497 penalty 0.063768
## itel 1 lambda 0.090000 stress 0.074874 penalty 0.059891
## itel 1 lambda 0.100000 stress 0.075299 penalty 0.055962
## itel 1 lambda 0.110000 stress 0.075765 penalty 0.052032
## itel 1 lambda 0.120000 stress 0.076269 penalty 0.048144
## itel 1 lambda 0.130000 stress 0.076806 penalty 0.044331
## itel 1 lambda 0.140000 stress 0.077368 penalty 0.040621
## itel 1 lambda 0.150000 stress 0.077951 penalty 0.037038
## itel 1 lambda 0.160000 stress 0.078548 penalty 0.033601
## itel 1 lambda 0.170000 stress 0.079152 penalty 0.030322
## itel 1 lambda 0.180000 stress 0.079756 penalty 0.027213
## itel 1 lambda 0.190000 stress 0.080355 penalty 0.024279
## itel 1 lambda 0.200000 stress 0.080942 penalty 0.021526
## itel 1 lambda 0.210000 stress 0.081511 penalty 0.018955
## itel 1 lambda 0.220000 stress 0.082058 penalty 0.016567
## itel 1 lambda 0.230000 stress 0.082578 penalty 0.014364
## itel 1 lambda 0.240000 stress 0.083067 penalty 0.012345
## itel 9 lambda 0.250000 stress 0.085317 penalty 0.003278
## itel 2 lambda 0.260000 stress 0.085541 penalty 0.002404
## itel 2 lambda 0.270000 stress 0.085724 penalty 0.001723
## itel 3 lambda 0.280000 stress 0.085924 penalty 0.001009
## itel 2 lambda 0.290000 stress 0.086017 penalty 0.000689
## itel 3 lambda 0.300000 stress 0.086111 penalty 0.000374
## itel 4 lambda 0.310000 stress 0.086177 penalty 0.000158
## itel 3 lambda 0.320000 stress 0.086202 penalty 0.000080
## itel 4 lambda 0.330000 stress 0.086218 penalty 0.000031
```

```
## itel    6 lambda    0.340000 stress 0.086226 penalty 0.000007
## itel    8 lambda    0.350000 stress 0.086228 penalty 0.000001
```



Regular Simplex

The regular simplex has all dissimilarities equal to one. We use an example with $n = 10$, for which the global minimum (as far as we know) of pMDS with $p = 2$ is a configuration with nine points equally spaced on a circle and one point in the center.

Using the Torgerson initial configuration in two dimensions, smacof finds a stress of 0.1110521776 after 872 iterations. This corresponds with the presumed global minimum.

A full-dimensional scaling with $p = 9$ gives stress $1.8681520461 \times 10^{-31}$ after 1 iterations. The singular values of the full dimensional solution are

```
## [1] +0.149071 +0.149071 +0.149071 +0.149071 +0.149071 +0.149071 +0.149071
## [8] +0.149071 +0.149071
```

and thus the Gower rank of these data is six.

If we our smacof iterations with the first two dimensions of the full dimensional solution stress is 0.1110521776 after 872 iterations, the same global solution as with the Torgerson start.

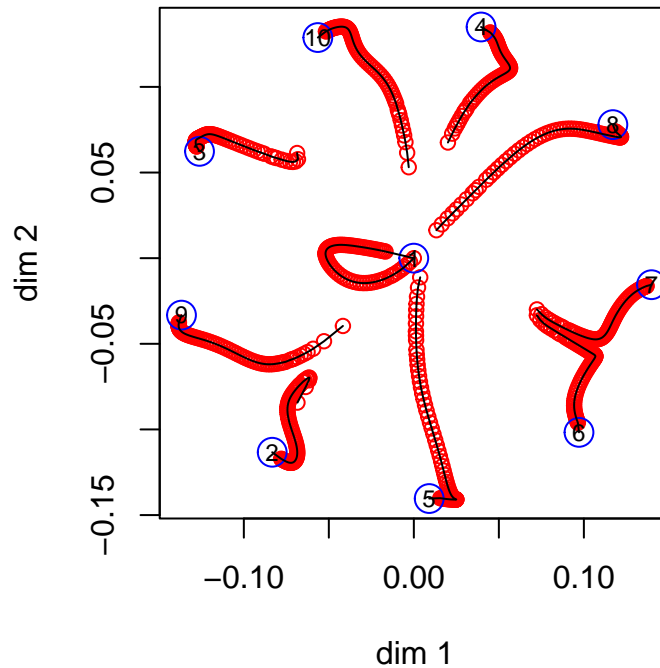
If we run smacof 1000 times with a random start we find three different local minima, and the one with the smallest stress is found in about 70% of the cases. Again, this is the global solution. There are, of course, many more local minima, because each permutation of the 10 solution points will give another local minimum with the same stress value.

```
##
```

```
## 0.10988 0.111052 0.111058
##      715      284      1
```

Our global optimization method with the default λ sequence also converges on the global solution.

```
## itel    1 lambda    0.000000 stress 0.000000 penalty 0.400000
## itel    7 lambda    0.010000 stress 0.000103 penalty 0.375240
## itel    5 lambda    0.020000 stress 0.000427 penalty 0.360212
## itel    3 lambda    0.030000 stress 0.000916 penalty 0.346937
## itel    2 lambda    0.040000 stress 0.001533 penalty 0.334923
## itel    2 lambda    0.050000 stress 0.002378 penalty 0.321650
## itel    2 lambda    0.060000 stress 0.003468 penalty 0.307513
## itel    2 lambda    0.070000 stress 0.004816 penalty 0.292842
## itel    1 lambda    0.080000 stress 0.005905 penalty 0.283148
## itel    1 lambda    0.090000 stress 0.007178 penalty 0.272814
## itel    2 lambda    0.100000 stress 0.009438 penalty 0.255499
## itel    1 lambda    0.110000 stress 0.011002 penalty 0.245389
## itel    1 lambda    0.120000 stress 0.012752 penalty 0.234937
## itel    1 lambda    0.130000 stress 0.014677 penalty 0.224281
## itel    1 lambda    0.140000 stress 0.016766 penalty 0.213553
## itel    1 lambda    0.150000 stress 0.019004 penalty 0.202867
## itel    1 lambda    0.160000 stress 0.021375 penalty 0.192317
## itel    1 lambda    0.170000 stress 0.023863 penalty 0.181981
## itel    1 lambda    0.180000 stress 0.026446 penalty 0.171921
## itel    1 lambda    0.190000 stress 0.029107 penalty 0.162184
## itel    1 lambda    0.200000 stress 0.031826 penalty 0.152804
## itel    1 lambda    0.210000 stress 0.034585 penalty 0.143802
## itel    1 lambda    0.220000 stress 0.037368 penalty 0.135189
## itel    1 lambda    0.230000 stress 0.040161 penalty 0.126967
## itel    1 lambda    0.240000 stress 0.042951 penalty 0.119133
## itel    1 lambda    0.250000 stress 0.045728 penalty 0.111680
## itel    1 lambda    0.260000 stress 0.048483 penalty 0.104597
## itel    1 lambda    0.270000 stress 0.051208 penalty 0.097873
## itel    1 lambda    0.280000 stress 0.053896 penalty 0.091496
## itel    1 lambda    0.290000 stress 0.056540 penalty 0.085455
## itel    1 lambda    0.300000 stress 0.059136 penalty 0.079739
## itel    1 lambda    0.310000 stress 0.061677 penalty 0.074338
## itel    1 lambda    0.320000 stress 0.064160 penalty 0.069241
## itel    1 lambda    0.330000 stress 0.066580 penalty 0.064438
## itel    1 lambda    0.340000 stress 0.068935 penalty 0.059918
## itel    1 lambda    0.350000 stress 0.071222 penalty 0.055668
## itel    1 lambda    0.700000 stress 0.109679 penalty 0.000427
## itel    1 lambda    0.710000 stress 0.109756 penalty 0.000320
## itel   92 lambda    0.720000 stress 0.109880 penalty 0.000000
```



Intelligence

These are correlations between eight intelligence tests, taken from the `smacof` package. We convert to dissimilarities by taking the negative logarithm of the correlations.

Using the Torgerson initial configuration in two dimensions, `smacof` finds a stress of 0.0075160651 after 28 iterations. This corresponds with the presumed global minimum.

A full-dimensional scaling with $p = 7$ gives stress 0.0049386956 after 1554 iterations. The singular values of the full dimensional solution are

```
## [1] +0.382699 +0.291593 +0.127580 +0.031521 +0.003190 +0.000000 +0.000000
```

and thus the Gower rank of these data is five.

If we our `smacof` iterations with the first two dimensions of the full dimensional solution stress is 0.0075160651 after 26 iterations, the same solution as with the Torgerson start.

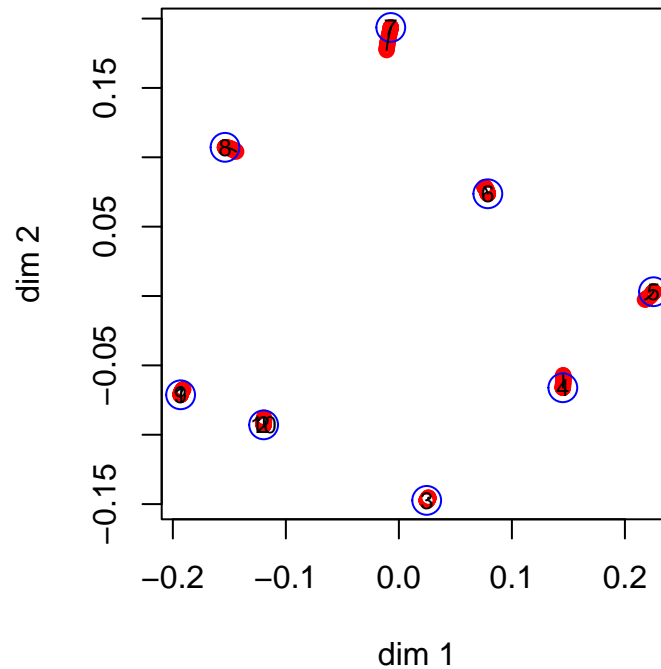
If we run `smacof` 1000 times with a random start we find two different local minima, and the one with the smallest stress is found in about 90% of the cases. Again, this is the same solution.

```
##
## 0.163122 1.24546
##      896      104
```

Our global optimization method with the default λ sequence also converges to the same solution.

```
## itel 1553 lambda    0.000000 stress 0.004939 penalty 0.368081
## itel    7 lambda    0.010000 stress 0.004956 penalty 0.031597
```

Iteration	lambda	stress	penalty
## itel 4	lambda 0.020000	stress 0.005001	penalty 0.028964
## itel 3	lambda 0.030000	stress 0.005070	penalty 0.026414
## itel 3	lambda 0.040000	stress 0.005181	penalty 0.023535
## itel 2	lambda 0.050000	stress 0.005286	penalty 0.021411
## itel 2	lambda 0.060000	stress 0.005420	penalty 0.019125
## itel 2	lambda 0.070000	stress 0.005580	penalty 0.016798
## itel 2	lambda 0.080000	stress 0.005760	penalty 0.014516
## itel 2	lambda 0.090000	stress 0.005954	penalty 0.012341
## itel 2	lambda 0.100000	stress 0.006155	penalty 0.010317
## itel 2	lambda 0.110000	stress 0.006356	penalty 0.008472
## itel 3	lambda 0.120000	stress 0.006631	penalty 0.006162
## itel 2	lambda 0.130000	stress 0.006795	penalty 0.004892
## itel 4	lambda 0.140000	stress 0.007060	penalty 0.002972
## itel 4	lambda 0.150000	stress 0.007241	penalty 0.001739
## itel 9	lambda 0.160000	stress 0.007438	penalty 0.000476
## itel 52	lambda 0.170000	stress 0.007516	penalty 0.000000
## itel 8	lambda 0.180000	stress 0.007516	penalty 0.000000
## itel 4	lambda 0.190000	stress 0.007516	penalty 0.000000
## itel 2	lambda 0.200000	stress 0.007516	penalty 0.000000
## itel 2	lambda 0.210000	stress 0.007516	penalty 0.000000



As in the chi-square example, the FDS and the 2MDS solution are very similar and the PMDS trajectories are short.

Countries

This is the `wish` dataset from the 'smacof' package, with similarities between 12 countries. They are converted to dissimilarities by subtracting each of them from seven.

```
data(wish, package = "smacof")
countries <- as.matrix(wish)
w <- matrix(1, 12, 12) - diag(12)
countries <- 7 * w - countries
countries <- 2 * countries / sqrt(sum(w * countries * countries))
```

Using the Torgerson initial configuration in two dimensions, smacof finds a stress of 0.0477490806 after 103 iterations.

A full-dimensional scaling with $p = 11$ gives stress 0.0159699675 after 763 iterations. The singular values of the full dimensional solution are

```
## [1] +0.256524 +0.226618 +0.163113 +0.109658 +0.081072 +0.040506 +0.000595
## [8] +0.000000 +0.000000 +0.000000 +0.000000
```

and thus the Gower rank of these data is seven.

If we start our smacof iterations with the first two dimensions of the full dimensional solution then stress is 0.0477490807 after 95 iterations, and we find the same solution as with the Torgerson start.

If we run smacof 1000 times with a random start we find sixteen different local minima, and the one with the smallest stress is found in about 35% of the cases. Note, however, that the Torgerson and full-dimensional solutions correspond with the second smallest local minimum, which is consequently certainly not the global minimum.

```
##
## 0.0474139 0.0477491 0.0494938 0.0513037 0.0669776 0.0671137 0.0691977 0.0698188
##      357      185      138      104      55      60      35      29
## 0.0898758 0.0909081 0.0913887 0.0914608 0.0917437 0.100849 0.102294 0.107119
##      4      7      4      5      3      6      7      1
```

Our global optimization method with the default λ sequence does converge to the solution with minimum stress 0.0474139053, which is our tentative global minimum.

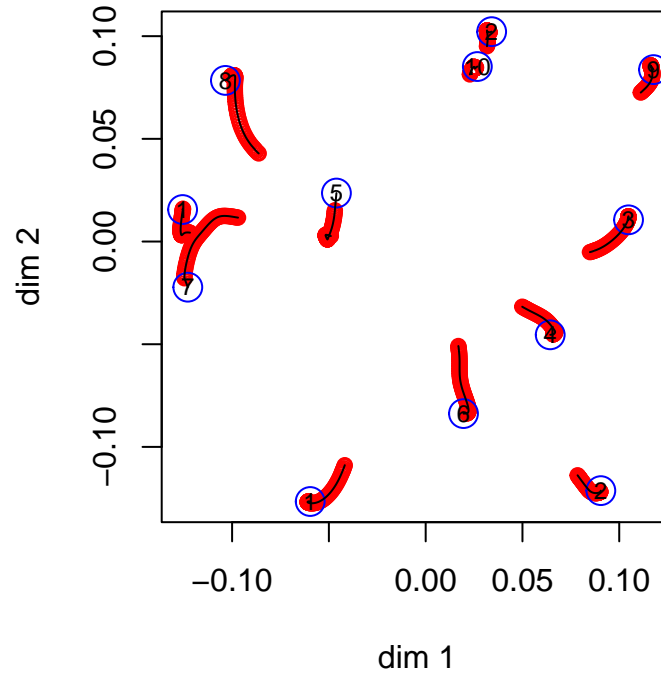
```
## itel 802 lambda 0.000000 stress 0.015970 penalty 0.367070
## itel 4 lambda 0.010000 stress 0.016010 penalty 0.134507
## itel 3 lambda 0.020000 stress 0.016140 penalty 0.128004
## itel 2 lambda 0.030000 stress 0.016331 penalty 0.122297
## itel 1 lambda 0.040000 stress 0.016515 penalty 0.118330
## itel 1 lambda 0.050000 stress 0.016780 penalty 0.113631
## itel 1 lambda 0.060000 stress 0.017127 penalty 0.108490
## itel 1 lambda 0.070000 stress 0.017557 penalty 0.103107
## itel 1 lambda 0.080000 stress 0.018068 penalty 0.097622
## itel 1 lambda 0.090000 stress 0.018654 penalty 0.092138
```

## itel	1	lambda	0.100000	stress	0.019310	penalty	0.086729
## itel	1	lambda	0.110000	stress	0.020028	penalty	0.081447
## itel	1	lambda	0.120000	stress	0.020801	penalty	0.076329
## itel	1	lambda	0.130000	stress	0.021620	penalty	0.071403
## itel	1	lambda	0.140000	stress	0.022479	penalty	0.066685
## itel	1	lambda	0.150000	stress	0.023369	penalty	0.062184
## itel	1	lambda	0.160000	stress	0.024283	penalty	0.057904
## itel	1	lambda	0.170000	stress	0.025215	penalty	0.053847
## itel	1	lambda	0.180000	stress	0.026159	penalty	0.050009
## itel	1	lambda	0.190000	stress	0.027110	penalty	0.046385
## itel	1	lambda	0.200000	stress	0.028062	penalty	0.042967
## itel	1	lambda	0.210000	stress	0.029011	penalty	0.039749
## itel	1	lambda	0.220000	stress	0.029953	penalty	0.036722
## itel	1	lambda	0.230000	stress	0.030885	penalty	0.033877
## itel	1	lambda	0.240000	stress	0.031802	penalty	0.031208
## itel	1	lambda	0.250000	stress	0.032702	penalty	0.028706
## itel	1	lambda	0.260000	stress	0.033582	penalty	0.026364
## itel	1	lambda	0.270000	stress	0.034440	penalty	0.024176
## itel	1	lambda	0.280000	stress	0.035273	penalty	0.022136
## itel	1	lambda	0.290000	stress	0.036079	penalty	0.020237
## itel	1	lambda	0.300000	stress	0.036857	penalty	0.018474
## itel	1	lambda	0.310000	stress	0.037604	penalty	0.016841
## itel	1	lambda	0.320000	stress	0.038319	penalty	0.015332
## itel	1	lambda	0.330000	stress	0.039003	penalty	0.013940
## itel	1	lambda	0.340000	stress	0.039654	penalty	0.012659
## itel	1	lambda	0.350000	stress	0.040272	penalty	0.011482
## itel	1	lambda	0.360000	stress	0.040858	penalty	0.010403
## itel	1	lambda	0.370000	stress	0.041413	penalty	0.009413
## itel	1	lambda	0.380000	stress	0.041936	penalty	0.008508
## itel	1	lambda	0.390000	stress	0.042428	penalty	0.007679
## itel	1	lambda	0.400000	stress	0.042892	penalty	0.006922
## itel	1	lambda	0.410000	stress	0.043326	penalty	0.006229
## itel	1	lambda	0.420000	stress	0.043734	penalty	0.005595
## itel	1	lambda	0.430000	stress	0.044116	penalty	0.005015
## itel	1	lambda	0.440000	stress	0.044472	penalty	0.004483
## itel	1	lambda	0.450000	stress	0.044803	penalty	0.003995
## itel	1	lambda	0.460000	stress	0.045112	penalty	0.003547
## itel	1	lambda	0.470000	stress	0.045397	penalty	0.003133
## itel	1	lambda	0.480000	stress	0.045660	penalty	0.002751
## itel	1	lambda	0.490000	stress	0.045902	penalty	0.002398
## itel	1	lambda	0.500000	stress	0.046124	penalty	0.002071
## itel	1	lambda	0.510000	stress	0.046325	penalty	0.001769
## itel	1	lambda	0.520000	stress	0.046506	penalty	0.001492
## itel	1	lambda	0.530000	stress	0.046668	penalty	0.001240
## itel	1	lambda	0.540000	stress	0.046811	penalty	0.001016

```

## itel      1 lambda    0.550000 stress 0.046934 penalty 0.000819
## itel      1 lambda    0.560000 stress 0.047040 penalty 0.000651
## itel      1 lambda    0.570000 stress 0.047128 penalty 0.000509
## itel      1 lambda    0.580000 stress 0.047200 penalty 0.000392
## itel      1 lambda    0.590000 stress 0.047258 penalty 0.000298
## itel      1 lambda    0.600000 stress 0.047303 penalty 0.000224
## itel     68 lambda    0.610000 stress 0.047414 penalty 0.000000

```



Dutch Political Parties

In 1967 one hundred psychology students at Leiden University judged the similarity of nine Dutch political parties, using the complete method of triads (De Grujter (1967)). Data were aggregated and converted to dissimilarities. We first print the matrix of dissimilarities.

```

##      KVP      PvdA      VVD      ARP      CHU      CPN      PSP      BP      D66
## KVP  +0.000 +0.209 +0.196 +0.171 +0.179 +0.281 +0.250 +0.267 +0.230
## PvdA +0.209 +0.000 +0.250 +0.210 +0.231 +0.190 +0.171 +0.269 +0.204
## VVD  +0.196 +0.250 +0.000 +0.203 +0.185 +0.302 +0.281 +0.257 +0.174
## ARP  +0.171 +0.210 +0.203 +0.000 +0.119 +0.292 +0.250 +0.271 +0.228
## CHU  +0.179 +0.231 +0.185 +0.119 +0.000 +0.290 +0.263 +0.259 +0.225
## CPN  +0.281 +0.190 +0.302 +0.292 +0.290 +0.000 +0.152 +0.236 +0.276
## PSP  +0.250 +0.171 +0.281 +0.250 +0.263 +0.152 +0.000 +0.256 +0.237
## BP   +0.267 +0.269 +0.257 +0.271 +0.259 +0.236 +0.256 +0.000 +0.274
## D66  +0.230 +0.204 +0.174 +0.228 +0.225 +0.276 +0.237 +0.274 +0.000

##      KVP      PvdA      VVD      ARP      CHU      CPN      PSP      BP      D66
## KVP  +0.000 +0.128 +0.099 +0.054 +0.066 +0.348 +0.242 +0.299 +0.179
## PvdA +0.128 +0.000 +0.241 +0.129 +0.185 +0.088 +0.054 +0.304 +0.115

```



```
## VVD +0.099 +0.241 +0.000 +0.114 +0.077 +0.438 +0.350 +0.263 +0.058
## ARP +0.054 +0.129 +0.114 +0.000 +0.004 +0.393 +0.242 +0.312 +0.175
## CHU +0.066 +0.185 +0.077 +0.004 +0.000 +0.387 +0.286 +0.270 +0.166
## CPN +0.348 +0.088 +0.438 +0.393 +0.387 +0.000 +0.029 +0.197 +0.331
## PSP +0.242 +0.054 +0.350 +0.242 +0.286 +0.029 +0.000 +0.260 +0.200
## BP +0.299 +0.304 +0.263 +0.312 +0.270 +0.197 +0.260 +0.000 +0.323
## D66 +0.179 +0.115 +0.058 +0.175 +0.166 +0.331 +0.200 +0.323 +0.000
```

This is mainly because these data, being averages over a heterogeneous group of individuals, regress to the mean and thus have a substantial additive constant.

Using the Torgerson initial configuration in two dimensions, smacof finds a stress of 0.0272187083 after 97 iterations.

A full-dimensional scaling with $p = 8$ gives stress 0.0218562551 after 68 iterations. The singular values of the full dimensional solution are

```
## [1] +0.389414 +0.216045 +0.132915 +0.037145 +0.000000 +0.000000 +0.000000
## [8] +0.000000
```

and thus the Gower rank of these (transformed) data is four.

If we start our smacof iterations with the first two dimensions of the full dimensional solution stress is 0.0272187083 after 78 iterations, the same solution as with the Torgerson start.

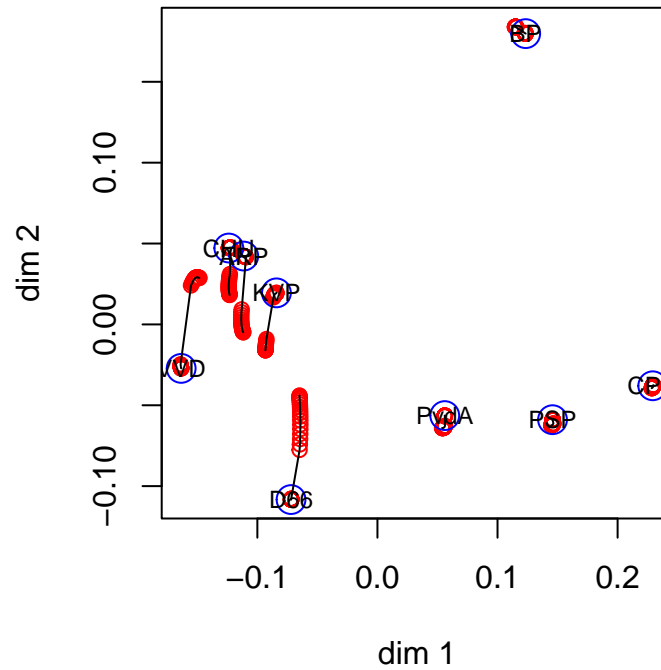
If we run smacof 1000 times with a random start we find two different local minima, and the one with the smallest stress is found in about 55% of the cases.

```
##
## 0.0272187 0.0301047
##      566      434
```

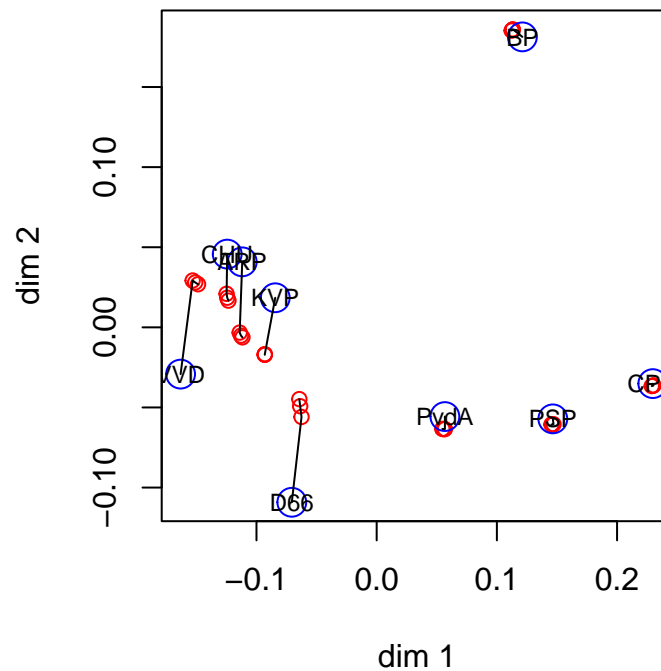
Our global optimization method with the default λ sequence also converges to the same smallest local minimum.

```
## itel 80 lambda 0.000000 stress 0.021856 penalty 0.453217
## itel 2 lambda 0.010000 stress 0.021865 penalty 0.041599
## itel 2 lambda 0.020000 stress 0.021899 penalty 0.040014
## itel 1 lambda 0.030000 stress 0.021938 penalty 0.038919
## itel 1 lambda 0.040000 stress 0.022002 penalty 0.037599
## itel 1 lambda 0.050000 stress 0.022091 penalty 0.036178
## itel 1 lambda 0.060000 stress 0.022205 penalty 0.034717
## itel 1 lambda 0.070000 stress 0.022342 penalty 0.033247
## itel 1 lambda 0.080000 stress 0.022499 penalty 0.031782
## itel 1 lambda 0.090000 stress 0.022675 penalty 0.030330
## itel 1 lambda 0.100000 stress 0.022868 penalty 0.028890
## itel 1 lambda 0.110000 stress 0.023077 penalty 0.027462
## itel 1 lambda 0.120000 stress 0.023299 penalty 0.026041
## itel 1 lambda 0.130000 stress 0.023532 penalty 0.024623
## itel 1 lambda 0.140000 stress 0.023776 penalty 0.023204
```

## itel	1	lambda	0.150000	stress	0.024028	penalty	0.021779
## itel	1	lambda	0.160000	stress	0.024286	penalty	0.020344
## itel	1	lambda	0.170000	stress	0.024548	penalty	0.018899
## itel	1	lambda	0.180000	stress	0.024811	penalty	0.017442
## itel	31	lambda	0.190000	stress	0.026907	penalty	0.001050
## itel	1	lambda	0.200000	stress	0.026926	penalty	0.000956
## itel	1	lambda	0.210000	stress	0.026948	penalty	0.000864
## itel	1	lambda	0.220000	stress	0.026971	penalty	0.000773
## itel	1	lambda	0.230000	stress	0.026995	penalty	0.000685
## itel	1	lambda	0.240000	stress	0.027018	penalty	0.000600
## itel	1	lambda	0.250000	stress	0.027042	penalty	0.000521
## itel	1	lambda	0.260000	stress	0.027064	penalty	0.000448
## itel	1	lambda	0.270000	stress	0.027084	penalty	0.000382
## itel	1	lambda	0.280000	stress	0.027104	penalty	0.000322
## itel	1	lambda	0.290000	stress	0.027121	penalty	0.000269
## itel	1	lambda	0.300000	stress	0.027137	penalty	0.000222
## itel	1	lambda	0.310000	stress	0.027152	penalty	0.000181
## itel	1	lambda	0.320000	stress	0.027164	penalty	0.000147
## itel	1	lambda	0.330000	stress	0.027175	penalty	0.000117
## itel	1	lambda	0.340000	stress	0.027184	penalty	0.000092
## itel	1	lambda	0.350000	stress	0.027191	penalty	0.000072
## itel	1	lambda	0.360000	stress	0.027197	penalty	0.000055
## itel	1	lambda	0.370000	stress	0.027202	penalty	0.000042
## itel	1	lambda	0.380000	stress	0.027206	penalty	0.000032
## itel	6	lambda	0.390000	stress	0.027217	penalty	0.000005
## itel	18	lambda	0.400000	stress	0.027219	penalty	0.000000
## itel	2	lambda	0.410000	stress	0.027219	penalty	0.000000
## itel	2	lambda	0.420000	stress	0.027219	penalty	0.000000



```
## itel 80 lambda 0.000000 stress 0.021856 penalty 0.453217
## itel 3 lambda 0.111111 stress 0.022908 penalty 0.029330
## itel 2 lambda 0.222222 stress 0.024912 penalty 0.020271
## itel 71 lambda 0.333333 stress 0.027219 penalty 0.000000
```



Ekman

The next example analyzes dissimilarities between 14 colors, taken from Ekman (1954). The original similarities s_{ij} , averaged over 31 subjects, were transformed to dissimilarities by

$$\delta_{ij} = 1 - s_{ij}.$$

Using the Torgerson initial configuration in two dimensions, smacof finds a stress of 0.0172132469 after 25 iterations.

A full-dimensional scaling with $p = 8$ gives stress $8.7569296565 \times 10^{-5}$ after 1204 iterations. The singular values of the full dimensional solution are

```
## [1] +0.254220 +0.205722 +0.119338 +0.109900 +0.068741 +0.055643 +0.033137
## [8] +0.022797 +0.011573 +0.001704 +0.000315 +0.000000 +0.000000
```

and thus the Gower rank of these (transformed) data is eleven.

If we start our smacof iterations with the first two dimensions of the full dimensional solution stress is 0.0172132469 after 25 iterations, the same solution as with the Torgerson start.

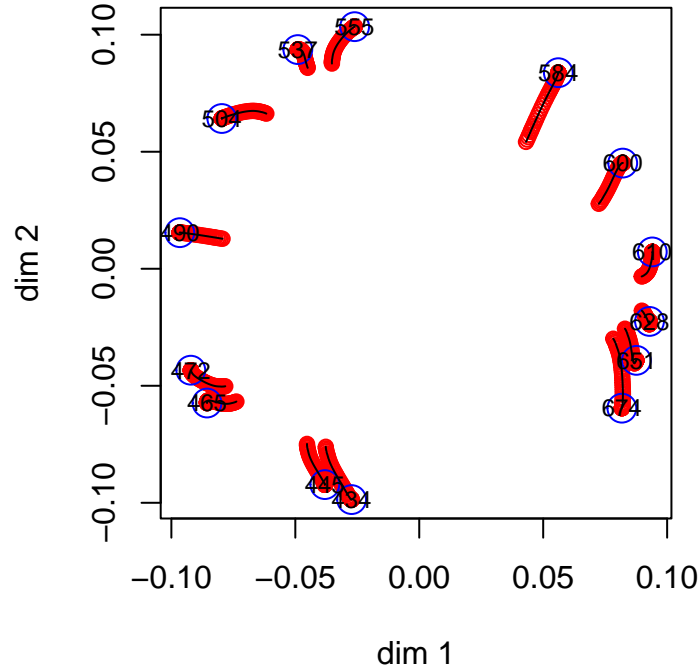
If we run smacof 1000 times with a random start we find ten different local minima, and the one with the smallest stress is found in about 80% of the cases.

```
##
## 0.0172132 0.0365429 0.0601969 0.0619843 0.063479 0.0662027 0.0674562 0.0678714
##      834      133      15      6      3      3      2      2
## 0.0758563 0.0759964
##      1      1
```

Our global optimization method with the default λ sequence also converges to the same smallest local minimum.

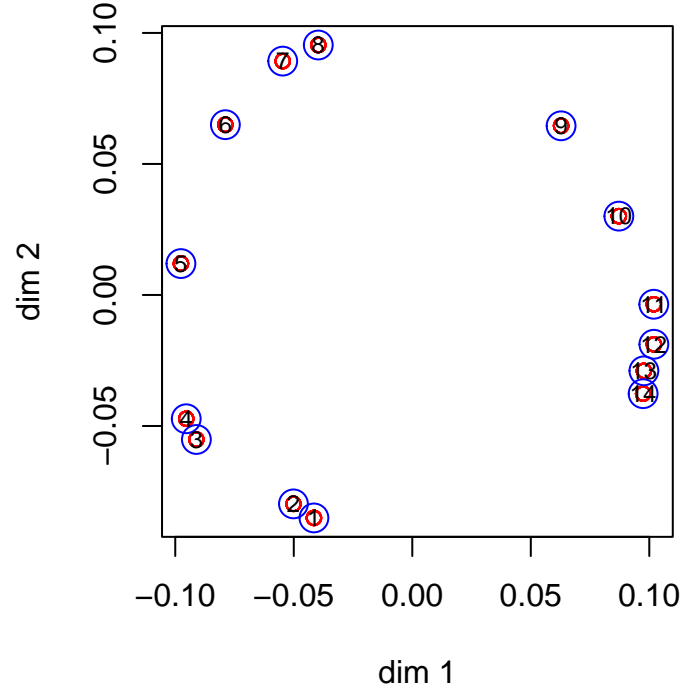
```
## itel 1482 lambda 0.000000 stress 0.000088 penalty 0.426110
## itel 5 lambda 0.010000 stress 0.000132 penalty 0.118988
## itel 3 lambda 0.020000 stress 0.000253 penalty 0.112777
## itel 2 lambda 0.030000 stress 0.000431 penalty 0.107184
## itel 2 lambda 0.040000 stress 0.000715 penalty 0.100691
## itel 1 lambda 0.050000 stress 0.000942 penalty 0.096656
## itel 1 lambda 0.060000 stress 0.001246 penalty 0.091999
## itel 1 lambda 0.070000 stress 0.001627 penalty 0.086941
## itel 1 lambda 0.080000 stress 0.002082 penalty 0.081645
## itel 1 lambda 0.090000 stress 0.002607 penalty 0.076232
## itel 1 lambda 0.100000 stress 0.003195 penalty 0.070791
## itel 1 lambda 0.110000 stress 0.003840 penalty 0.065388
## itel 1 lambda 0.120000 stress 0.004534 penalty 0.060073
## itel 1 lambda 0.130000 stress 0.005267 penalty 0.054886
## itel 1 lambda 0.140000 stress 0.006033 penalty 0.049859
## itel 1 lambda 0.150000 stress 0.006820 penalty 0.045021
## itel 1 lambda 0.160000 stress 0.007622 penalty 0.040395
## itel 1 lambda 0.170000 stress 0.008427 penalty 0.036003
## itel 1 lambda 0.180000 stress 0.009228 penalty 0.031864
## itel 1 lambda 0.190000 stress 0.010014 penalty 0.027995
## itel 1 lambda 0.200000 stress 0.010778 penalty 0.024407
```

## itel	1	lambda	0.210000	stress	0.011510	penalty	0.021111
## itel	1	lambda	0.220000	stress	0.012203	penalty	0.018110
## itel	1	lambda	0.230000	stress	0.012852	penalty	0.015406
## itel	1	lambda	0.240000	stress	0.013451	penalty	0.012993
## itel	1	lambda	0.250000	stress	0.013997	penalty	0.010863
## itel	1	lambda	0.260000	stress	0.014489	penalty	0.009003
## itel	1	lambda	0.270000	stress	0.014926	penalty	0.007396
## itel	2	lambda	0.280000	stress	0.015618	penalty	0.004939
## itel	1	lambda	0.290000	stress	0.015890	penalty	0.004012
## itel	1	lambda	0.300000	stress	0.016125	penalty	0.003230
## itel	2	lambda	0.310000	stress	0.016484	penalty	0.002072
## itel	1	lambda	0.320000	stress	0.016620	penalty	0.001651
## itel	1	lambda	0.330000	stress	0.016734	penalty	0.001305
## itel	2	lambda	0.340000	stress	0.016904	penalty	0.000807
## itel	1	lambda	0.350000	stress	0.016966	penalty	0.000632
## itel	1	lambda	0.360000	stress	0.017017	penalty	0.000491
## itel	1	lambda	0.370000	stress	0.017059	penalty	0.000378
## itel	1	lambda	0.380000	stress	0.017093	penalty	0.000289
## itel	1	lambda	0.390000	stress	0.017121	penalty	0.000219
## itel	1	lambda	0.400000	stress	0.017142	penalty	0.000165
## itel	2	lambda	0.410000	stress	0.017172	penalty	0.000092
## itel	1	lambda	0.420000	stress	0.017183	penalty	0.000068
## itel	1	lambda	0.430000	stress	0.017190	penalty	0.000050
## itel	1	lambda	0.440000	stress	0.017196	penalty	0.000037
## itel	2	lambda	0.450000	stress	0.017204	penalty	0.000019
## itel	2	lambda	0.460000	stress	0.017209	penalty	0.000010
## itel	2	lambda	0.470000	stress	0.017211	penalty	0.000005
## itel	3	lambda	0.480000	stress	0.017212	penalty	0.000002
## itel	4	lambda	0.490000	stress	0.017213	penalty	0.000000
## itel	4	lambda	0.500000	stress	0.017213	penalty	0.000000
## itel	4	lambda	0.510000	stress	0.017213	penalty	0.000000
## itel	2	lambda	0.520000	stress	0.017213	penalty	0.000000



If we transform the Ekman similarities by $\delta_{ij} = (1 - s_{ij})^3$ then it is known (De Leeuw (2016)) that the Gower rank of the transformed data is equal to two. Thus the FDS solution has rank 2, and the 2MDS solution always the global minimum.

```
## itel 99 lambda 0.000000 stress 0.011025 penalty 0.433456
## itel 1 lambda 0.010000 stress 0.011025 penalty 0.000000
## itel 1 lambda 0.020000 stress 0.011025 penalty 0.000000
## itel 1 lambda 0.030000 stress 0.011025 penalty 0.000000
## itel 1 lambda 0.040000 stress 0.011025 penalty 0.000000
## itel 1 lambda 0.050000 stress 0.011025 penalty 0.000000
## itel 1 lambda 0.060000 stress 0.011025 penalty 0.000000
## itel 1 lambda 0.070000 stress 0.011025 penalty 0.000000
## itel 1 lambda 0.080000 stress 0.011025 penalty 0.000000
## itel 1 lambda 0.090000 stress 0.011025 penalty 0.000000
## itel 1 lambda 0.100000 stress 0.011025 penalty 0.000000
## itel 1 lambda 0.110000 stress 0.011025 penalty 0.000000
## itel 1 lambda 0.120000 stress 0.011025 penalty 0.000000
## itel 1 lambda 0.130000 stress 0.011025 penalty 0.000000
```



Morse in Two

Next, we use dissimilarities between 36 Morse code signals (Rothkopf (1957)). We used the symmetrized version `morse` from the `smacof` package (De Leeuw and Mair (2009)).

Using the Torgerson initial configuration in two dimensions, `smacof` finds a stress of 0.0899492031 after 238 iterations.

A full-dimensional scaling with $p = 31$ gives stress $7.6345177128 \times 10^{-4}$ after 1347 iterations. The singular values of the full dimensional solution are

```
## [1] +0.100442 +0.090156 +0.078222 +0.067738 +0.061768 +0.060456 +0.055417
## [8] +0.049762 +0.047491 +0.045725 +0.044117 +0.039553 +0.036242 +0.033715
## [15] +0.032183 +0.025291 +0.022152 +0.019617 +0.018309 +0.013419 +0.012242
## [22] +0.006957 +0.000575 +0.000071 +0.000014 +0.000000 +0.000000 +0.000000
## [29] +0.000000 +0.000000 +0.000000 +0.000000 +0.000000 +0.000000 +0.000000
```

and thus the Gower rank of these (transformed) data is 25.

If we start our `smacof` iterations with the first two dimensions of the full dimensional solution stress is 0.0899492031 after 238 iterations, the same solution as with the Torgerson start.

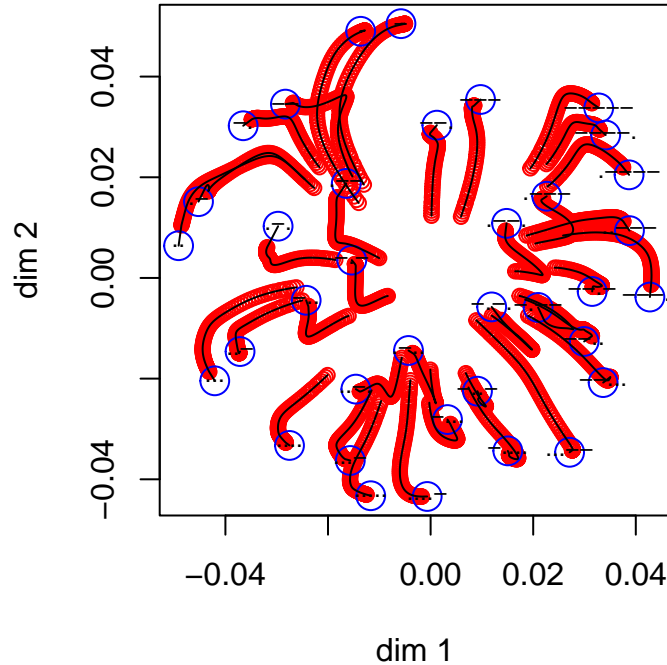
If we run `smacof` 1000 times with a random start we find a huge number of local minima. It is difficult to draw the line on which local minima are the same and which are different, but our usual procedure says there are 684 of them. It would be silly to print them out here. Suffices it to say that the smallest stress is 0.0899492025, which is found in only about 7% of the cases. It is the same as stress for the previous Torgerson and FDS solutions.

Our global optimization method with the default λ sequence also converges to the same

smallest local minimum.

## itel	1461	lambda	0.000000	stress	0.000763	penalty	0.472254
## itel	6	lambda	0.010000	stress	0.000858	penalty	0.322181
## itel	4	lambda	0.020000	stress	0.001147	penalty	0.308335
## itel	3	lambda	0.030000	stress	0.001622	penalty	0.294777
## itel	2	lambda	0.040000	stress	0.002207	penalty	0.283003
## itel	2	lambda	0.050000	stress	0.003018	penalty	0.269965
## itel	1	lambda	0.060000	stress	0.003720	penalty	0.261134
## itel	2	lambda	0.070000	stress	0.005058	penalty	0.245658
## itel	1	lambda	0.080000	stress	0.006058	penalty	0.236320
## itel	1	lambda	0.090000	stress	0.007232	penalty	0.226392
## itel	1	lambda	0.100000	stress	0.008576	penalty	0.216089
## itel	1	lambda	0.110000	stress	0.010085	penalty	0.205587
## itel	1	lambda	0.120000	stress	0.011750	penalty	0.195030
## itel	1	lambda	0.130000	stress	0.013559	penalty	0.184543
## itel	1	lambda	0.140000	stress	0.015500	penalty	0.174225
## itel	1	lambda	0.150000	stress	0.017554	penalty	0.164160
## itel	1	lambda	0.160000	stress	0.019705	penalty	0.154411
## itel	1	lambda	0.170000	stress	0.021934	penalty	0.145029
## itel	1	lambda	0.180000	stress	0.024222	penalty	0.136047
## itel	1	lambda	0.190000	stress	0.026550	penalty	0.127488
## itel	1	lambda	0.200000	stress	0.028903	penalty	0.119364
## itel	1	lambda	0.210000	stress	0.031265	penalty	0.111677
## itel	1	lambda	0.220000	stress	0.033621	penalty	0.104423
## itel	1	lambda	0.230000	stress	0.035962	penalty	0.097591
## itel	1	lambda	0.240000	stress	0.038277	penalty	0.091167
## itel	1	lambda	0.250000	stress	0.040558	penalty	0.085132
## itel	1	lambda	0.260000	stress	0.042799	penalty	0.079467
## itel	1	lambda	0.270000	stress	0.044995	penalty	0.074148
## itel	1	lambda	0.280000	stress	0.047143	penalty	0.069155
## itel	1	lambda	0.290000	stress	0.049240	penalty	0.064466
## itel	1	lambda	0.300000	stress	0.051285	penalty	0.060060
## itel	1	lambda	0.310000	stress	0.053275	penalty	0.055920
## itel	1	lambda	0.320000	stress	0.055211	penalty	0.052027
## itel	1	lambda	0.330000	stress	0.057092	penalty	0.048366
## itel	1	lambda	0.340000	stress	0.058916	penalty	0.044923
## itel	1	lambda	0.350000	stress	0.060683	penalty	0.041685
## itel	1	lambda	0.360000	stress	0.062393	penalty	0.038640
## itel	1	lambda	0.370000	stress	0.064045	penalty	0.035778
## itel	1	lambda	0.380000	stress	0.065639	penalty	0.033088
## itel	1	lambda	0.390000	stress	0.067175	penalty	0.030562
## itel	1	lambda	0.400000	stress	0.068653	penalty	0.028190
## itel	1	lambda	0.410000	stress	0.070072	penalty	0.025963
## itel	1	lambda	0.420000	stress	0.071433	penalty	0.023875

## itel	1	lambda	0.430000	stress	0.072735	penalty	0.021917
## itel	1	lambda	0.440000	stress	0.073980	penalty	0.020083
## itel	1	lambda	0.450000	stress	0.075168	penalty	0.018366
## itel	1	lambda	0.460000	stress	0.076298	penalty	0.016761
## itel	1	lambda	0.470000	stress	0.077371	penalty	0.015263
## itel	1	lambda	0.480000	stress	0.078389	penalty	0.013865
## itel	1	lambda	0.490000	stress	0.079351	penalty	0.012564
## itel	1	lambda	0.500000	stress	0.080258	penalty	0.011356
## itel	1	lambda	0.510000	stress	0.081112	penalty	0.010237
## itel	1	lambda	0.520000	stress	0.081912	penalty	0.009204
## itel	1	lambda	0.530000	stress	0.082661	penalty	0.008252
## itel	1	lambda	0.540000	stress	0.083360	penalty	0.007379
## itel	1	lambda	0.550000	stress	0.084009	penalty	0.006580
## itel	1	lambda	0.560000	stress	0.084610	penalty	0.005853
## itel	1	lambda	0.570000	stress	0.085165	penalty	0.005194
## itel	1	lambda	0.580000	stress	0.085676	penalty	0.004598
## itel	1	lambda	0.590000	stress	0.086144	penalty	0.004061
## itel	1	lambda	0.600000	stress	0.086572	penalty	0.003578
## itel	1	lambda	0.610000	stress	0.086962	penalty	0.003145
## itel	1	lambda	0.620000	stress	0.087316	penalty	0.002757
## itel	1	lambda	0.630000	stress	0.087637	penalty	0.002411
## itel	1	lambda	0.640000	stress	0.087927	penalty	0.002102
## itel	1	lambda	0.650000	stress	0.088188	penalty	0.001827
## itel	1	lambda	0.660000	stress	0.088423	penalty	0.001582
## itel	1	lambda	0.670000	stress	0.088633	penalty	0.001365
## itel	1	lambda	0.680000	stress	0.088822	penalty	0.001173
## itel	1	lambda	0.690000	stress	0.088990	penalty	0.001003
## itel	1	lambda	0.700000	stress	0.089140	penalty	0.000854
## itel	1	lambda	0.710000	stress	0.089273	penalty	0.000723
## itel	1	lambda	0.720000	stress	0.089390	penalty	0.000608
## itel	1	lambda	0.730000	stress	0.089493	penalty	0.000509
## itel	1	lambda	0.740000	stress	0.089583	penalty	0.000422
## itel	1	lambda	0.750000	stress	0.089660	penalty	0.000348
## itel	1	lambda	0.760000	stress	0.089726	penalty	0.000284
## itel	1	lambda	0.770000	stress	0.089782	penalty	0.000230
## itel	1	lambda	0.780000	stress	0.089828	penalty	0.000185
## itel	1	lambda	0.790000	stress	0.089867	penalty	0.000147
## itel	1	lambda	0.800000	stress	0.089898	penalty	0.000116
## itel	1	lambda	0.810000	stress	0.089923	penalty	0.000090
## itel	1	lambda	0.820000	stress	0.089943	penalty	0.000070
## itel	1	lambda	0.830000	stress	0.089958	penalty	0.000053
## itel	1	lambda	0.840000	stress	0.089970	penalty	0.000040
## itel	197	lambda	0.850000	stress	0.089949	penalty	0.000000



Vegetables

Our first one-dimensional example uses paired comparisons of 9 vegetables, originating with Guilford (1954) and taken from the `psychTools` package (Revelle (2023)). The proportions are transformed to dissimilarities by using the normal quantile function, i.e. $\delta_{ij} = |\Phi^{-1}(p_{ij})|$.

Using the Torgerson initial configuration in two dimensions, `smacof` finds a stress of 0.0353011713 after 3 iterations.

A full-dimensional scaling with $p = 8$ gives stress 0.0136747248 after 1378 iterations. The singular values of the full dimensional solution are

```
## [1] +0.406637 +0.209179 +0.100334 +0.002692 +0.000000 +0.000000 +0.000000
## [8] +0.000000
```

and thus the Gower rank of these (transformed) data is 4.

If we start our `smacof` iterations with the first dimension of the full dimensional solution stress is 0.0353011713 after 3 iterations, the same solution as with the Torgerson start.

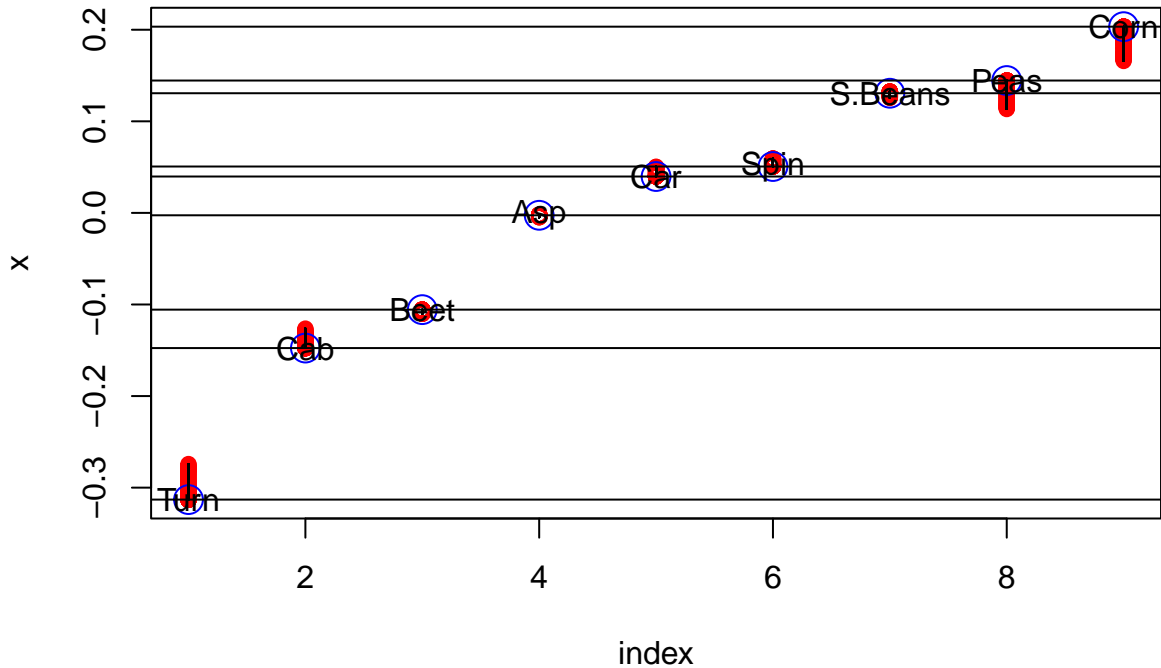
If we run `smacof` 1000 times with a random start we find a huge number of local minima. It is difficult to draw the line on which local minima are the same and which are different, but our usual procedure says there are 665 of them. It would be silly to print them all out here. Suffices it to say that the smallest stress is 0.0353011713, which is found in only about 10% of the cases. It is the same as stress for the previous Torgerson and FDS solutions.

Our global optimization method with the default λ sequence also converges to the same smallest local minimum.

```
## itel 1412 lambda 0.000000 stress 0.013675 penalty 0.269308
```

## itel	5	lambda	0.010000	stress	0.013716	penalty	0.114786
## itel	3	lambda	0.020000	stress	0.013831	penalty	0.108848
## itel	2	lambda	0.030000	stress	0.014001	penalty	0.103509
## itel	2	lambda	0.040000	stress	0.014271	penalty	0.097345
## itel	1	lambda	0.050000	stress	0.014489	penalty	0.093513
## itel	1	lambda	0.060000	stress	0.014782	penalty	0.089107
## itel	1	lambda	0.070000	stress	0.015148	penalty	0.084346
## itel	1	lambda	0.080000	stress	0.015587	penalty	0.079388
## itel	1	lambda	0.090000	stress	0.016095	penalty	0.074352
## itel	1	lambda	0.100000	stress	0.016666	penalty	0.069327
## itel	1	lambda	0.110000	stress	0.017295	penalty	0.064380
## itel	1	lambda	0.120000	stress	0.017973	penalty	0.059564
## itel	1	lambda	0.130000	stress	0.018693	penalty	0.054914
## itel	1	lambda	0.140000	stress	0.019446	penalty	0.050458
## itel	1	lambda	0.150000	stress	0.020224	penalty	0.046215
## itel	1	lambda	0.160000	stress	0.021019	penalty	0.042197
## itel	1	lambda	0.170000	stress	0.021824	penalty	0.038412
## itel	1	lambda	0.180000	stress	0.022630	penalty	0.034863
## itel	1	lambda	0.190000	stress	0.023432	penalty	0.031550
## itel	1	lambda	0.200000	stress	0.024224	penalty	0.028472
## itel	1	lambda	0.210000	stress	0.025000	penalty	0.025622
## itel	1	lambda	0.220000	stress	0.025755	penalty	0.022994
## itel	1	lambda	0.230000	stress	0.026487	penalty	0.020581
## itel	1	lambda	0.240000	stress	0.027191	penalty	0.018373
## itel	1	lambda	0.250000	stress	0.027865	penalty	0.016360
## itel	1	lambda	0.260000	stress	0.028506	penalty	0.014530
## itel	1	lambda	0.270000	stress	0.029114	penalty	0.012872
## itel	1	lambda	0.280000	stress	0.029687	penalty	0.011375
## itel	1	lambda	0.290000	stress	0.030225	penalty	0.010027
## itel	1	lambda	0.300000	stress	0.030728	penalty	0.008815
## itel	1	lambda	0.310000	stress	0.031196	penalty	0.007730
## itel	1	lambda	0.320000	stress	0.031630	penalty	0.006761
## itel	1	lambda	0.330000	stress	0.032030	penalty	0.005896
## itel	1	lambda	0.340000	stress	0.032399	penalty	0.005127
## itel	1	lambda	0.350000	stress	0.032736	penalty	0.004445
## itel	1	lambda	0.360000	stress	0.033044	penalty	0.003840
## itel	1	lambda	0.370000	stress	0.033324	penalty	0.003307
## itel	1	lambda	0.380000	stress	0.033578	penalty	0.002836
## itel	1	lambda	0.390000	stress	0.033806	penalty	0.002423
## itel	1	lambda	0.400000	stress	0.034011	penalty	0.002060
## itel	1	lambda	0.410000	stress	0.034195	penalty	0.001744
## itel	1	lambda	0.420000	stress	0.034358	penalty	0.001469
## itel	1	lambda	0.430000	stress	0.034502	penalty	0.001230
## itel	1	lambda	0.440000	stress	0.034629	penalty	0.001024
## itel	1	lambda	0.450000	stress	0.034739	penalty	0.000847

## itel	1	lambda	0.460000	stress	0.034835	penalty	0.000696
## itel	1	lambda	0.470000	stress	0.034918	penalty	0.000567
## itel	1	lambda	0.480000	stress	0.034989	penalty	0.000459
## itel	1	lambda	0.490000	stress	0.035049	penalty	0.000369
## itel	1	lambda	0.500000	stress	0.035099	penalty	0.000293
## itel	1	lambda	0.510000	stress	0.035141	penalty	0.000231
## itel	1	lambda	0.520000	stress	0.035175	penalty	0.000181
## itel	1	lambda	0.530000	stress	0.035204	penalty	0.000140
## itel	1	lambda	0.540000	stress	0.035226	penalty	0.000107
## itel	1	lambda	0.550000	stress	0.035244	penalty	0.000081
## itel	1	lambda	0.560000	stress	0.035258	penalty	0.000061
## itel	1	lambda	0.570000	stress	0.035269	penalty	0.000045
## itel	1	lambda	0.580000	stress	0.035278	penalty	0.000033
## itel	1	lambda	0.590000	stress	0.035284	penalty	0.000024
## itel	1	lambda	0.600000	stress	0.035289	penalty	0.000017
## itel	1	lambda	0.610000	stress	0.035293	penalty	0.000012
## itel	1	lambda	0.620000	stress	0.035295	penalty	0.000009
## itel	1	lambda	0.630000	stress	0.035297	penalty	0.000006
## itel	1	lambda	0.640000	stress	0.035298	penalty	0.000004
## itel	1	lambda	0.650000	stress	0.035299	penalty	0.000003
## itel	1	lambda	0.660000	stress	0.035300	penalty	0.000002
## itel	1	lambda	0.670000	stress	0.035300	penalty	0.000001
## itel	1	lambda	0.680000	stress	0.035301	penalty	0.000001
## itel	1	lambda	0.690000	stress	0.035301	penalty	0.000001
## itel	1	lambda	0.700000	stress	0.035301	penalty	0.000000
## itel	1	lambda	0.710000	stress	0.035301	penalty	0.000000
## itel	4	lambda	0.720000	stress	0.035301	penalty	0.000000
## itel	3	lambda	0.730000	stress	0.035301	penalty	0.000000



```
## itel 1412 lambda 0.000000 stress 0.013675 penalty 0.269308
## itel 5 lambda 0.010000 stress 0.013716 penalty 0.114786
## itel 5 lambda 0.100000 stress 0.016719 penalty 0.069309
## itel 23 lambda 1.000000 stress 0.035301 penalty 0.000000
```

This example was previously analyzed in De Leeuw (2005) using enumeration of all permutations. He found 14354 isolated local minima, and a global minimum equal to the one we computed here.

Plato

Mair, Groenen, and De Leeuw (2022) use seriation of the works of Plato, from the data collected by Cox and Brandwood (1959), as an example of unidimensional scaling.

Using the Torgerson initial configuration in one dimension, smacof finds a stress of 0.1436303539 after 2 iterations.

A full-dimensional scaling with $p = 6$ gives stress $1.525336516 \times 10^{-31}$ after 1 iterations. The singular values of the full dimensional solution are

```
## [1] +0.367934 +0.238357 +0.211808 +0.155444 +0.130119 +0.086998
```

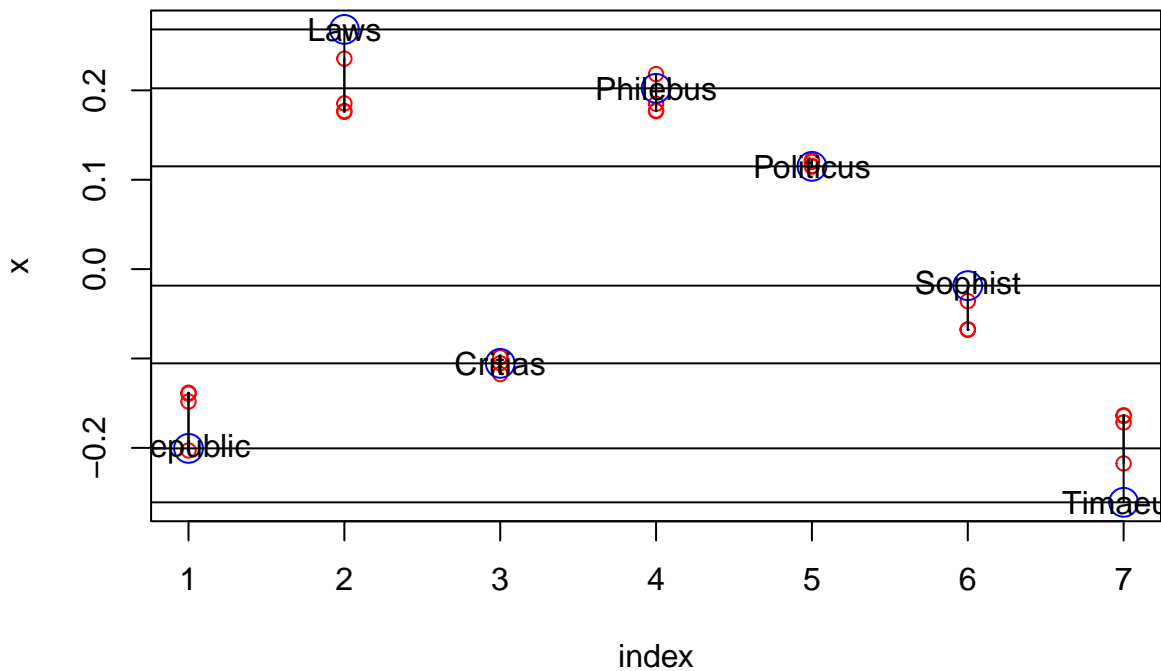
and thus the Gower rank of these (transformed) data is 6.

If we start our smacof iterations with the first dimension of the full dimensional solution stress is 0.1436303539 after 2 iterations, the same solution as with the Torgerson start.

If we run smacof 1000 times with a random start we find 793 local minima. Suffices it to say that the smallest stress is 0.1287689224, which is found in only one of the 1000 runs. It is smaller than stress for the previous Torgerson and FDS solutions.

Now start the global method with a short λ sequence.

```
## itel 169 lambda 0.000000 stress 0.000000 penalty 0.410927
## itel 3 lambda 0.010000 stress 0.000062 penalty 0.255246
## itel 3 lambda 0.100000 stress 0.005117 penalty 0.194993
## itel 4 lambda 1.000000 stress 0.106058 penalty 0.019675
## itel 9 lambda 10.000000 stress 0.139462 penalty 0.000000
```



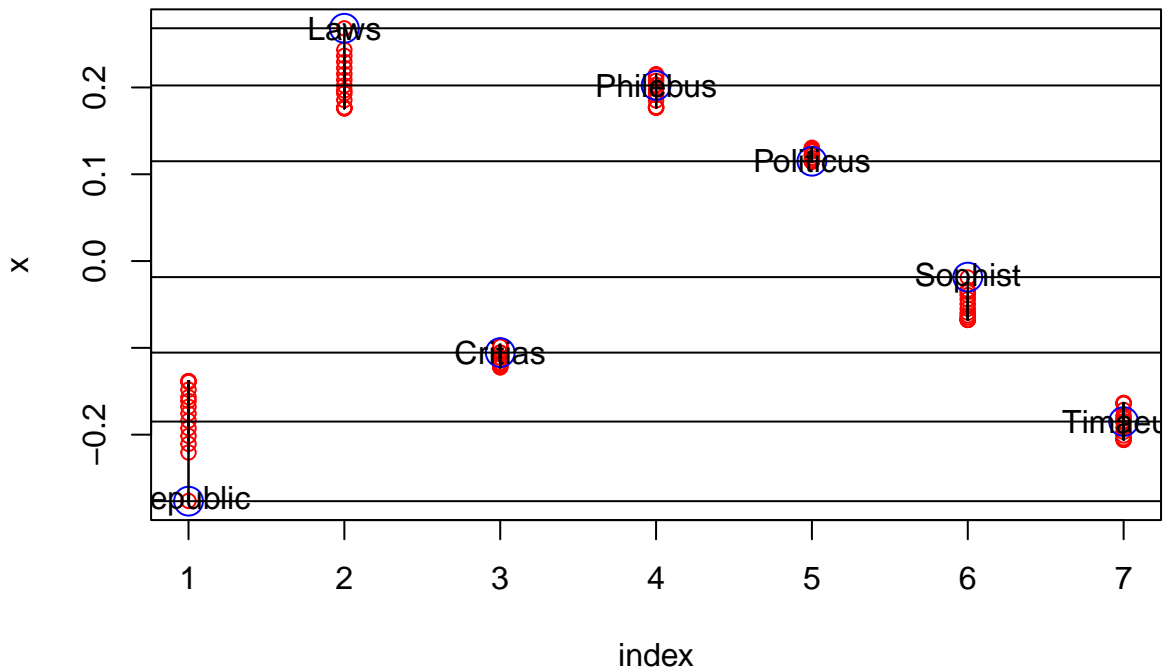
This gives the order

```
##      [,1]
## [1,] "Timaeus"
## [2,] "Republic"
## [3,] "Critias"
## [4,] "Sophist"
## [5,] "Politicus"
## [6,] "Philebus"
## [7,] "Laws"
```

which is different from the order at the global minimum, which has Republic before Timaeus. Thus we have recovered a local minimum, and it seems our sequence of λ values was not fine enough to do the job properly. So we try a longer and finer sequence.

```
## itel 169 lambda 0.000000 stress 0.000000 penalty 0.410927
## itel 3 lambda 0.000100 stress 0.000000 penalty 0.263015
## itel 3 lambda 0.001000 stress 0.000001 penalty 0.262280
## itel 3 lambda 0.010000 stress 0.000064 penalty 0.255078
## itel 3 lambda 0.100000 stress 0.005123 penalty 0.194945
## itel 2 lambda 0.200000 stress 0.016184 penalty 0.147493
```

```
## itel    1 lambda    0.300000 stress 0.026997 penalty 0.119323
## itel    1 lambda    0.400000 stress 0.040023 penalty 0.093615
## itel    1 lambda    0.500000 stress 0.053688 penalty 0.072330
## itel    1 lambda    0.600000 stress 0.066833 penalty 0.055452
## itel    1 lambda    0.700000 stress 0.078832 penalty 0.042269
## itel    1 lambda    0.800000 stress 0.089439 penalty 0.032019
## itel    1 lambda    0.900000 stress 0.098557 penalty 0.024079
## itel    1 lambda    1.000000 stress 0.106135 penalty 0.017940
## itel    6 lambda    2.000000 stress 0.130789 penalty 0.000148
## itel   13 lambda    3.000000 stress 0.131135 penalty 0.000000
```



Now the order is

```
##      [,1]
## [1,] "Republic"
## [2,] "Timaeus"
## [3,] "Critias"
## [4,] "Sophist"
## [5,] "Politicus"
## [6,] "Philebus"
## [7,] "Laws"
```

which does indeed correspond to the global minimum.

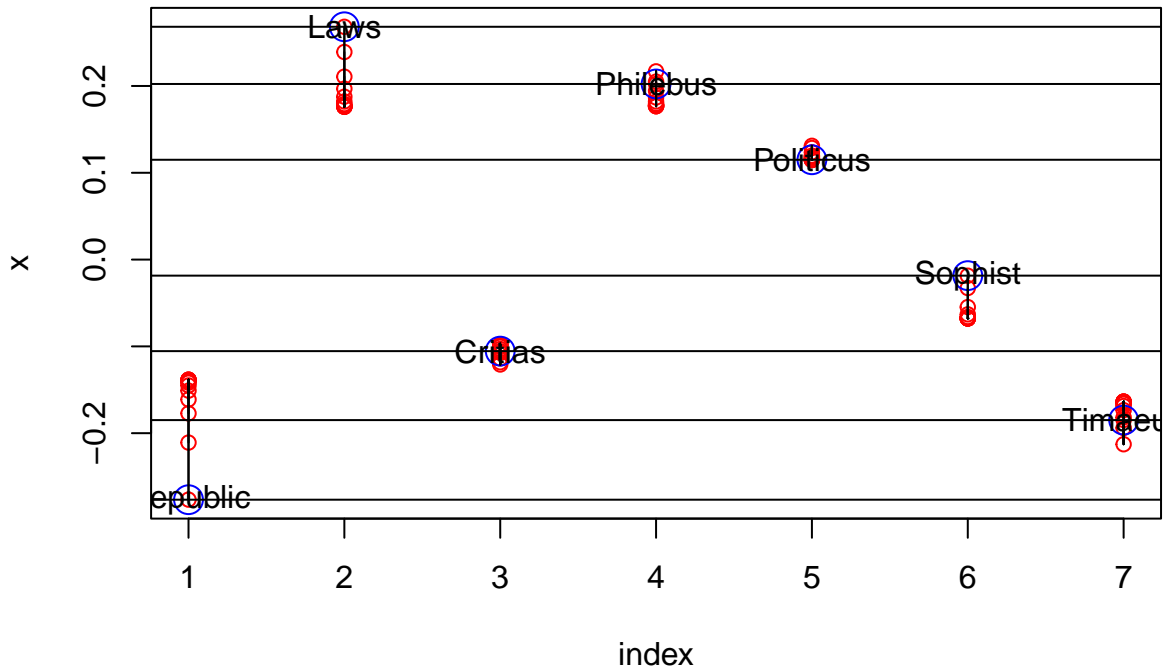
With a different λ sequence we find the same solution.

```
## itel  169 lambda    0.000000 stress 0.000000 penalty 0.410927
## itel    3 lambda    0.001000 stress 0.000001 penalty 0.262296
## itel    2 lambda    0.002000 stress 0.000003 penalty 0.261483
```

```

## itel    2 lambda    0.004000 stress 0.000010 penalty 0.259872
## itel    2 lambda    0.008000 stress 0.000041 penalty 0.256690
## itel    2 lambda    0.016000 stress 0.000159 penalty 0.250470
## itel    2 lambda    0.032000 stress 0.000613 penalty 0.238574
## itel    2 lambda    0.064000 stress 0.002266 penalty 0.216785
## itel    2 lambda    0.128000 stress 0.007791 penalty 0.180067
## itel    2 lambda    0.256000 stress 0.023483 penalty 0.127006
## itel    2 lambda    0.512000 stress 0.056940 penalty 0.067948
## itel    3 lambda    1.024000 stress 0.107743 penalty 0.017937
## itel    8 lambda    2.048000 stress 0.131059 penalty 0.000032
## itel    9 lambda    4.096000 stress 0.131135 penalty 0.000000

```



The order is

```

##      [,1]
## [1,] "Republic"
## [2,] "Timaeus"
## [3,] "Critias"
## [4,] "Sophist"
## [5,] "Politicus"
## [6,] "Philebus"
## [7,] "Laws"

```

Our global optimization method with the default λ sequence also converges to the same smallest local minimum.

```

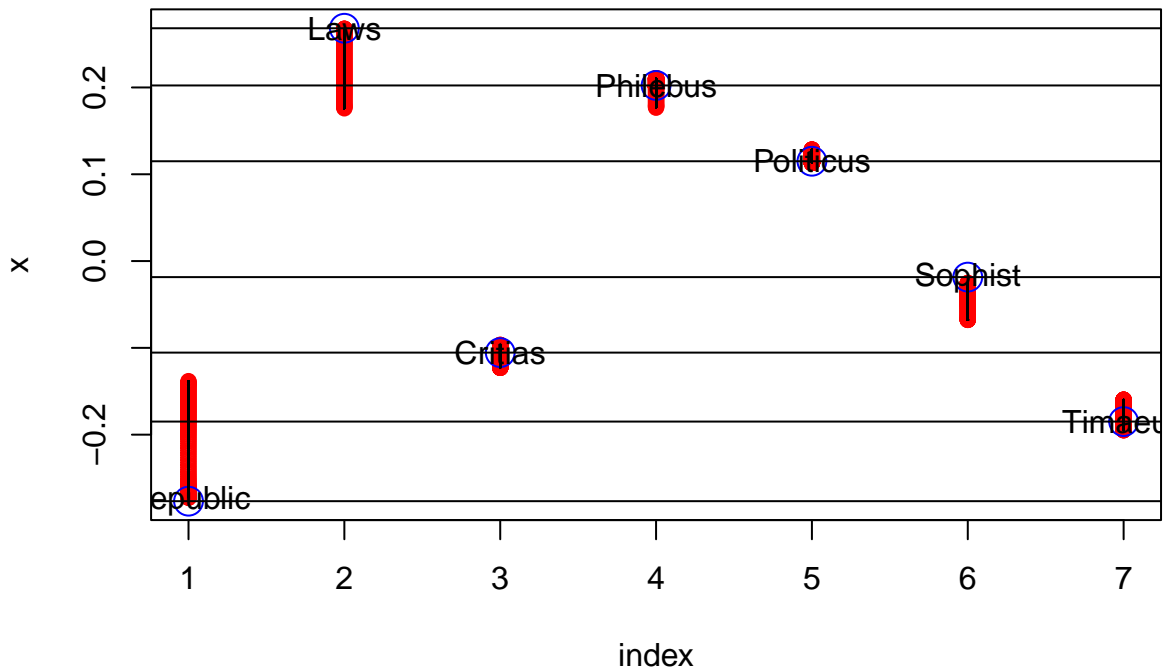
## itel  169 lambda    0.000000 stress 0.000000 penalty 0.410927
## itel    3 lambda    0.010000 stress 0.000062 penalty 0.255246
## itel    2 lambda    0.020000 stress 0.000245 penalty 0.247505

```


## itel	1	lambda	0.030000	stress	0.000488	penalty	0.241370
## itel	1	lambda	0.040000	stress	0.000859	penalty	0.234376
## itel	1	lambda	0.050000	stress	0.001366	penalty	0.226943
## itel	1	lambda	0.060000	stress	0.002009	penalty	0.219341
## itel	1	lambda	0.070000	stress	0.002784	penalty	0.211742
## itel	1	lambda	0.080000	stress	0.003684	penalty	0.204257
## itel	1	lambda	0.090000	stress	0.004698	penalty	0.196954
## itel	1	lambda	0.100000	stress	0.005816	penalty	0.189875
## itel	1	lambda	0.110000	stress	0.007025	penalty	0.183041
## itel	1	lambda	0.120000	stress	0.008317	penalty	0.176460
## itel	1	lambda	0.130000	stress	0.009682	penalty	0.170134
## itel	1	lambda	0.140000	stress	0.011110	penalty	0.164057
## itel	1	lambda	0.150000	stress	0.012594	penalty	0.158222
## itel	1	lambda	0.160000	stress	0.014128	penalty	0.152618
## itel	1	lambda	0.170000	stress	0.015704	penalty	0.147236
## itel	1	lambda	0.180000	stress	0.017317	penalty	0.142064
## itel	1	lambda	0.190000	stress	0.018962	penalty	0.137093
## itel	1	lambda	0.200000	stress	0.020635	penalty	0.132311
## itel	1	lambda	0.210000	stress	0.022331	penalty	0.127710
## itel	1	lambda	0.220000	stress	0.024047	penalty	0.123281
## itel	1	lambda	0.230000	stress	0.025780	penalty	0.119015
## itel	1	lambda	0.240000	stress	0.027525	penalty	0.114904
## itel	1	lambda	0.250000	stress	0.029281	penalty	0.110941
## itel	1	lambda	0.260000	stress	0.031045	penalty	0.107118
## itel	1	lambda	0.270000	stress	0.032814	penalty	0.103430
## itel	1	lambda	0.280000	stress	0.034587	penalty	0.099869
## itel	1	lambda	0.290000	stress	0.036361	penalty	0.096430
## itel	1	lambda	0.300000	stress	0.038136	penalty	0.093107
## itel	1	lambda	0.310000	stress	0.039909	penalty	0.089895
## itel	1	lambda	0.320000	stress	0.041680	penalty	0.086788
## itel	1	lambda	0.330000	stress	0.043447	penalty	0.083781
## itel	1	lambda	0.340000	stress	0.045209	penalty	0.080870
## itel	1	lambda	0.350000	stress	0.046965	penalty	0.078050
## itel	1	lambda	0.360000	stress	0.048715	penalty	0.075317
## itel	1	lambda	0.370000	stress	0.050458	penalty	0.072666
## itel	1	lambda	0.380000	stress	0.052193	penalty	0.070095
## itel	1	lambda	0.390000	stress	0.053920	penalty	0.067598
## itel	1	lambda	0.400000	stress	0.055637	penalty	0.065173
## itel	1	lambda	0.410000	stress	0.057344	penalty	0.062816
## itel	1	lambda	0.420000	stress	0.059042	penalty	0.060524
## itel	1	lambda	0.430000	stress	0.060728	penalty	0.058295
## itel	1	lambda	0.440000	stress	0.062404	penalty	0.056125
## itel	1	lambda	0.450000	stress	0.064068	penalty	0.054012
## itel	1	lambda	0.460000	stress	0.065721	penalty	0.051953
## itel	1	lambda	0.470000	stress	0.067362	penalty	0.049946

## itel	1	lambda	0.480000	stress	0.068991	penalty	0.047990
## itel	1	lambda	0.490000	stress	0.070607	penalty	0.046082
## itel	1	lambda	0.500000	stress	0.072211	penalty	0.044222
## itel	1	lambda	0.510000	stress	0.073802	penalty	0.042410
## itel	1	lambda	0.520000	stress	0.075378	penalty	0.040646
## itel	1	lambda	0.530000	stress	0.076938	penalty	0.038931
## itel	1	lambda	0.540000	stress	0.078479	penalty	0.037266
## itel	1	lambda	0.550000	stress	0.079999	penalty	0.035654
## itel	1	lambda	0.560000	stress	0.081496	penalty	0.034096
## itel	1	lambda	0.570000	stress	0.082965	penalty	0.032594
## itel	1	lambda	0.580000	stress	0.084405	penalty	0.031150
## itel	1	lambda	0.590000	stress	0.085811	penalty	0.029765
## itel	1	lambda	0.600000	stress	0.087182	penalty	0.028440
## itel	1	lambda	0.610000	stress	0.088516	penalty	0.027174
## itel	1	lambda	0.620000	stress	0.089811	penalty	0.025966
## itel	1	lambda	0.630000	stress	0.091066	penalty	0.024817
## itel	1	lambda	0.640000	stress	0.092280	penalty	0.023725
## itel	1	lambda	0.650000	stress	0.093453	penalty	0.022689
## itel	1	lambda	0.660000	stress	0.094584	penalty	0.021706
## itel	1	lambda	0.670000	stress	0.095675	penalty	0.020775
## itel	1	lambda	0.680000	stress	0.096726	penalty	0.019895
## itel	1	lambda	0.690000	stress	0.097737	penalty	0.019062
## itel	1	lambda	0.700000	stress	0.098709	penalty	0.018275
## itel	1	lambda	0.710000	stress	0.099645	penalty	0.017532
## itel	1	lambda	0.720000	stress	0.100545	penalty	0.016830
## itel	1	lambda	0.730000	stress	0.101411	penalty	0.016166
## itel	1	lambda	0.740000	stress	0.102244	penalty	0.015540
## itel	1	lambda	0.750000	stress	0.103046	penalty	0.014947
## itel	1	lambda	0.760000	stress	0.103818	penalty	0.014387
## itel	1	lambda	0.770000	stress	0.104563	penalty	0.013856
## itel	1	lambda	0.780000	stress	0.105280	penalty	0.013354
## itel	1	lambda	0.790000	stress	0.105972	penalty	0.012877
## itel	1	lambda	0.800000	stress	0.106641	penalty	0.012425
## itel	1	lambda	0.810000	stress	0.107286	penalty	0.011995
## itel	1	lambda	0.820000	stress	0.107909	penalty	0.011587
## itel	1	lambda	0.830000	stress	0.108510	penalty	0.011198
## itel	1	lambda	0.840000	stress	0.109092	penalty	0.010828
## itel	1	lambda	0.850000	stress	0.109655	penalty	0.010476
## itel	1	lambda	0.860000	stress	0.110199	penalty	0.010140
## itel	1	lambda	0.870000	stress	0.110726	penalty	0.009819
## itel	1	lambda	0.880000	stress	0.111236	penalty	0.009513
## itel	1	lambda	0.890000	stress	0.111731	penalty	0.009220
## itel	1	lambda	0.900000	stress	0.112210	penalty	0.008940
## itel	1	lambda	0.910000	stress	0.112676	penalty	0.008672
## itel	1	lambda	0.920000	stress	0.113129	penalty	0.008414

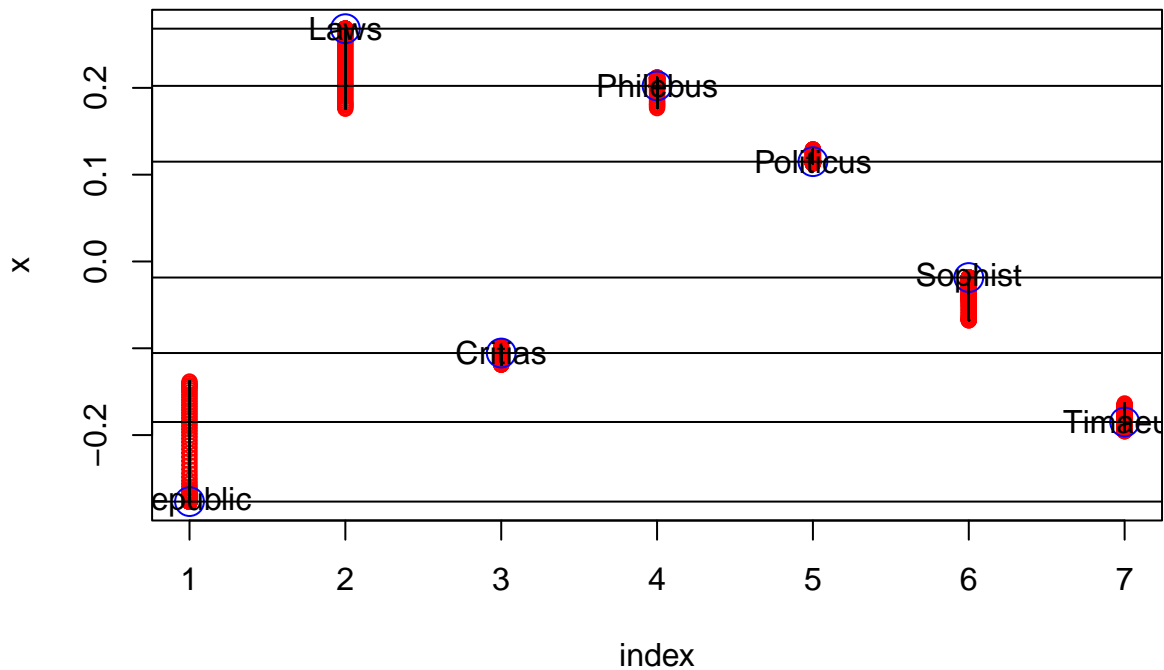
```
## itel    1 lambda    0.930000 stress 0.113570 penalty 0.008166
## itel    1 lambda    0.940000 stress 0.114001 penalty 0.007928
## itel    1 lambda    0.950000 stress 0.114422 penalty 0.007697
## itel    1 lambda    0.960000 stress 0.114834 penalty 0.007474
## itel    1 lambda    0.970000 stress 0.115238 penalty 0.007258
## itel    1 lambda    0.980000 stress 0.115637 penalty 0.007047
## itel    1 lambda    0.990000 stress 0.116030 penalty 0.006841
## itel    1 lambda    1.000000 stress 0.116419 penalty 0.006639
## itel    1 lambda    2.000000 stress 0.124820 penalty 0.002891
## itel   11 lambda    5.000000 stress 0.131135 penalty 0.000000
```



```
## itel   169 lambda    0.000000 stress 0.000000 penalty 0.410927
## itel    3 lambda    0.020000 stress 0.000241 penalty 0.247687
## itel    2 lambda    0.040000 stress 0.000932 penalty 0.233031
## itel    1 lambda    0.060000 stress 0.001815 penalty 0.221839
## itel    1 lambda    0.080000 stress 0.003123 penalty 0.209450
## itel    1 lambda    0.100000 stress 0.004849 penalty 0.196723
## itel    1 lambda    0.120000 stress 0.006964 penalty 0.184174
## itel    1 lambda    0.140000 stress 0.009420 penalty 0.172104
## itel    1 lambda    0.160000 stress 0.012159 penalty 0.160674
## itel    1 lambda    0.180000 stress 0.015125 penalty 0.149951
## itel    1 lambda    0.200000 stress 0.018264 penalty 0.139947
## itel    1 lambda    0.220000 stress 0.021533 penalty 0.130640
## itel    1 lambda    0.240000 stress 0.024892 penalty 0.121988
## itel    1 lambda    0.260000 stress 0.028312 penalty 0.113946
## itel    1 lambda    0.280000 stress 0.031767 penalty 0.106464
## itel    1 lambda    0.300000 stress 0.035236 penalty 0.099496
```

## itel	1	lambda	0.320000	stress	0.038704	penalty	0.092999
## itel	1	lambda	0.340000	stress	0.042155	penalty	0.086932
## itel	1	lambda	0.360000	stress	0.045579	penalty	0.081260
## itel	1	lambda	0.380000	stress	0.048967	penalty	0.075949
## itel	1	lambda	0.400000	stress	0.052311	penalty	0.070970
## itel	1	lambda	0.420000	stress	0.055605	penalty	0.066295
## itel	1	lambda	0.440000	stress	0.058844	penalty	0.061900
## itel	1	lambda	0.460000	stress	0.062024	penalty	0.057764
## itel	1	lambda	0.480000	stress	0.065139	penalty	0.053866
## itel	1	lambda	0.500000	stress	0.068188	penalty	0.050188
## itel	1	lambda	0.520000	stress	0.071166	penalty	0.046713
## itel	1	lambda	0.540000	stress	0.074073	penalty	0.043424
## itel	1	lambda	0.560000	stress	0.076907	penalty	0.040305
## itel	1	lambda	0.580000	stress	0.079668	penalty	0.037343
## itel	1	lambda	0.600000	stress	0.082358	penalty	0.034526
## itel	1	lambda	0.620000	stress	0.084977	penalty	0.031847
## itel	1	lambda	0.640000	stress	0.087524	penalty	0.029303
## itel	1	lambda	0.660000	stress	0.089994	penalty	0.026899
## itel	1	lambda	0.680000	stress	0.092378	penalty	0.024644
## itel	1	lambda	0.700000	stress	0.094662	penalty	0.022549
## itel	1	lambda	0.720000	stress	0.096833	penalty	0.020623
## itel	1	lambda	0.740000	stress	0.098879	penalty	0.018867
## itel	1	lambda	0.760000	stress	0.100796	penalty	0.017278
## itel	1	lambda	0.780000	stress	0.102584	penalty	0.015847
## itel	1	lambda	0.800000	stress	0.104247	penalty	0.014562
## itel	1	lambda	0.820000	stress	0.105793	penalty	0.013409
## itel	1	lambda	0.840000	stress	0.107231	penalty	0.012374
## itel	1	lambda	0.860000	stress	0.108569	penalty	0.011444
## itel	1	lambda	0.880000	stress	0.109818	penalty	0.010607
## itel	1	lambda	0.900000	stress	0.110985	penalty	0.009852
## itel	1	lambda	0.920000	stress	0.112078	penalty	0.009168
## itel	1	lambda	0.940000	stress	0.113105	penalty	0.008546
## itel	1	lambda	0.960000	stress	0.114072	penalty	0.007979
## itel	1	lambda	0.980000	stress	0.114987	penalty	0.007459
## itel	1	lambda	1.000000	stress	0.115855	penalty	0.006980
## itel	1	lambda	1.020000	stress	0.116682	penalty	0.006536
## itel	1	lambda	1.040000	stress	0.117474	penalty	0.006122
## itel	1	lambda	1.060000	stress	0.118237	penalty	0.005732
## itel	1	lambda	1.080000	stress	0.118975	penalty	0.005363
## itel	1	lambda	1.100000	stress	0.119693	penalty	0.005010
## itel	1	lambda	1.120000	stress	0.120395	penalty	0.004671
## itel	1	lambda	1.140000	stress	0.121086	penalty	0.004343
## itel	1	lambda	1.160000	stress	0.121766	penalty	0.004024
## itel	1	lambda	1.180000	stress	0.122438	penalty	0.003713
## itel	1	lambda	1.200000	stress	0.123101	penalty	0.003410

## itel	1	lambda	1.220000	stress	0.123755	penalty	0.003114
## itel	1	lambda	1.240000	stress	0.124397	penalty	0.002826
## itel	1	lambda	1.260000	stress	0.125024	penalty	0.002549
## itel	1	lambda	1.280000	stress	0.125632	penalty	0.002283
## itel	1	lambda	1.300000	stress	0.126217	penalty	0.002029
## itel	1	lambda	1.320000	stress	0.126775	penalty	0.001790
## itel	1	lambda	1.340000	stress	0.127301	penalty	0.001567
## itel	1	lambda	1.360000	stress	0.127793	penalty	0.001360
## itel	1	lambda	1.380000	stress	0.128249	penalty	0.001170
## itel	1	lambda	1.400000	stress	0.128667	penalty	0.000997
## itel	1	lambda	1.420000	stress	0.129046	penalty	0.000841
## itel	1	lambda	1.440000	stress	0.129386	penalty	0.000702
## itel	1	lambda	1.460000	stress	0.129687	penalty	0.000580
## itel	1	lambda	1.480000	stress	0.129950	penalty	0.000473
## itel	1	lambda	1.500000	stress	0.130177	penalty	0.000382
## itel	1	lambda	1.520000	stress	0.130371	penalty	0.000304
## itel	1	lambda	1.540000	stress	0.130533	penalty	0.000239
## itel	1	lambda	1.560000	stress	0.130667	penalty	0.000186
## itel	1	lambda	1.580000	stress	0.130776	penalty	0.000142
## itel	1	lambda	1.600000	stress	0.130864	penalty	0.000107
## itel	1	lambda	1.620000	stress	0.130933	penalty	0.000080
## itel	1	lambda	1.640000	stress	0.130987	penalty	0.000059
## itel	1	lambda	1.660000	stress	0.131028	penalty	0.000042
## itel	1	lambda	1.680000	stress	0.131058	penalty	0.000030
## itel	1	lambda	1.700000	stress	0.131081	penalty	0.000021
## itel	1	lambda	1.720000	stress	0.131098	penalty	0.000015
## itel	1	lambda	1.740000	stress	0.131109	penalty	0.000010
## itel	1	lambda	1.760000	stress	0.131118	penalty	0.000007
## itel	1	lambda	1.780000	stress	0.131123	penalty	0.000004
## itel	1	lambda	1.800000	stress	0.131127	penalty	0.000003
## itel	1	lambda	1.820000	stress	0.131130	penalty	0.000002
## itel	1	lambda	1.840000	stress	0.131132	penalty	0.000001
## itel	1	lambda	1.860000	stress	0.131133	penalty	0.000001
## itel	1	lambda	1.880000	stress	0.131134	penalty	0.000000
## itel	1	lambda	1.900000	stress	0.131134	penalty	0.000000
## itel	1	lambda	1.920000	stress	0.131134	penalty	0.000000
## itel	1	lambda	1.940000	stress	0.131134	penalty	0.000000
## itel	1	lambda	1.960000	stress	0.131135	penalty	0.000000
## itel	1	lambda	1.980000	stress	0.131135	penalty	0.000000
## itel	1	lambda	2.000000	stress	0.131135	penalty	0.000000
## itel	1	lambda	2.020000	stress	0.131135	penalty	0.000000
## itel	1	lambda	2.040000	stress	0.131135	penalty	0.000000

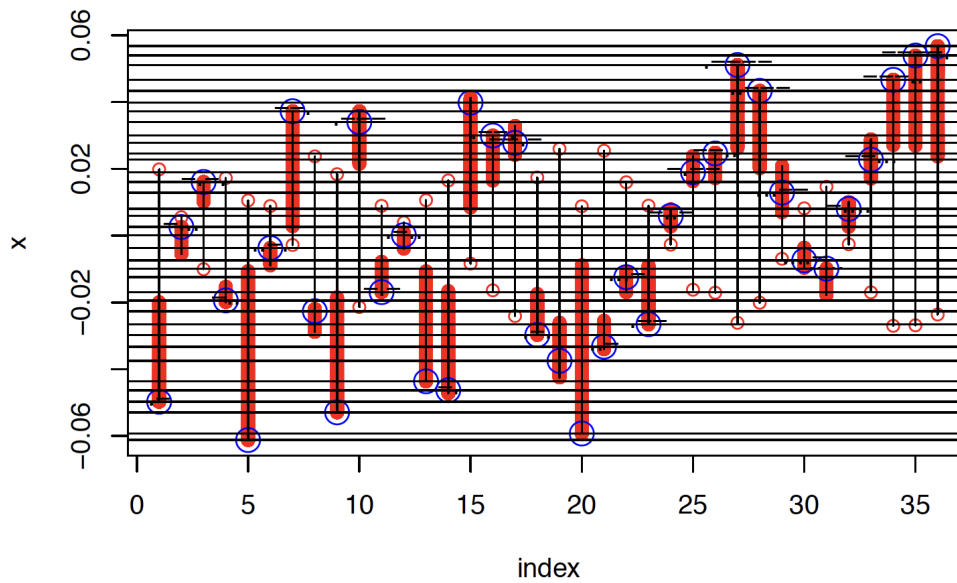


Morse in One

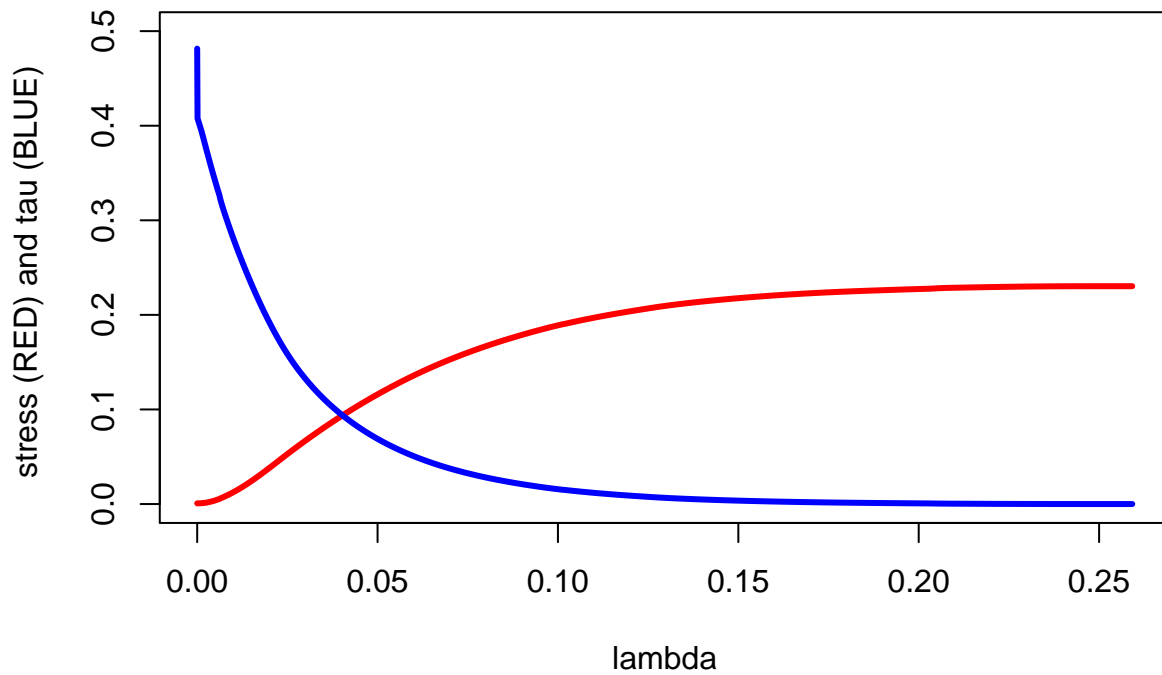
Now for a more challenging example. The Morse code data have been used to try out exact unidimensional MDS techniques, for example by Palubeckis (2013). We will enter the global minimum contest by using 10,000 values of λ , in an equally spaced sequence from 0 to 10. This is not as bad as it sounds. For the 10,000 FDS solutions `system.time()` tells us

```
## user system elapsed
## 26.729 1.063 28.107
```

The one-dimensional plot show quite a bit of movement, but much of it seems to be contained in the very first change of λ .



We can also plot stress and the penalty term as functions of λ . Again, note the big change in the penalty term when λ goes from zero to 0.001.



After the first 2593 values of λ the penalty term is zero and we stop, i.e. we estimate λ_+ is 2.593. At that point we have run a total of 5013 FDS iterations, and thus on average about two iterations per λ value. Stress has increased from 0.0007634501 to 0.2303106976 and the penalty value has decreased from 0.4815136419 to 0.0000000001. We find the following order of the points on the dimension.

```
##      [,1]
## [1,] ". "
```

```

## [2,] "-"
## [3,] ".."
## [4,] "-."
## [5,] "-."
## [6,] "--"
## [7,] "... "
## [8,] "... "
## [9,] "... "
## [10,] "... "
## [11,] "... "
## [12,] "... "
## [13,] "... "
## [14,] "... "
## [15,] "... "
## [16,] "... "
## [17,] "... "
## [18,] "... "
## [19,] "... "
## [20,] "... "
## [21,] "... "
## [22,] "... "
## [23,] "... "
## [24,] "... "
## [25,] "... "
## [26,] "... "
## [27,] "... "
## [28,] "... "
## [29,] "... "
## [30,] "... "
## [31,] "... "
## [32,] "... "
## [33,] "... "
## [34,] "... "
## [35,] "... "
## [36,] "... "

```

Our order, and consequently our solution, is the same as the exact global solution given by Palubeckis (2013). See his table 4, reproduced below. The difference is that computing our solution takes 10 seconds, while his takes 494 seconds. But of course we would not know we actually found the global minimum if the exact exhaustive methods had not analyzed the same data before.

TABLE 4: Optimal solutions for the full Morse code dissimilarity matrix and th

i	Morse code full	
	$p(i)$	$x_{p(i)}$
1	(E) •	0.00000
2	(T) –	5.83333
3	(I) ••	24.77778
4	(A) •–	34.08333
5	(N) –•	44.11111
6	(M) – –	52.33333
7	(S) •••	70.30556
8	(U) ••–	83.13889
9	(R) •–•	93.47222
10	(W) •– –	102.97222
11	(H) ••••	114.44444
12	(D) –••	124.30556
13	(K) –•–	131.52778
14	(V) •••–	145.00000
15	(5) •••••	152.44444
16	(4) •••• –	159.91667
17	(F) ••–•	170.75000
18	(L) •–••	182.25000
19	(B) –•••	189.52778
20	(X) –••–	199.55556
21	(6) –••••	205.66667
22	(3) •••– –	220.13889
23	(C) –•–•	229.47222
24	(Y) –•– –	238.41667
25	(7) – –•••	249.30556
26	(Z) – –••	254.88889
27	(Q) – –•–	264.38889
28	(P) •– –•	270.83333
29	(J) •– – –	282.94444
30	(G) – –•	292.47222
31	(O) – – –	300.22222
32	(2) ••– – –	310.50000
33	(8) – – –••	320.36111
34	(1) •– – – –	333.50000
35	(9) – – – –•	341.86111
36	(0) – – – – –	350.27778

Discussion

There is one surprising (to me, at least) finding from all our examples. There is a value, say λ_+ , such that the penalty $\tau(Y)$ is zero for all PFDS solutions with $\lambda \geq \lambda_+$. In other

words, our penalty function acts like a *smooth exact penalty function*. The precise reason for exactness in our case (if there is one) is not entirely clear to me yet, but it is obviously a topic for further research, using for example the recent theoretical framework of Dolgopolik (2016a), Dolgopolik (2016b), Dolgopolik (2017), Dolgopolik (n.d.).

In our two dimensional examples we always start our plots with the first two dimensions of the FDS configuration. These two-dimensional configurations are usually small (all points relatively close to the origin), because so much variation is still in the higher dimensions. If λ increases the growth of the configurations is one important aspect of configuration change.

In our iteration counts with short sequences of λ we see relatively small increases in stress and small decreases in the penalty term, until we get closer to λ_+ , when we suddenly see a sudden change and a larger number of iterations. This is also reflected in the figures, where generally the change to the last solution (with the largest λ) makes the largest jump. This suggest a finer sequence near λ_+ and perhaps an adaptive strategy for choosing λ . Or to use brute force, as in the unidimensional Morse code example. With such longer and finer sequences convergence becomes more smooth.

Another all-important aspect of the method discussed here is that it assumes computation of the global minimum for each λ . Since we cannot expect a result as nice as the one for FDS (all local minima are global) for $\lambda > 0$ our method remains somewhat heuristic. We have seen that some sequences of λ can take us to a non-global local minimum. Of course the fact that we start with a global minimum for $\lambda = 0$ is of some help, but we do not know how far it will take us in general. Jumps near λ_+ may indicate bifurcations to other local minima.

We have not stressed in the paper that minimizing the penalty function is a *continuation method* (Allgower and George (1979)). This means that probably better methods are available to follow the trajectory of solutions along $\lambda > 0$. There are also possibilities in exploring the fact that the maximum over Z of the penalty function (12) is a concave function of the single variable λ , which is a constant function for all $\lambda > \lambda_+$. There is a duality theory associated with these Courant penalty functions, which we have not used or explored so far.

Appendix A: Exterior Penalty Methods

Suppose $\mathcal{X} \subseteq \mathbb{R}^n$ and $f : \mathbb{R}^n \Rightarrow \mathbb{R}$ is continuous. Define

$$\mathcal{X}_\star = \operatorname{argmin}_{x \in \mathcal{X}} f(x)$$

Suppose \mathcal{X}_\star is non-empty and that x_\star is any element of \mathcal{X}_\star , and

$$f_\star = f(x_\star) = \min_{x \in \mathcal{X}} f(x).$$

The following convergence analysis of external linear penalty methods is standard and can be found in many texts (for example, Zangwill (1969), section 12.2).

The penalty term $g : \mathbb{R}^n \Rightarrow \mathbb{R}^+$ is continuous and satisfies $g(x) = 0$ if and only if $x \in \mathcal{X}$. For each $\lambda > 0$ we define the (linear, external) penalty function

$$h(x, \lambda) = f(x) + \lambda g(x). \tag{18}$$

Suppose $\{\lambda_k\}$ is a strictly increasing sequence of positive real numbers. Define

$$\mathcal{X}_k = \operatorname{argmin}_{x \in \mathcal{X}} h(x, \lambda_k). \quad (19)$$

Suppose all \mathcal{X}_k are nonempty and contained in a compact subset of \mathcal{X} . Choose $x_k \in \mathcal{X}_k$ arbitrarily.

Lemma 2: [Basic]

$$1: h(x_k, \lambda_k) \leq h(x_{k+1}, \lambda_{k+1}).$$

$$2: g(x_k) \geq g(x_{k+1}).$$

$$3: f(x_k) \leq f(x_{k+1}).$$

$$4: f_\star \geq h(x_k, \lambda_k) \geq f(x_k).$$

Proof:

1: We have the chain

$$h(x_{k+1}, \lambda_{k+1}) = f(x_{k+1}) + \lambda_{k+1}g(x_{k+1}) \geq f(x_{k+1}) + \lambda_k g(x_{k+1}) \geq f(x_k) + \lambda_k g(x_k) = h(x_k, \lambda_k).$$

2: Both

$$f(x_k) + \lambda_k g(x_k) \leq f(x_{k+1}) + \lambda_k g(x_{k+1}), \quad (20)$$

$$f(x_{k+1}) + \lambda_{k+1}g(x_{k+1}) \leq f(x_k) + \lambda_{k+1}g(x_k). \quad (21)$$

Adding inequalities (20) and (21) gives

$$\lambda_k g(x_k) + \lambda_{k+1}g(x_{k+1}) \leq \lambda_k g(x_{k+1}) + \lambda_{k+1}g(x_k),$$

or

$$(\lambda_k - \lambda_{k+1})g(x_k) \leq (\lambda_k - \lambda_{k+1})g(x_{k+1}),$$

and thus $g(x_k) \geq g(x_{k+1})$.

3: First

$$f(x_{k+1}) + \lambda_k g(x_{k+1}) \geq f(x_k) + \lambda_k g(x_k). \quad (22)$$

We just proved that $g(x_{k+1}) \geq g(x_k)$, and thus

$$f(x_k) + \lambda_k g(x_k) \geq f(x_k) + \lambda_k g(x_{k+1}). \quad (23)$$

Combining inequalities (22) and (23) gives $f(x_{k+1}) \geq f(x_k)$.

4: We have the chain

$$f_\star = f(x_\star) + \lambda_k g(x_\star) \geq f(x_k) + \lambda_k g(x_k) \geq f(x_k).$$

■

Theorem 3: Suppose the sequence $\{\lambda_k\}_{k \in K}$ diverges to ∞ and $x_{\star\star}$ is the limit of any convergent subsequence $\{x_\ell\}_{\ell \in L}$. Then $x_{\star\star} \in \mathcal{X}_\star$, and $f(x_{\star\star}) = f_\star$, and $g(x_{\star\star}) = 0$.

Proof: Using part 4 of lemma 2

$$\lim_{\ell \in L} h(x_\ell, \lambda_\ell) = \lim_{\ell \in L} \{f(x_\ell) + \lambda_\ell g(x_\ell)\} = f(x_{**}) + \lim_{\ell \in L} \lambda_\ell g(x_\ell) \leq f(x_*) .$$

Thus $\{h(x_\ell, \lambda_\ell)_{\ell \in L}\}$ is a bounded increasing sequence, which consequently converges, and $\lim_{\ell \in L} \lambda_\ell g(x_\ell)$ also converges. Since $\{\lambda_\ell\}_{\ell \in L} \rightarrow \infty$ it follows that $\lim_{\ell \in L} g(x_\ell) = g(x_{**}) = 0$. Thus $x_{**} \in \mathcal{X}$. Since $f(x_\ell) \leq f_*$ we see that $f(x_{**}) \leq f_*$, and thus $x_{**} \in \mathcal{X}_*$ and $f(x_{**}) = f_*$. ■

Appendix B: Code

penalty.R

```
source("smacofBasics.R")

smacofComplement <- function(y, v) {
  return(sum(v * tcrossprod(y)) / 4)
}

smacofPenalty <-
  function(w,
    delta,
    p = 2,
    lbd = 0,
    zold = columnCenter(diag(nrow(delta))),
    itmax = 10000,
    eps = 1e-10,
    verbose = FALSE) {
  itel <- 1
  n <- nrow(zold)
  vmat <- smacofVmat(w)
  vinv <- solve(vmat + (1 / n)) - (1 / n)
  dold <- as.matrix(dist(zold))
  mold <- sum(w * delta * dold) / sum(w * dold * dold)
  zold <- zold * mold
  dold <- dold * mold
  yold <- zold[, (p + 1):n]
  sold <- smacofLoss(dold, w, delta)
  bold <- smacofBmat(dold, w, delta)
  told <- smacofComplement(yold, vmat)
  uold <- sold + lbd * told
  repeat {
    znew <- smacofGuttman(zold, bold, vinv)
    ynew <- znew[, (p + 1):n] / (1 + lbd)
```

```

znew[, (p + 1):n] <- ynew
xnew <- znew[, 1:p]
dnew <- as.matrix(dist(znew))
bnew <- smacofBmat(dnew, w, delta)
tnew <- smacofComplement(ynew, vmat)
snew <- smacofLoss(dnew, w, delta)
unew <- snew + lbd * tnew
if (verbose) {
  cat(
    "itel ",
    formatC(itel, width = 4, format = "d"),
    "sold ",
    formatC(
      sold,
      width = 10,
      digits = 6,
      format = "f"
    ),
    "snew ",
    formatC(
      snew,
      width = 10,
      digits = 6,
      format = "f"
    ),
    "told ",
    formatC(
      told,
      width = 10,
      digits = 6,
      format = "f"
    ),
    "tnew ",
    formatC(
      tnew,
      width = 10,
      digits = 6,
      format = "f"
    ),
    "uold ",
    formatC(
      uold,
      width = 10,
      digits = 6,

```

```

        format = "f"
    ),
    "unew ",
    formatC(
        unew,
        width = 10,
        digits = 6,
        format = "f"
    ),
    "\n"
)
}
if (((uold - unew) < eps) || (itel == itmax)) {
    break
}
itel <- itel + 1
zold <- znew
bold <- bnew
sold <- snew
told <- tnew
uold <- unew
}
zpri <- znew %*% svd(znew)$v
xpri <- zpri[, 1:p]
return(list(
    x = xpri,
    z = zpri,
    b = bnew,
    l = lbd,
    s = snew,
    t = tnew,
    itel = itel
))
}

```

runPenalty.R

```

runPenalty <-
function(w,
    delta,
    lbd,
    p = 2,
    itmax = 10000,
    eps = 1e-10,

```

```

        cut = 1e-8,
        write = TRUE,
        verbose = FALSE) {
m <- length(lbd)
hList <- as.list(1:m)
hList[[1]] <-
  smacofPenalty(
    w,
    delta,
    p,
    lbd = lbd[1],
    itmax = itmax,
    eps = eps,
    verbose = verbose
  )
for (j in 2:m) {
  hList[[j]] <-
    smacofPenalty(
      w,
      delta,
      p,
      zold = hList[[j - 1]]$z,
      lbd = lbd[j],
      itmax = itmax,
      eps = eps,
      verbose = verbose
    )
}
mm <- m
for (i in 1:m) {
  if (write) {
    cat(
      "itel",
      formatC(hList[[i]]$itel, width = 4, format = "d"),
      "lambda",
      formatC(
        hList[[i]]$l,
        width = 10,
        digits = 6,
        format = "f"
      ),
      "stress",
      formatC(
        hList[[i]]$s,

```

```

        width = 8,
        digits = 6,
        format = "f"
    ),
    "penalty",
    formatC(
        hList[[i]]$t,
        width = 8,
        digits = 6,
        format = "f"
    ),
    "\n"
)
}
if (hList[[i]]$t < cut) {
    mm <- i
    break
}
}
return(hList[1:mm])
}

```

```

writeSelected <- function(hList, ind) {
    m <- length(hList)
    n <- length(ind)
    mn <- sort(union(union(1:3, ind), m - (2:0)))
    for (i in mn) {
        if (i > m) {
            next
        }
        cat(
            "itel",
            formatC(hList[[i]]$itel, width = 4, format = "d"),
            "lambda",
            formatC(
                hList[[i]]$l,
                width = 10,
                digits = 6,
                format = "f"
            ),
            "stress",
            formatC(
                hList[[i]]$s,
                width = 8,

```



```

        digits = 6,
        format = "f"
    ),
    "penalty",
    formatC(
        hList[[i]]$t,
        width = 8,
        digits = 6,
        format = "f"
    ),
    "\n"
)
}
}

```

matchMe.R

```

matchMe <- function (x,
                      itmax = 100,
                      eps = 1e-10,
                      verbose = FALSE) {
  m <- length (x)
  y <- sumList (x) / m
  itel <- 1
  fold <- sum (sapply (x, function (z)
    (z - y) ^ 2))
  repeat {
    for (j in 1:m) {
      u <- crossprod (x[[j]], y)
      s <- svd (u)
      r <- tcrossprod (s$u, s$v)
      x[[j]] <- x[[j]] %*% r
    }
    y <- sumList (x) / m
    fnew <- sum (sapply (x, function (z)
      (z - y) ^ 2))
    if (verbose) {
      }
    if (((fold - fnew) < eps) || (itel == itmax))
      break
    itel <- itel + 1
    fold <- fnew
  }
}

```

```

    return (x)
}

sumList <- function (x) {
  m <- length (x)
  y <- x[[1]]
  for (j in 2:m) {
    y <- y + x[[j]]
  }
  return (y)
}

```

plotMe.R

```

plotMe2 <- function(hList, labels, s = 1, t = 2) {
  n <- nrow(hList[[1]]$x)
  m <- length (hList)
  par(pty = "s")
  hMatch <- matchMe(lapply (hList, function(r)
    r$x))
  hMat <- matrix(0, 0, 2)
  for (j in 1:m) {
    hMat <- rbind(hMat, hMatch[[j]][, c(s, t)])
  }
  plot(
    hMat,
    xlab = "dim 1",
    ylab = "dim 2",
    col = c(rep("RED", n * (m - 1)), rep("BLUE", n)),
    cex = c(rep(1, n * (m - 1)), rep(2, n))
  )
  for (i in 1:n) {
    hLine <- matrix(0, 0, 2)
    for (j in 1:m) {
      hLine <- rbind(hLine, hMatch[[j]][i, c(s, t)])
    }
    lines(hLine)
  }
  text(hMatch[[m]], labels, cex = .75)
}

plotMe1 <- function(hList, labels) {
  n <- length(hList[[1]]$x)

```

```

m <- length(hList)
blow <- function(x) {
  n <- length(x)
  return(matrix(c(1:n, x), n, 2))
}
hMat <- matrix(0, 0, 2)
for (j in 1:m) {
  hMat <- rbind(hMat, blow(hList[[j]]$x))
}
plot(
  hMat,
  xlab = "index",
  ylab = "x",
  col = c(rep("RED", n * (m - 1)), rep("BLUE", n)),
  cex = c(rep(1, n * (m - 1)), rep(2, n))
)
for (i in 1:n) {
  hLine <- matrix(0, 0, 2)
  for (j in 1:m) {
    hLine <- rbind(hLine, blow(hList[[j]]$x)[i,])
    lines(hLine)
  }
}
text(blow(hList[[m]]$x), labels, cex = 1.00)
for (i in 1:n) {
  abline(h = hList[[m]]$x[i])
}
}

```

checkUni.R

```

checkUni <- function (w, delta, x) {
  x <- drop (x)
  n <- length (x)
  vinv <- solve (smacofVmat (w) + (1 / n)) - (1 / n)
  return (drop (vinv %*% rowSums (w * delta * sign (outer (x, x, "-")))))
}

```

References

- Allgower, E. L., and K. George. 1979. *Introduction to Numerical Continuation Methods*. Wiley.
- Cox, D. R., and L. Brandwood. 1959. "On a Discriminatory Problem Connected with the

- Works of Plato.” *Journal of the Royal Statistical Society, Series B* 21: 195–200.
- De Gruijter, D. N. M. 1967. “The Cognitive Structure of Dutch Political Parties in 1966.” Report E019-67. Psychological Institute, University of Leiden.
- De Leeuw, J. 1977. “Applications of Convex Analysis to Multidimensional Scaling.” In *Recent Developments in Statistics*, edited by J. R. Barra, F. Brodeau, G. Romier, and B. Van Cutsem, 133–45. Amsterdam, The Netherlands: North Holland Publishing Company.
- . 1984. “Differentiability of Kruskal’s Stress at a Local Minimum.” *Psychometrika* 49: 111–13.
- . 1993. “Fitting Distances by Least Squares.” Preprint Series 130. Los Angeles, CA: UCLA Department of Statistics. <https://jansweb.netlify.app/publication/deleeuw-r-93-c/deleeuw-r-93-c.pdf>.
- . 2005. “Unidimensional Scaling.” In *The Encyclopedia of Statistics in Behavioral Science*, edited by B. S. Everitt and D. Howell, 4:2095–97. New York, N.Y.: Wiley.
- . 2014. “Bounding, and Sometimes Finding, the Global Minimum in Multidimensional Scaling.” UCLA Department of Statistics. <https://jansweb.netlify.app/publication/deleeuw-u-14-b/deleeuw-u-14-b.pdf>.
- . 2016. “Gower Rank.” 2016. <https://jansweb.netlify.app/publication/deleeuw-e-16-k/deleeuw-e-16-k.pdf>.
- . 2017. “Shepard Non-metric Multidimensional Scaling.” 2017.
- De Leeuw, J., P. Groenen, and P. Mair. 2016. “Full-Dimensional Scaling.” 2016. <https://jansweb.netlify.app/publication/deleeuw-groenen-mair-e-16-e/deleeuw-groenen-mair-e-16-e.pdf>.
- De Leeuw, J., and P. Mair. 2009. “Multidimensional Scaling Using Majorization: SMACOF in R.” *Journal of Statistical Software* 31 (3): 1–30. <https://www.jstatsoft.org/article/view/v031i03>.
- Dolgopolik, M. V. 2016a. “A Unifying Theory of Exactness of Linear Penalty Functions.” *Optimization* 65 (6): 1167–1202.
- . 2016b. “Smooth Exact Penalty Functions: a General Approach.” *Optimization Letters* 10: 635–48.
- . 2017. “A Unifying Theory of Exactness of Linear Penalty Functions II: Parametric Penalty Functions.” *Optimization* 66 (10): 1577–1622.
- . n.d. “Smooth Exact Penalty Functions: a General Approach.”
- Ekman, G. 1954. “Dimensions of Color Vision.” *Journal of Psychology* 38: 467–74.
- Guilford, J. P. 1954. *Psychometric Methods*. McGraw-Hill.
- Kruskal, J. B. 1964a. “Multidimensional Scaling by Optimizing Goodness of Fit to a Nonmetric Hypothesis.” *Psychometrika* 29: 1–27.
- . 1964b. “Nonmetric Multidimensional Scaling: a Numerical Method.” *Psychometrika* 29: 115–29.
- Mair, P., P. J. F. Groenen, and J. De Leeuw. 2022. “More on Multidimensional Scaling in R: smacof Version 2.” *Journal of Statistical Software* 102 (10): 1–47. <https://www.jstatsoft.org/article/view/v102i10>.
- Palubeckis, G. 2013. “An Improved Exact Algorithm for Least-Squares Unidimensional Scaling.” *Journal of Applied Mathematics*, 1–15.
- Revelle, W. 2023. *psychTools: Tools to Accompany the ‘psych’ Package for Psychological Research*. R package version 2.3.4. <https://CRAN.R-project.org/package=psychTools>.

- Rockafellar, R. T. 1970. *Convex Analysis*. Princeton University Press.
- Rothkopf, E. Z. 1957. "A Measure of Stimulus Similarity and Errors in some Paired-associate Learning." *Journal of Experimental Psychology* 53: 94–101.
- Shepard, R. N. 1962a. "The Analysis of Proximities: Multidimensional Scaling with an Unknown Distance Function. I." *Psychometrika* 27: 125–40.
- . 1962b. "The Analysis of Proximities: Multidimensional Scaling with an Unknown Distance Function. II." *Psychometrika* 27: 219–46.
- Zangwill, W. I. 1969. *Nonlinear Programming: a Unified Approach*. Englewood-Cliffs, N.J.: Prentice-Hall.