Smacof at 50: A Manual
Part 4: smacofRO - Non-metric Smacof for Rank Order Data

Jan de Leeuw - University of California Los Angeles

Started May 12 2024, Version of May 20, 2024

Contents

1 Introduction

The smacofRO program is a non-metric MDS program designed to handle a partial order over dissimilarities of pairs of objects selected from a single set. In the terminology of Coombs (1964) we are dealing with QIVa similarity data. Typically the partial order is derived from a symmetric numerical matrix of dissimilarities or from a ranking (with or without ties) of the dissimilarities.

1.1 Parameters

The smacofRO function in R has the following parameters (with default values).

```
smacofRO <- function(data,
                     ndim,
                     xold = NULL,
                     labels = NULL,
                     width = 15,
                     precision = 10,
                     itmax = 1000,
                     eps = 1e-10,
                     verbose = TRUE,
                     kitmax = 5,
                     keps = 1e-10,
                     kverbose = FALSE,
                     init = 1,
                     ties = 1)
```

- If xold is non-null then an initial configuration matrix must be provided.
- If labels is non-null then a character vector of plot labels must be provided.
- width and precision are relevant for the format of (optional) major iteration output.
- itmax and eps determine when the major iterations stop.
- If verbose = TRUE itel and stress for each major iteration are written to stdout.
- kitmax and keps determine the number of inner Guttman transform iterations between two monotone regressions.
- If kverbose = TRUE then itel and stress for each inner iteration are written to stdout.
- If init = 1 the Torgerson initial configuration is computed, if init = 2 the maximum sum initial configuration is used, if init = 3 a random initial configuration is used.
- Ties is either 1, 2, or 3 indicating if the primary, secondary, or tertiary apprach to ties should be used.

1.2 Input

The data are entered in a five-column table. Here are the first five rows for the Gruijter example, analyzed below.

```
     i j delta     weight ties
[1,] 7 6  3.20 0.02777778    1
```

```
[2,] 2 1  4.08 0.02777778    2
[3,] 3 2  4.59 0.02777778    3
[4,] 6 5  4.60 0.02777778    4
[5,] 8 4  4.67 0.02777778    5
```

The first two columns of data give the indices of the dissimilarities. Always $1 \leq j < i \leq n$. Column three gives the value of delta, which can be an observed numerical value or a rank number. The data are in non-increasing order within their column. Column four gives the weights, adding up to one over all observations. And the last column codes tieblocks. Data is allowed to have are fewer than $\frac{1}{2}n(n-1)$ rows, but each index pair should occur at most one time. Missing pairs are computationally equivalent to rows with weight equal to zero.

Typically we create the data by using the utility function smacofMakeRankOrderData(), which is part of the smacofRO package.

```r
smacofMakeRankOrderData <-
  function(delta, weights = NULL, tieblocks = TRUE)
```

Here delta is a symmetric matrix or an object of class dist containing the dissimilarities. If weights is NULL all weights are assumed to be equal.

1.3 Algorithm

- The ALS algorithm alternates a number of inner Guttman iterations with a single monotone regression (with one of the three options for ties, cf De Leeuw (1977)).
- The maximum number of inner iterations is kitmax, if the stress in an inner iteration changes less than keps the inner iterations are also stopped.
- Guttman iterations are normalized explicitly, using $\sum\sum w_{ij}d_{ij}^2(X) = 1$.
- The maximum number of outer iterations is itmax, if the stress in an outer iteration changes less than eps the outer iterations are also stopped.
- For monotone regression we use the pava algorithm from De Leeuw (2017), with some additional code for the tertiary approach to ties.

1.4 Output

The program returns a list with the following elements.

```r
  h <- list(
    nobj = nobj,
    ndim = ndim,
    snew = snew,
    itel = itel,
    xnew = xnew,
    delta = delta,
    evec = evec,
    dvec = dvec,
    wvec = wvec,
    labels = labels
  )
```

- nobj is the number of objects.
- ndim is the number of dimensions.
- snew is the final value of stress.
- itel is the number of major iterations.
- xnew is a matrix with the final configuration.
- delta is a vector with the dissimilarities.
- wvec is a vector with the weights.
- evec is a vector with the final disparities.
- dvec is a vector with the final distances.
- labels is the character vector of labels (or NULL).

The elements of delta, wvec, evec, and dvec are all in the order of the input data, i.e. sorted by increasing delta.

1.5 Plots

The output list of smacofRO can be passed to the three utility programs that make plots. In addition to the list with smacofRO output the plot functions have a number of the standard graphical parameters. smacofConfigurationPlot() uses labels (if available) to label the points. The fitlines argument of smacofShepardPlot() and smacofDistDhatplot() connects pairs of points in the plot that coincide if stress is zero. The sum of the squared lengths of the fitlines is equal to the stress.

```
smacofShepardPlot <-
  function(h,
           main = "ShepardPlot",
           fitlines = TRUE,
           colline = "RED",
           colpoint = "BLUE",
           resolution = 100,
           lwd = 2,
           cex = 1,
           pch = 16)

smacofConfigurationPlot <-
  function(h,
           main = "ConfigurationPlot",
           dim1 = 1,
           dim2 = 2,
           pch = 16,
           col = "RED",
           cex = 1.5)
```

```r
smacofDistDhatPlot <- function(h,
                               fitlines = TRUE,
                               colline = "RED",
                               colpoint = "BLUE",
                               main = "Dist-Dhat Plot",
                               cex = 1,
                               lwd = 2,
                               pch = 16)
```

2 Examples

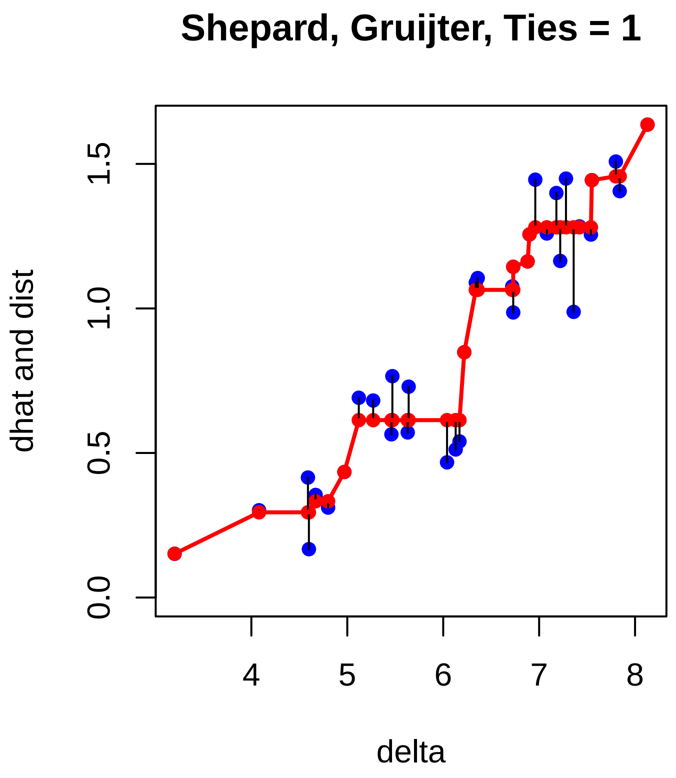
2.1 De Gruijter (1967)

In this example the number of tieblocks is 35, which means there is only a single tieblock of two observations, the other 34 observations are untied. We expect that the three different ways of handling ties will not make much of a difference.

```r
h1 <- smacofRO(gruijterData, 2, ties = 1, labels = labels, verbose = FALSE)
h2 <- smacofRO(gruijterData, 2, ties = 2, labels = labels, verbose = FALSE)
h3 <- smacofRO(gruijterData, 2, ties = 3, labels = labels, verbose = FALSE)
```

- Ties = 1 uses 88 iterations and stops at stress 0.0042180140
- Ties = 2 uses 65 iterations and stops at stress 0.0042573281
- Ties = 3 uses 69 iterations and stops at stress 0.0040850898

And indeed the solutions are practically the same, although the number of iterations needed for convergence differs somewhat between options. We only show one Shepardplot, because the other two are virtually the same.

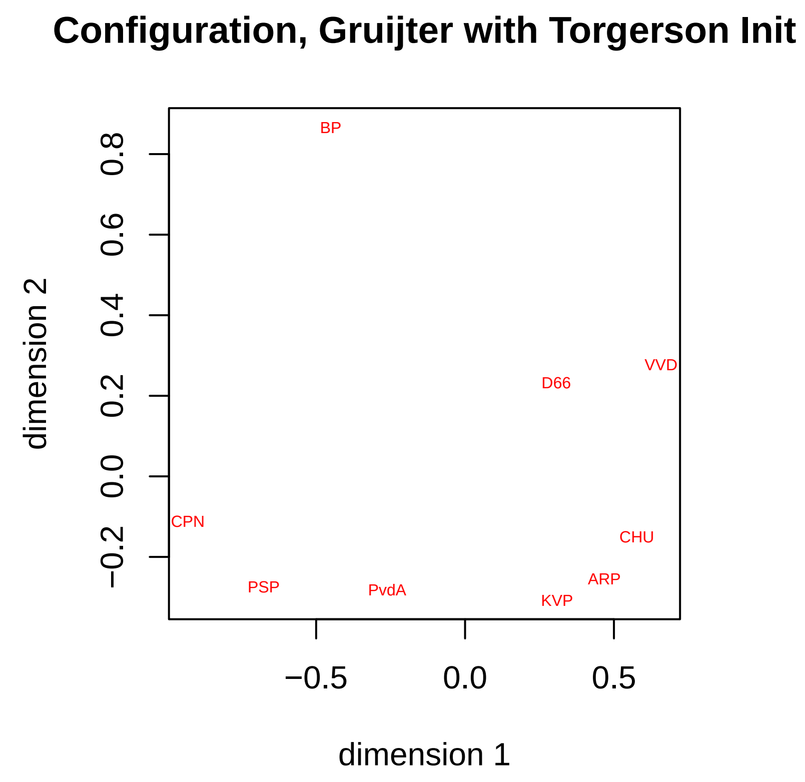


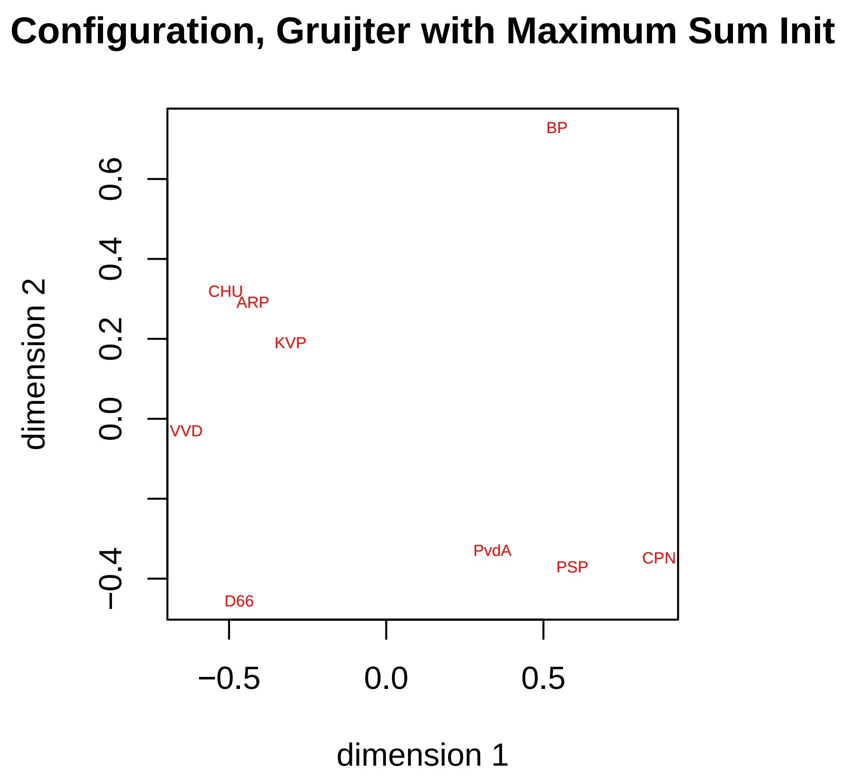
We can also use this example to show the effect of using different initial configurations.

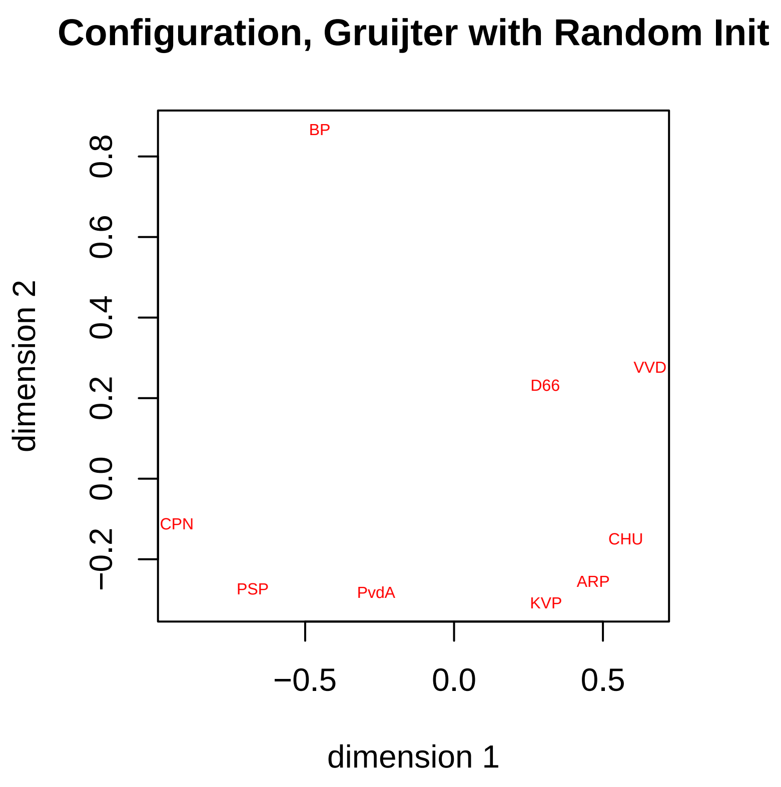
```r
h1 <- smacofRO(gruijterData, 2, init = 1, labels = labels, verbose = FALSE)
h2 <- smacofRO(gruijterData, 2, init = 2, labels = labels, verbose = FALSE)
h3 <- smacofRO(gruijterData, 2, init = 3, labels = labels, verbose = FALSE)
```

- Init = 1 uses 88 iterations and stops at stress 0.0042180140

- Init = 2 uses 76 iterations and stops at stress 0.0039894695
- Init = 3 uses 69 iterations and stops at stress 0.0060410599







Using different initial configurations in this example makes a huge difference. All three plots show the (CPN, PvdA, PSP) leftist cluster, the liberal (VVD, D'66) cluster, the protest BP outlier which is its own cluster, and the (KVP, ARP, CHU) christian democrat cluster. But in the three plots the clusters are distributed quite differently over the plane.

2.2 Ekman (1954)

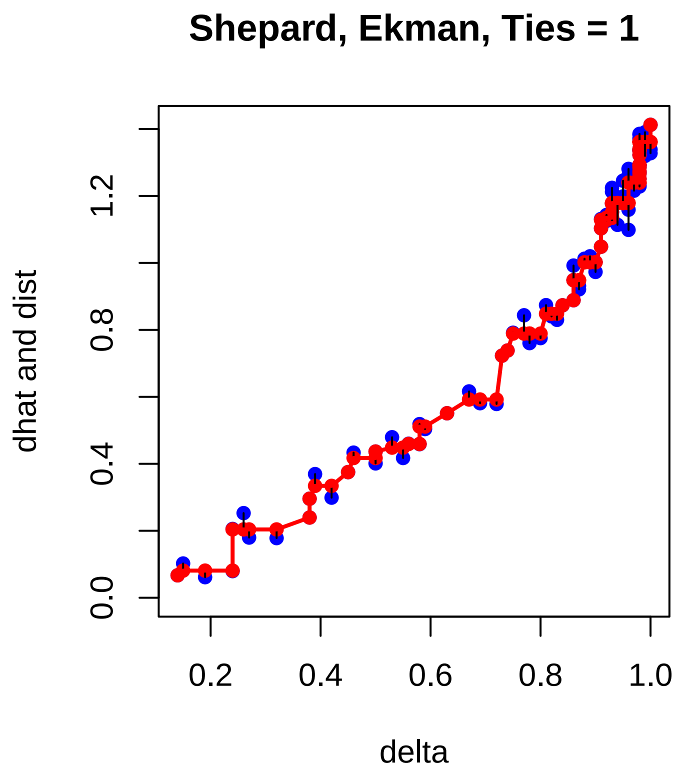
In the Ekman data there are 47 tieblocks out of 91 observations, and we expect the ties option to make some difference.

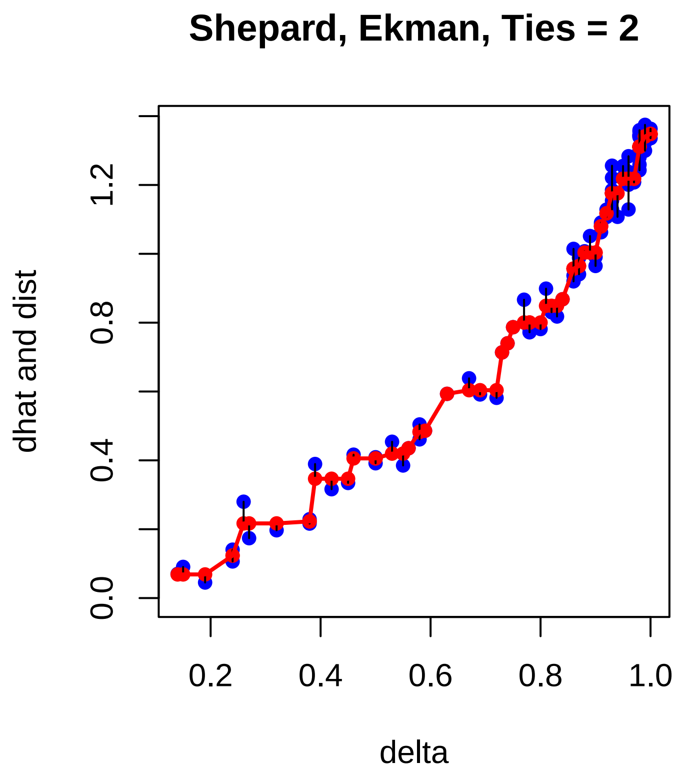
```
h1 <- smacofRO(ekmanData, 2, ties = 1, verbose = FALSE, labels = labels, itmax = 10000)
h2 <- smacofRO(ekmanData, 2, ties = 2, verbose = FALSE, labels = labels, itmax = 10000)
h3 <- smacofRO(ekmanData, 2, ties = 3, verbose = FALSE, labels = labels, itmax = 10000)
```

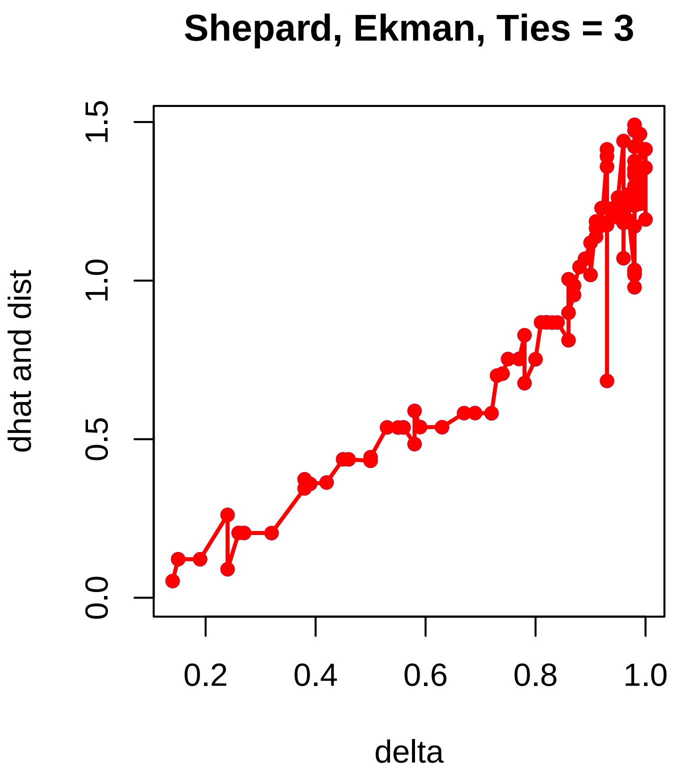
- Ties = 1 uses 40 iterations and stops at stress 0.0002668633
- Ties = 2 uses 32 iterations and stops at stress 0.0004988332
- Ties = 3 uses 1400 iterations and stops at stress 0.0000000320

Although the stress values are indeed different the solutions for ties = 1 and ties = 2 are very similar. Also note that for ties = 3 a huge number of iterations is needed to bring stress very close to zero, indicating perfect fit and very slow, and possibly sublinear, convergence. Remember that for ties = 3 there is only stress between tieblocks, no stress within tieblocks.

We now look at the Shepard plots and the configuration plots.


Shepard, Ekman, Ties = 1
dhat and dist
0.0
0.4
0.8
1.2
0.2
0.4
0.6
0.8
1.0
delta

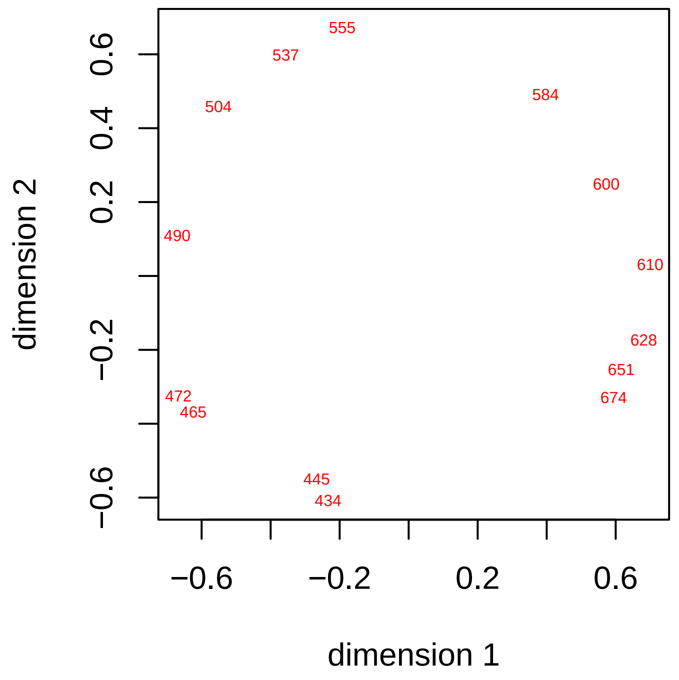

Shepard, Ekman, Ties = 2
dhat and dist
0.0
0.4
0.8
1.2
0.2
0.4
0.6
0.8
1.0
delta



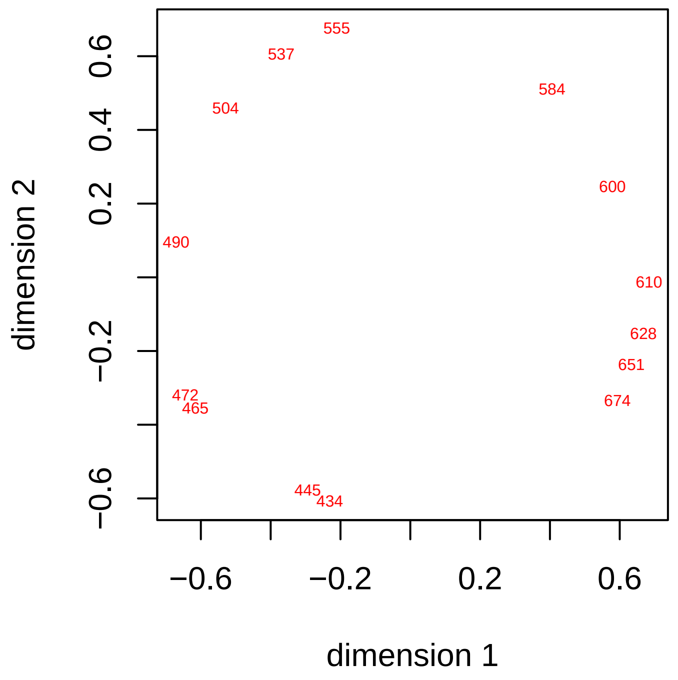
When looking at the Shepard plots we have to remember that for ties = 1 and ties = 3 we can have different disparity values for the same dissimilarity value. Thus tieblocks are represented as intervals on the vertical axis, and strictly spoken we do not have a functional relationship between delta and dhat. This is clear from the plot for the tertiary approach, which shows some of the intervals, mostly for the larger dissimilarities. It reinforces the idea that the tertiary approach is mostly useful if there are many small tieblocks, in which case it will be quite similar to the primary and secondary approach.

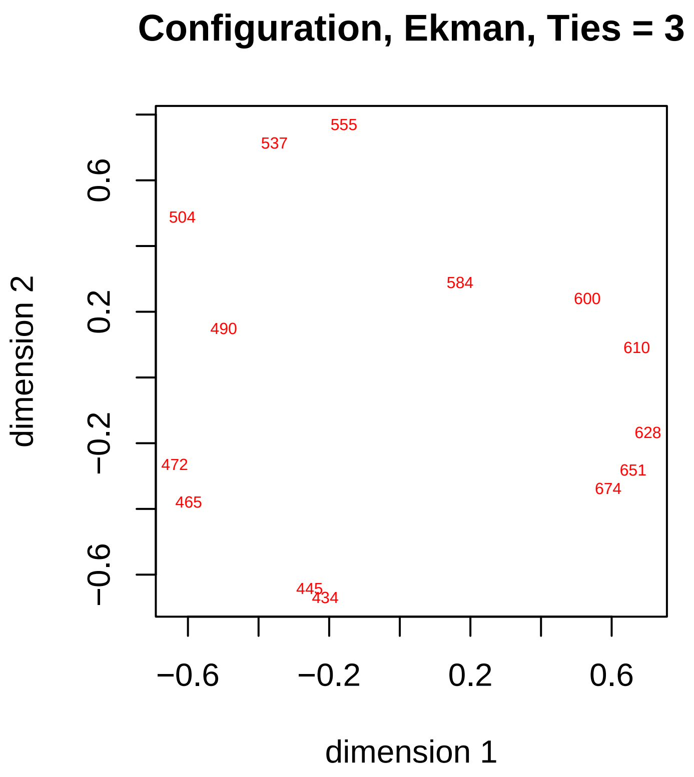
If we compare the three Shepard plots we see that ties = 1 and ties = 2 are very similar, but ties = 3 is different. There are no blue points in the plot for ties = 3, for the simple reason that they are identical to the red points. The stress is practically zero, which means that smacof is able to scale the 47 tieblock averages in the correct order. Plots for ties = 1 and ties = 2 show the black fitlines, which indicate less than perfect fit. There are no fitlines in the plot for ties = 3.

Configuration, Ekman, Ties = 1



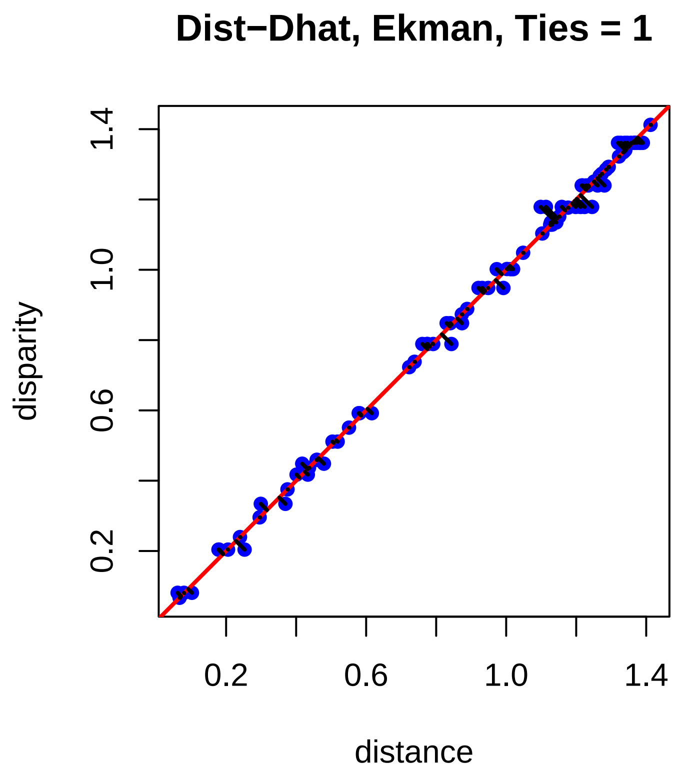
Configuration, Ekman, Ties = 2

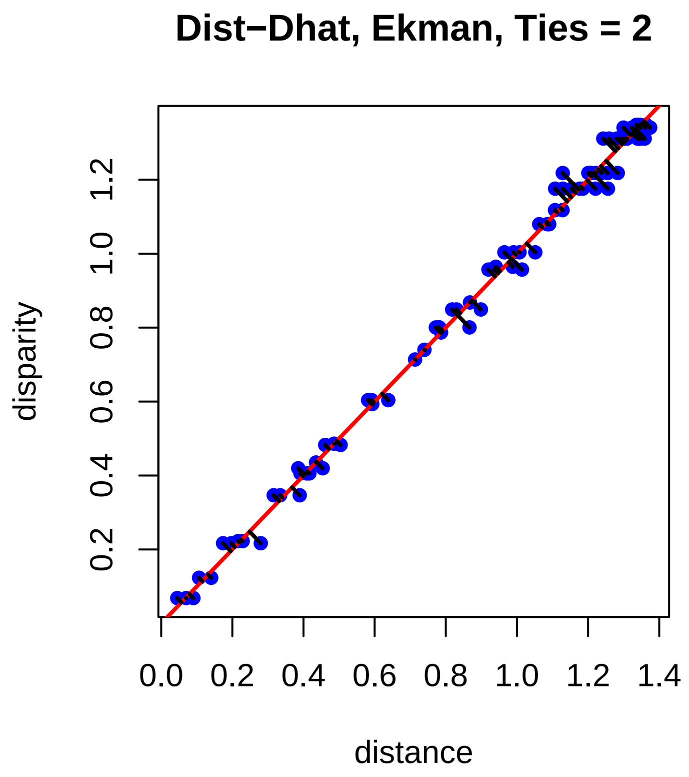


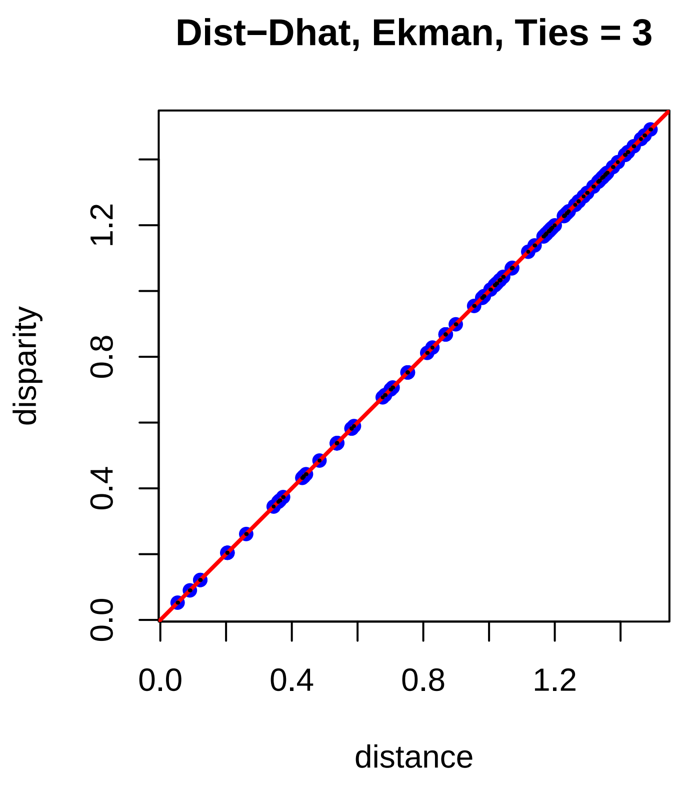


In configuration plots for ties = 1 and ties = 2 are very similar, and they show the color circle in all its glory. For ties = 3 the circle has some serious dents, which are presumably needed to force stress to zero.

For completeness we also give the dist-dhat plots corresponding with the three ties options. They again show the perfect fit with ties = 3 and the similarity of the fits for ties = 1 and ties = 2.


Dist−Dhat, Ekman, Ties = 1
disparity
distance
0.2
0.6
1.0
1.4
0.2
0.6
1.0
1.4


Dist−Dhat, Ekman, Ties = 2
disparity
distance
0.2
0.4
0.6
0.8
1.0
1.2
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4

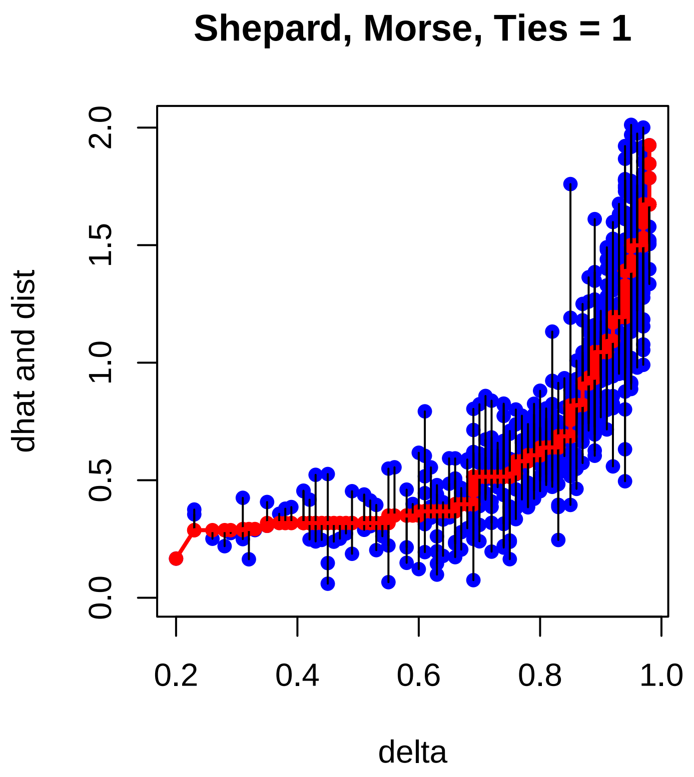


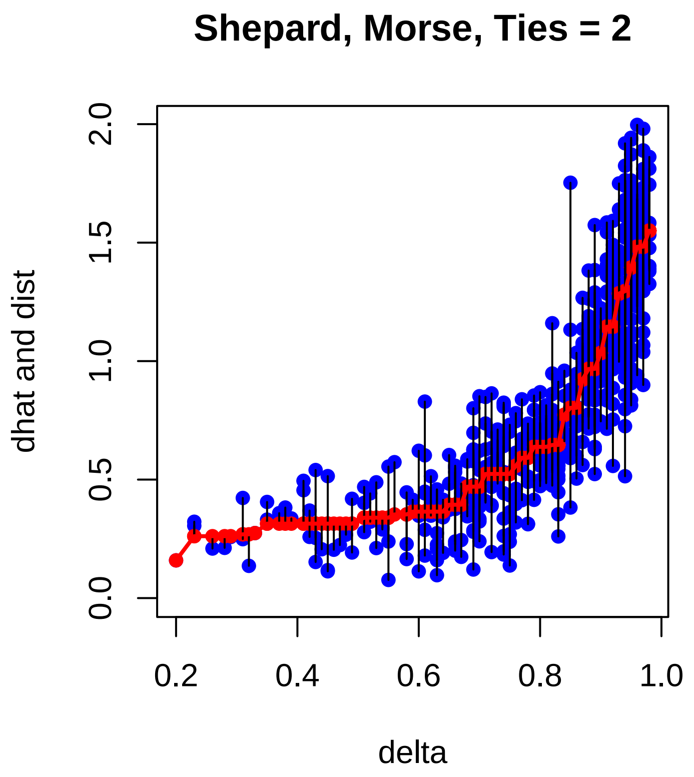
2.3 Rothkopf (1957)

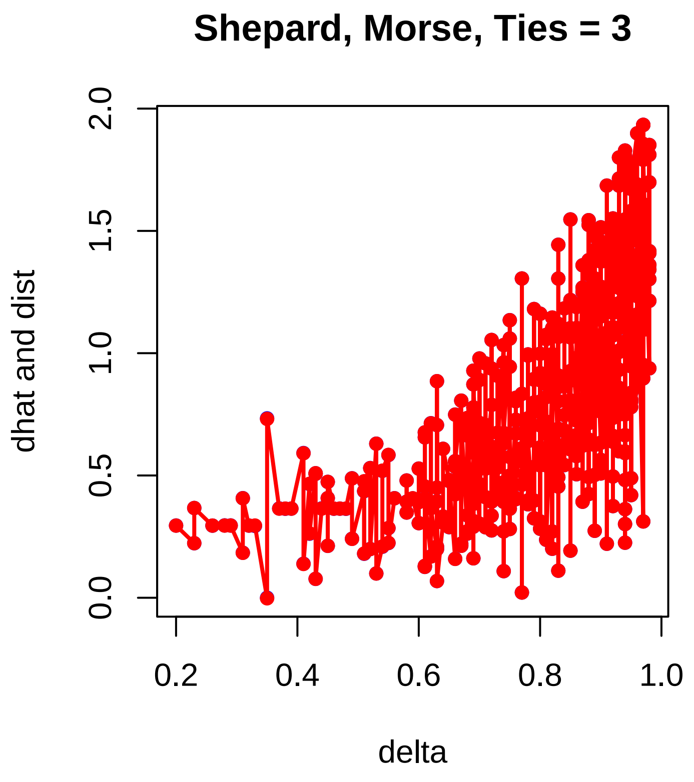
In the morse code data there are 68 tieblocks out of 630 observations. Thus almost all observations are tied.

```r
h1 <- smacofRO(morseData, 2, ties = 1, verbose = FALSE, itmax = 10000, labels = labels)
h2 <- smacofRO(morseData, 2, ties = 2, verbose = FALSE, itmax = 10000, labels = labels)
h3 <- smacofRO(morseData, 2, ties = 3, verbose = FALSE, itmax = 10000, labels = labels)
```

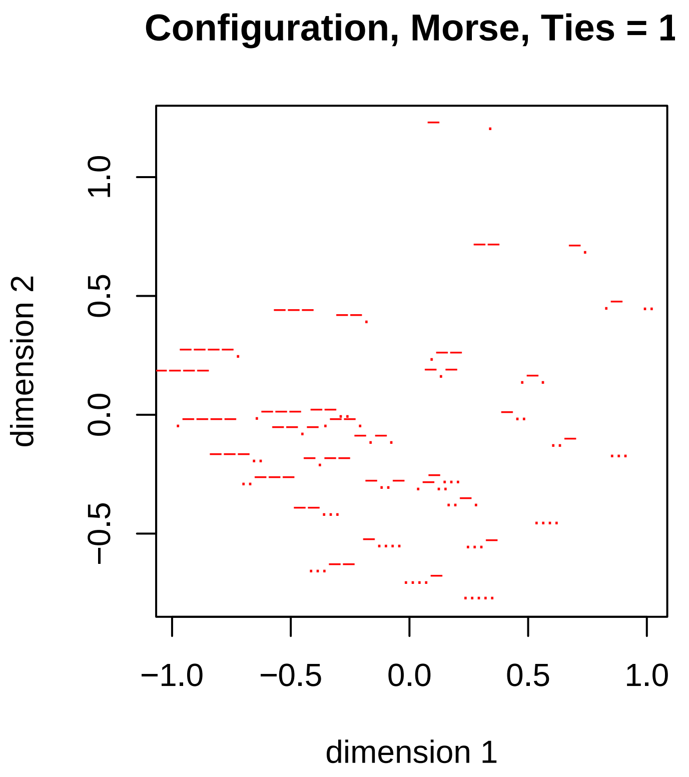
- Ties = 1 uses 34 iterations and stops at stress 0.0163278414
- Ties = 2 uses 30 iterations and stops at stress 0.0203202336
- Ties = 3 uses 1129 iterations and stops at stress 0.0000000226

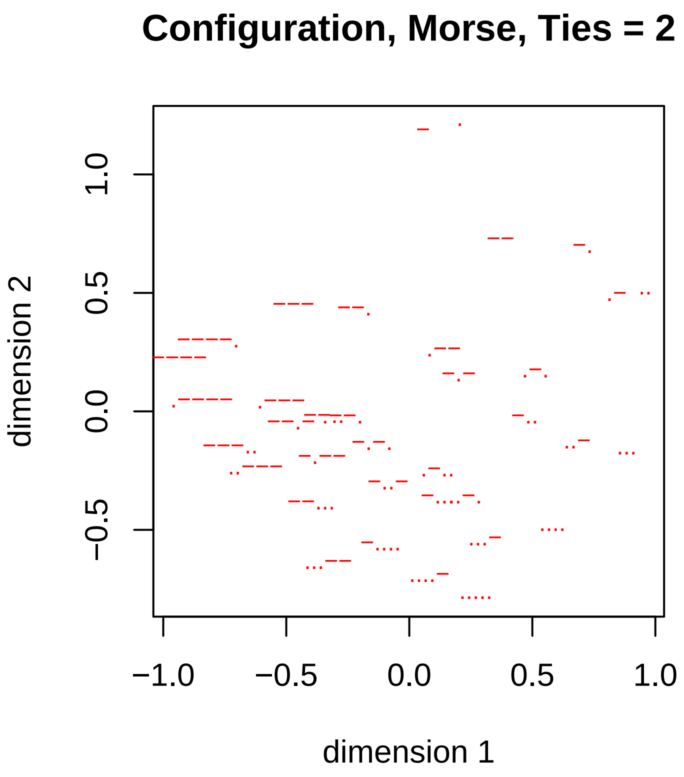

Shepard, Morse, Ties = 1
dhat and dist
delta
0.0
0.5
1.0
1.5
2.0
0.2
0.4
0.6
0.8
1.0

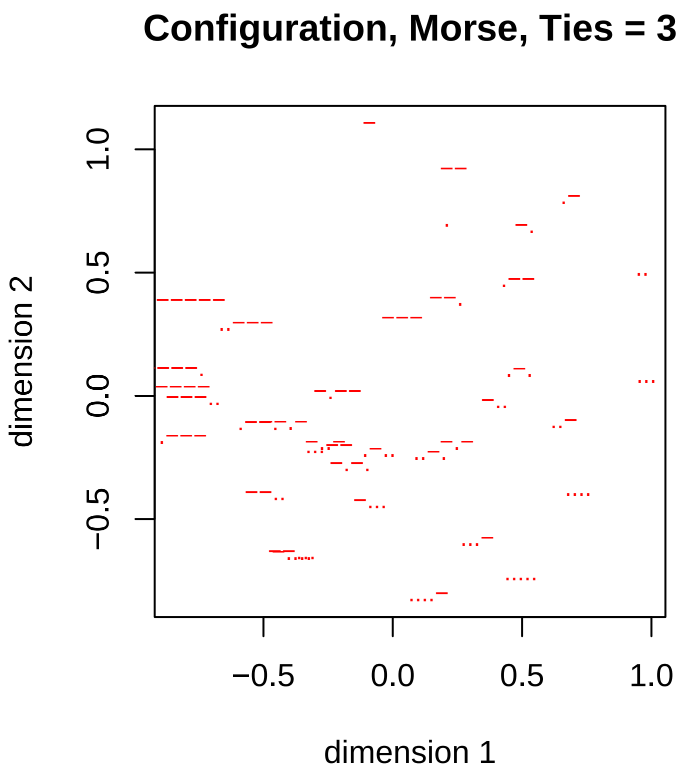

Shepard, Morse, Ties = 2
dhat and dist
delta
0.0
0.5
1.0
1.5
2.0
0.2
0.4
0.6
0.8
1.0



We see the same types of results in the iterations and the Shepard plots as before. For ties = 3 stress is again close to zero and we require a huge number of iterations. The Shepard plot for ties = 3 is messy, because there are so many intervals and intervals can be wide because within tieblocks there is no stress.







Configuration plots for ties = 1 and ties = 2 are similar and show parallel lines for morse codes with one, two, three, four, or five symbols. On those five parallel lines codes are ordered from all dashes to all dots. For ties = 3 that nice raster structure is again disturbed.

We also run morse with different values of kitmax, the number of inner Guttman iterations. The results are

```
h1 <- smacofRO(morseData, 2, kitmax = 1, verbose = FALSE, itmax = 10000)
h2 <- smacofRO(morseData, 2, kitmax = 5, verbose = FALSE, itmax = 10000)
h3 <- smacofRO(morseData, 2, kitmax = 10, verbose = FALSE, itmax = 10000)
```

- Kitmax = 1 uses 138 major iterations and stops at stress 0.0163278417
- Kitmax = 5 uses 34 major iterations and stops at stress 0.0163278414
- Kitmax = 10 uses 22 major iterations and stops at stress 0.0163278414

We see that the resulting stress values are pretty much the same, but the number of outer iterations differs. This may be significant, because it means that using only one Guttman iteration means doing 138 monotone regressions, while having ten Guttman iterations per major iteration only uses 22 monotone regressions. On the other hand for kitmax 1, 5, 10 we have to do respectively 138, 170, and 220 inner Guttman iterations. Which of the three options is faster will depend on the relative time required by Guttman transforms and monotone regressions. The timing for the current implementation, with 100 replications in microbenchmark (Mersmann (2023)), is as follows.

```
## Warning in microbenchmark(smacofRO(morseData, 2, kitmax = 1, verbose = FALSE, :
## less accurate nanosecond times to avoid potential integer overflows

## Unit: milliseconds
##                                                                     expr       min
##   smacofRO(morseData, 2, kitmax = 1, verbose = FALSE, itmax = 10000) 1336.8770
##   smacofRO(morseData, 2, kitmax = 5, verbose = FALSE, itmax = 10000)  484.2788
##  smacofRO(morseData, 2, kitmax = 10, verbose = FALSE, itmax = 10000)  418.8768
##         lq      mean    median        uq       max neval
##  1360.1857 1372.0015 1368.2405 1379.6018 1506.5266   100
##   494.0313  498.0338  496.7758  500.0796  538.6924   100
##   429.6171  434.6207  432.5188  437.1846  454.1709   100
```

It seems advantageous to have a fairly large number of Guttman transforms between monotone regressions. In smacofRO for the Ekman data (with default options) kitmax = 10 is about three times faster than kitmax = 1. This may very well be example and implementation dependent.

References

Coombs, C. H. 1964. *A Theory of Data*. Wiley.

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. "Correctness of Kruskal's Algorithms for Monotone Regression with Ties." *Psychometrika* 42: 141–44.

———. 2017. "Exceedingly Simple Monotone Regression (with Ties)." 2017. https://jansweb.netlify.app/publication/deleeuw-e-17-h/deleeuw-e-17-h.pdf.

Ekman, G. 1954. "Dimensions of Color Vision." *Journal of Psychology* 38: 467–74.

Mersmann, O. 2023. *microbenchmark: Accurate Timing Functions*. https://CRAN.R-project.org/package=microbenchmark.

Rothkopf, E. Z. 1957. "A Measure of Stimulus Similarity and Errors in some Paired-associate Learning." *Journal of Experimental Psychology* 53: 94–101.