An algorithm for Replicated Nonmetric MDS that uses SMACOF and LSMT

1. The (metric) replicated multidimensional scaling (RMDS) situation involves fitting *D,* an (n x n) matrix of Euclidean distances, to several (n x n) matrices O_k of dissimilarity data, where each matrix of data is obtained from subject k, there being m (k=1,m) subjects.
2. *D* has distances derived from *X*, an (n x r) matrix of coordinates of a configuration of points in an r-dimensional Euclidean space.
3. The *nonmetric* RMDS situation involves fitting *D* to several monotonically transformed matrices $t_k[O_k]$, there being a different monotonic transformation t_k for each subject k.
4. In these notes we outline an algorithm for optimizing Stress, our index of fit, which is defined as

$$S = \sqrt{\sum_{k=1}^{m} \left[\frac{\sum_i \sum_j (d_{ij} - t_k[o_{ijk}])^2}{\sum_i \sum_j t_k[o_{ijk}]^2} \right]} .$$

5. We assume the denominator is a constant, which we denote A_k. Furthermore, we assume it is the same constant *A* for all k=1,m. Thus we can restate the problem as one of optimizing:

$$S^2 = \frac{1}{A} \left[\sum_k \sum_i \sum_j (d_{ij} - t_k[o_{ijk}])^2 \right]$$

6. We note that the numerator involves summing differences twice, so we can further restate the problem as one of optimizing

$$S^2 = \frac{2}{A} \left[\sum_k \sum_{j<i} (d_{ij} - t_k[o_{ijk}])^2 \right] .$$

7. To be explicit, S^2 is optimized by estimating its parameters. These parameters include the Euclidean coordinates X and the m separate transformations t_k. That is, we wish to minimize S^2 ove X and t_k:

$$S^2[X, t_k] = \frac{2}{A}\left\{\sum_{k=1}^{m}\sum_{i<j}^{n(n-1)/2}\left[\left(\sum_{a=1}^{r}(x_{ia}-x_{ja})^2\right)^{1/2} - t_k[o_{ijk}]\right]^2\right\}.$$

We minimize S^2 by constructing an alternating-improved-squares (AIS) algorithm consisting of two phases.

- In the first phase, which we call the "optimal scaling" phase, we assume that we know X and we wish to solve for the t_k which conditionally minimizes $S^2[t_k|X]$. We use Kruskal's least squares monotonic transformation to do this.
- In the second phase, which we call the "model estimation" phase, we assume that we know t_k and that we wish to solve for the X that conditionally reduces $S^2[X|t_k]$. The SMACOF algorithm is used here. Note that SMACOF is not least squares: It reduces $S^2[X|t_k]$, but doesn't minimize it.

We call our algorithm alternating-improved-squares rather than alternating-least-squares, since the model estimation phase is not least squares, but improved squares.

We turn to each of these phases now.

Optimal Scaling

We note that

$$S^2[t_k|X] = \frac{2}{A}\sum_k \sum_{i<i} (d_{ij} - t_k[o_{ijk}])^2 = \frac{2}{A}\left[\sum_k S_k^2\right] ,$$

Since Stress can be separated into a sum of independent terms, it is the case that the overall function can be minimized by minimizing each of its terms. Thus, S can be minimized by minimizing each S_k. In turn, each S_k can be minimized as described in the notes for Kruskal's least squares transformation, using the distances D, and the data O_k for subject k.

Model Estimation

1. Following the development given in previous notes for SMACOF, we note that

$$S^2[X|t_k] = \frac{2}{A}\sum_k \sum_{i<i} (d_{ij} - t_k[o_{ijk}])^2$$

can be expanded to obtain

$$\begin{aligned} S^*[X|t_k] &= \frac{S^2[X|t_k]A}{2} \\ &= \sum_k \left[\sum_{i<j} t_k[o_{ijk}]^2 + \sum_{i<j} d_{ij}^2 - 2\sum_{i<j} d_{ij}t_k[o_{ijk}]\right] \\ &= \sum_k \sum_{i<j} t_k[o_{ijk}]^2 + \sum_k \sum_{i<j} d_{ij}^2 - 2\sum_k \sum_{i<j} d_{ij}t_k[o_{ijk}] \\ &= \frac{mA}{2} + \sum_k \sum_{i<j} d_{ij}^2 - 2\sum_k \sum_{i<j} d_{ij}t_k[o_{ijk}] \\ &= [a + b - 2c] \end{aligned}$$

where the three terms are equal to a, b, and c, respectively.

2. The function can now be rewritten in matrix form, where

$$b = \sum_{k}^{m} \sum_{i<j} d_{ij}^{\,2} = m \sum_{i<j} d_{ij}^{\,2} = m\left(\sum_{i<j} (x_i - x_j)(x_i - x_j)' \right) = m(\mathrm{tr}[XVX']) \quad ,$$

and

$$\begin{aligned} c &= \sum_{k} \sum_{i<j} d_{ij} t_k[o_{ijk}] = \sum_{i<j} d_{ij} \sum_{k} t_k[o_{ijk}] \\ &= \sum_{i<j} \left(\frac{\sum_{k} t_k[o_{ijk}]}{d_{ij}} \right) d_{ij}^2 = \sum_{i<j} \left(\frac{\sum_{k} t_k[o_{ijk}]}{d_{ij}} \right) (x_i - x_j)(x_i - x_j)' \\ &= \mathrm{tr}\left[X \left(\sum_{k} R_k \right) X' \right] \end{aligned}$$

where X is an (n x n) matrix of coordinates of points on all n dimensions, rather than the (n x r) matrix we have become accustomed to, with the n-dimensional coordinates for point i being in row i. In these equations V and R_k are (n x n) matrices with elements

$$v_{ij} = \begin{cases} -1 & (i \neq j) \\ n-1 & (i = j) \end{cases} \quad \text{and} \quad r_{ijk} = \begin{cases} \dfrac{-t[o_{ijk}]}{d_{ij}} & (i \neq j) \\ 0, \text{ if } d_{ij} = 0 & \\ -\sum\limits_{h \neq i}^{n} r_{ihk} & (j = i) \end{cases}$$

Stress can now be written as the matrix equation:

$$S^* = a + m(\mathrm{tr}[XVX']) - 2\mathrm{tr}\left[X \left(\sum_{k} R_k \right) X' \right] \quad .$$

3. Following de Leeuw, a majorizing function T^* for S^* is given by

$$T^* = a + m(\mathrm{tr}[XVX']) - 2\mathrm{tr}\left[X\left(\sum_k R_k^y \right) Y' \right]$$

where $\boldsymbol{R_k^y}$ is a function of the distances between points at point Y in optimization space, which is written similarly to R above:

$$r_{ijk} = \begin{cases} \dfrac{-t[o_{ijk}]}{d_{ij}[y]} & (i \neq j) \\ 0, \text{ if } d_{ij}[y] = 0 & \\ \quad n & \\ -\sum_{h \neq i} r_{ihk} & (j = i) \end{cases},$$

where $d_{ij}[y] = \sqrt{(y_i - y_j)(y_i - y_j)'}$

4. Finally, to minimize T^* we solve

$$\frac{\partial T^*}{\partial X} = 2mVX - 2\left(\sum_k R_k^y \right) Y = 0 \ .$$

Noting that V has rank n-1 (its row-sums are all zero), we use the Moore-Penrose inverse (left pseudo-inverse) to solve the equation:

$$X = (V'V)^{-1} V' \left(\sum_k R_k^y \right) Y \ .$$

5. The algorithm is:

$$X = \frac{1}{nm} \left(\sum_k R_k^y \right) Y$$

where Y is our current (nxr) matrix of coordinates of n points on r dimensions, X is the (nxr) matrix of coordinates on the next iteration, and where $\boldsymbol{R_k^y}$ has elements defined above.