Notes on the C Version of Smacof

Jan de Leeuw - University of California Los Angeles

Started October 10 2023, Version of November 07, 2023

Abstract

TBD

Contents

Note: This is a working paper which will be expanded/updated frequently. All suggestions for improvement are welcome. All Rmd, tex, html, pdf, R, and C files are in the public domain. Attribution will be appreciated, but is not required. The files can be found at https://github.com/deleeuw/mdsStruct.

1 Introduction

The loss function in (metric, least squares, Euclidean, symmetric) Multidimensional Scaling (MDS) is

$$\sigma(X) := \frac{1}{2}\mathop{\sum\sum}_{1\leq j<i\leq n} w_{ij}(\delta_{ij}-d_{ij}(X))^2.$$

This assumes symmetry and it uses all elements below the diagonal of both W, Δ, and $D(X)$. For missing data we set $w_{ij}=0$.

2 Example

Here is a small input example.

3 smacofStructure

We use this example as input for the R version of *smacofSort()*, which results in the *smacofStructure*

The *dist* element in the data frame is zero, because we have not computed distances yet. Given a configuration X the *row* and *col* elements in the data frame allow from straightforward computation of distances. In the case of nonmetric MDS, or more generally in MDS with transformed dissimilarities, the *dhat* column will be filled as well. The fact that preprocessing with *smacofSort()* gives the ordered dissimilarities, as well as the tie blocks, is especially useful in the ordinal case, both with monotone polynomials and monotone splines. In the metric (ratio) case there is no need for the *dhat* column.

It is obvious how this *smacofStructure* can be adapted if there are multiway data.If we have row-conditional or matrix-conditional data, then we use one of these smacofStructures for each row or each matrix.

4 Appendix: Code

We use *qsort* to sort the rows of the input data frame by increasing delta. The sorting is done in C, the R version is a wrapper around the compiled C code. The C code also contains a main which analyzes the same small example as we have used in the text. Compile with "clang -o runner -O2 smacofSort.c" and then start "runner" in the shell. The C code uses the .C() interface in R, which can be improved using .Call(), but probably in this case with little gain.

5 smacofMaximumSum

Maximize

$$\sum\sum_{1\leq j<i\leq n} w_{ij}\delta_{ij}^2 d_{ij}^2(X)$$

over X with tr $(X'X)^2 = 1$. This gives $BX = X\Lambda$, with

$$B = \sum\sum_{1\leq j<i\leq n} w_{ij}\delta_{ij}^2 A_{ij}$$

There is also a non-metric version. Maximize over tr $(X'X)^2$

$$\sum\sum_{1\leq j<i\leq n}\sum\sum_{1\leq k<l\leq n} w_{ij,kl}\text{sign}(\delta_{ij}-\delta_{kl})(d_{ij}^2(X)-d_{kl}^2(X))$$

which simplifies to

$$2\sum\sum_{1\leq j<i\leq n} d_{ij}^2(X)\sum\sum_{1\leq k<l\leq n} w_{ij,kl}\text{sign}(\delta_{ij}-\delta_{kl})$$

Simplifies more $w_{ij,kl} = w_{ij}$ or $_{ij,kl} = w_{ij}w_{kl}$

Also note

$$\rho(X) = \sum\sum_{1\leq j<i\leq n} w_{ij}\delta_{ij}d_{ij}(X) = \sum\sum_{1\leq j<i\leq n}\frac{w_{ij}}{\delta_{ij}d_{ij}(X)}\delta_{ij}^2 d_{ij}^2(X) \approx \sum\sum_{1\leq j<i\leq n}\frac{w_{ij}}{\delta_{ij}^2}\delta_{ij}^2 d_{ij}^2(X) = \eta^2(X)$$

6 smacofElegant

7 smacofAdjustDiagonal

8 smacofImpute

9 smacofHildreth

Hildreth (1957)

Consider the QP problem of minimizing $f(x) = \frac{1}{2}(x-y)'W(x-y)$ over all $x \in \mathbb{R}^n$ satisfying $Ax \geq 0$, where A is $m \times n$. Wlg we can assume $a_j'Wa_j = 1$. The Lagrangian is

$$\mathcal{L}(x,\lambda) = \frac{1}{2}(x-y)'W(x-y) - \lambda' Ax.$$

$$\max_{\lambda\geq 0}\mathcal{L}(x,\lambda) = \begin{cases}\frac{1}{2}(x-y)'W(x-y) & \text{if } Ax \geq 0,\\ +\infty & \text{otherwise.}\end{cases}$$

and thus

$$\min_{x\in\mathbb{R}^n}\max_{\lambda\geq 0}\mathcal{L}(x,\lambda) = \min_{Ax\geq 0}\frac{1}{2}(x-y)'W(x-y).$$

By duality
$$\min_{x\in\mathbb{R}^n}\max_{\lambda\geq 0}\mathcal{L}(x,\lambda)=\max_{\lambda\geq 0}\min_{x\in\mathbb{R}^n}\mathcal{L}(x,\lambda).$$
The inner minimum over x is attained for
$$x=y+W^{-1}A'\lambda,$$
and is equal to
$$\min_{x\in\mathbb{R}^n}\mathcal{L}(x,\lambda)=-\frac{1}{2}\lambda'AW^{-1}A'\lambda-\lambda'Ay.$$
Thus
$$\max_{\lambda\geq 0}\min_{x\in\mathbb{R}^n}\mathcal{L}(x,\lambda)=-\frac{1}{2}\min_{\lambda\geq 0}\left\{(Wy+A'\lambda)'W^{-1}(Wy+A'\lambda)-y'Wy\right\}$$
We minimize $h(\lambda)=(Wy+A'\lambda)'W^{-1}(Wy+A'\lambda)$ with coordinate descent. Let $\lambda_j(\epsilon)=\lambda+\epsilon e_j$. Then
$$h(\lambda_j(\epsilon))=(Wy+A'\lambda+\epsilon a_j)'W^{-1}()=\epsilon^2a_j'W^{-1}a_j+2\epsilon a_j'W^{-1}(y+W^{-1}A'\lambda)+$$
which must be minimized over $\epsilon\geq-\lambda_j$. So the minimum is attained at
$$\epsilon=-\frac{a_j'W^{-1}x}{a_j'W^{-1}a_j}$$
with $x=y+W^{-1}A'\lambda$ (cf ...), provided ... satisfies .. Otherwise $\epsilon=-\lambda_j$. Now update both λ and x, and go to the next j.

10 smacofDykstra

11 smacofJacobi

taken from De Leeuw (2017)

partial jacobi

12 smacofIndividualDifferenceModels.c

$$\begin{aligned}
X_k&=X, &(1)\\
X_k&=X\Lambda_k \text{ with } \Lambda_k \text{ diagonal}, &(2)\\
X_k&=XC_k, &(3)\\
X_k&=X\Lambda_kY' \text{ with } \Lambda_k \text{ diagonal}, &(4)\\
X_k&=XC_kY', &(5)\\
X_k&=X\Lambda_kY_k' \text{ with } \Lambda_k \text{ diagonal} &(6)\\
X_k&=XC_kY' \text{ with } C_k=\sum_{s=1}^r z_{ks}H_s. &(7)
\end{aligned}$$

13 smacofSort

14 Code

14.1 smacofSort.R

14.2 smacofSort.c

```
```

References

De Leeuw, J. 2017. “Jacobi Eigen in R/C with Lower Triangular Column-wise Compact Storage.” 2017.

Hildreth, C. 1957. “A Quadratic Programming Procedure.” *Naval Research Logistic Quarterly* 14 (79–85).