Least Squares Unidimensional Scaling

Jan de Leeuw - University of California Los Angeles

Started December 12 2022, Version of July 22, 2023

Abstract

This paper we discuss characteristics of the least squares loss function for unidimensional scaling. Some properties of both its local and global minima and the corresponding minimizers.

Contents

Note: This is an unfinished working paper which will be expanded/updated. All suggestions for improvement are welcome.

1 Introduction

A statistical technique is a map from data-space into representation-space. In the technique discussed in this paper the **data** are a pair of matrices containing, respectively, **weights** and **dissimilarities**. Both weights W and dissimilarities Δ are elements of $\mathbb{H}^{n\times n}$, the set of symmetric, non-negative, and hollow (zero diagonal) matrices of order n. The **representations** are elements of $\underline{\mathbb{R}}^n$, the $p(n-1)$dimensional linear space of all vectors with elements adding up to zero.

Unidimensional Scaling (UDS) of n objects is defined in this paper as the minimization of the loss function $\sigma(\bullet)$, defined for all $X \in \underline{\mathbb{R}}^n$ by

$$\sigma(x) := \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(\delta_{ij} - d_{ij}(X))^2, \tag{1}$$

over $X \in \underline{\mathbb{R}}^n$, where

$$d_{ij}(x) := |x_i - x_j| \tag{2}$$

is the distance between elements i and j of x. Following Kruskal (1964a) and Kruskal (1964b) we will refer to this loss function as **stress**, so that equation (1) defines the stress of X. By the way, we use the symbol $:=$ for definitions.

We assume the MDS problem is **non-trivial**, by which we mean $w_{ij}\delta_{ij} > 0$ for at least one pair (i, j) with $i \neq j$. We also assume that W is **irreducible**. A matrix W of order n is **reducible** if there exists a permutation matrix P and two matrices W_1 and W_2 of orders n_1 and n_2, with $n_1 > 0$, $n_2 > 0$, and $n_1 + n_2 = n$, such that

$$PWP' = \begin{bmatrix} W_1 & 0 \\ 0 & W_2 \end{bmatrix}, \tag{3}$$

If W is reducible then the UDS problem separates into two smaller UDS problems. Thus we can assume without any real loss of generality that W is not reducible, i.e. **irreducible**.

An UDS problem is **unweighted** if $w_{ij} = 1$ for all $i \neq j$. An UDS problem is **positive** if $w_{ij}\delta_{ij} > 0$ for all $i \neq j$. Thus in a positive unweighted UDS problem all off-diagonal dissimilarities are positive.

2 Theory of UDS

2.1 Some Shorthand

We expand the square in definition (1) and use some Multidimensional scaling (MDS) notation originally introduced by De Leeuw (1977) and De Leeuw and Heiser (1977) to simplify our expressions. The same notation is used in the MDS textbooks of Borg and Groenen (2005) and Borg, Groenen, and Mair (2018), and in the documentation for the R package smacof (De Leeuw and Mair (2009), Mair, Groenen, and De Leeuw (2022)).

First assume, without loss of generality, that

$$\frac{1}{2}\sum_{i=1}^n\sum_{j=1}^n w_{ij}\delta_{ij}^2 = 1. \tag{4}$$

Now define

$$\rho(x) := \frac{1}{2}\sum_{i=1}^n\sum_{j=1}^n w_{ij}\delta_{ij}|x_i - x_j|, \tag{5}$$

and

$$\eta^2(x) := \frac{1}{2}\sum_{i=1}^n\sum_{j=1}^n w_{ij}(x_i - x_j)^2. \tag{6}$$

Both ρ and $\eta := \sqrt{\eta^2}$ are convex and homogeneous of degree one, equal to zero on $\underline{\mathbb{R}}^n$ if and only if $x = 0$. Thus they are norms.

With definitions (5) and (6) we can write

$$\sigma(x) = 1 - 2\rho(x) + \eta^2(x). \tag{7}$$

The quadratic form in definition (6) can be simplified by defining the matrix V as

$$v_{ij} := \begin{cases} -w_{ij} & \text{if } i \neq j, \\ \sum_{j=1}^{n} w_{ij} & \text{if } i = j. \end{cases} \tag{8}$$

Then

$$\eta^2(x) = x'Vx, \tag{9}$$

and thus

$$\sigma(x) = 1 - 2\rho(x) + x'Vx. \tag{10}$$

V has non-positive off-diagonal elements and its rows and columns add up to zero (i.e. V is **doubly-centered**). By a result usually attributed to Taussky (1949) the irreducibility of W implies that V is positive semi-definite of rank $n-1$, with only vectors proportional to e in its null space. If $x \in \underline{\mathbb{R}}^n$ then $x'Vx \geq 0$, with $x'Vx = 0$ if and only if $x = 0$. Thus V is positive definite on $\underline{\mathbb{R}}^n$.

Note that in the unweighted case we have $V = nJ$, where $J := I - n^{-1}ee'$ is the **centering matrix** that maps $\mathbb{R}^n$ into $\underline{\mathbb{R}}^n$. Thus in the unweighted case

$$\sigma(x) = 1 - 2\rho(x) + nx'x. \tag{11}$$

$V = D - W$, with $W \geq 0$. Now $D - W \gtrsim 0$ or $U := D^{-\frac{1}{2}}WD^{-\frac{1}{2}} \leq I$.Also $(D-W)e = 0$ and thus $D^{\frac{1}{2}}e$ is an eigenvector with eigenvalue $+1$. $D^{-\frac{1}{2}}VD^{-\frac{1}{2}} = I - U$ If $U = K\Lambda K'$ then $I + U + U^2 + \cdots + U^p = K(I + \Lambda + \Lambda^2 + \Lambda^p)K' = K\Phi K'$ with $\phi_1 = p$ and $\phi_i = \frac{1-\lambda_i^{p+1}}{1-\lambda_i}$ Thus $(I + U + U^2 + \cdots + U_p) - pk_1k_1'$ converges to $K'(I-\Lambda)^+K = (I-U)^+$. Now $V^+ = D^{-\frac{1}{2}}(I-U)^+D^{-\frac{1}{2}}$

$V_\epsilon = D_\epsilon - W$ where $D_\epsilon = D + \epsilon I$ ## The Shape of Stress

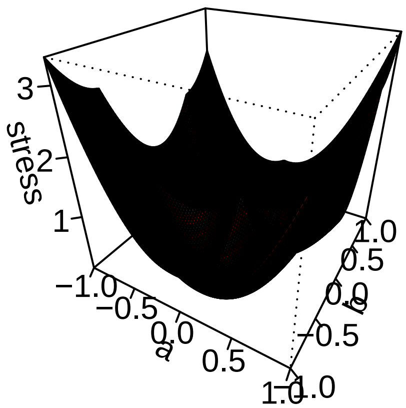
2.1.1 Global Shape

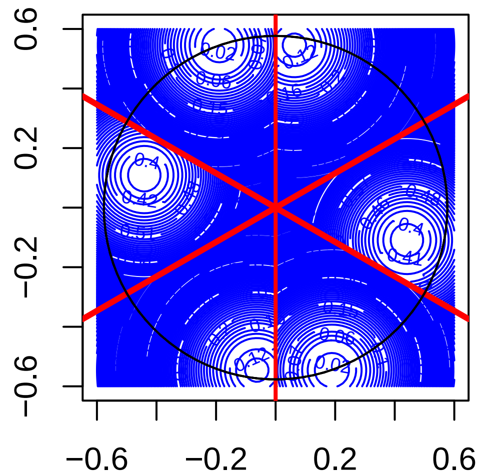
From (1) we see that stress is a continuous function of x, bounded below by zero and unbounded above. It is also **even** with $\sigma(x) = \sigma(-x)$ for all x, and **coercive**, which means that $\sigma(x) \rightarrow \infty$ as $\|x\| \rightarrow \infty$. Combining these characteristics indicates that stress, at least as seen from afar, is bowl-shaped.

```
par(mfrow = c(1, 2))
pplot(delta1)
```

```
## NULL
```

```
cplot(delta1)
```





On any half-line starting from the origin stress is a convex quadratic.

Not bowl shaped Local maximum at zero. Only local maximum.

2.1.2 Stress is DC

In optimization we are always looking for convexity, wherever we can find it. Unfortunately stress is clearly not convex.

From equations (10) and (11) we see that stress is a *DC-function*, i.e. the difference of two convex functions. The function $1 + \rho$ is non-smooth and not differentiable on any hyperplane with $x_i = x_j$ for at least one pair (i, j). The function η^2 is quadratic, strictly convex on $\mathbb{R}^n$, and infinitely many times continuously differentiable.

It follows from the DC property that stress is locally Lipschitz and almost everywhere twice differentiable (Hiriart-Urruty (1988), Bacak and Borwein (2011)).

2.2 Building Blocks

In this section we discuss some tools that all formalize orders of points on the line. Why ?

2.2.1 Signatures

It is clear from what we have discussed so far that the ridges in the stress surface are the areas where ρ is non-smooth. We will make this precise in later sections. The quadratic η^2 is as smooth as possible.

Further analysis of ρ. Our next step is to simplify the expression (5) for ρ. For this we use **signatures**.

If $x \in \mathbb{R}^n$ then $S(x)$ is the signature of x if $s_{ij}(x) = \text{sign}(x_i - x_j)$. A signature S is a hollow anti-symmetric matrix with elements $-1, 0$, and $+1$. Thus $S' = -S$ and $\text{diag}(S) = 0$. Signatures also must satisfy transitivity: if $s_{ih} = s_{hj} = 1$ for some h then $s_{ij} = 1$. Or, equivalenty, for a signature S there is a permutation matrix P such that PSP' has all $+1$ elements above the diagonal and thus all -1 below the diagonal.

The set of all $n \times n$ signatures is $\mathcal{S}_n$. Now n objects can be partitioned into r tie-blocks in $S(n,r)$ different ways, where $S(n,r)$ is the Stirling number of the second kind. The r tie blocks can be ordered in $r!$ different ways, and so the number of different signatures is the Fubini number $F(n) = \sum_{r=0}^{n} r!S(n,r)$ (see, for example, Good (1975)).

A signature S is **strict** if there are no ties, i.e. if $s_{ij} = \pm 1$ for all $i \neq j$. Strict signatures define the set $\mathcal{S}_n^+$. There are $n!$ different strict signatures, corresponding with the $n!$ permutations of $I_n := \{1, 2, \cdots, n\}$. For later use we also define the set $\mathcal{S}_a^n$ of all hollow antisymmetric matrices with off-diagonal elements ± 1 which are not necessarily transitive and the set $\mathcal{S}_b^n$ of all hollow matrices with elements ± 1 which are not necessarily transitive or antisymmetric. Here are the number of elements in each of these sets for $n = 1, \cdots, 10$. I left out the last three elements in the final column because they would blow the table off the page.

$\mathcal{S}^n$	$\mathcal{S}_+^n$	$\mathcal{S}_a^n$	$\mathcal{S}_b^n$
1	1	1	1
3	2	2	4
13	6	8	64
75	24	64	4 096
541	120	1 024	1 048 576
4 683	720	32 768	1 073 741 824
47 293	5 040	2 097 152	4 398 046 511 104
545 835	40 320	268 435 456	
7 087 261	362 880	68 719 476 736	
102 247 563	3 628 800	35 184 372 088 832	

2.2.2 Permutations and Rankings

There is a one-one correspondence between strict signatures and permutations π of $\{1, 2, \cdots, n\}$. There is also a one-one correspondence between signatures and rankings which may have ties.

2.2.3 Isotone Cones

Suppose π is a permutation of $\{1, 2, \cdots, n\}$, and $K(\pi)$ is the set of all $x \in \underline{\mathbb{R}}^n$ such that

$$x_{\pi(1)} \leq x_{\pi(2)} \leq \cdots \leq x_{\pi(n)}. \tag{12}$$

$K(\pi)$ is a pointed polyhedral closed convex cone in $\underline{\mathbb{R}}^n$ with apex at the origin. There are $n!$ such **isotone cones**. Their union is all of $\underline{\mathbb{R}}^n$.

The extreme rays of $K(\pi)$ are the $n-1$ half-lines defined by

$$x_{\pi(1)} < x_{\pi(2)} = x_{\pi(3)} = \cdots = x_{\pi(n-1)} = x_{\pi(n)}, \tag{13}$$

$$x_{\pi(1)} = x_{\pi(2)} < x_{\pi(3)} = \cdots = x_{\pi(n-1)} = x_{\pi(n)}, \tag{14}$$

$$\vdots \tag{15}$$

$$x_{\pi(1)} = x_{\pi(2)} = x_{\pi(3)} = \cdots = x_{\pi(n-1)} < x_{\pi(n)}. \tag{16}$$

```r
a <- matrix(0, 4, 5)
diag(a) <- -1
a[outer(1:4,1:5,function(x, y) x - y == -1)] <- 1
print(a)
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]   -1    1    0    0    0
## [2,]    0   -1    1    0    0
## [3,]    0    0   -1    1    0
## [4,]    0    0    0   -1    1
```

```r
g <- scdd(makeH(-a, c(0,0,0,0), c(1,1,1,1,1), 0))
b <- g$output[,3:7]
print(b)
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,] -4.0  1.0    1  1.0  1.0
## [2,] -1.5 -1.5    1  1.0  1.0
## [3,] -1.0 -1.0   -1  1.5  1.5
## [4,] -1.0 -1.0   -1 -1.0  4.0
```

```r
h <- scdd(makeH(-a,c(0,0,0,0),c(1,1,1,1,1),0))
print(h)
```

```
## $output
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7]
## [1,]    0    0 -4.0  1.0    1  1.0  1.0
## [2,]    0    0 -1.5 -1.5    1  1.0  1.0
## [3,]    0    0 -1.0 -1.0   -1  1.5  1.5
## [4,]    0    0 -1.0 -1.0   -1 -1.0  4.0
## attr(,"representation")
## [1] "V"
```

The cone $K(\pi)$ has an interior $K^\circ(\pi)$, consisting of all x with

$$x_{\pi(1)} < x_{\pi(2)} < \cdots < x_{\pi(n)}, \tag{17}$$

and it has a boundary $\partial K(\pi)$, consisting of the faces for which one or more of the elements of x are equal. Different cones are not necessarily disjoint, because some cones have some of their faces in common. For example an x with $x_1 = x_2 < x_3$ is both in $K(1,2,3)$ and $K(2,1,3)$.

(Negative Polar Cone)

$$\frac{1}{n-1}(x_2+\cdots+x_n)\leq x_1, \tag{18}$$

$$\frac{1}{n-2}(x_3+\cdots+x_n)\leq x_2, \tag{19}$$

$$\vdots \tag{20}$$

$$\frac{1}{2}(x_{n-1}+x_n)\leq x_{n-2}, \tag{21}$$

$$x_n\leq x_{n-1}. \tag{22}$$

2.3 Fixed Order

Consider the problem of findfing the infimum of σ over all x in the interior $\mathcal{K}_k^0$ of one the isotone cones.

$$\inf_{x\in\mathcal{K}_K^0} = \min_{x\in\mathcal{K}_k}$$

2.4 Augmentation and Majorization

ρ is piecewise linear, σ is piecewise quadratic

For all $x\in\underline{\mathbb{R}}^n$ we have

$$|x_i-x_j|=s_{ij}(x)(x_i-x_j) \tag{23}$$

and for all (x,y) in $\underline{\mathbb{R}}^n\otimes\underline{\mathbb{R}}^n$

$$|x_i-x_j|\geq s_{ij}(y)(x_i-x_j) \tag{24}$$

with equality if and only if either $x_i=x_j$ or $s_{ij}(x)=s_{ij}(y)$ or both, i.e. if and only $s_{ij}^2(x)=s_{ij}(x)s_{ij}(y)$. We can also combine (23) and (24) to

$$|x_i-x_j|=\max_{y\in\underline{\mathbb{R}}^n} s_{ij}(y)(x_i-x_j). \tag{25}$$

$$|x_i-x_j|=\max_{-1\leq s_{ij}\leq+1} s(x_i-x_j)=\max_{s\in\{-1,+1\}} s(x_i-x_j). \tag{26}$$

The basic difference between UDS and MDS in more than one dimension is that (26) must be replaced by

$$\|x_i-x_j\|=\max_{y'y=1} y'(x_i-x_j) \tag{27}$$

Notation

ξ, μ, τ

Rho

Tau is a step function.

Bounds on τ

$V^+ = (V + \frac{1}{n}ee')^{-1} - \frac{1}{n}ee'$

Now define, using signatures, a vector valued function $\mu : \underline{\mathbb{R}}^n \Rightarrow \underline{\mathbb{R}}^n$ as

$$\mu_i(x) := \sum_{j=1}^n w_{ij}\delta_{ij}s_{ij}(x) = \sum_j \{w_{ij}\delta_{ij} \mid x_j < x_i\} - \sum_j \{w_{ij}\delta_{ij} \mid x_j > x_i\}, \tag{28}$$

and another such function $\tau(x) := V^+\mu(x)$, with V^+ the Moore-Penrose inverse of V. In the unweighted case $\tau(x) = \mu(x)/n$. Using the elementwise (or Hadamard) matrix product $\times$ we see that $\mu(x)$ is the vector of row-sums of the anti-symmetric matrix $W \times \Delta \times S(x)$. Thus

$$\tau(x) = V^+(W \times \Delta \times S(x))e. \tag{29}$$

In MDS we call $\tau(x)$ the **Guttman transform** of x to honor Louis Guttman, who first defined it for the unweighted UDS case in Guttman (1968, section 23).

The elements of both $\mu(x)$ and $\tau(x)$ always add up to zero, and both $\mu(x)$ and $\tau(x)$ only depend on the order of the elements of x, not on their actual numerical values. Thus if $S(x) = S(y)$ we also have $\tau(x) = \tau(y)$. It consequenty makes sense to define τ_S for each $S \in \mathcal{S}_n$ as

$$\tau_S := V^+(W \times \Delta \times S)e. \tag{30}$$

Some more shorthand is useful. For each pair (x, y) we define

$$\xi(x, y) := \frac{1}{2}\sum_{i=1}^n\sum_{j=1}^n w_{ij}\delta_{ij}s_{ij}(y)(x_i - x_j) = \sum_{i=1}^n x_i\mu_i(x). \tag{31}$$

Then

$$\rho(x) = \xi(x, x) = x'V\tau(x), \tag{32}$$
$$\rho(x) \geq \xi(x, y) = x'V\tau(y), \tag{33}$$

with equality in (33) if and only if $s_{ij}(x)s_{ij}(y) = +1$ for all (i, j) with $w_{ij}\delta_{ij}s_{ij}(x) \neq 0$. In a positive UDS problem with a strict $S(x)$ this means we have equality if and only if $S(x) = S(y)$.

The corresponding result for stress is that for all (x, y)

$$\sigma(x) = 1 - 2x'V\tau(x) + x'Vx, \tag{34}$$
$$\sigma(x) \leq 1 - 2x'V\tau(y) + x'Vx, \tag{35}$$

or, equivalently,

$$\sigma(x) = \min_y 1 - 2x'V\tau(y) + x'Vx. \tag{36}$$

This representation is somewhat wasteful, because the minimum is over all $y \in \underline{\mathbb{R}}^n$. But since $\tau(y)$ only depends on the order of the y we may as well use

$$\sigma(x) = \min_{S\in\mathcal{S}_n} 1 - 2x'V\tau_S + x'Vx. \tag{37}$$

where τ_S is given by definition (30). In fact, we can even sharpen equation (37) to

$$\sigma(x) = \min_{S \in \mathcal{S}_n^+} 1 - 2x'V\tau_S + x'Vx, \tag{38}$$

because if $x_i = x_j$ it does not matter what we use for $s_{ij}(x)$ in (24), so we might as well choose S in $\mathcal{S}_n^+$.

2.4.1 Defays Theorem

Define ω, the squared length of the Guttman transform, as

$$\omega(x) := \tau(x)'V\tau(x). \tag{39}$$

Theorem 2.1 (Defays (1978))**.** *If $\overline{x}$ maximizes ω then $\overline{x}$ minimizes σ. Also $\sigma(\overline{x}) = 1 - \omega(\overline{x})$.*

Proof. Completing the square gives

$$\sigma(x) = 1 + (x - \tau(x))'V(x - \tau(x)) - \theta(x). \tag{40}$$

If there is an x that gives the global minimum of the second term $\eta^2(x - \tau(x))$ and the global maximum of ω then that x certainly gives the global minimum of σ. We first maximizing the third term θ, which gives us $\tau(x_+)$, and then minimize the second term by setting $x_+ = \tau(x_+)$. □

There were inklings of theorem 2.1 in Guttman (1968) and there is an informal version of it in De Leeuw and Heiser (1977), but the first complete analysis is the influential paper of Defays (1978). Theorem 2.1 separates the UDS problem into two steps. The first step finds the optimal signature, i.e. the optimal order of the x_i, and the second step trivially assigns optimal numerical values to the x_i.

2.4.2 Minimin

$$\begin{aligned}\min_x \sigma(x) = &\min_x \min_y 1 + x'Vx - 2x'V\tau(y) = \\ &\min_y \min_x 1 + x'Vx - 2x'V\tau(y) = \\ &\min_y \; 1 - \tau(y)'V^+\tau(y) = 1 - \max_y \tau(y)'V^+\tau(y).\end{aligned} \tag{41}$$

Toland duality

Subdiff of ρ

$$\rho^\star(y) = \max 2_x y'x - \rho(x) = \max_{k=1}^N \max_{x \in K_k} x'(y - t_k) = \max_{k=1}^N \begin{cases} 0 & \text{if } (y - t_k) \in K_k^\circ, \\ +\infty & \text{otherwise.}\end{cases}$$

$$\rho^\star(y)=\begin{cases}0&\text{if }(y-t_k)\in\bigcap_{k=1}^K K_k^\circ,\\+\infty&\text{otherwise.}\end{cases}$$

$$(\eta^2)^\star(y)=\max_x y'x-x'Vx=\frac{1}{4}y'V^+y$$

The polar cone of the monotone cone is the Shur cone, which codes the majorization order.

If X is a basis for the monotone code, then $Y=-X^+=-X(X'X)^{-1}$ is a basis for the polar cone.

2.5 Analysis

Instead of combinatorial now calculus.

The absolute value function is not differentiable at zero in the usual sense. Consequently $d_{ij}(x)=|x_i-x_j$ and σ are not differentiable at x when $x_i=x_j$ for one or more pairs (i,j). Thus the Fermat rule that at local minimum x_+ the derivative of σ is zero does not make sense if some of the elements of x are equal. In order to find necessary conditions for a local minimum we switch to (one-sided) directional derivatives, defined as

$$D\sigma(x;y)=\lim_{\epsilon\downarrow 0}\frac{\sigma(x+\epsilon y)-\sigma(x)}{\epsilon},\tag{42}$$

where ϵ approaches zero from the right, i.e. ϵ only takes positive values.

Theorem 2.2 (De Leeuw (1984)). *The directional derivative of σ at x in the direction y is*

$$D\sigma(x;y)=-2\left\{y'(\xi(x)-Vx)+\sum_{i=1}^n\sum_{j=1}^n\{w_{ij}\delta_{ij}|y_i-y_j|\mid d_{ij}(x)=0\}\right\}.\tag{43}$$

Proof. We have the result

$$\eta^2(x+\epsilon y)=\eta^2(x)+2\epsilon y'Vx+\epsilon^2y'Vy.\tag{44}$$

Thus

$$D\eta^2(x;y)=2y'Vx$$

and for small enough ϵ we have the exact result

$$\rho(x+\epsilon y)=\rho(x)+\epsilon y'\xi(x)+\epsilon\sum_i\sum_j\{w_{ij}\delta_{ij}|y_i-y_j|\mid x_i=x_j\}\tag{45}$$

$$D\rho(x;y)=y'V\tau(x)+\sum_i\sum_j\{w_{ij}\delta_{ij}|y_i-y_j|\mid x_i=x_j\}$$

Combining equations (44) and (45) gives the required result. □

2.5.1 Stationary Points

An x for which $\xi(x) - Vx = 0$, or $x = \tau(x)$, is a **stationary point**.

Theorem 2.3 (De Leeuw(1977))**.** *If x is a stationary point then $\eta^2(x) \leq 1$.*

Proof. If $\xi(x) - Vx = 0$ then $\rho(x) = \eta^2(x)$. Thus $\sigma(x) = 1 - \eta^2(x)$ and thus $\eta^2(x) \leq 1$. □

Theorem 2.3 is important because it shows that minimizing σ is the same thing as minimizing σ over the compact convex set $\{x|\eta^2(x) \leq 1\}$.

Theorem 2.4. *Stress has a single local maximum at $x = 0$*

Proof. From … $D\sigma(0, y) > 0$ for all $y \neq 0$. Thus stress has a strict local maximum at zero. If there was a local maximum at $x \neq 0$ then σ would also have a local maximum at the line through z and the origin. But on that line $\sigma(\lambda x) = 1 - 2\lambda\rho(x) + \lambda^2\eta^2(x)$, which is a convex quadratic with no local maxima. □

2.5.2 Local Minima

Stress has a **local minimum** at x if there is a neighborhood $N(x)$ such that $\sigma(x) \leq \sigma(y)$ for all $y \in N(x)$. The local minimum is **strict** if actually $\sigma(x) < \sigma(y)$ for all $y \in N(x)$. A local minimum at x is **isolated** if there is a neighborhood $N(x)$ which contains no other local minima. Al isolated local minima are strict.

The following necessary condition for a local minimum of stress is due to De Leeuw (1984), who proves it for MDS. It is a key MDS result, and even more so for UDS.

Theorem 2.5 (De Leeuw(1984))**.** *If stress has a local minimum at x if and only if*

1. *x is a stationary point, and*
2. *$d_{ij}(x) > 0$ for all (i, j) for which $w_{ij}\delta_{ij} > 0$.*

Proof. If x is not stationary, i.e. if $x \neq \tau(x)$, then there is an y such that $y'V(\tau(x) - x)) > 0$. And thus by (43) $D\sigma(x; y) < 0$.

If σ has a local minimum at x we must have $D\sigma(x; y) \geq 0$ for all y. Thus also

$$D\sigma(x; y) + D\sigma(x; -y) = -\sum_{i=1}^{n}\sum_{j=1}^{n}\{w_{ij}\delta_{ij}|y_i - y_j| \mid d_{ij}(x) = 0\} \geq 0 \tag{46}$$

for all y, which implies that for all k we must have $w_{ij}\delta_{ij} = 0$ for all i, j with $d_{ij}(x) = 0$.. □

All local minima of stress are isolated.

There are at most $n!$ local minima.

If stress has a local minimum at x then $d_{ij}(x) = 0$ only if $w_{ij}\delta_{ij} = 0$. Note we do **not** say that $w_{ij}\delta_{ij} = 0$ implies that at a local minimum $d_{ij}(x) = 0$.

Corollary 2.1. *In UDS problems σ is differentiable at all local minima.*

Proof. This is immediate for positive UDS problems. But even if a problem is not positive we have

$$\rho(x) = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\{w_{ij}\delta_{ij}d_{ij}(x) \mid w_{ij}\delta_{ij} > 0\}.$$

By the same reasoning as before we now have $d_{ij}(x) > 0$ at local minima for all (i,j) for which $w_{ij}\delta_{ij} > 0$. If $d_{ij}(x) = 0$ then $w_{ij}\delta_{ij} = 0$ and the (i,j) term simply does not enter into the summation defining $\rho(x)$. □

Local minima are stationary points, but stationary points are not necessarily local minima. Suppose we have an unweighted UDS with

$$\Delta = \begin{bmatrix} 0 & 2 & 1 \\ 2 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}.$$

Consider $x_1 = x_2 = -\frac{1}{3}$ and $x_3 = \frac{2}{3}$. Then

$$S(x) = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 0 & -1 \\ 1 & 1 & 0 \end{bmatrix},$$

and $\tau(x) = (-\frac{1}{3}, -\frac{1}{3}, \frac{2}{3})$. Thus $x = \tau(x)$ and x is stationary, but it cannot be a local minimum because $x_1 = x_2$. If, for example, we choose a direction y as $(-1,1,0)$ then $D\sigma(x;y) = -2\delta_{12}|y_1 - y_2| = -4$ and thus y is a descent direction at x.

Suppose x_s minimizes q_s and is on the boundary. If $d_{ij}(x) = 0$ then $\delta_{ij} = 0$. Suppose S_s and S_t are adjacent. Then $t_s = t_t$ and thus q_s and q_t are the same. Thus strict local minimum.

$$\Delta = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

$x = (+\frac{1}{3}, \frac{2}{3}, +\frac{1}{3})$. Then

$$S(x) = \begin{bmatrix} 0 & +1 & 0 \\ -1 & 0 & -1 \\ 0 & +1 & 0 \end{bmatrix}$$

Now $\tau(x) = x$ and thus x is a stationary point, even though it has $d_{13}(x) = 0$.

From equation (38) we see that σ is the minimum of $N = n!$ strictly convex quadratic functions, all of them with the same quadratic part $x'Vx$.

2.6 Subdifferentials

$$\partial\rho(x) = \{y \mid y = (W \times \Delta \times S(x))e\}$$

where S is any matrix with

$$s_{ij}(x) = \begin{cases} \text{sign}(x_i - x_j) & \text{if } x_i \neq x_j, \\ -1 \leq s_{ij}(x) \leq +1 & \text{if } x_i = x_j. \end{cases}$$

Conjugate

$$\rho(x) = \begin{cases} x' t_1 & \text{if } x \in K_1, \\ & \vdots \\ x' t_N & \text{if } x \in K_N \end{cases}$$

$$\rho^\star(y) = \sup_x x'y - \rho(x) = \max_{k=1}^N \sup_{x \in K_k} x'V(y - \tau_k)$$

$$\sup_{x \in K_k} x'V(y - \tau_k) = \begin{cases} 0 & \text{if } y - \tau_k \in K_k^0, \\ +\infty & \text{otherwise.} \end{cases}$$

Thus

$$\rho^\star(y) = 0 \text{ iff } y \in \bigcup_{k=1}^n \{t_k + K_k^\circ\}$$

In terms of projections on the cone.

In terms of polar norm.

$$\rho(x) = \max_{k=1}^N x' t_k$$

$$\partial\rho(x) = \text{conv}\{t_k | x' t_k = \rho(x)\}$$

Polar Norm

$$\rho^\circ(y) = \sup_x \frac{x'Vy}{\rho(x)} = \max_{k=1}^K \sup_{x \in \mathcal{K}_k} \frac{x'Vy}{x'V\tau_k}$$

2.7 Moreau-Yoshida regularization of rho

$$e_\lambda(y) = \inf_x \rho(x) + \frac{1}{2}\frac{1}{\lambda}(x - y)'V(x - y) = \min_{k=1}^N \inf_{x \in K_k} x'V\tau_k + \frac{1}{2}\frac{1}{\lambda}(x - y)'V(x - y)$$

$$\min_{x \in K_k} \frac{1}{2}\frac{1}{\lambda}\{(x - y)'V(x - y) + 2\lambda x'V\tau_k\} = \min_{x \in K_k} \frac{1}{2}\frac{1}{\lambda}\{x - (y - \lambda\tau_k))'V(x - (y - \lambda\tau_k))\} - \frac{1}{2}\lambda\tau_k'V\tau_k$$

$$x_k = \text{proj}_k(y - \lambda\tau_k)$$

2.8 Ratio of Norms

De Leeuw (1977)

3 Algorithms for UDS

3.1 Local Optimization

3.1.1 Majorization (MM) Algorithm

De Leeuw (1994) Heiser (1995) Lange, Hunter, and Yang (2000) Lange (2016)

Using a superscipt for iteration number we$lange can now define the majorization algorithm for UDS with the update formula

$$x^{(k+1)} = \underset{x}{\text{argmin}}\ \{1 - 2x'\xi(x^{(k)}) + x'Vx\} = \tau(x^{(k)}), \tag{47}$$

unless $x^{(k)} = \tau(x^{(k)})$, in which case we stop the algorithm.

Theorem 3.1 (De Leeuw and Heiser (1977))**.** *The majorization algorithm* (47) *converges to a stationary point in a finite number of steps.*

Proof. We have $s_{ij}(y)(x_i - x_j) \leq s_{ij}(x)(x_i - x_j)$. Thus, for all (x, y).

$$\rho(x) \geq x't(y). \tag{48}$$

It follows that

$$\sigma(x) \leq 1 - 2x'V\tau(y) + x'Vx. \tag{49}$$

for all pairs (x, y), and thus

$$\sigma(x^{(k+1)}) \leq 1 - 2\{x^{(k+1)}\}'V\tau(x^{(k)}) + \eta^2(x^{(k)}). \tag{50}$$

Because of the minimization in the update rule (47)

$$1 - 2\{x^{(k+1)}\}'V\tau(x^{(k)}) + \eta^2(x^{(k)}) < 1 - 2\{x^{(k)}\}'V\tau(x^{(k)}) + \eta^2(x^{(k)}) = \sigma(x^{(k)}). \tag{51}$$

unless $x^{(k)} = \tau(x^{(k)})$, in which we case we stop anyway. Thus either we stop or, from inequalities (50) and (51),

$$\sigma(x^{(k+1)}) < \sigma(x^{(k)}). \tag{52}$$

Since $\tau(x)$ only depends on the order of x, and since by result (52) an order can never repeat, this ends the proof. □

3.1.2 Projection Algorithm

3.2 Global Optimization

3.2.1 Integer Programming

3.2.2 Permutation Algorithm

De Leeuw (2005) Mair and De Leeuw (2015)

Theorem 3.2 (De Leeuw and Heiser (1977))**.** *Suppose*

$$\vartheta(\pi)$$

$$x(\pi) = \underset{x\in K(\pi)}{\mathit{argmin}}\, (x-\tau(\pi))'V(x-\tau(\pi))$$

then

$$\min_{x\in\underline{\mathbb{R}}^n} \sigma(x) = \min_{\pi\in\mathcal{P}} 1 + d(\pi) - \tau(\pi)'V\tau(\pi)$$

If x is in the interior of the cone K then

$$\sigma(x) = 1 - 2x'V\tau_K + x'Vx = 1 + (x-\tau_K)'V(x-\tau_K) + \tau_K'V\tau_K$$

If x is on the boundary of the cone

Thus minimizing x over int(K) can be done by minimizing

$$\eta^2(x-\tau_K) = (x-\tau_K)'V(x-\tau_K).$$

$$\inf_{x\in\text{int}(K)} \eta^2(x-\tau_K) = \min_{x\in K} \eta^2(x-\tau_K)$$

and the last problem is a weighted least squares monotone regression problem, which has a unique solution $P_K(\tau_K)$.

3.3 Smoothing

Pliner (1986)

Moreau-Yoshida

3.4 Penalizing Full-dimensional Scaling

4 Small Examples

In this section we illustrate the developments so far with five small examples $\Delta_1, \cdots, \Delta_5$ of dissimilarity matrices. They are

$$\begin{bmatrix}0&1&2\\1&0&3\\2&3&0\end{bmatrix}\begin{bmatrix}0&1&0\\1&0&1\\0&1&0\end{bmatrix}\begin{bmatrix}0&1&1\\1&0&1\\1&1&0\end{bmatrix}\begin{bmatrix}0&1&2\\1&0&4\\2&4&0\end{bmatrix}\begin{bmatrix}0&1&0\\1&0&0\\0&0&0\end{bmatrix}$$

In all examples the UDS is unweighted, but Δ_2 and Δ_5 are not positive. The matrices are scaled using (4), so that all stationary points are in the region with $\eta^2(x) \leq 1$. We use two orthonormal basis vectors z_1 and z_2 that span $\underline{\mathbb{R}}^3$, and use for the coordinates of our plots the coefficients of the linear combinations (α, β) in $x = \alpha z_1 + \beta z_2$. Thus $\eta^2(x) = 3(\alpha^2 + \beta^2)$.

We make two plots for each example. The first plot shows the circle $\eta^2(x) = 1$ and the six cones corresponding with the six permutations of the three coordinates. We then show a single iteration of the majorization algorithm. Six starting points (small black points) are chosen. Each of the six x_k is then connected with $\tau(x_k)$ (large blue point). Prefect fit with $\sigma(x) = 0$ is indicated by a point on the circle, the closer the (α, β) point is to the circle, the better the fit.

The second plot is a straightforward contour plot of σ, again with the six cones drawn in. Because of our choice of coordinates in each cone σ is a circular quadratic around its minimum. in the middle of the circle arc where the six cones intersect the unit circle. and the set of x with $\eta^2(x) = 1$ is the circle of radius $\frac{1}{3}\sqrt{3}$.

Dissimilarities Δ_1 can be fitted exactly in one dimension (with x_1 between x_2 and x_3). Figures 1 and 2 show six local minima, in three mirror image pairs, in one of each of the six cones. The pair on the circle are the global minima, with σ equal to zero.

INSERT FIGURES 1 AND 2 ABOUT HERE

Δ_2 can also be fitted perfectly in one dimension, but with $x_1 = x_3$, because $\delta_{13} = 0$. Figures 3 and 4 show four local minima, with the global minima on the circle. The global minima are in the intersection of two of the cones, on the line with $x_1 = x_3$. This illustrates that local minima are not necessarily in the interior of one of the cones.

INSERT FIGURES 3 AND 4 ABOUT HERE

Δ_3 has a perfect MDS fit in two dimensions (an equilateral triangle), but not in one dimension. It does have complete symmetry, which means that all six cones produce local minima with the same value of σ and the same numerical values in x (in different permutations). This is shown in figures 5 and 6.

INSERT FIGURES 5 AND 6 ABOUT HERE

Matrix Δ_4 violates the triangle inequality because $\delta_{23} > \delta_{21} + \delta_{13}$. There is no MDS perfect fit in any dimensionality. It looks from 7 if the point in the cone $x_2 < x_1 < x_3$ is on the circle, and thus must have zero stress, but actually $\sigma(x) = .0159$ and $\eta^2(x) = .9841$. Close, but no cigar. Also the point on the half line $x_1 = x_2 < x3$ is not a stationary point, and thus not a local minimum. The next step of the majorization algorithm will move to the point in $x_2 < x_1 < x_3$, which is both stationary and a local minimum. The contour plot in figure 8 shows the irregularities at the intersections of the cones, where stress is not differentiable and has a ridge of saddle points. Thus for Δ_4 there are only four local minima.

INSERT FIGURES 7 AND 8 ABOUT HERE

Dissimilarties Δ_5 also violates the triangle inequality. Figures 9 and 10 show there is only one mirror pair of local minima, which are thus the global minima. The majorization algorithm finds a global minimum in one step, no matter where we start. Note that in this example stress is differentiable everywhere, except on the line $x_1 = x_2$.

INSERT FIGURES 9 AND 10 ABOUT HERE

5 Extensions

5.1 Inverse Unidimensional Scaling

Thus, more formally,

$$\text{MDS}_{(n,p)}n : \mathbb{H}^{n\times n} \otimes \mathbb{H}^{n\times n} \Rightarrow \underline{\mathbb{R}}^{n\times p} \tag{53}$$

is defined by

$$MDS_n(W, \Delta) := \underset{x\in\underline{\mathbb{R}}^n}{\text{argmin}}\,\sigma(x) = \{y \in \underline{\mathbb{R}}^n \mid \sigma(y) = \min_{x\in\underline{\mathbb{R}}^n} \sigma(x)\}. \tag{54}$$

Suppose σ has a local minimum at x.

$$\{\Delta \mid Vx = t(x)\}$$

$$\{W, \Delta \mid Vx = t(x)\}$$

Example $n = 3$, $w_{ij} = 1$

$(-1, 0, 1)$ is a stationary point for any Δ

$$\Delta(\theta) = \begin{bmatrix} 0 & 3-\theta & \theta \\ 3-\theta & 0 & 3-\theta \\ \theta & 3-\theta & 0 \end{bmatrix}$$

for any $0 \leq \theta \leq 3$. If $\theta = 2$ then $\Delta = D(x)$ and $\sigma(x) = 0$

$(-1, -1, 2)$ is a stationary point for

$$\Delta(\theta) = \begin{bmatrix} 0 & \theta & 3 \\ \theta & 0 & 3 \\ 3 & 3 & 0 \end{bmatrix}$$

for any $\theta \geq 0$. If $\theta = 0$ then $\Delta = D(x)$ and $\sigma(x) = 0$. This is not a counterexample to De Leeuw (1984), because his result requires $\theta > 0$. But it does show that we can have $d_{ij}(x) = 0$ at a local minimum. It also shows that stationary points may not be local minima.

Question: are there more local minima for these $\Delta(\theta)$. Consider the 3 permutations $x_1 < x_2 < x_3$, $x_1 < x_3 < x_2$, and $x_3 < x_1 < x_2$

Suppose

$$\Delta = \begin{bmatrix} 0 & a & b \\ a & 0 & c \\ b & c & 0 \end{bmatrix}$$

What are all possible local minima ? Assume wlg that $a = 1$.

And stationary points ?

$x_1 < x_2 < x_3$

$$S = \begin{bmatrix} 0 & -1 & -1 \\ +1 & 0 & -1 \\ +1 & +1 & 0 \end{bmatrix}$$

$$-a - b < a - c < b + c$$

```
a1 <- matrix(c(-2,-1,1,1,-1,-2),2,3,byrow = TRUE)
v1 <- scdd(makeH(rbind(-diag(ncol(a1)),a1),rep(0, ncol(a1) + nrow(a1))))[[1]]
attr(v1, "representation") <- NULL
print(v1[, -(1:2)])
```

```
##      [,1] [,2] [,3]
## [1,]    0    1    0
## [2,]    1    1    0
## [3,]    2    0    1
## [4,]    1    0    2
## [5,]    0    1    1
```

$$\begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 2 & 0 \\ 2 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 2 \\ 0 & 2 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

$x_1 < x_3 < x_2$

$$S = \begin{bmatrix} 0 & -1 & -1 \\ +1 & 0 & +1 \\ +1 & -1 & 0 \end{bmatrix}.$$

$$-a - b < b - c < a + c$$

```
a2 <- matrix(c(-1,-2,1,-1,1,-2), 2, 3, byrow = TRUE)
scdd(makeH(rbind(-diag(ncol(a2)),a2),rep(0, ncol(a2) + nrow(a2))))
```

```
## $output
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    0    0    0    2    1
```

```
## [2,]    0    0    1    0    0
## [3,]    0    0    1    1    0
## [4,]    0    0    1    0    1
## [5,]    0    0    0    1    2
## attr(,"representation")
## [1] "V"
```

$x_3 < x_1 < x_2$

$$S = \begin{bmatrix} 0 & -1 & +1 \\ +1 & 0 & +1 \\ -1 & -1 & 0 \end{bmatrix}$$

$$-b - c < -a + b < a + c$$

```r
a3 <- matrix(c(1,-2,-1,-2,1,-1),2,3,byrow = TRUE)
scdd(makeH(rbind(-diag(ncol(a3)),a3),rep(0, ncol(a3) + nrow(a3))))
```

```
## $output
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    0    0    0    0    1
## [2,]    0    0    2    1    0
## [3,]    0    0    1    0    1
## [4,]    0    0    1    2    0
## [5,]    0    0    0    1    1
## attr(,"representation")
## [1] "V"
```

```r
a4 <- matrix(c(-2,-1,-1,1,1,0,
               1,-2,0,-2,-1,1,
               0,1,-1,1,-1,-2),3,6,byrow = TRUE)
v4 <- scdd(makeH(rbind(-diag(ncol(a4)),a4),rep(0, ncol(a4) + nrow(a4))))[[1]]
attr(v4, "representation") <- NULL
v4 <- v4[,-(1:2)]
```

```r
a52 <- matrix(c(-1,-1,-1,0,0,0,
              1,0,0,-1,-1,0,
              0,1,0,1,0,-1,
              0,0,1,0,1,1), 4, 6, byrow = TRUE)
b52 <- c(-12,-4,4,12)
h5<-makeH(rbind(-diag(6)),rep(0, 6),a52,b52)
v5 <- scdd(h5)
print(h5)
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
##  [1,]    1  -12    1    1    1    0    0    0
##  [2,]    1   -4   -1    0    0    1    1    0
```

```
## [3,]    1    4    0   -1    0   -1    0    1
## [4,]    1   12    0    0   -1    0   -1   -1
## [5,]    0    0    1    0    0    0    0    0
## [6,]    0    0    0    1    0    0    0    0
## [7,]    0    0    0    0    1    0    0    0
## [8,]    0    0    0    0    0    1    0    0
## [9,]    0    0    0    0    0    0    1    0
## [10,]   0    0    0    0    0    0    0    1
## attr(,"representation")
## [1] "H"
```

```
print(v5)
```

```
## $output
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
## [1,]    0    1   12    0    0   16    0   12
## [2,]    0    1   12    0    0    4   12    0
## [3,]    0    1    8    4    0    0   12    0
## [4,]    0    1    0    4    8    0    4    0
## [5,]    0    1    0   12    0    0    4    8
## [6,]    0    1    0   12    0    4    0   12
## [7,]    0    1    0    0   12    4    0    0
## attr(,"representation")
## [1] "V"
```

De Leeuw (2019)

5.2 Nonmetric Unidimensional Scaling

$$\Delta = \sum_{s=1}^{r} \theta_s T_s$$

with $\theta_s \geq 0$. When is x a solution. If the row sums of $\sum_{s=1}^{r} \theta_s (W \times T_s \times S(x)$ are monotone with x. This gives a bunch of homogeneous linear inequalities in θ.

$$\sigma(x,\theta) := \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(\delta_{ij}(\theta) - |x_i - x_j|)^2$$

Additive constant, as in Heiser et al.

6 Data Examples

6.1 Vegetables

```
data (vegetables, package = "psychTools")
veg <- abs(qnorm(as.matrix(veg)))
w <- 1 - diag(9)
veg <- 2 * veg / sqrt(sum(w * veg * veg))
```

```
vstress <- NULL
vitel <- NULL
vx <- matrix(0, 0, 9)
for (i in 1:10000) {
  h <- udsmm(veg, w, sample(1:9, 9), verbose = FALSE)
  vstress <- c(vstress, h$f)
  vitel <- c(vitel, h$itel)
  vx <- rbind(vx, h$x)
}
```

The minimum value of stress found was 0.035301, which was found 940 times.

INSERT FIGURE 11 ABOUT HERE

```
## vitel
##    2    3    4    5    6    7    8    9   10
##  421 2529 3094 2163 1065  478  125   76   49
```

The minimum stress is 0.035301. In 15484 out of the 362880 starts we found stationary points. Because the UDS problem is positive these are all local minima and they are all in the interior of different cones !!not so!!deltas

6.2 Genes

```
genes <- as.matrix(read.table("nextperm/genes.R"))
weights <- as.matrix(read.table("nextperm/weights.R"))/1000
genes <- genes + t(genes)
weights <- weights + t(weights)
```

The minimum value of stress found was 0.007077, which was found 559 times.

INSERT FIGURE 12 ABOUT HERE

```
## gitel
##    2    3    4    5
##  592 7663 1744    1
```

The minimum stress is 0.007077. In 2380 out of the 40320 starts we found stationary points.

6.3 Plato

```
data(Plato7, package = "smacof")
plato <- as.matrix(dist(sqrt(t(Plato7))))
weights <- 1 - diag(7)
```

The minimum value of stress found was 0.121515, which was found 10 times.

INSERT FIGURE 13 ABOUT HERE

```
## pitel
##    2    3
## 9942   58
```

The minimum stress is 0.121515. In 5016 out of the 5040 starts we found stationary points.

6.4 Morse Code

A Figures

B Appendix: Code

B.1 udsplots.R

```
delta1 <- matrix(c(0, 1, 2, 1, 0, 3, 2, 3, 0), 3, 3)
delta2 <- matrix(c(0, 1, 0, 1, 0, 1, 0, 1, 0), 3, 3)
delta3 <- 1 - diag(3)
delta4 <- matrix(c(0, 1, 2, 1, 0, 4, 2, 4, 0), 3, 3)
delta5 <- matrix(c(0, 1, 0, 1, 0, 0, 0, 0, 0), 3, 3)

pplot <- function(delta) {
  delta <- delta / sqrt(sum(delta ^ 2) / 2)
  z <- matrix(c(-1, 1, 0, -1, -1 , 2), 3, 2)
  z <- apply(z, 2, function(x)
    x / sqrt(sum(x ^ 2)))
  a <- seq(-1, 1, length = 100)
  b <- seq(-1, 1, length = 100)
  funk <- function(a, b) {
    n <- length(a)
    m <- length(b)
    s <- matrix(0, n, m)
    for (i in 1:n) {
      for (j in 1:m) {
        x <- a[i] * z[, 1] + b[j] * z[, 2]
```

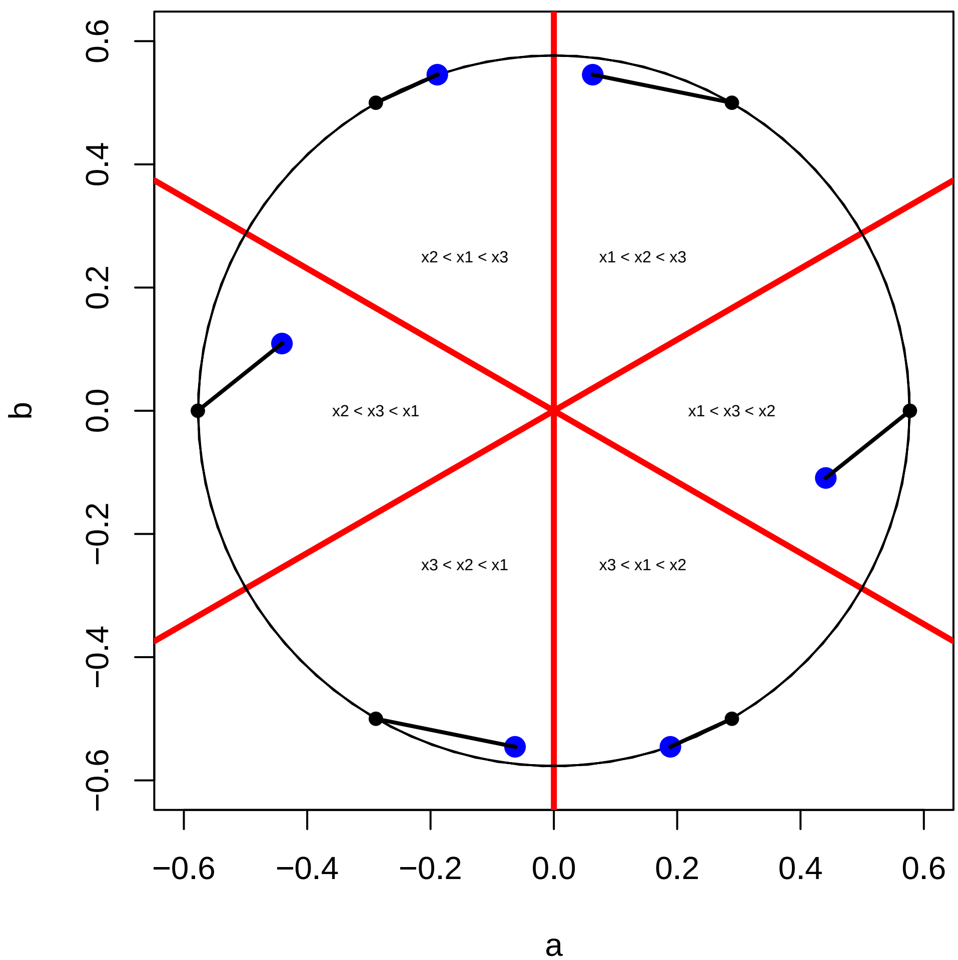


Figure 1: Cones for Delta1

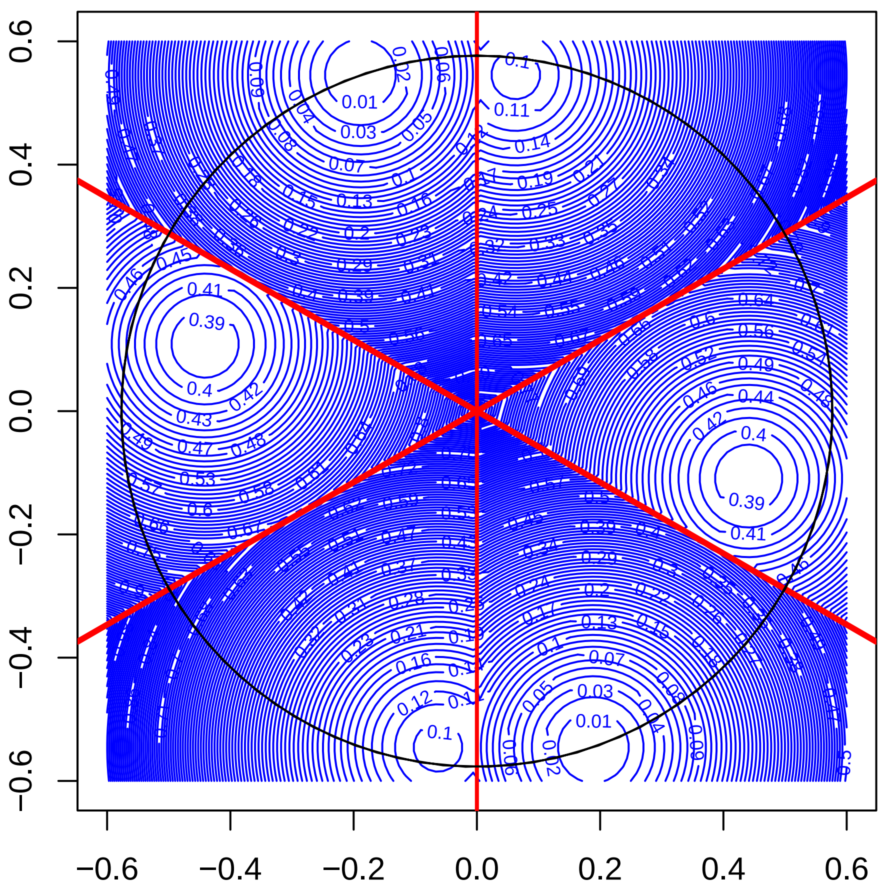


Figure 2: Local Minima for Delta1

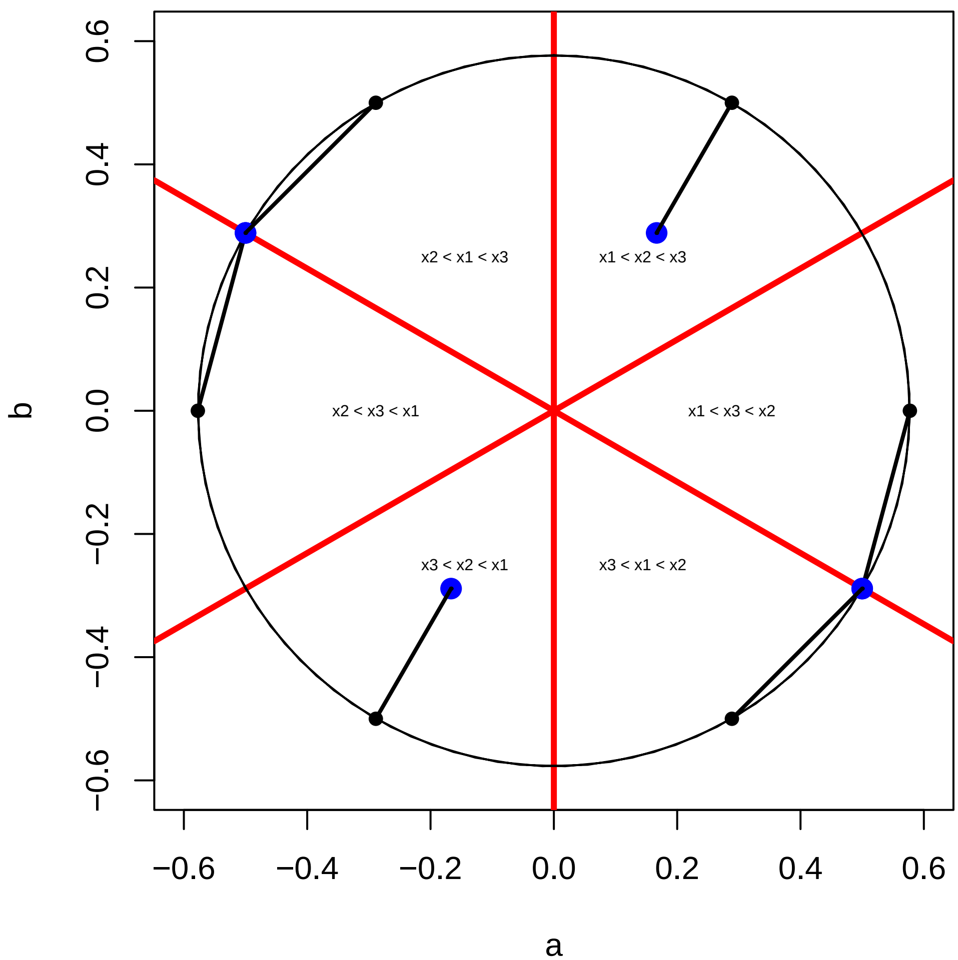


Figure 3: Cones for Delta2

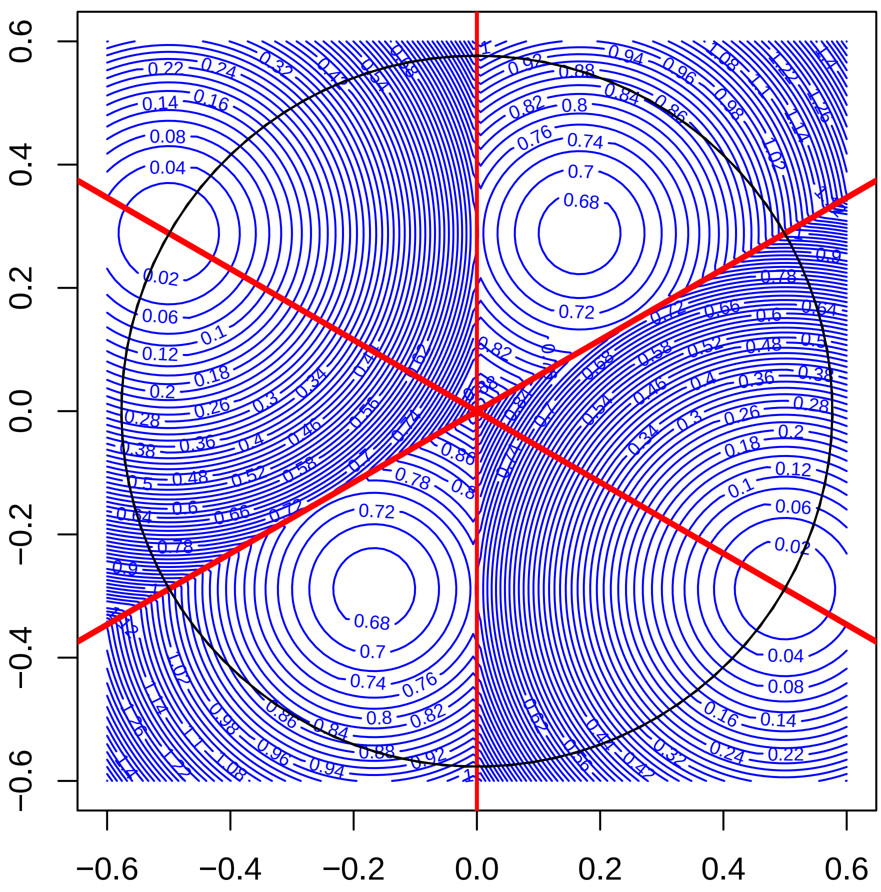


Figure 4: Local Minima for Delta2

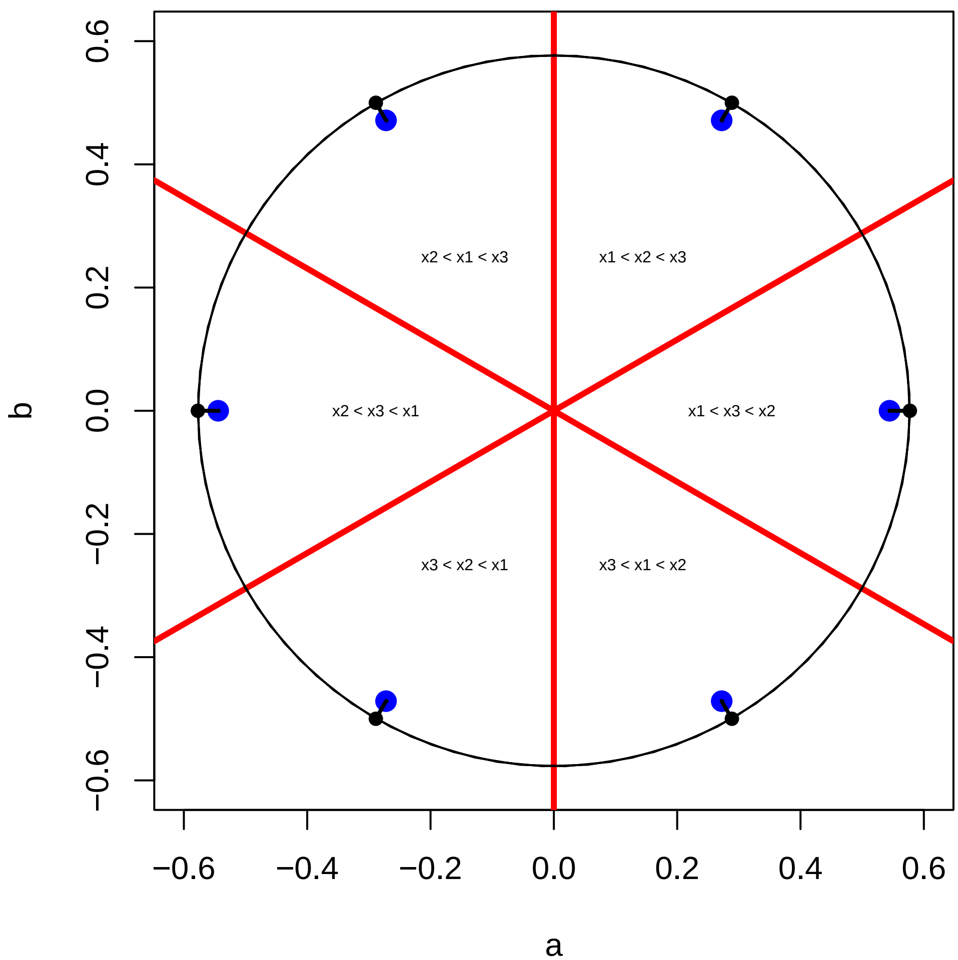


Figure 5: Cones for Delta3

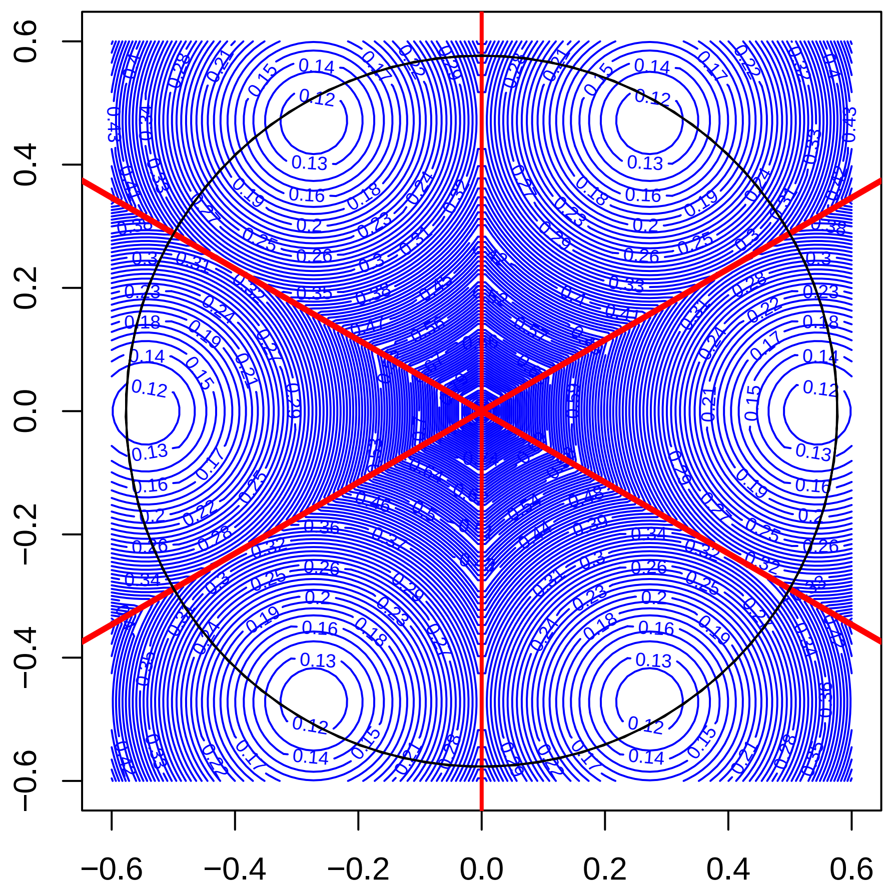


Figure 6: Local Minima for Delta3

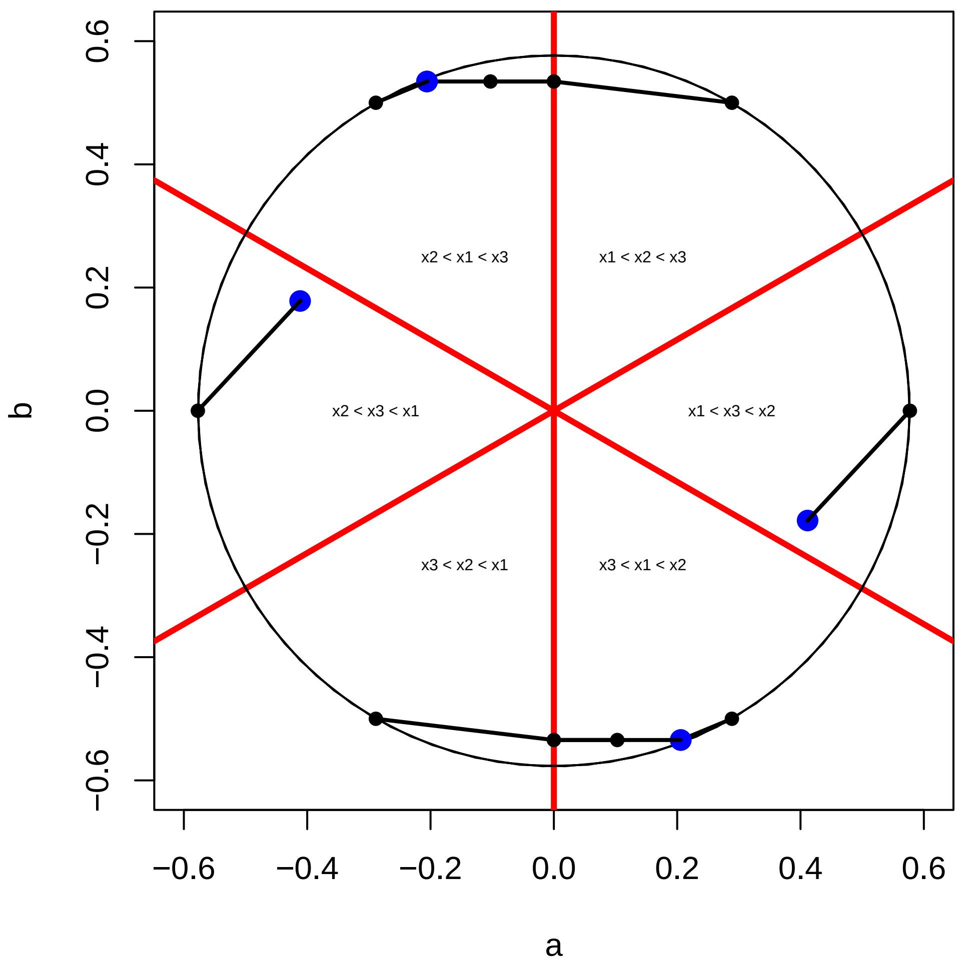


Figure 7: Cones for Delta4

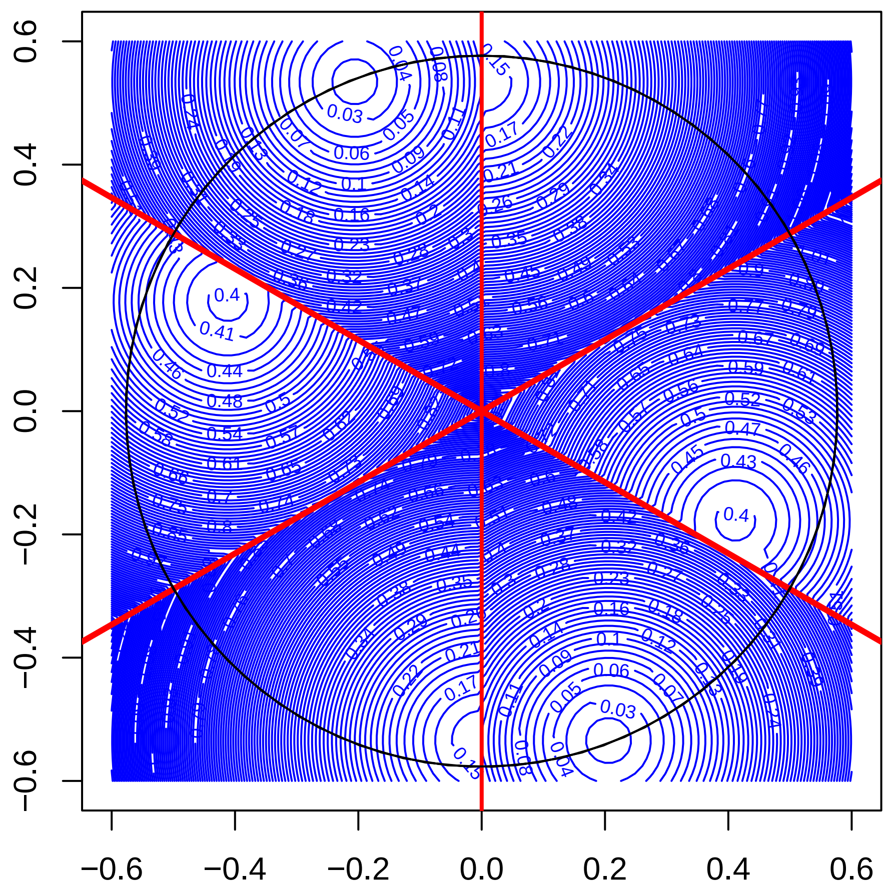


Figure 8: Local Minima for Delta4

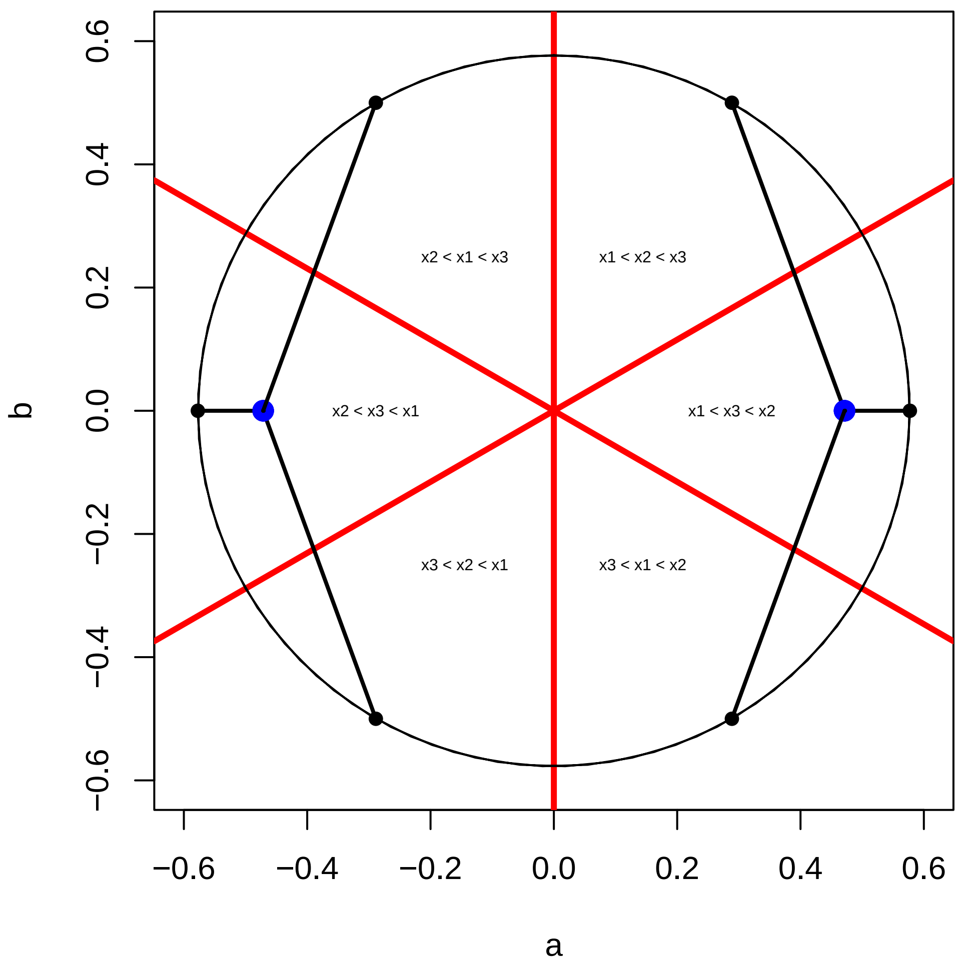


Figure 9: Cones for Delta5

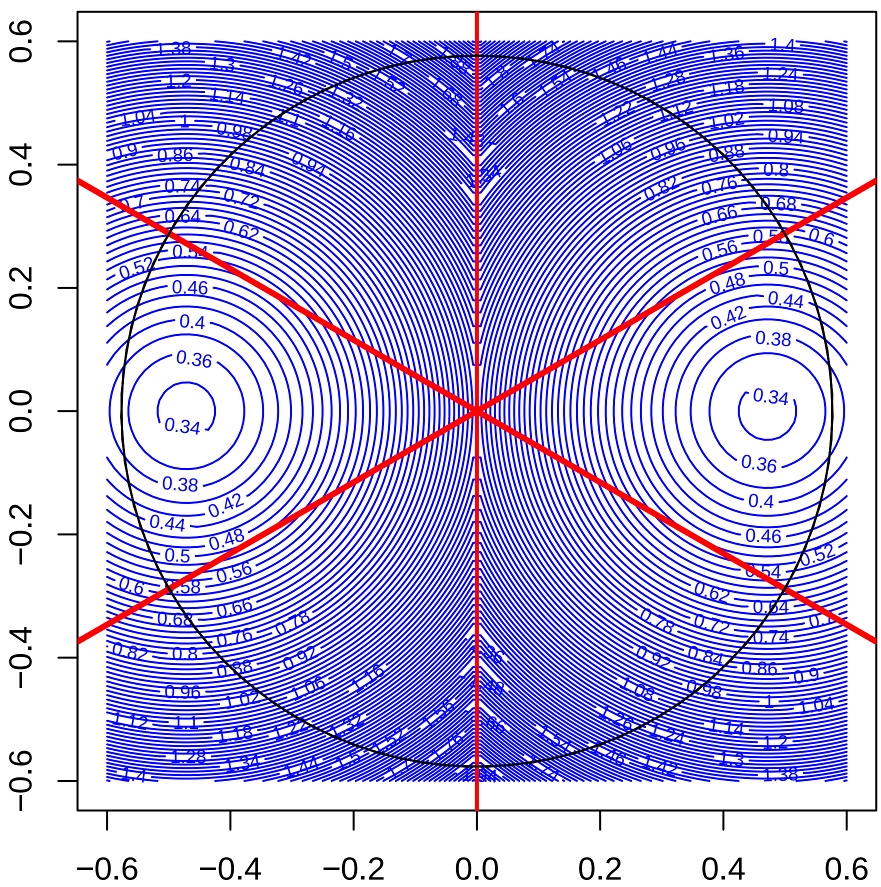


Figure 10: Local Minima for Delta5

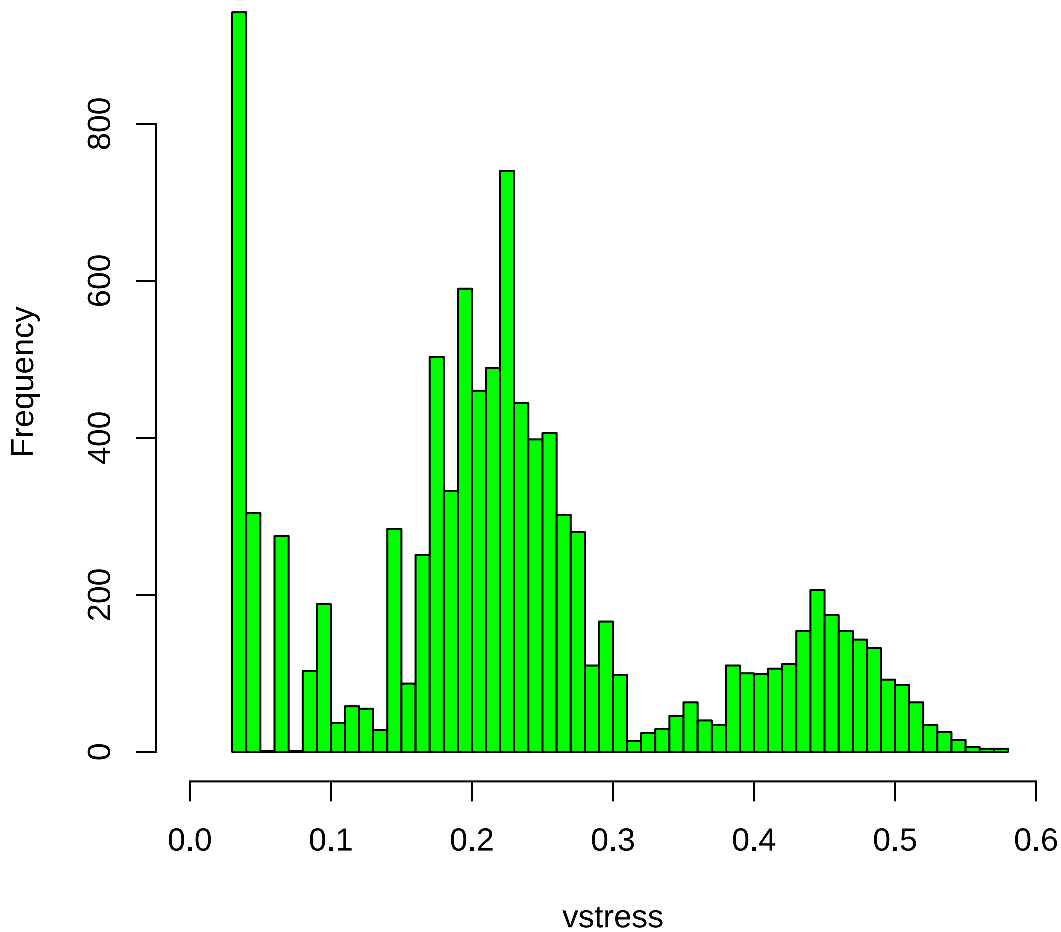


Figure 11: Stress Histogram for Vegetables, 10000 runs

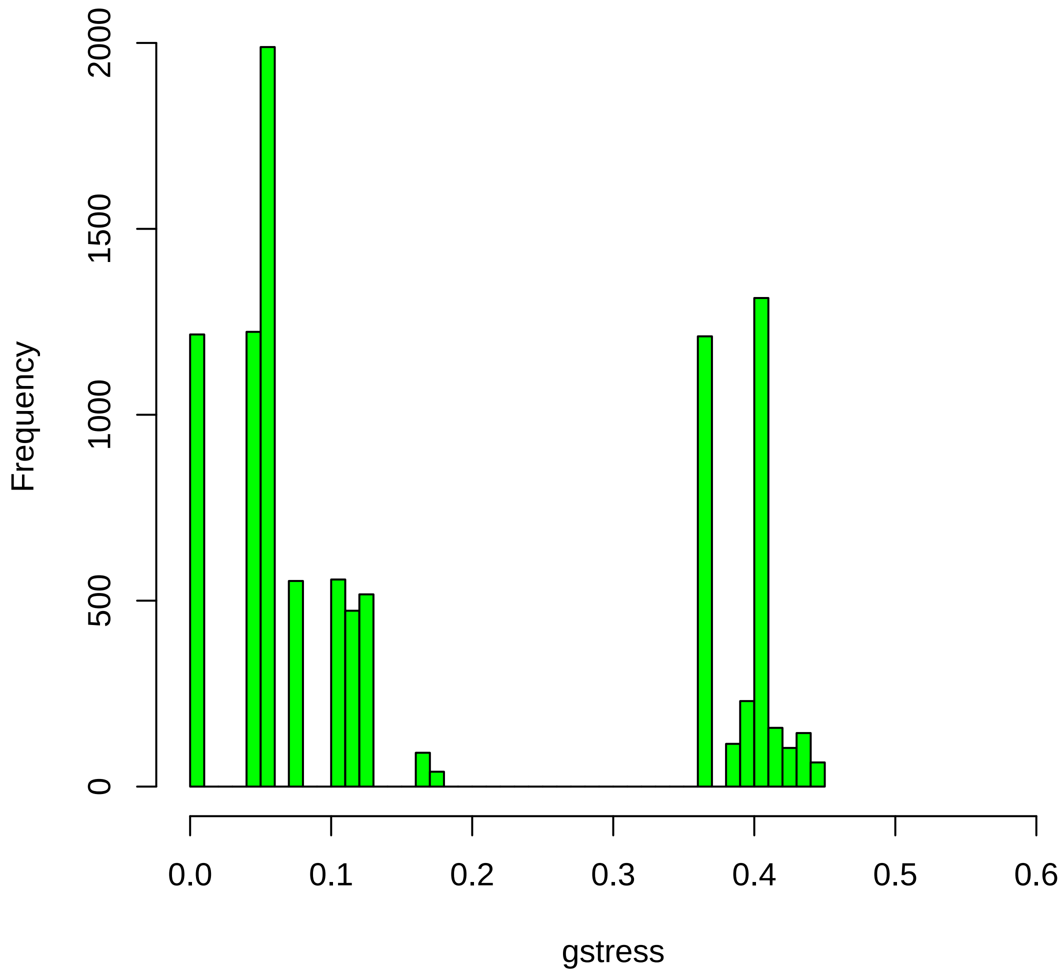


Figure 12: Stress Histogram for Genes, 10000 runs

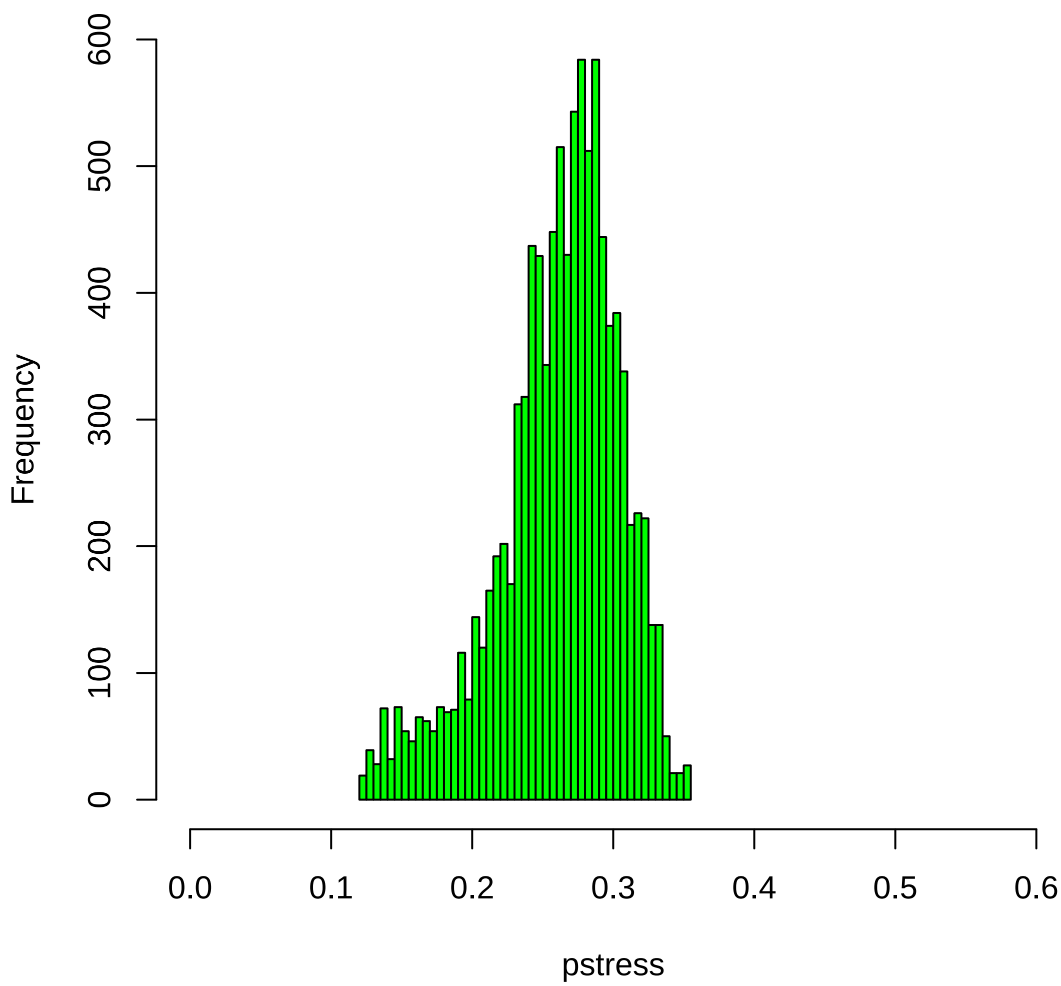


Figure 13: Stress Histogram for Plato, 10000 runs

```
        d <- abs(outer(x, x, "-"))
        s[i, j] <- sum((delta - d) ^ 2) / 2
      }
    }
    return(s)
  }
  s <- funk(a, b)
  persp(
    a,
    b,
    s,
    theta = 30,
    phi = 30,
    zlab = "stress",
    col = "RED",
    ticktype = "detailed"
  )
  return()
}

mplot <- function(delta) {
  delta <- delta / sqrt(sum(delta ^ 2) / 2.0)
  par(pty = "s")
  plot(
    0,
    xlim = c(-.6, .6),
    ylim = c(-.6, .6),
    xlab = "a",
    ylab = "b",
    type = "n"
  )
  abline(0, -sqrt(3) / 3, col = "RED", lwd = "3")
  abline(0, sqrt(3) / 3, col = "RED", lwd = "3")
  abline(v = 0, col = "RED", lwd = "3")
  z <- matrix(c(-1, 1, 0, -1, -1 , 2), 3, 2)
  z <- apply(z, 2, function(x)
    x / sqrt(sum(x ^ 2)))
  y <- matrix(0, 6, 2)
  saa <- sqrt(1 / 3)
  sbb <- saa / 2
  y[2, 1] <- saa
  y[5, 1] <- -saa
  y[1, ] <- c(sbb, .5)
  y[3, ] <- c(sbb, -.5)
```

```
  y[6, ] <- c(-sbb, .5)
  y[4, ] <- c(-sbb, -.5)
  points(y, pch = 16)
  text(.5 * y[1, 1], .5 * y[1, 2], "x1 < x2 < x3", cex = .5)
  text(.5 * y[2, 1], .5 * y[2, 2], "x1 < x3 < x2", cex = .5)
  text(.5 * y[3, 1], .5 * y[3, 2], "x3 < x1 < x2", cex = .5)
  text(.5 * y[4, 1], .5 * y[4, 2], "x3 < x2 < x1", cex = .5)
  text(.5 * y[5, 1], .5 * y[5, 2], "x2 < x3 < x1", cex = .5)
  text(.5 * y[6, 1], .5 * y[6, 2], "x2 < x1 < x3", cex = .5)
  cir <- seq(-2 * pi, 2 * pi, length = 100)
  lines(sqrt(1 / 3.0) * sin(cir), sqrt(1 / 3.0) * cos(cir))
  x <- tcrossprod(z, y)
  for (i in 1:6) {
    v <- x[, i]
    s <- sign(outer(v, v, "-"))
    t <- rowSums(s * delta) / 3.0
    a <- drop(t %*% z)
    s <- sign(outer(t, t, "-"))
    t <- rowSums(s * delta) / 3.0
    b <- drop(t %*% z)
    s <- sign(outer(t, t, "-"))
    t <- rowSums(s * delta) / 3.0
    c <- drop(t %*% z)
    points(matrix(a, 1, 2), pch = 16)
    points(matrix(b, 1, 2), pch = 16)
    points(matrix(c, 1, 2),
           col = "BLUE",
           cex = 1.5,
           pch = 16)
    lines(rbind(y[i,], a), lwd = 2)
    lines(rbind(a, b), lwd = 2)
    lines(rbind(a, c), lwd = 2)
  }
}


cplot <- function(delta) {
  par(pty = "s")
  delta <- delta / sqrt(sum(delta ^ 2) / 2.0)
  a <- seq(-.6, .6, length = 100)
  b <- seq(-.6, .6, length = 100)
  h <- matrix(0, 100, 100)
  z <- matrix(c(-1, 1, 0, -1, -1 , 2), 3, 2)
  z <- apply(z, 2, function(x)
```

```
    x / sqrt(sum(x ^ 2)))
  for (i in 1:100) {
    for (j in 1:100) {
      x <- a[i] * z[, 1] + b[j] * z[, 2]
      d <- abs(outer(x, x, "-"))
      h[i, j] <- sum((delta - d) ^ 2) / 2
    }
  }
  contour(a, b, h, nlevels = 100, col = "BLUE")
  abline(v = 0, col = "RED", lwd = 2)
  abline(0, -sqrt(3) / 3, col = "RED", lwd = "3")
  abline(0, sqrt(3) / 3, col = "RED", lwd = "3")
  cir <- seq(-2 * pi, 2 * pi, length = 100)
  lines(sqrt(1 / 3.0) * sin(cir), sqrt(1 / 3.0) * cos(cir))
}
```

B.2 udsmm.R

```
udsmm <- function (delta, w , x, itmax = 10, eps = 1e-10, verbose = TRUE) {
  n <- length(x)
  delta <- delta / sqrt(sum(w * (delta ^ 2)) / 2)
  xold <- x - mean(x)
  dold <- abs(outer(xold, xold, "-"))
  fold <- sum(w * (delta - dold) ^ 2) / 2
  v <- -w
  diag(v) <- -rowSums(v)
  vv <- solve(v + (1 / n)) - (1 / n)
  itel <- 1
  repeat {
    sold <- sign(outer(xold, xold, "-"))
    xnew <- drop(vv %*% rowSums(w * delta * sold))
    dnew <- abs(outer(xnew, xnew, "-"))
    fnew <- sum(w * (delta - dnew) ^ 2) / 2
    tnrm <- sum(xnew * (v %*% xnew))
    if (verbose) {
      cat("itel ", formatC(itel, digits = 2, format = "d"),
          "fold ", formatC(fold, digits = 10, format = "f"),
          "fnew ", formatC(fnew, digits = 10, format = "f"),
          "tnrm ", formatC(tnrm, digits = 10, format = "f"),
          "\n"
          )
    }
    if (((fold - fnew) < eps) || (itel == itmax)) {
```

```
      break
    }
    itel <- itel + 1
    xold <- xnew
    fold <- fnew
  }
  return(list(x = xnew, f = fnew, itel = itel))
}
```

B.3 nextperm.R

```
next.perm <-
  function(x)
    .C("permNext", as.double(x), as.integer(length(x)))[[1]]

are.monotone <- function(delta, w, x, y) {
  n <- nrow(delta)
  mon <- TRUE
  for (j in 1:(n - 1)) {
    for (i in (j + 1):n) {
      dx <- sign(x[i] - x[j])
      dy <- sign(y[i] - y[j])
      if ((w[i, j] * delta[i, j] * dx * dy) < 0) {
        mon <- FALSE
      }
    }
  }
  return(mon)
}

uniscale <-
  function(delta,
           w = 1 - diag(nrow(delta)),
           verbose = FALSE) {
    n <- nrow(delta)
    delta <- delta / sqrt(sum(w * (delta ^ 2)) / 2)
    m <- 0
    k <- 0
    fmin <- Inf
    x <- 1:n
    v <- -w
    diag(v) <- -rowSums(v)
    v <- solve(v + (1 / n)) - (1 / n)
    repeat {
```

```r
      k <- k + 1
      s <- sign(outer(x, x, "-"))
      t <- drop(v %*% rowSums(delta * w * s))
      if (are.monotone(delta, w, x, t)) {
        m <- m + 1
        d <- abs(outer(t, t, "-"))
        f <- sum(w * (delta - d) ^ 2) / 2
        if (verbose) {
          print(c(m, f, fmin, max(t), min(t)))
        }
        if (f < fmin) {
          fmin <- f
          xmin <- t
        }
      }
      if (all(x == (n:1)))
        return(list(
          xmin = xmin,
          fmin = fmin,
          m = m,
          k = k
        ))
      x <- next.perm(x)
    }
  }
```

B.4 nextPerm.c

```c

void swap(double *, int, int);
void permNext(double *, int *);

void swap(double *x, int i, int j) {
    double temp;
    temp = x[i];
    x[i] = x[j];
    x[j] = temp;
}

void permNext(double *x, int *nn) {
    int n = *nn;
    int i = n - 1;
    while (x[i - 1] >= x[i]) i--;
    if (i == 0) return;
```

```
    int j = n;
    while (x[j - 1] <= x[i - 1]) j--;
    swap(x, i - 1, j - 1);
    j = n;
    i++;
    while (i < j) {
        swap(x, i - 1, j - 1);
        j--;
        i++;
    }
}
```

References

Bacak, M., and J. M. Borwein. 2011. "On Difference Convexity of Locally Lipschitz Functions." *Optimization* 60 (8-9): 961–78.

Borg, I., and P. J. F. Groenen. 2005. *Modern Multidimensional Scaling*. Second Edition. Springer.

Borg, I., P. J. F. Groenen, and P. Mair. 2018. *Applied Multidimensional Scaling and Unfolding*. Springer.

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.

———. 1994. "Block Relaxation Algorithms in Statistics." In *Information Systems and Data Analysis*, edited by H. H. Bock, W. Lenski, and M. M. Richter, 308–24. Berlin: Springer Verlag. https://jansweb.netlify.app/publication/deleeuw-c-94-c/deleeuw-c-94-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.

———. 2019. "Global Minima by Penalized Full-dimensional Scaling." 2019. https://jansweb.netlify.app/publication/deleeuw-e-19-e/deleeuw-e-19-e.pdf.

De Leeuw, J., and W. J. Heiser. 1977. "Convergence of Correction Matrix Algorithms for Multidimensional Scaling." In *Geometric Representations of Relational Data*, edited by J. C. Lingoes, 735–53. Ann Arbor, Michigan: Mathesis Press.

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.

Defays, D. 1978. "A Short Note on a Method of Seriation." *British Journal of Mathematical and Statistical Psychology* 31: 49–53.

Good, I. J. 1975. "The Number of Orderings of n Candidates when Ties Are Permitted." *Fibonacci Quarterly* 13 (1): 11–18.

Guttman, L. 1968. "A General Nonmetric Technique for Fitting the Smallest Coordinate Space for a Configuration of Points." *Psychometrika* 33: 469–506.

Heiser, W. J. 1995. "Convergent Computing by Iterative Majorization: Theory and Applications in Multidimensional Data Analysis." In *Recent Advantages in Descriptive Multivariate Analysis*, edited by W. J. Krzanowski, 157–89. Oxford: Clarendon Press.

Hiriart-Urruty, J.-B. 1988. "Generalized Differentiability / Duality and Optimization for Problems Dealing with Differences of Convex Functions." In *Convexity and Duality in Optimization*, edited by Ponstein. J., 37–70. Lecture Notes in Economics and Mathematical Systems 256. Springer.

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.

Lange, K. 2016. *MM Optimization Algorithms*. SIAM.

Lange, K., D. R. Hunter, and I. Yang. 2000. "Optimization Transfer Using Surrogate Objective Functions." *Journal of Computational and Graphical Statistics* 9: 1–20.

Mair, P., and J. De Leeuw. 2015. "Unidimensional Scaling." In *Wiley StatsRef: Statistics Reference Online*, 1–3. Wiley.

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.

Pliner, V. 1986. "The Problem of Multidimensional Metric Scaling." *Automation and Remote Control* 47: 560–67.

Taussky, O. 1949. "A Recurring Theorem on Determinants." *American Mathematical Monthly* 56: 672–76.