Robust Least Squares Multidimensional Scaling

Jan de Leeuw

October 2, 2024

We use an iteratively reweighted version of the smacof algorithm to minimize various robust multidimensional scaling loss functions. Our results use a general theorem on sharp quadratic majorization of De Leeuw and Lange (2009). We relate this theorem to earlier results in robust statistics, localization theory, and sparse recovery. Code in R is included.

```
source("smacofRobust.R")
```

1 Introduction

The title of this chapter seems something paradoxical. Least squares estimation is typically not robust, it is sensitive to outliers and pays a lot of attention to fitting the larger observations. What we mean by robust least squares MDS, however, is using the smacof machinery designed to minimize loss of the form

$$\sigma_2(X) := \sum w_k(\delta_k - d_k(X))^2, \tag{1}$$

to minimize robust loss functions. The prototypical robust loss function is

$$\sigma_1(X) := \sum w_k|\delta_k - d_k(X)|, \tag{2}$$

which we will call *strife*, because stress, sstress, and strain are already taken.

Strife is not differentiable at configurations X for which there is at least one k for which either $d_k(X) = \delta_k$ or $d_k(X) = 0$ (or both). This lack of differentiability complicates the minimization problem. Moreover experience with one-dimensional and city block MDS suggests that having many points where the loss function is not differentiable leads to (many) additional local minima.

In this chapter we will discuss (and implement) various variations of σ_1 from (2). They can be interpreted in two different ways. On the one hand we use smoothers of the absolute value function, and consequently of strife. We want to eliminate the problems with differentiability, at least the ones caused by $\delta_k = d_k(X)$. If this is our main goal, then we want to choose the smoother in such a way that it is as close to the absolute value function as possible. This is not unlike the distance smoothing used by Pliner (1996) and Groenen, Heiser, and Meulman (1999) in the global minimization of σ_2 from (1).

On the other hand our modified loss function can be interpreted as more robust versions of the least squares loss function, and consequently of stress. Our goal here is to combines the robustness of the absolute value function with the efficiency and computational ease of least squares. If that is our goal then there is no reason to stay as close to the absolute value function as possible.

Our robust or smooth loss functions are all of the form

$$\sigma(X) := \sum w_k \, f(\delta_k - d_k(X)), \tag{3}$$

for a suitable choice of the real valued function f. We will define what we mean by "suitable" later on. For now, note that loss (1) is the special case with $f(x) = x^2$ and loss (2) is the special case with $f(x) = |x|$.

2 Majorizing Strife

The pioneering work in strife minimization using smacof is Heiser (1988), building on earlier work in Heiser (1987). It is based on a creative use of the Arithmetic Mean-Geometric Mean (AM/GM) inequality to find a majorizer of the absolute value function. For the general theory of majorization algorithms (now more commonly known as MM algorithms) we refer to their original introduction in De Leeuw (1994) and to the excellent book by Lange (2016).

The AM/GM inequality says that for all non-negative x and y we have

$$|x||y| = \sqrt{x^2y^2} \leq \frac{1}{2}(x^2 + y^2), \tag{4}$$

with equality if and only if $x = y$. If $y > 0$ we can write (4) as

$$|x| \leq \frac{1}{2}\frac{1}{|y|}(x^2 + y^2), \tag{5}$$

and this provides a quadratic majorization of $|x|$ at y. There is no quadratic majorization of $|x|$ at $y = 0$, which is a nuisance we must deal with.

Using the majorization (5), and assuming $\delta_k \neq d_k(Y)$ for all k, we define

$$\omega_1(X) := \frac{1}{2}\sum w_k \frac{1}{|\delta_k - d_k(Y)|}((\delta_k - d_k(Y))^2 + (\delta_k - d_k(X))^2). \tag{6}$$

Now $\sigma_1(X) \leq \omega_1(X)$ for all X and $\sigma_1(Y) = \omega_1(Y)$. Thus ω_1 majorizes σ_1 at Y.

2.1 Algorithm

Define

$$w_k(Y) := w_k \frac{1}{|\delta_k - d_k(Y)|}. \tag{7}$$

Reweighted smacof to minimize strife computes $X^{(k+1)}$ by decreasing

$$\sum w_k(X^{(k)})(\delta_k - d_k(X^{(k)}))^2, \tag{8}$$

using a standard smacof step. It then computes the new weights $w_k(X^{(k+1)})$ from (7) and uses them in the next smacof step to update $X^{(k+1)}$. And so on, until convergence.

A straightforward variation of the algorithm does a number of smacof steps before upgrading the weights. This still leads to a monotone, and thus convergent, algorithm. How many smacof steps we have to take in the inner iterations is something that needs further study. It is likely to depend on the fit of the data, on the shape of the function near the local minimum, and on how far the iterations are from the local minimum.

2.2 Zero Residuals

It may happen that for some k we have $d_k(X^{(k)}) = \delta_k$ while iterating. There have been various proposals to deal with such an unfortunate event, and we will discuss some of them further on. Even more importantly we will see that that the minimizer of the absolute value loss usually satisfies $d_k(X) = \delta_k$ for quite a few elements, which means that near convergence the algorithm will become unstable because the weights from (7) become very large.

To illustrate the problems with differentiability we compute the directional derivatives of strife.

Let $s_k(X) := w_k|d_k(X) - \delta_k|$.

1. If $\delta_k = 0$ and $d_k(X) = 0$ then $ds_k(X;Y) = w_k d_k(Y)$.
2. If $\delta_k > 0$ and $d_k(X) = 0$ then $ds(X;Y) = -w_k d_k(Y)$.
3. If $d_k(X) > 0$ and $d_k(X) - \delta_k > 0$ then $ds_k(X;Y) = w_k \frac{1}{d_k(X)}\text{tr } X'A_kY$.
4. If $d_k(X) > 0$ and $d_k(X) - \delta_k < 0$ then $ds_k(X;Y) = -w_k \frac{1}{d_k(X)}\text{tr } X'A_kY$.
5. If $d_k(X) > 0$ and $d_k(X) - \delta_k = 0$ then $ds_k(X;Y) = w_k \frac{1}{d_k(X)}|\text{tr } X'A_kY|$.

The directional derivative of σ_1 is consequently the sum of five terms, corresponding with each of these five cases.

In the case of stress the directional derivatives could be used to prove that if $w_k\delta_k > 0$ for all k then stress is differentiable at each local minimum (De Leeuw (1984)). For strife to be differentiable we would have to prove that at a local minimum both $d_k(X) > 0$ and $(d_k(X) - \delta_k) \neq 0$. for all k with $w_k > 0$.

But this is impossible by the following argument. In the one-dimensional case we can partition $\mathbb{R}^n$ into $n!$ polyhedral convex cones corresponding with the permutations of x. Within each cone the distances are a linear function of x. Each cone can be partitioned by intersecting it with the $2^{\binom{n}{2}}$ polyhedra defined by the inequalities $\delta_k - d_k(x) \geq 0$ or $\delta_k - d_k(x) \leq 0$. Some of these intersections can and will obviously be empty. Within each of these non-empty polyhedral regions strife is a linear function of x. Thus it attains its minimum at a vertex of the region, which is a solution for which some distances are zero and some residuals are zero. There can be no
minima, local or global, in the interior of one of the polyhedral regions. We have shown that in one dimension strife is not differentiable at a local minimum, and that there is presumably a large number of them. Of course even for moderate n the number of regions, which is maximally $n!2^{\binom{n}{2}}$, is too large to actually compute or draw.

In the multidimensional case linearity goes out the window. The set of configurations $d_k(X) = \delta_k$ is an ellipsoid and $d_k(X) = 0$ is a hyperplane. Strife is not differentiable at all intersections of these ellipsoids and hyperplanes. The partitioning of $\mathbb{R}^n$ by these ellipsoids and hyperplanes

is not simple to describe. It has convex and non-convex cells, and within each cell strife is the difference of two weighted sums of distances. Anything can happen.

3 Generalizing Strife

A function g *majorizes* a function f at y if $g(x) \geq f(x)$ for all x and $g(y) = f(y)$. Majorization is *strict* if $g(x) > f(x)$ for all $x \neq y$. If $\mathfrak{H}$ is a family of functions that all majorize f at y then $h \in \mathfrak{H}$ is a *sharp majorization* if $h(x) \leq g(x)$ for all $g \in \mathfrak{H}$. The AM/GM inequality was used in the previous section to construct a quadratic majorization of strife.

We are specifically interested in this chapter in sharp quadratic majorization, in which $\mathfrak{H}$ is the set of all convex quadratics that majorize f at y. This case has been studied in detail (in the case of real-valued functions on the line) by De Leeuw and Lange (2009). Their Theorem 4.5 on page 2478 says

> Theorem 4.5: Suppose $f(x)$ is an even, differentiable function on $\mathbb{R}$ such that the ratio $f'(x)/x$ is non-increasing on $(0, \infty)$. Then the even quadratic
>
> $$g(x) = \frac{f'(y)}{2y}(x^2 - y^2) + f(y) \tag{9}$$
>
> is a sharp quadratic majorizer of f at the point y.
>
> Theorem 4.6. The ratio $f'(x)/x$ is decreasing on $(0, \infty)$ if and only $f(\sqrt{(}x))$ is concave. The set of functions satisfying this condition is closed under the formation of (a) positive multiples, (b) convex combinations, (c) limits, and (d) composition with a concave increasing function $g(x)$.

Note that these theorems give a sufficient condition for quadratic majorization (in fact, for sharp quadratic majorization) and not a necessary one. Quadratic majorization may still be possible if the conditions in the theorem are violated.

Although De Leeuw and Lange (2009) give no references

We now apply this theorem to functions of the form

$$\sigma_f(X) := \sum w_k \, f(\delta_k - d_k(X)), \tag{10}$$

where f satisfies the conditions in the theorem. If

$$\omega_f(X) := \sum w_k \frac{f'(\delta_k - d_k(Y))}{2(\delta_k - d_k(Y))}\{(\delta_k - d_k(X))^2 - (\delta_k - d_k(Y))^2\} + f(\delta_k - d_k(Y)), \tag{11}$$

then ω_f is a sharp quadratic majorization at Y.

Although the absolute value is not differentiable at the origin the theorem can still be applied. It just does not give a majorizer at $y = 0$. If $f(x) = |x|$ then

$$g(x) = \frac{1}{2|y|}(x^2 - y^2) + |y| = \frac{1}{2|y|}(x^2 + y^2), \tag{12}$$

which is the same as (5). Thus the AM/GM method gives a sharp quadratic majorization.

In iteration k the robust smacof algorithm does a smacof step towards minimization of ω_f over X. We can ignore the parts of (11) that only depend on Y, and minimize

$$\sum w_k(X^{(k)})(\delta_k - d_k(X))^2, \tag{13}$$

with

$$w_k(X^{(k)}) := w_k \frac{f'(\delta_k - d_k(X^{(k)}))}{2(\delta_k - d_k(Y))}. \tag{14}$$

It then recomputes the weights $w_k(X^{(k+1)})$ and goes to the smacof step again. This can be thought of as iteratively reweighted least squares (IRLS), and also as nested majorization, with the smacof majorization within the sharp quadratic majorization of the loss function.

4 On IRLS

The history of iterative reweighted least squares (IRLS), as applied to fitting functions to data, is rather complicated. It is mostly used in fitting linear models, and in minimizing ℓ_p power loss.

Weiszfeld

5 Charbonnier loss

The first, and perhaps most obvious, choice for smoothing the absolute value function is

$$f_c(x) = \sqrt{x^2 + c^2}. \tag{15}$$

This smoother was previously used by De Leeuw (2018) in least absolute value regression and in De Leeuw (2020) in what was called least squares absolute value regression.

In the figure below we show the loss function for $c = 1$ (black), $c = 0.1$ (red), $c = 0.01$ (blue), and $c = 0.001$ (green).

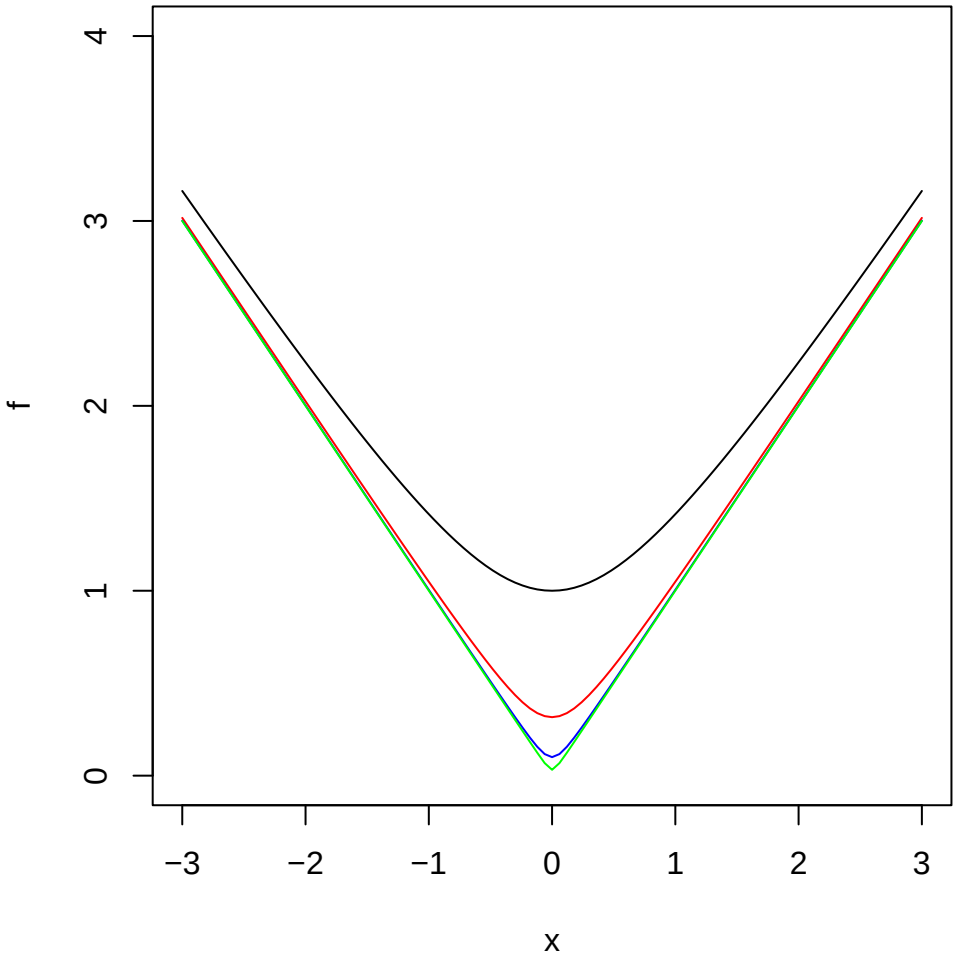

Figure 1: Chardonnier Loss

For $c > 0$ we have $f_c(x) > |x|$. If $c \to 0$ then $f_c(x)$ decreases monotonically to $|x|$. Also $\min_x |f_c(x) - |x|| = c$ attained at $x = 0$, which implies uniform convergence of f_c to $|x|$.

In the engineering literature (15) is known as Charbonnier loss, after Charbonnier et al. (1994), who were possibly the first engineers to use it. x Ramirez et al. (2014) argue (15) is also the "most computationally efficient smooth approximation to $|x|$".

By l'Hôpital

$$\lim_{x\to 0}\frac{\sqrt{x^2+c^2}-c}{\frac{1}{2}x^2}=1.$$

Of course also

$$\lim_{x\to\infty}\frac{\sqrt{x^2+c^2}}{|x|}=1$$

and

$$\lim_{x\to\pm\infty}\sqrt{x^2+c^2}-|x|=0$$

Thus if x is much smaller than c loss is approximately a quadratic in x, and if x is much larger than c then loss is approximately the absolute value.

Loss function (15) is infinitely many times differentiable. Its first derivative is

$$f_c'(x)=\frac{1}{\sqrt{x^2+c^2}}x,$$

which converges, again in the sup-norm and uniformly, to the sign function if $c\to 0$. The IRLS weights are

$$w_c(x)=\frac{1}{\sqrt{x^2+c^2}}$$

which is clearly a decreasing function of x on $\mathbb{R}^+$.

$$\sigma_c(X):=\sum w_k\sqrt{(\delta_k-d_k(X))^2+c^2}$$

Now majorization using

$$\omega(X,X^{(k)}):=\sum w_k w_c(\delta_k-d_k(X^{(k)})(\delta_k-d_k(X))^2$$

6 Generalized Charbonnier Loss

The loss function $(x^2+c^2)^{\frac{1}{2}}$ smoothes $|x|$. IN the same generalized Charbonnier loss smoothes ℓ_p loss $|x|^q$.

$$f_{c,q}(x) := (x^2+c^2)^{\frac{1}{2}q}$$

$$w_{c,q}(x) = q(x^2+c^2)^{\frac{1}{2}q-1}$$

which is non-increasing for $q \leq 2$. Note that we do not assume that $q > 0$, and consequently … provides us with much more flexibility than Charbonnier loss …

There are a large number of generalizations of these types of loss functions in the engineering community, and in their maze of conference publications. We will discuss the nice generalization in Barron (2019).

$$f_{\alpha,c}(x) = \frac{|\alpha-2|}{\alpha}\left(\left(\frac{(x/c)^2}{|\alpha-2|}+1\right)^{\alpha/2}-1\right)$$

7 Convolution Smoothers

General

7.1 Huber Loss

The Huber function (Huber (1964)) is

$$f(x) = \begin{cases} \frac{1}{2}x^2 & \text{if } |x| < c, \\ c|x| - \frac{1}{2}c^2 & \text{otherwise.} \end{cases}$$

Because … Chardonnier loss is also known as Pseudo-Huber loss.

The Huber function is differentiable, although not twice diffentiable. Its derivative is

$$f'(x) = \begin{cases} c & \text{if } x \geq c, \\ x & \text{if } |x| \leq c, \\ -c & \text{if } x \leq -c. \end{cases}$$

$$w(x) = \begin{cases} \frac{c}{x} & \text{if } x \geq c, \\ 1 & \text{if } |x| \leq c, \\ -\frac{c}{x} & \text{if } x \leq -c. \end{cases}$$

The Huber function is even and differentiable. Moreover $f'(x)/x$ decreases from. Thus the theorem applies and the sharp quadratic majorizer at y is

$$g(x) = \{$$

$$\sigma_k(X) = \begin{cases} \frac{1}{2}(\delta_k - d_k(X))^2 & \text{if } |\delta_k - d_k(X)| < c, \\ c|\delta_k - d_k(X)| - \frac{1}{2}c^2 & \text{if } |\delta_k - d_k(X)| \geq c. \end{cases}$$

$$\omega_k(x, y) = \begin{cases} \frac{1}{2}\frac{c}{|y|}(x^2 - y^2) - cy - \frac{1}{2}c^2 & \text{if } y \leq -c, \\ \frac{1}{2}x^2 & \text{if } |y| < c, \\ \frac{1}{2}\frac{c}{|y|}(x^2 - y^2) + cy - \frac{1}{2}c^2 & \text{if } y \geq +c. \end{cases}$$

Now $x = \delta_k - d_k(X)$ and $y = \delta_k - d_k(Y)$

$$\omega_k(X;Y)=\begin{cases}\frac{1}{2}\frac{c}{|\delta_k-d_k(Y)|}\{(\delta_k-d_k(X))^2+(d_k(Y)-\delta_k)^2\}-c(\delta_k-d_k(Y))-\frac{1}{2}c^2 & \text{if } \delta_k-d_k(Y)\leq\\ \frac{1}{2}(\delta_k-d_k(X))^2 & \text{if } |\delta_k-d_k(Y)|<\\ \frac{1}{2}\frac{c}{|\delta_k-d_k(Y)|}\{(\delta_k-d_k(X))^2+(d_k(Y)-\delta_k)^2\}+c(\delta_k-d_k(Y))-\frac{1}{2}c^2 & \text{if } \delta_k-d_k(Y)\geq\end{cases}$$

Thus the MDS majorization algorithm for the Huber loss is to update Y by minimizing (or by performing one smacof step to decrease)

$$\sum w_k(Y)(\delta_k-d_k(X))^2$$

where

$$w_k(Y)=\begin{cases}w_k & \text{if } |\delta_k-d_k(Y)|<c,\\ \frac{cw_k}{|\delta_k-d_k(Y)|} & \text{otherwise.}\end{cases}$$

7.2 Gaussian Convolution

In De Leeuw (2018) we also used the convolution smoother proposed by Voronin, Ozkaya, and Yoshida (n.d.). The idea is to use the convolution of the absolute value function and a *mollifier* as the smoothed function.

> A smooth function $\psi:\mathbb{R}\to\mathbb{R}$ is said to be a pdf if it is non-negative, and has area $\int\psi(x)dx=1$. For any pdf ψ and any $c>0$, define the parametric function $\psi_c:\mathbb{R}\to\mathbb{R}$ by: $\psi_c(x):=\frac{1}{c}\psi(\frac{1}{c})$, for all $x\in\mathbb{R}$. Then $\{\psi_c:c>0\}$ is a family of pdf's, whose support decreases as $c\to 0$, but the volume under the graph always remains equal to one.

choose a Gaussian pdf.

$$f(x)=\frac{1}{c\sqrt{2\pi}}\int_{-\infty}^{+\infty}|x-y|\exp\left\{-\frac{1}{2}(\frac{y}{c})^2\right\}dy$$

Carrying out the integration gives

$$f(x)=x\{2\Phi(x/c)-1\}+2c\phi(x/c).$$

The derivative is

$$f'(x)=2\Phi(x/c)-1$$

It may not be immediately obvious in this case that $f'(x)/x$ is decreasing. We prove that its derivative is negative on $(0,+\infty)$. The derivative of $f'(x)/x$ has the sign of $xf''(x)-f'(x)$,

which is $z\phi(z) - \Phi(z) + 1/2$, with $z = x/c$. It remains to show that $\Phi(z) - z\phi(z) \geq \frac{1}{2}$, or equivalently that $\int_0^z \phi(x)dx - z\phi(z) \geq 0$. Now if $0 \leq x \leq z$ then $\phi(x) \geq \phi(z)$ and thus $\int_0^z \phi(x)dx \geq \phi(z) \int_0^z dx = z\phi(z)$, which completes the proof.

$$w_k(Y) = \frac{\Phi((\delta_k - d_k(Y))/c) - \frac{1}{2}}{\delta_k - d_k(Y)}$$

Convolution with rectangular between c and -c gives the Huber function.

$$f(x) = \frac{1}{2c} \int_{-c}^{+c} |x - y| dy$$

$$f(x) = \frac{1}{2c} \int_{-c}^{+c} |x - y| dy = \begin{cases} \frac{1}{2c}(x^2 + c^2) & \text{if } |x| \leq c, \\ |x| & \text{otherwise.} \end{cases}$$

$$f'(x) = \begin{cases} \frac{1}{c}x & \text{if } |x| \leq c, \\ \text{sign}(x) & \text{otherwise.} \end{cases}$$

$$w(x) = \begin{cases} \frac{1}{c} & \text{if } |x| \leq c, \\ \frac{1}{|x|} & \text{otherwise.} \end{cases}$$

It is also clear that we can use any scale family of probability densities to define convolution smoothers. There is an infinite number of possible choices, with finite or infinite support, smooth or nonsmooth, using splines or wavelets, and so on.

8 Downweighters

8.1 Tukey Loss

8.2 Welsch Loss

$$f(x) = 1 - \exp(-\{\frac{x}{c}\}^2)$$

$$f'(x) = \frac{2}{c^2}\exp(-\{\frac{x}{c}\}^2)x$$

8.3 Cauchy Loss

$$f(x) = \log(\{\frac{x}{c}\}^2 + 1)$$

$$f'(x) = \frac{1}{c^2}x\frac{1}{\{\frac{x}{c}\}^2 + 1}$$

$$w(x) = \frac{1}{c^2}\frac{1}{\{\frac{x}{c}\}^2 + 1}$$

which is non-increasing on $\mathbb{R}^+$.

9 Example

The example we use are dissimilarities between nine Dutch political parties, collected by De Gruijter (1967).

```
delta
```

```
      KVP PvdA  VVD  ARP  CHU  CPN  PSP   BP  D66
KVP  0.00 5.63 5.27 4.60 4.80 7.54 6.73 7.18 6.17
PvdA 5.63 0.00 6.72 5.64 6.22 5.12 4.59 7.22 5.47
VVD  5.27 6.72 0.00 5.46 4.97 8.13 7.55 6.90 4.67
ARP  4.60 5.64 5.46 0.00 3.20 7.84 6.73 7.28 6.13
CHU  4.80 6.22 4.97 3.20 0.00 7.80 7.08 6.96 6.04
```

```
CPN  7.54 5.12 8.13 7.84 7.80 0.00 4.08 6.34 7.42
PSP  6.73 4.59 7.55 6.73 7.08 4.08 0.00 6.88 6.36
BP   7.18 7.22 6.90 7.28 6.96 6.34 6.88 0.00 7.36
D66  6.17 5.47 4.67 6.13 6.04 7.42 6.36 7.36 0.00
```

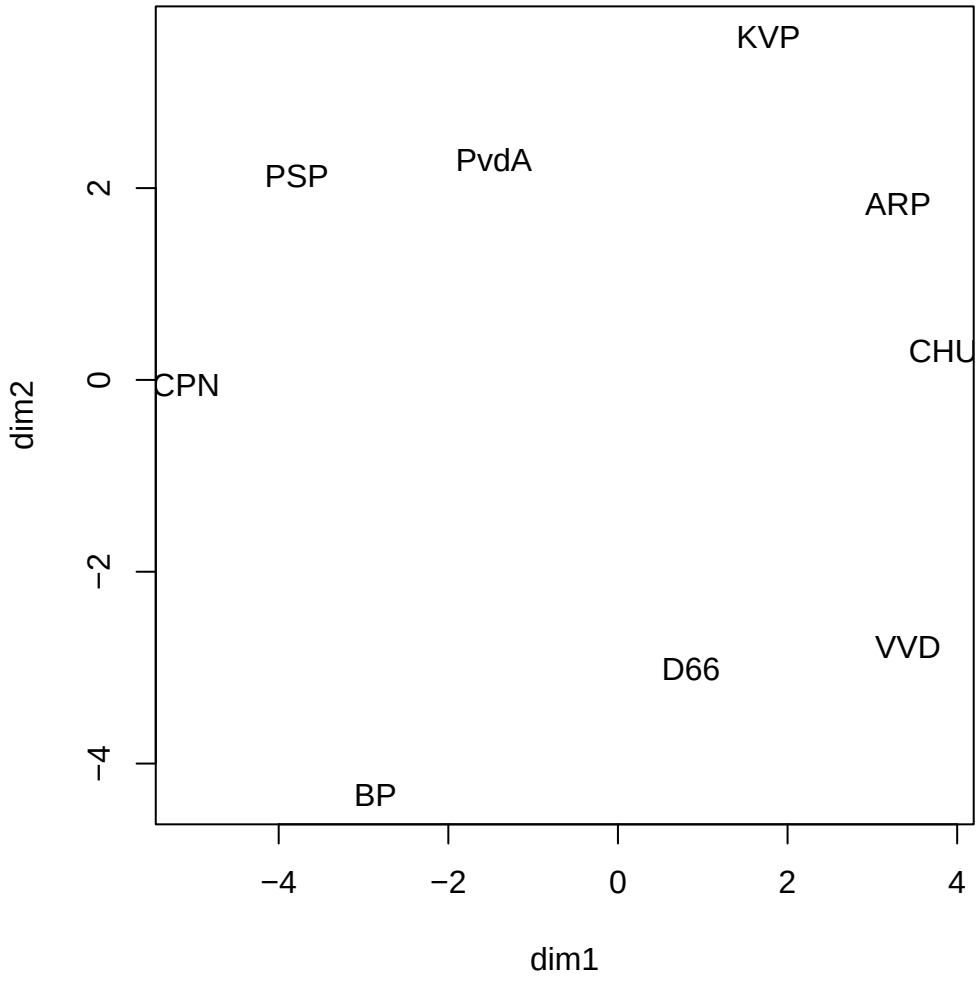

Figure 2: Configuration Least Squares

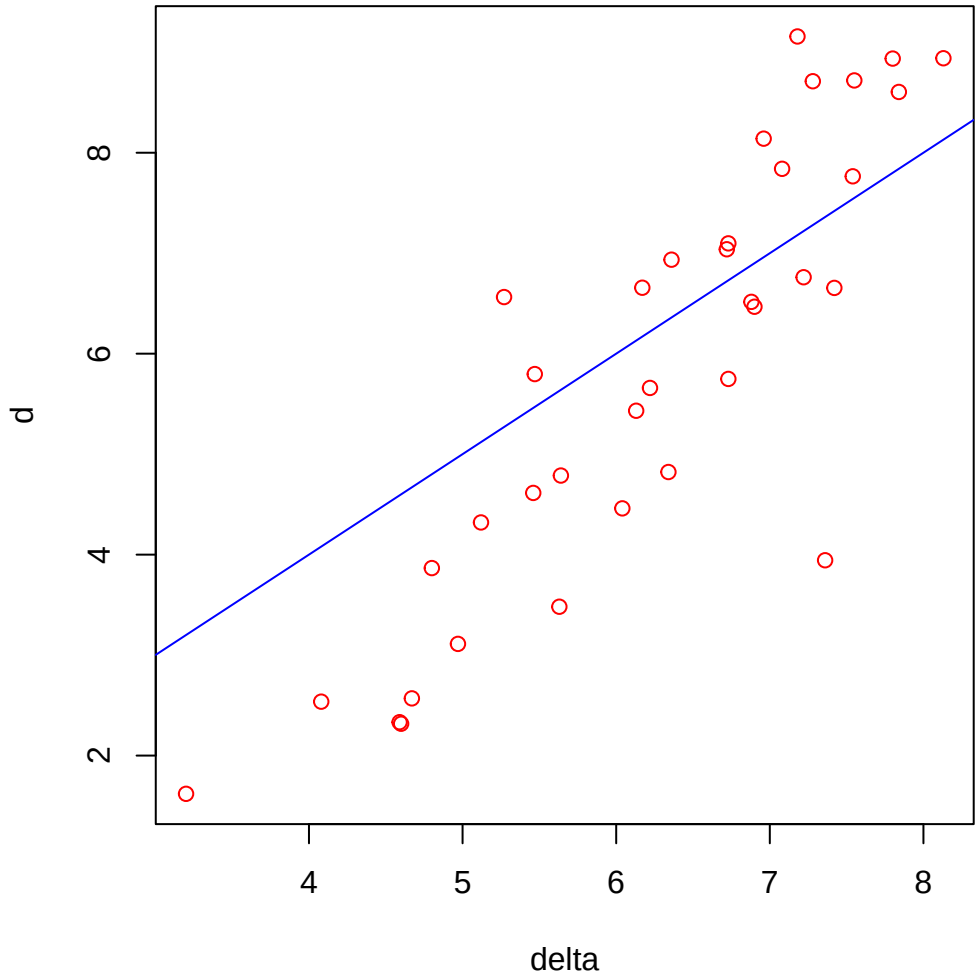

Figure 3: Shepard Plot Least Squares

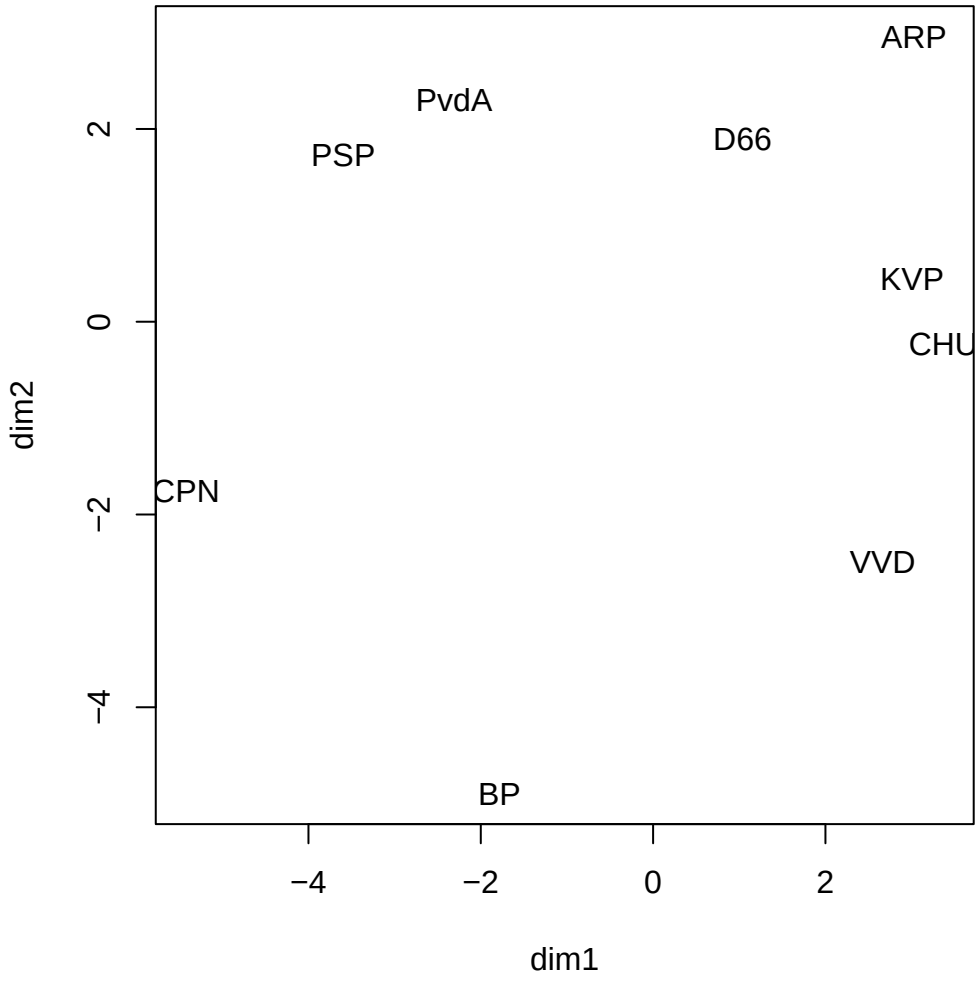

Figure 4: Configuration Least Absolute Value

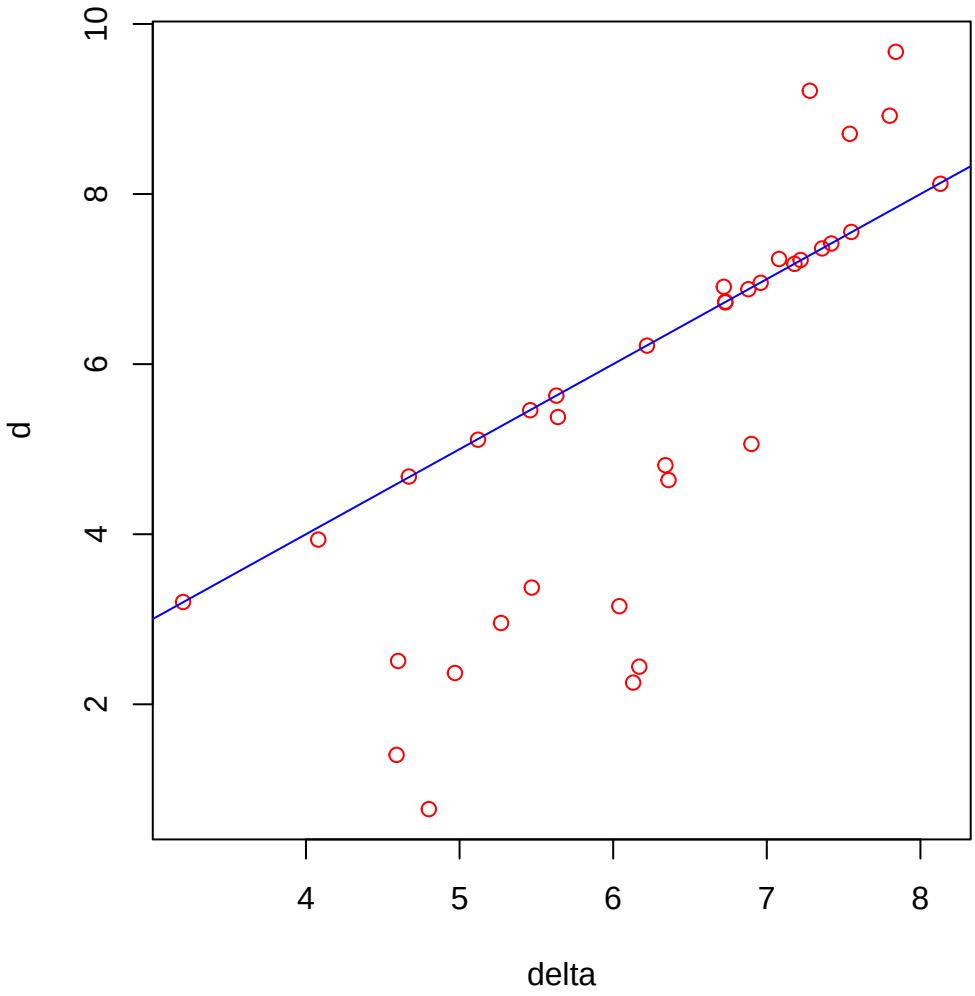

Figure 5: Shepard Plot Least Absolute Value

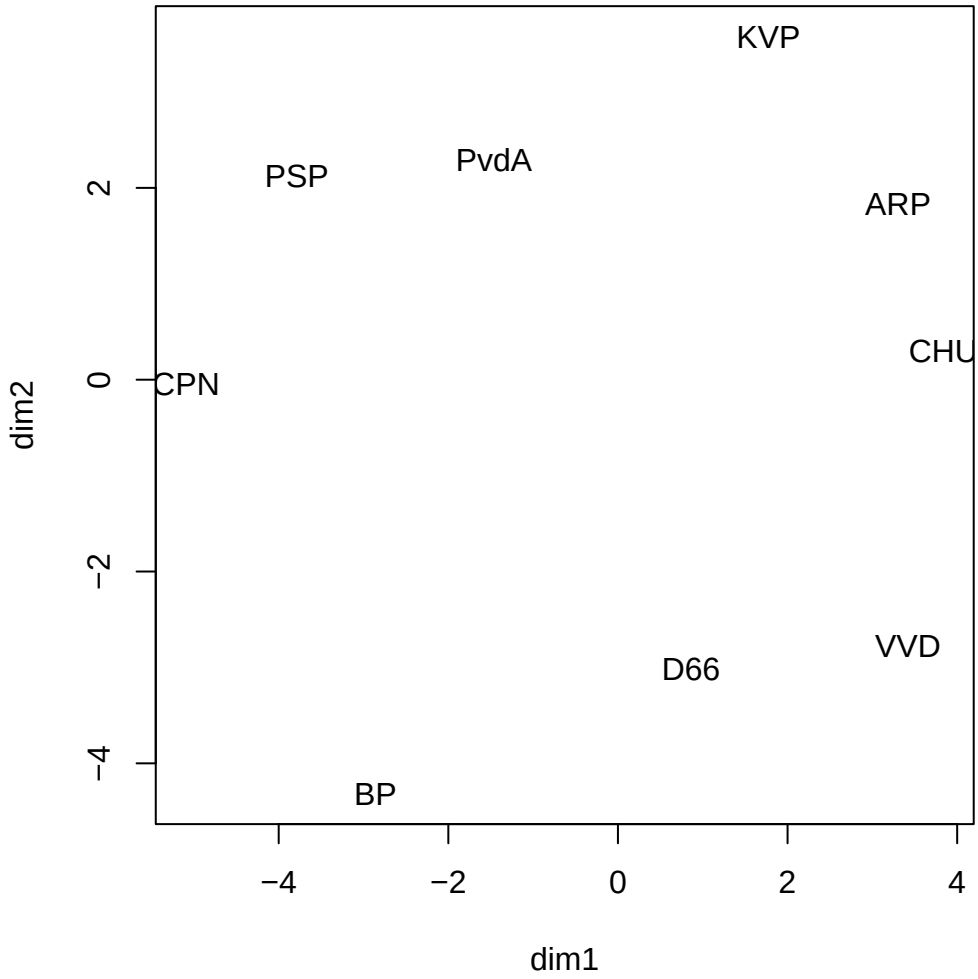

Figure 6: Configuration Median Huber

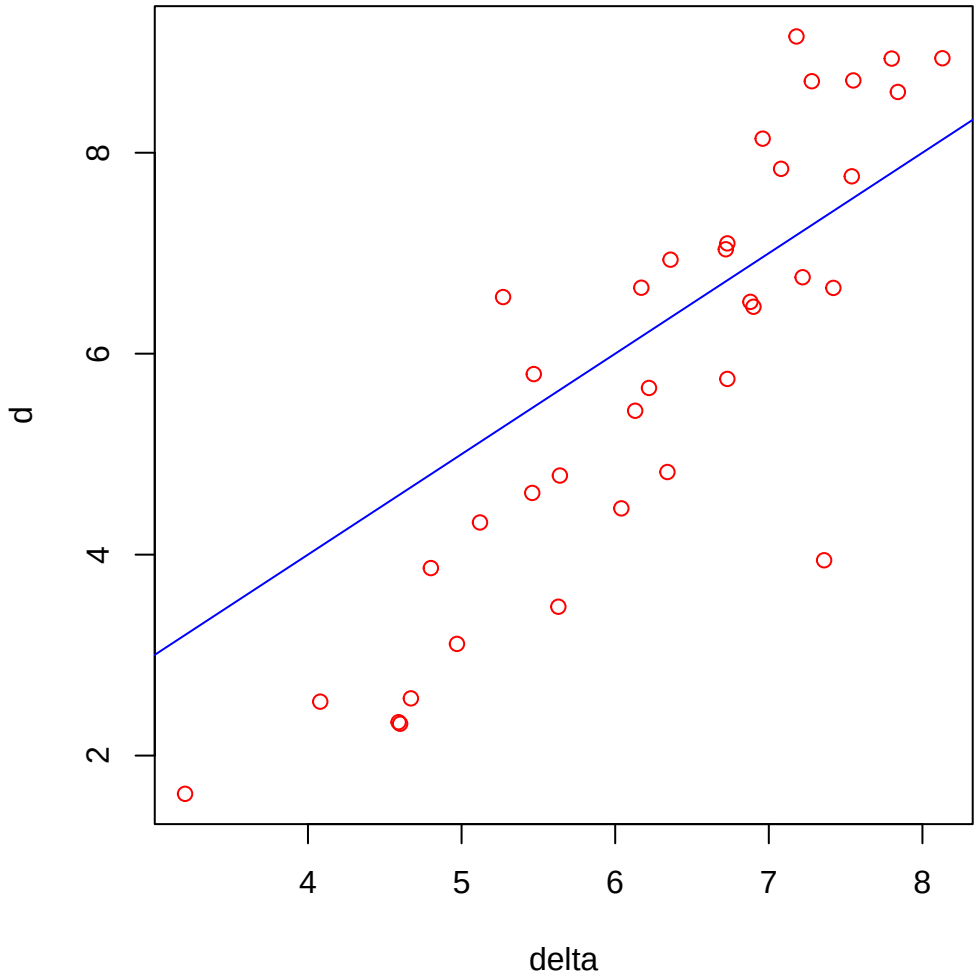

Figure 7: Shepard Plot Median Huber

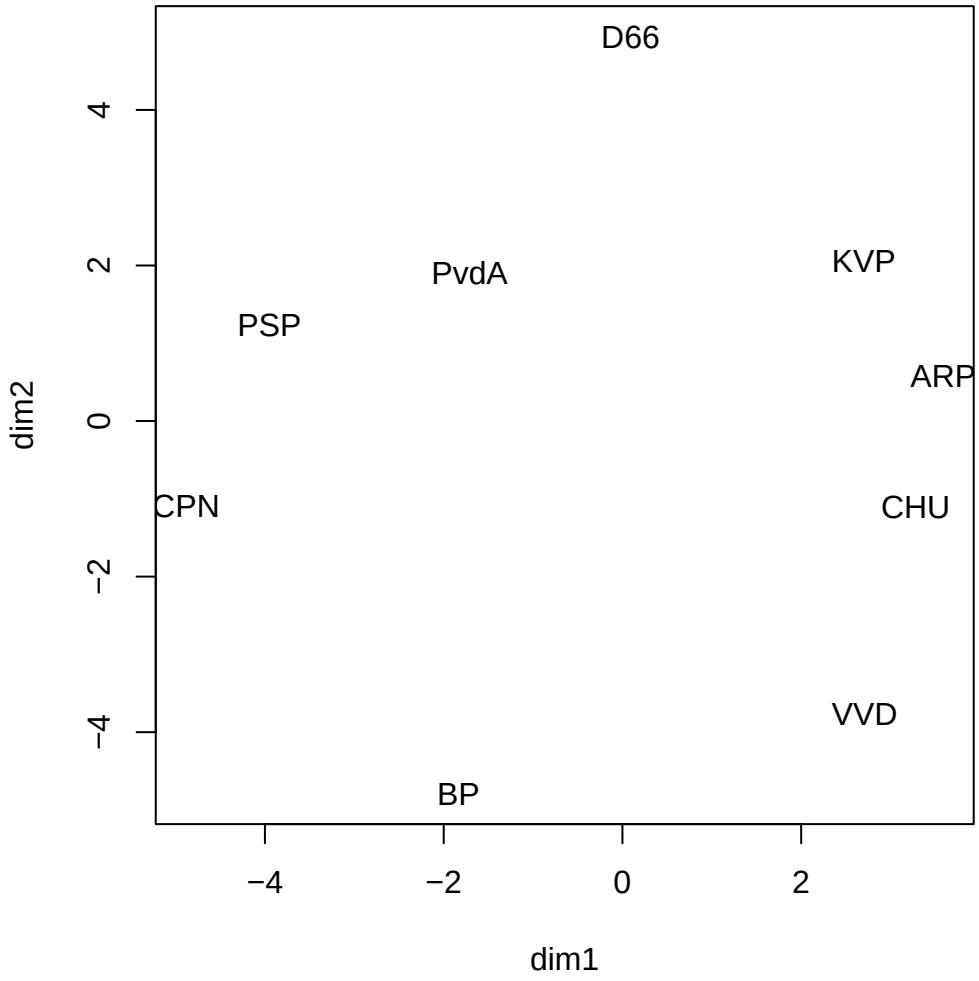

Figure 8: Configuration Median Tukey

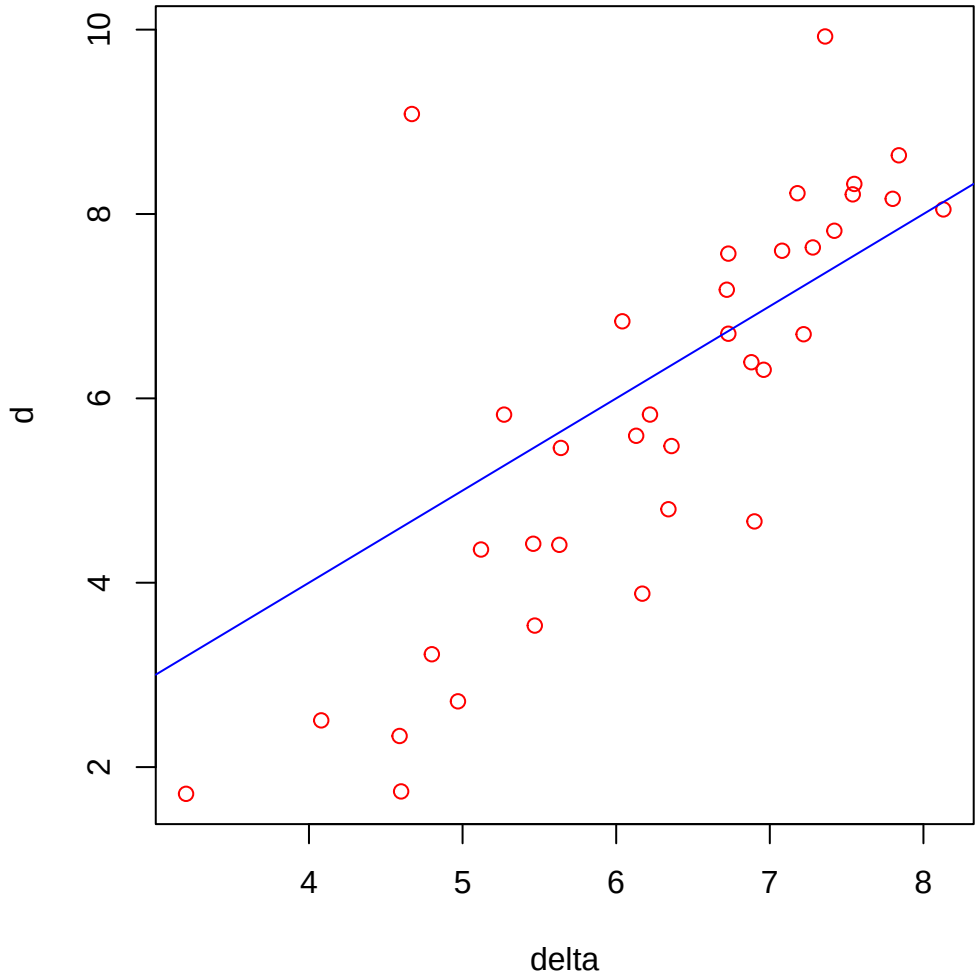

Figure 9: Shepard Plot Median Tukey

10 Discussion

Fixed weights

Minimize

$$\sum w_k \, f(\delta_k - d_k(X))$$

if $f''(x) \leq K$.

$$f(\delta_k - d_k(X)) \leq f(\delta_k - d_k(Y)) + f'(\delta_k - d_k(Y))(d_k(Y) - d_k(X)) + \frac{1}{2}K(d_k(Y) - d_k(X))^2$$

Minimize

$$\left[d_k(X) - \{d_k(Y)) - K^{-1} f'(\delta_k - d_k(Y))\}\right]^2$$

11 Code

The function smacofRobust has a parameter "engine", which can be equal to smacofAV, smacofHuber, smacofTukey, or smacofConvolution. These four small modules compute the respective loss function values and weights for the IRLS procedure. This makes it easy to add additional robust loss functions.

```
smacofRobust <- function(delta,
                         weights = 1 - diag(nrow(delta)),
                         ndim = 2,
                         xold = smacofTorgerson(delta, ndim),
                         engine = smacofAV,
                         cons = 0,
                         itmax = 1000,
                         eps = 1e-15,
                         verbose = TRUE) {
  nobj <- nrow(delta)
  wmax <- max(weights)
  dold <- as.matrix(dist(xold))
  h <- engine(nobj, weights, delta, dold, cons)
  rold <- h$resi
  wold <- h$wght
  sold <- h$strs
  itel <- 1
  repeat {
    vmat <- -wold
    diag(vmat) <- -rowSums(vmat)
    vinv <- solve(vmat + (1 / nobj)) - (1 / nobj)
    bmat <- -wold * delta / (dold + diag(nobj))
    diag(bmat) <- -rowSums(bmat)
    xnew <- vinv %*% (bmat %*% xold)
    dnew <- as.matrix(dist(xnew))
    h <- engine(nobj, weights, delta, dnew, cons)
    rnew <- h$resi
    wnew <- h$wght
    snew <- h$strs
```

```
    if (verbose) {
      cat(
        "itel ",
        formatC(itel, width = 4, format = "d"),
        "sold ",
        formatC(sold, digits = 10, format = "f"),
        "snew ",
        formatC(snew, digits = 10, format = "f"),
        "\n"
      )
    }
    if ((itel == itmax) || ((sold - snew) < eps)) {
      break
    }
    xold <- xnew
    dold <- dnew
    sold <- snew
    wold <- wnew
    rold <- rnew
    itel <- itel + 1
  }
  return(list(
    x = xnew,
    s = snew,
    d = dnew,
    r = rnew,
    itel = itel
  ))
}

smacofTorgerson <- function(delta, ndim) {
  dd <- delta ^ 2
  rd <- apply(dd, 1, mean)
  md <- mean(dd)
  sd <- -.5 * (dd - outer(rd, rd, "+") + md)
  ed <- eigen(sd)
  return(ed$vectors[, 1:ndim] %*% diag(sqrt(ed$values[1:ndim])))
}

smacofAV <- function(nobj, wmat, delta, dmat, cons) {
```

```
  resi <- sqrt((delta - dmat) ^ 2 + cons)
  resi <- ifelse(resi < 1e-10, 2 * max(wmat), resi)
  rmin <- sqrt(cons)
  wght <- wmat / (resi + diag(nobj))
  strs <- sum(wmat * resi) - rmin * sum(wmat)
  return(list(resi = resi, wght = wght, strs = strs))
}

smacofLP <- function(nobj, wmat, delta, dmat, cons) {
  resi <- ((delta - dmat) ^ 2 + cons[1]) ^ cons[2]
  rmin <- cons[1] ^ cons[2]
  wght <- wmat * ((delta - dmat) ^ 2 + cons[1] + diag(nobj)) ^ (cons[2] - 1)
  strs <- sum(wmat * resi) - rmin * sum(wmat)
  return(list(resi = resi, wght = wght, strs = strs))
}

smacofConvolution <- function(nobj, wmat, delta, dmat, cons) {
  difi <- delta - dmat
  resi <- difi * (2 * pnorm(difi / cons) - 1) + 2 * cons * dnorm(difi / cons)
  rmin <- 2 * cons * dnorm(0)
  wght <- wmat * (pnorm(difi / cons) - 0.5) / (difi + diag(nobj))
  strs <- sum(wmat * resi) - rmin * sum(wmat)
  return(list(resi = resi, wght = wght, strs = strs))
}

smacofHuber <- function(nobj, wmat, delta, dmat, cons) {
  difi <- delta - dmat
  resi <- ifelse(abs(difi) < cons, (difi ^ 2) / 2, cons * abs(difi) - ((cons ^ 2) / 2)
  wght <- ifelse(abs(difi) < cons, wmat,
                 wmat * sign(difi - cons) * cons / (difi + diag(nobj)))
  strs <- sum(wmat * resi)
  return(list(resi = resi, wght = wght, strs = strs))
}

smacofTukey <- function(nobj, wmat, delta, dmat, cons) {
  cans <- (cons ^ 2) / 6
  difi <- delta - dmat
  resi <- ifelse(abs(difi) < cons,
                 cans * (1 - (1 - (difi / cons) ^ 2) ^ 3), cans)
  wght <- wmat * ifelse(abs(difi) < cons,
```

```
                  (1 - (difi / cons) ^ 2) ^ 2, 0)
  strs <- sum(wmat * resi)
  return(list(resi = resi, wght = wght, strs = strs))
}

smacofCauchy <- function(nobj, wmat, delta, dmat, cons) {
  difi <- delta - dmat
  resi <- log((difi / cons) ^ 2 + 1)
  wght <- wmat * (1 / ((difi / cons) ^ 2 + 1))
  strs <- sum(wmat * resi)
  return(list(resi = resi, wght = wght, strs = strs))
}

smacofWelsch <- function(nobj, wmat, delta, dmat, cons) {
  difi <- delta - dmat
  resi <- 1 - exp(-(difi / cons) ^ 2)
  wght <- wmat * exp(-(difi / cons) ^ 2)
  strs <- sum(wmat * resi)
  return(list(resi = resi, wght = wght, strs = strs))
}
```

References

Barron, J. T. 2019. "A General and Adaptive Robust Loss Function." In *Proceedings 2019 IEEE/CVF Conferehce on Computer Vision and Pattern Recognition*, 4331–39. https://openaccess.thecvf.com/content_CVPR_2019/papers/Barron_A_General_and_Adaptive_Robust_Loss_Function_CVPR_2019_paper.pdf.

Charbonnier, P., L. Blanc-Feraud, G. Aubert, and M. Barlaud. 1994. "Two deterministic half-quadratic regularization algorithms for computed imaging." *Proceedings of 1st International Conference on Image Processing* 2: 168–72. https://doi.org/10.1109/icip.1994.413553.

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. 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.

———. 2018. "MM Algorithms for Smoothed Absolute Values." 2018. https://jansweb.netlify.app/publication/deleeuw-e-18-f/deleeuw-e-18-f.pdf.

———. 2020. "Least Squares Absolute Value Regression." 2020. https://jansweb.netlify.app/publication/deleeuw-e-20-b/deleeuw-e-20-b.pdf.

De Leeuw, J., and K. Lange. 2009. "Sharp Quadratic Majorization in One Dimension." *Computational Statistics and Data Analysis* 53: 2471–84.

Groenen, P. J. F., W. J. Heiser, and J. J. Meulman. 1999. "Global Optimization in Least-Squares Multidimensional Scaling by Distance Smoothing." *Journal of Classification* 16: 225–54.

Heiser, W. J. 1987. "Correspondence Analysis with Least Absolute Residuals." *Computational Statistica and Data Analysis* 5: 337–56.

———. 1988. "Multidimensional Scaling with Least Absolute Residuals." In *Classification and Related Methods of Data Analysis*, edited by H. H. Bock, 455–62. North-Holland Publishing Co.

Huber, P. J. 1964. "Robust Estimation of a Location Parameter." *Annals of Mathematical Statistics* 35 (1): 73–101.

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

Pliner, V. 1996. "Metric Unidimensional Scaling and Global Optimization." *Journal of Classification* 13: 3–18.

Ramirez, C., R. Sanchez, V. Kreinovich, and M. Argaez. 2014. "$\sqrt{x^2+\mu}$ is the Most Computationally Efficient Smooth Approximation to |x|." *Journal of Uncertain Systems* 8: 205–10.

Voronin, S., G. Ozkaya, and Y. Yoshida. n.d. "Convolution Based Smooth Approximations to the Absolute Value Function with Application to Non-smooth Regularization."