Robust Least Squares Multidimensional Scaling

Jan de Leeuw

October 6, 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, location theory, and sparse recovery. Code in R is included.

Table of contents

List of Figures

1 Introduction

The title of this paper 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 paper 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.

A large number of somewhat ad-hoc solutions have been proposed to deal with the problem of zero residuals, both in location analysis and in the statistical literature. We tend to agree with Aftab and Hartley (2015).

> .. attempts to analyze this difficulty [caused by infinite weights of IRLS for the ℓ_p-loss] have a long history of proofs and counterexamples to incorrect claims.

Schlossmacher (1973) is the first discussion of the majorization method in the statistical literature (for LAV linear regression). His proposal is to simply set a weight equal to zero if the corresponding residual is less than some small positive value ϵ. A similar approach, also used in location analysis, is to cap the weights at some large positive value. In Heiser (1988) all residuals smaller than this epsilon get a weight equal to the weighted average of all these small residuals. Phillips (2002) assumes double-exponential errors in LAV regression and then concludes that the EM algorithm gives the majorization method we have discussed. He uses (7) throughout if all residuals are larger than ϵ. If one of more residuals are smaller than epsilon then the weight for those residuals is set equal to one, while for the remaining residuals the weight is set to epsilon divided by the absolute value of the residual. Often we get the assurance that the problem is not really important in practice, because it is very rare, and by just wiggling we will get to the unique solution anyway. But both in location analysis and in LAV regression the loss function is convex, while this is certainly not the case in robust MDS. In this paper we try to follow a more systematic approach that uses smooth approximations to the absolute value function.

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.

2.3 ℓ_0 loss

A somewhat extreme special case of Equation (3) has

$$f(x) = \begin{cases} 0 & \text{if } x = 0, \\ 1 & \text{otherwise.} \end{cases}$$

This is ℓ_0 loss. Minimizing ℓ_0 loss means maximizing the number of cases with perfect fit, i.e. with $\delta_k = d_k(X)$. The reason we mention it here is that the work of Donoho and Elad (2003) and Candes and Tao (2005) suggests that the minimizer of ℓ_1 loss, i.e. absolute value loss, gives a good approximation to the minimizer of ℓ_0 loss, at least in a number of special cases. In MDS we do not have linearity or convexity, but nevertheless the reulsr are suggestive. By computing the directional derivatives we have seen that at least in the one-dimensional MDS case a number of residuals will indeed be zero at the optimum LAV solution.

3 Generalizing Strife

We have seen that Heiser (1988) applied majorization to minimize strife, using the AM/GM inequality. We now generalize this approach so that it can easily deal with other robust loss functions. A great number of loss functions will appear below. The intention is not to confuse the reader by requiring them to choose from impossibly large number of alternatives with rather limited information. We show all these loss functions as examples of the general principle of algorithm construction and as examples of loss functions that have been used in statistics, location analysis, image analysis and engineering over the years. They are all implemented in our software smacofRobust.R.

3.1 Majorization

First some definitions. A function g *majorizes* a function f at y if $g(x) \geq f(x)$ for all x and $g(y) = f(y)$. The AM/GM inequality was used in the previous section to construct a quadratic majorization of strife. 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 sharp majorization is by definition unique.

We are specifically interested in this paper 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 Theorems 4.5 and 4.6 on pages 2478-2479 says

Theorem 3.1. *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 3.2. *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 (see discussion).

We now apply Theorem 3.1 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 (Van Ruitenburg (2005)). 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 the 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 based on the Cauchy-Schwartz inequality within the sharp quadratic majorization of the loss function based on the AM/GM inequality.

3.2 Literature

The literature on results like Theorem 3.1 and Theorem 3.2 is an absolute shambles. There are various reasons for that. Relevant results have been published in robust statistics, comptational statistics, optimization, location theory, image restortation, sparse recovery. There are not many references between fields, almost everything is within. Moreover much of it is hidden in the usual caves of engineering conference proceedings. Also, in most cases, the authors have specific applications in mind, which they then embed in a likelihood, Bayesian, linear regression, facility location, or EM framework and language.

De Leeuw and Lange (2009) give some references to previous work on results like Theorem 3.1, notably Groenen, Giaquinto, and Kiers (2003), Jaakkola and Jordan (2000), and Hunter and Li (2005). In these earlier papers we do not find Theorem 3.1 in its full generality. In Groenen, Giaquinto, and Kiers (2003) majorization of the log logistic function is considered. Besides requiring equality of the function and the majorizing quadratic at the support point y they also require equality at $-y$ and then check that the resulting quadratic is indeed a majorizer. In Jaakkola and Jordan (2000) also consider a symmetrized version of the log logistic function. They note that the resulting function is a convex funcion of x^2, and use a linear majorizer at x^2 to obtain a quadratic majorization. Hunter and Li (2005) come closest to Theorem 3.1. In their proposition 3.1 they approximate the general penalty function used in varable selection at y by a quadratic with coefficient $f'(y)/2y$, and then show that it provides a quadratic majorization. In neither of the three papers there is a notion of sharp quadratic majorization.

Van Ruitenburg (2005)

In robust statistics it has been known for a long time that iterative reweighted least squares with weights $f'(x)/x$ gives a quadratic majorization algorithm.

4 Power Smoothers

We first discuss a class of smoothers of the absolute value function that maintain most of its structure. They have a shift parameter c that takes care of the non-differentiability. And some of them also have a power parameter q that determines the shape of the loss function bowl.

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

For $c > 0$ we have $f_c(x) > |x|$. If $c \rightarrow 0$ 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 researchers to use it in image restauration. Ramirez et al. (2014) count the number of elementary computer operations and argue (15) is also the "most computationally efficient smooth approximation to $|x|$".

By l'Hôpital

$$\lim_{x\rightarrow 0}\frac{\sqrt{x^2+c^2}-c}{\frac12 x^2} = 1. \tag{16a}$$

Of course also

$$\lim_{x\rightarrow\infty}\frac{\sqrt{x^2+c^2}}{|x|} = 1 \tag{16b}$$

and

$$\lim_{x\rightarrow\pm\infty}\sqrt{x^2+c^2} - |x| = 0 \tag{16c}$$

Thus if x is much smaller than c then 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

$$f_c'(x) = \frac{1}{\sqrt{x^2+c^2}}x, \tag{17}$$

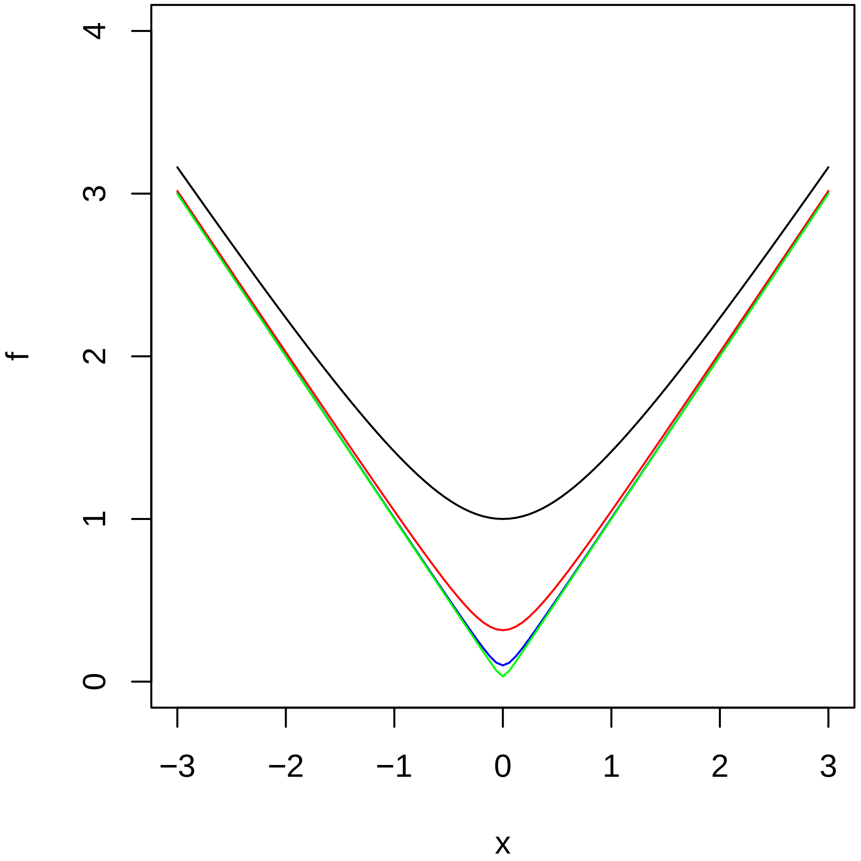

Figure 1: Charbonnier Loss

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

$$w_c(x) = \frac{1}{\sqrt{x^2 + c^2}} \tag{18}$$

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

4.2 Generalized Charbonnier Loss

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

$$f_{c,q}(x) := (x^2 + c^2)^{\frac{1}{2}q} \tag{19}$$

$$w_{c,q}(x) = q(x^2 + c^2)^{\frac{1}{2}q-1} \tag{20}$$

which is non-increasing for $q \leq 2$. Note that we do not assume that $q > 0$, and consequently (19) provides us with much more flexibility than Charbonnier loss (15).

In figure Figure 2 we have plotted (19) for $c = 1$ and α equal to -5 (black), -1 (red), $-.5$ (blue), and $-.1$ (green).

We see that for $\alpha \rightarrow -\infty$ generalized Charbonnier loss aproximates ℓ_0 loss.

4.3 Barron Loss

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

$$f_{\alpha,c}(x) = \frac{|\alpha - 2|}{\alpha}\left(\left(\frac{(x/c)^2}{|\alpha - 2|} + 1\right)^{\alpha/2} - 1\right). \tag{21}$$

> Here $\alpha \in \mathbb{R}$ is a shape parameter that controls the robust-ness of the loss and $c > 0$ is a scale parameter that controls the size of the loss's quadratic bowl near $x = 0$.

There are a number of interesting special cases of (21) by selecting various values of the parameters. For $\alpha = 1$ it becomes Charbonnier loss, and for $\alpha = -2$ it is Geman-McLure loss. For $\alpha \rightarrow 2$ it becomes squared error loss, for $\alpha \rightarrow 0$ it becomes Cauchy loss, and for $\alpha \rightarrow -\infty$ it becomes Cauchy loss.

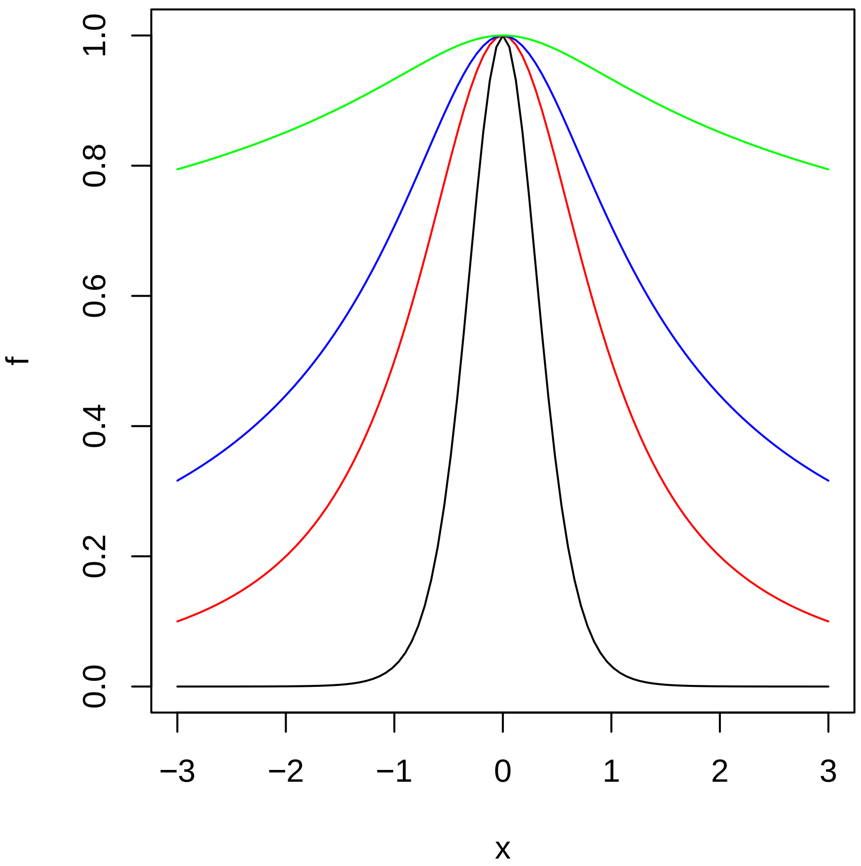

Figure 2: Generalized Charbonnier Loss

5 Convolution Smoothers

Suppose π is a probability density, symmetric around zero, with finite or infinite support, expectation zero, and variance one. Define the convolution

$$f_c(x) := \frac{1}{c}\int_{-\infty}^{+\infty} |x-y|\,\pi(\frac{y}{c})dy.$$

Now $c^{-1}\pi(y/c)$ is still a symmetric probability density integrating to one, with expectation zero, but it now has variance c^2. Thus if $c \rightarrow 0$ it becomes more and more like the Dirac delta function and $f_c(x)$ converges to the absolute value function.

It is 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. We give two quite different examples.

5.1 Huber Loss

Take

$$\pi(x) = \begin{cases} \frac12 & \text{if } |x| \leq 1, \\ 0 & \text{otherwise.} \end{cases}$$

Then

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

The Huber function (Huber (1964)) is

$$f_c(x) = \begin{cases} \frac12 x^2 & \text{if } |x| < c, \\ c|x| - \frac12 c^2 & \text{otherwise.} \end{cases}$$

Because Charbonnier loss behaves like absolute value loss for large x and as squared loss for small x 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 Theorem 3.1 applies.

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

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

5.2 Gaussian Convolution

In De Leeuw (2018) we also discussed 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 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

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

The derivative is

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

It may not be immediately obvious in this case that the weight function $f'(x)/x$ is non-increasing on $\mathbb{R}^+$. 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.

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

6 A Bouquet of Loss Functions

In the early seventies, after the pioneering mostly theoretical work in robust statistics of Huber, Hampel, and Tukey, the mainframe computer allowed statisticians to make large-scale comparisons of many robust loss functions. The most impressive of such comparisons was the Princeton Robustness Study (Andrews et al. (1972)).

In Holland and Welsch (1977) the computer package ROSEPACK was introduced that made it relatively easy to compute robust estimators using several different loss functions. Eight different weight functions were implemented as options. Somewhat later Coleman et al. (1980) made an more modern computer implementation available, using the same eight weight functions, which was not limited to mainframes.

We have implemented the same eight weight functions in smacofRobust, plus some additional ones that came out of our discussion of the power and convolution families. Below we give formulas for the loss function, the influence function, and the weight function. We also graph the loss function for selected values of the parameters.

6.1 Andrews

$$f(x)=\begin{cases}c^2(1-\cos(x/c)) & \text{if } |x|\leq\pi c,\\ 2c^2 & \text{otherwise.}\end{cases}\tag{22}$$

$$f'(x)=\begin{cases}c\sin(x/c) & \text{if } |x|\leq\pi c,\\ 0 & \text{otherwise.}\end{cases}\tag{23}$$

$$w(x)=\begin{cases}(x/c)^{-1}\sin(x/c) & \text{if } |x|\leq\pi c,\\ 0 & \text{otherwise.}\end{cases}\tag{24}$$

6.2 Tukey

$$f(x) = \begin{cases} \frac{c^2}{6}\left(1-(1-(x/c)^2)^3\right) & \text{if } |x| \leq c, \\ \frac{c^2}{6} & \text{otherwise.} \end{cases} \tag{25}$$

$$f'(x) = \begin{cases} x\left(1-(1-(x/c)^2)^2\right) & \text{if } |x| \leq c, \\ 0 & \text{otherwise.} \end{cases} \tag{26}$$

$$w(x) = \begin{cases} \left(1-(1-(x/c)^2)^2\right) & \text{if } |x| \leq c, \\ 0 & \text{otherwise.} \end{cases} \tag{27}$$

6.3 Hinich

6.4 Cauchy

$$f(x) = \log(\{\frac{x}{c}\}^2 + 1), \tag{28}$$

$$f'(x) = \frac{1}{c^2} x \frac{1}{\{\frac{x}{c}\}^2 + 1}, \tag{29}$$

$$w(x) = \frac{1}{c^2} \frac{1}{\{\frac{x}{c}\}^2 + 1} \tag{30}$$

6.5 Fair

6.6 Huber

6.7 Logistic

6.8 Talwar

6.9 Welsch

$$f(x) = 1 - \exp(-\{\frac{x}{c}\}^2), \tag{31}$$

$$f'(x) = \tag{32}$$

$$w(x) = \frac{2}{c^2} \exp(-\{\frac{x}{c}\}^2) \tag{33}$$

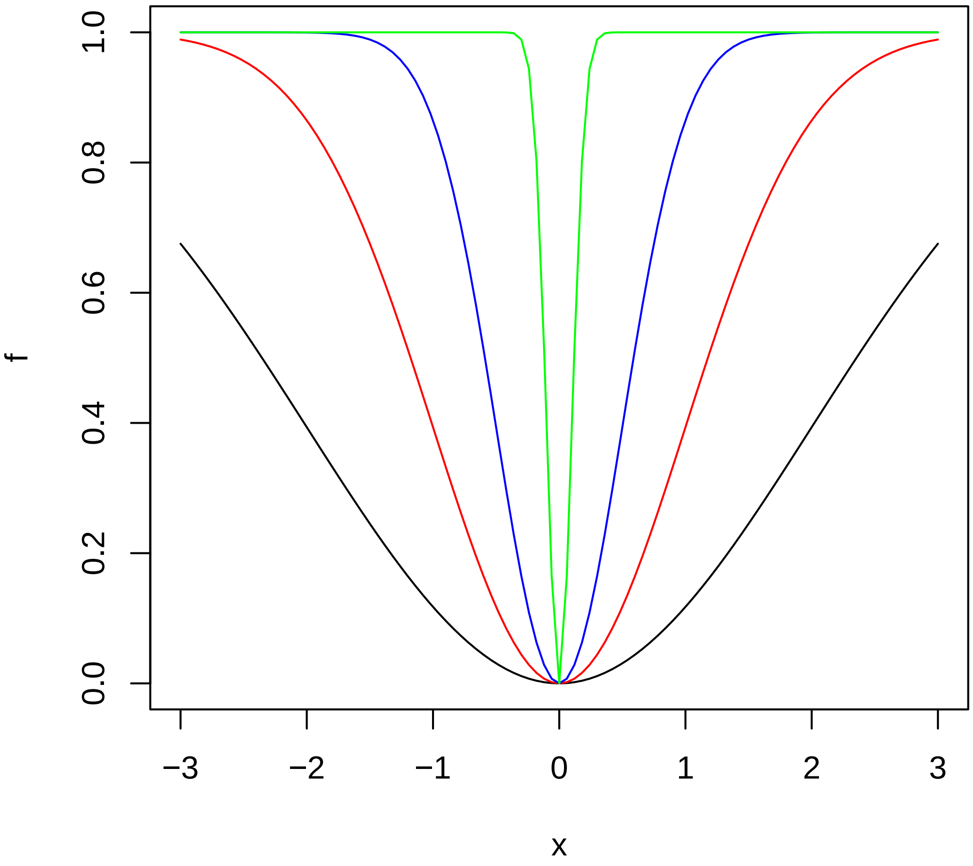

Figure 3: Welsch Loss

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

6.10 Geman-McLure Loss

$$f_c(x) = \frac{2(x/c)^2}{(x/c)^2 + 4}.$$

7 Examples

7.1 Gruijter

The example we use are dissimilarities between nine Dutch political parties, collected by De Gruijter (1967). They are averages over a politically heterogenous group of 100 introductory psychology students, and consequently they regress to the mean. Any reasonable MDS analysis of these data would at least allow for an additive constant.

Some background on Dutch politics around that time may be useful.

- CPN - Communists.
- PSP - Pacifists, left-wing.
- PvdA - Labour, Democratic Socialists.
- D'66 - Pragmatists, nether left-wing fish nor right-wing flesh, brand new in 1967.
- KVP - Christian Democrats, catholic.
- ARP - Christian Democrats, protestant.
- CHU - Christian Democrats, protestants, different flavor.
- VVD - Liberals, European flavour, right-wing.
- BP - Farmers, protest party, right-wing.

The dissimilarities are in the table below.

```
      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
```

The reason we have chosen this example is partly because CPN and BP are outliers, and we can expect the robust loss functions to handle outlying dissimilarities differently from the bulk of the data.

Unless otherwise indicated we run smacofRobust() with a maximum of 10,000 iterations, and we decide that we have comvergence if the difference between consecutive stress values is less than 10^{-15}. We perform a single smacof iteration between the updates of the weights. For each analysis we show the configuration plot and the Shepard plot. In the Shepard plot points corresponding to the eight CPN-dissimilarities arelabeled “C”, while BP-dissimilarities are “B”.

7.1.1 Least Squares

We start with a least squares analysis, actually with Huber loss with $c = 10$, which for these data is equivalent to least squares. The process converges in 859 iterations.

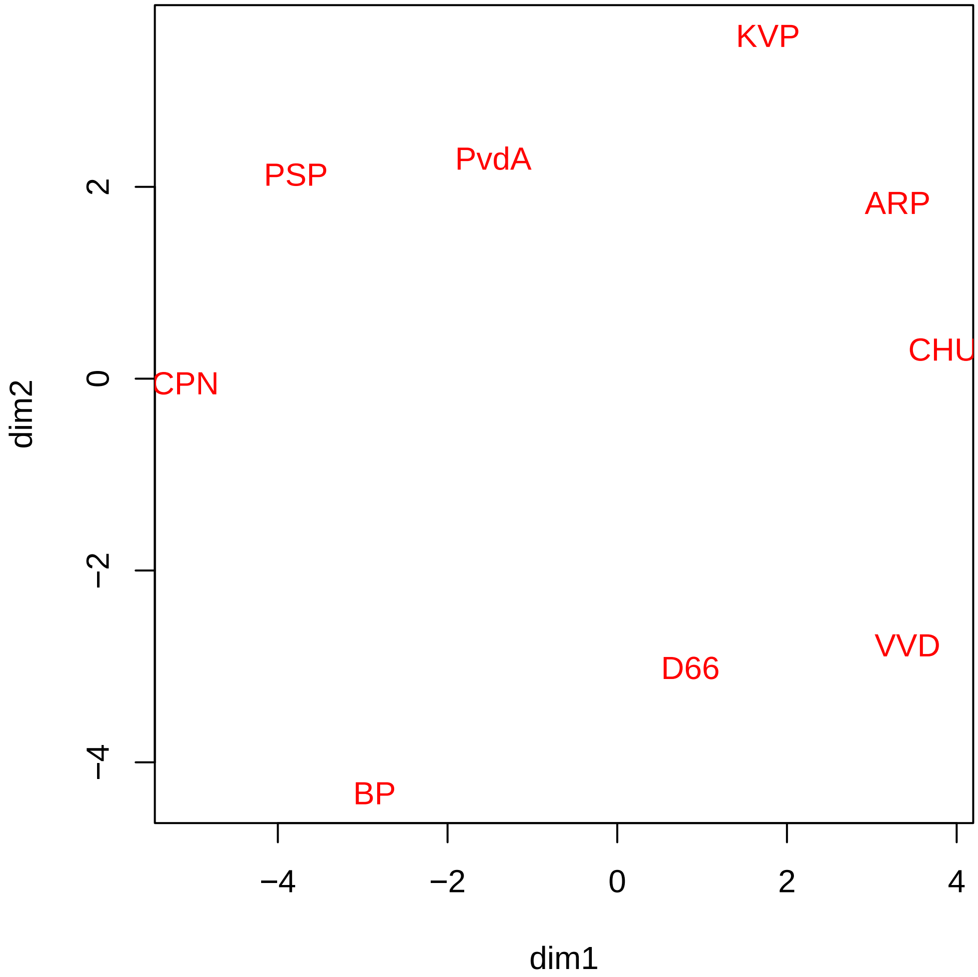

Figure 4: Configuration Least Squares

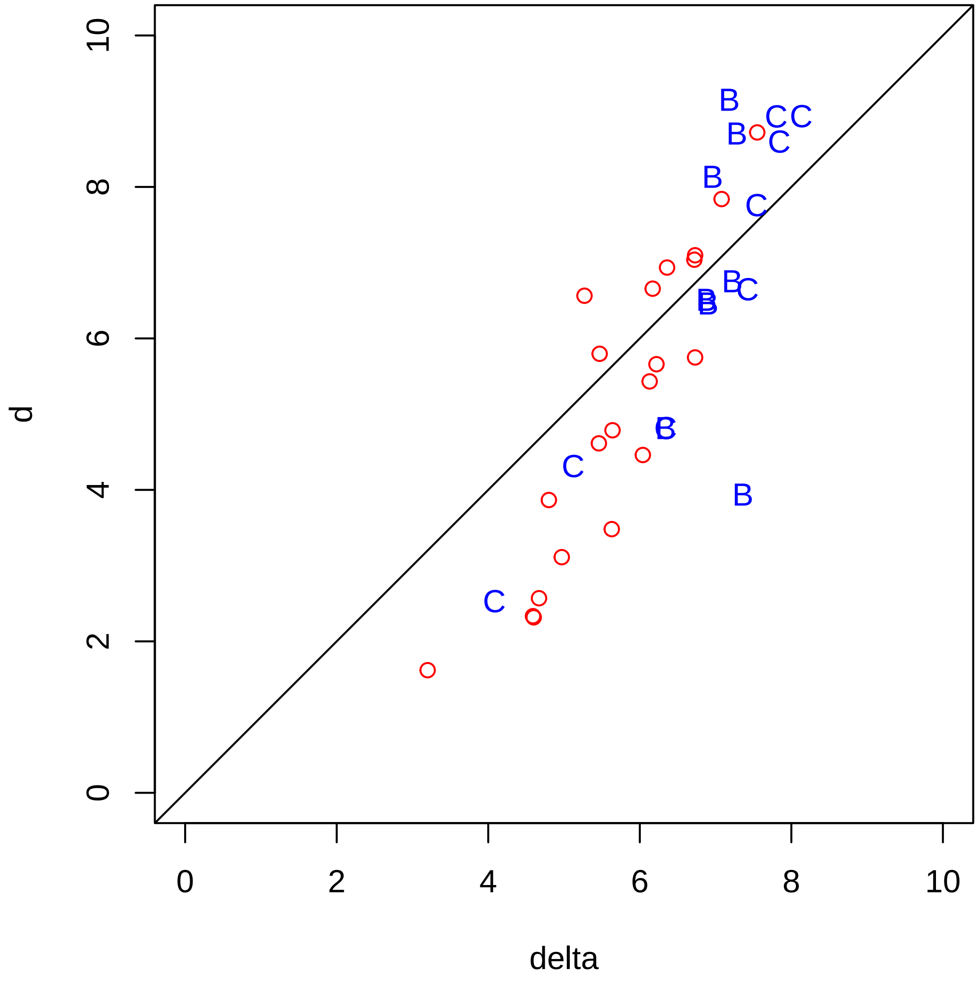

Figure 5: Shepard Plot Least Squares

The Shepard plot clearly shows why an additive constant would be very beneficial in this case.

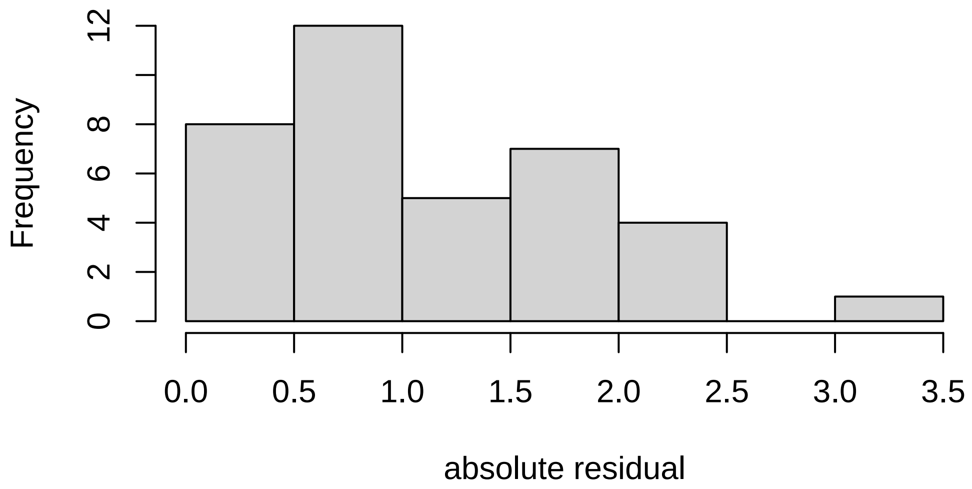

Figure 6: Histogram Least Squares Residuals

7.1.2 Least Absolute Value

For our least absolute value smacof we use engine smacofAV with $c = .001$. We have convergence in 637 iterations.

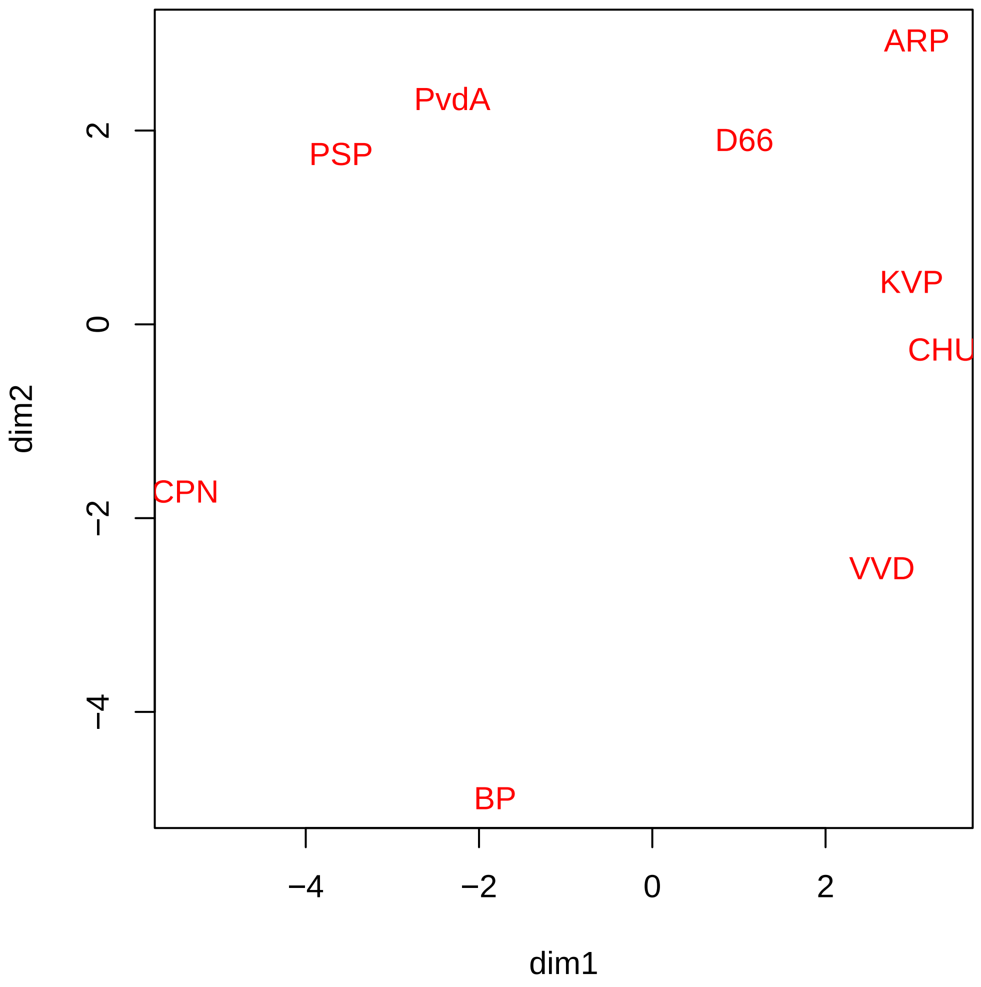

Figure 7: Configuration Least Absolute Value

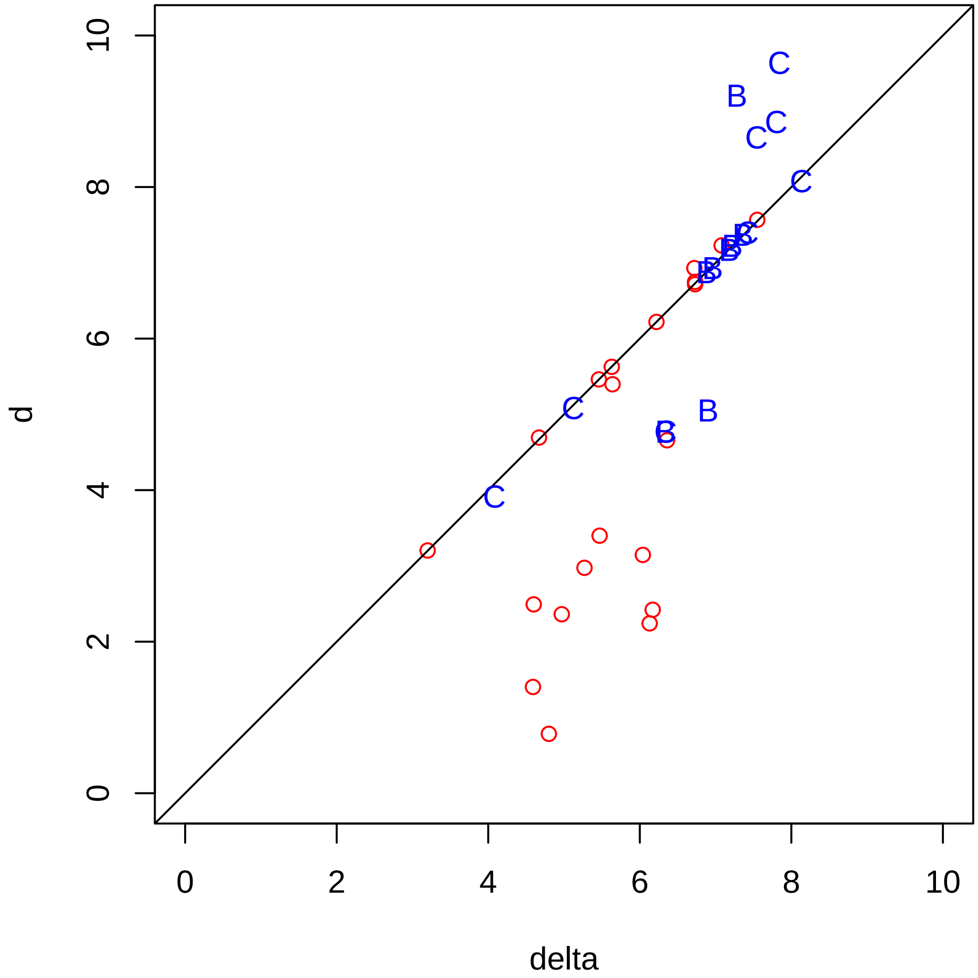

Figure 8: Shepard Plot Least Absolute Value

In the Shepard plot we see that there are a number of dissimilarities which are fitted exactly. If we count them there are about 15. Note that configurations have $(n-1)+(n-2)=2n-3$ degrees of freedom, which is 15 in this case. Thus if we take the 15 dissimilarities which are fitted exactly, give them weight one, give all other 21 dissimilarities weight zero, and do a smacof analysis using these weights, then we will have perfect fit in two dimensions, and the solution will be the least absolute value solution. All this is easier said than done, because it presumes Charbonnier loss with $c=0$ and the ability to decide what exact equality is. It also shows the possibility of a huge number of local minima in the least absolute value case, because there are so many ways to pick 15 out of 36 dissimilarities.

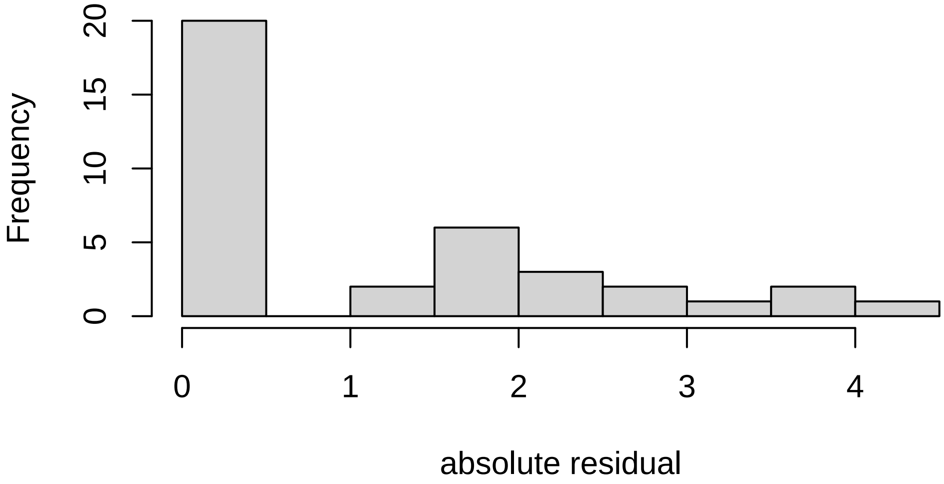

Figure 9: Histogram Least Absolute Value Residuals

7.1.3 Huber

smacofHuber with $c = 1$ converges in 165 iterations.

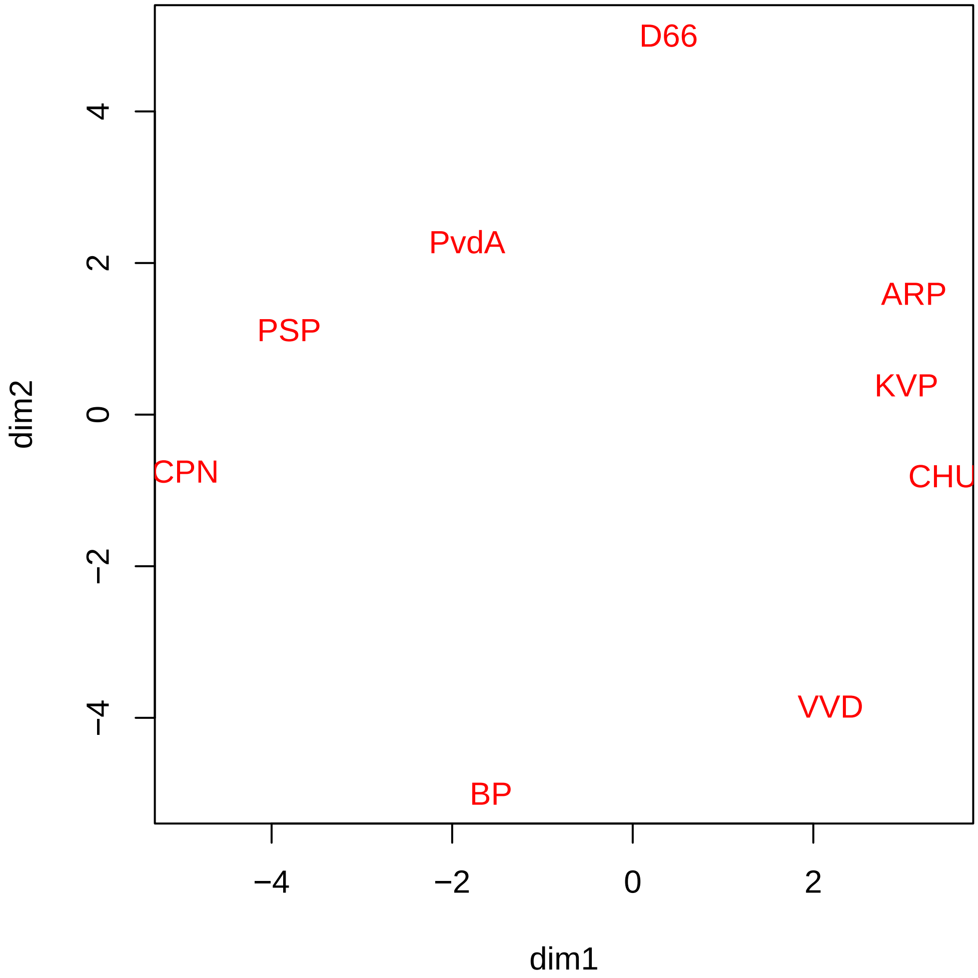

Figure 10: Configuration Huber c = 1

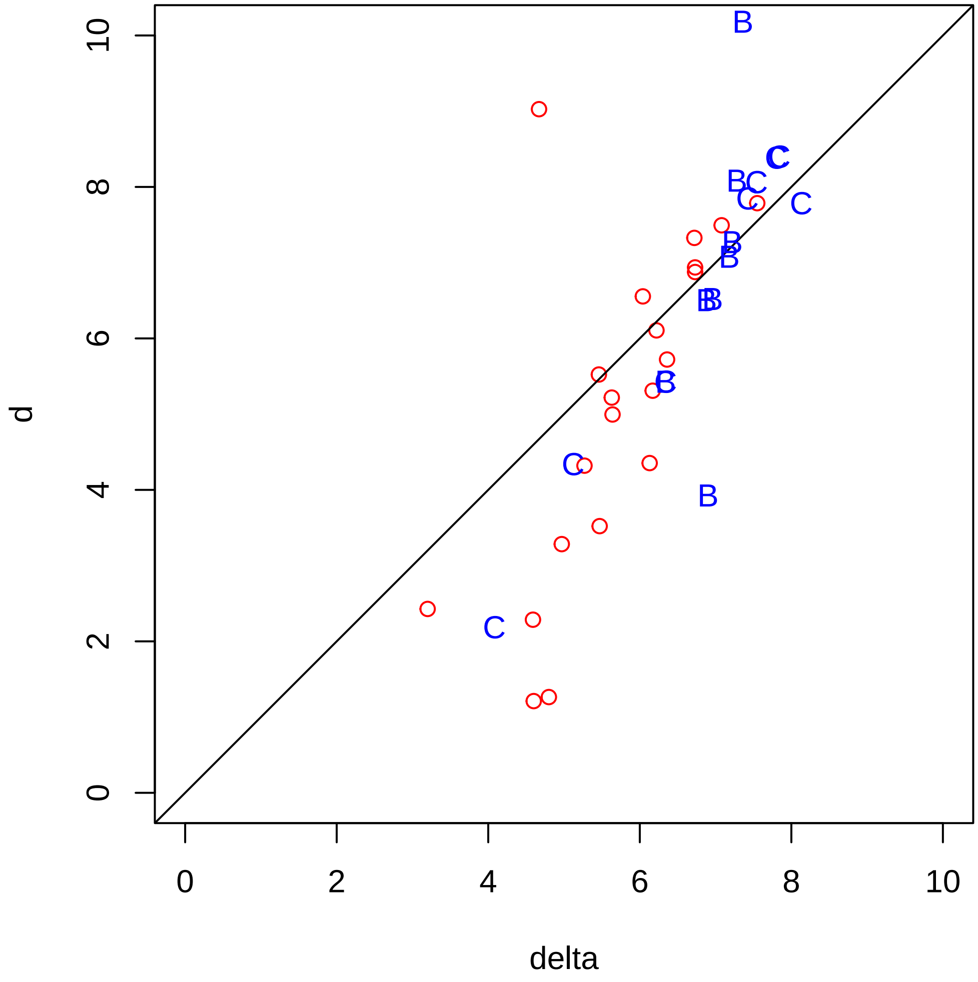

Figure 11: Shepard Plot Huber c = 1

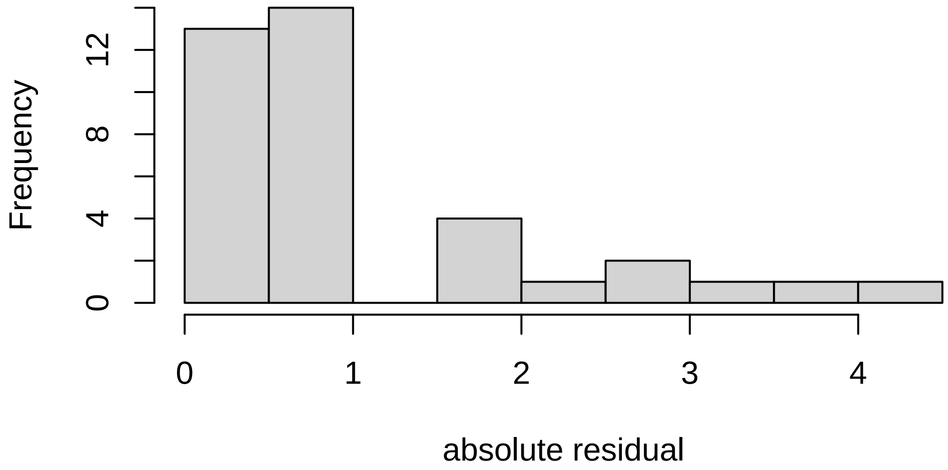

Figure 12: Histogram Huber Residuals

7.1.4 Tukey

Converges in 180 iterations.

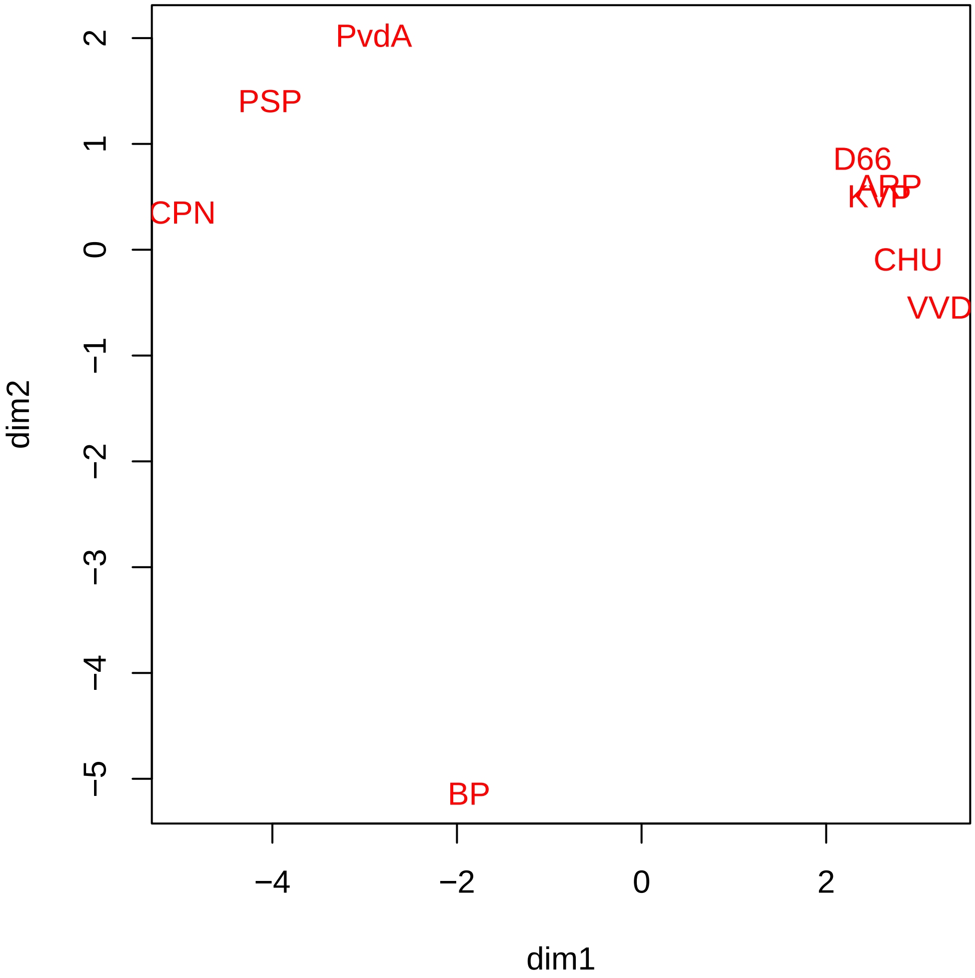

Figure 13: Configuration Tukey c = 2

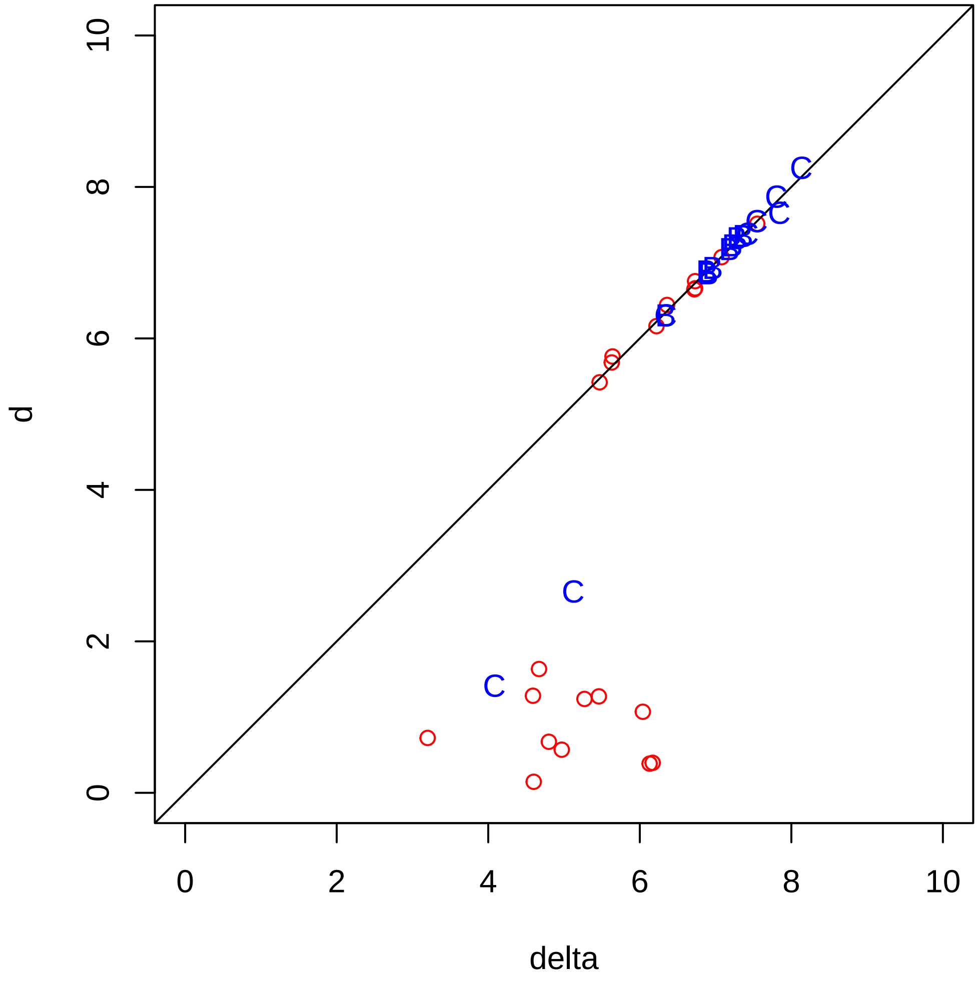

Figure 14: Shepard Plot Tukey c = 2

```r
hist(abs(delta[iall] - htu$d[iall]), main = "", xlab = "absolute residual")
```

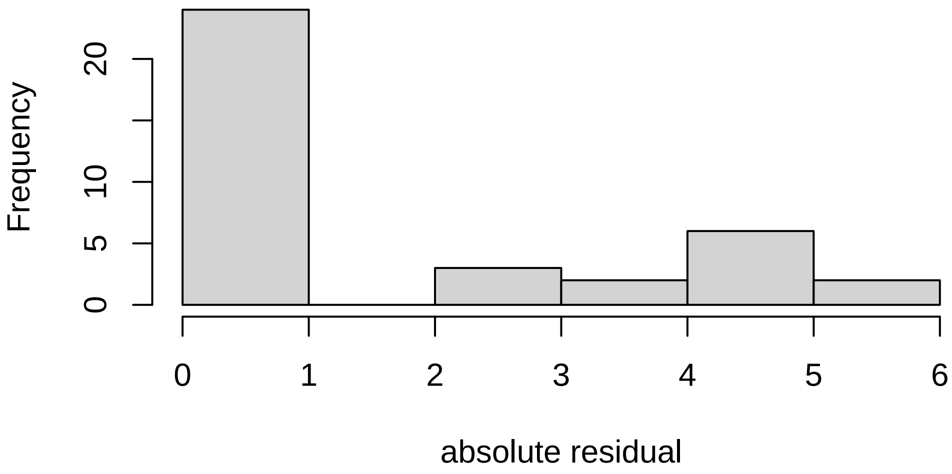

Figure 15: Histogram Tukey Residuals

7.2 Rothkopf

Our seond example are the Rothkopf Morse data (Rothkopf (1957)), which are better behaved as the Gruijter data. We used the asymetric confusion matrix and defined a dissimilarity by the Shepard-Luce like formula

$$\delta_{ij}=-\log\frac{p_{ij}p_{ji}}{p_{ii}p_{jj}}.$$

The fivenum for these data is 0.2220619, 2.5633455, 4.027474, 5.3988089, 8.3503118.

7.2.1 Least Squares

For least squares we use the smacofHuber engine with $c=25$, well outside the range of the residuals. We have convergence in 213 iterations.

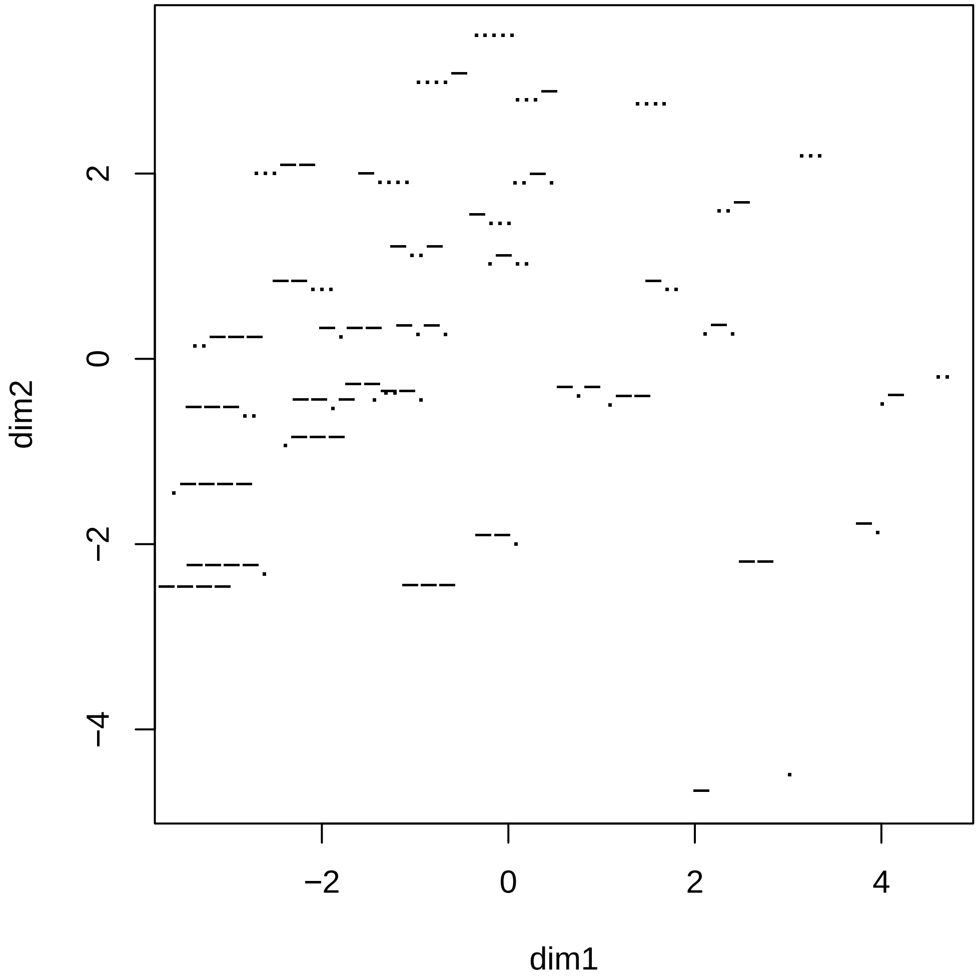

Figure 16: Configuration Least Squares

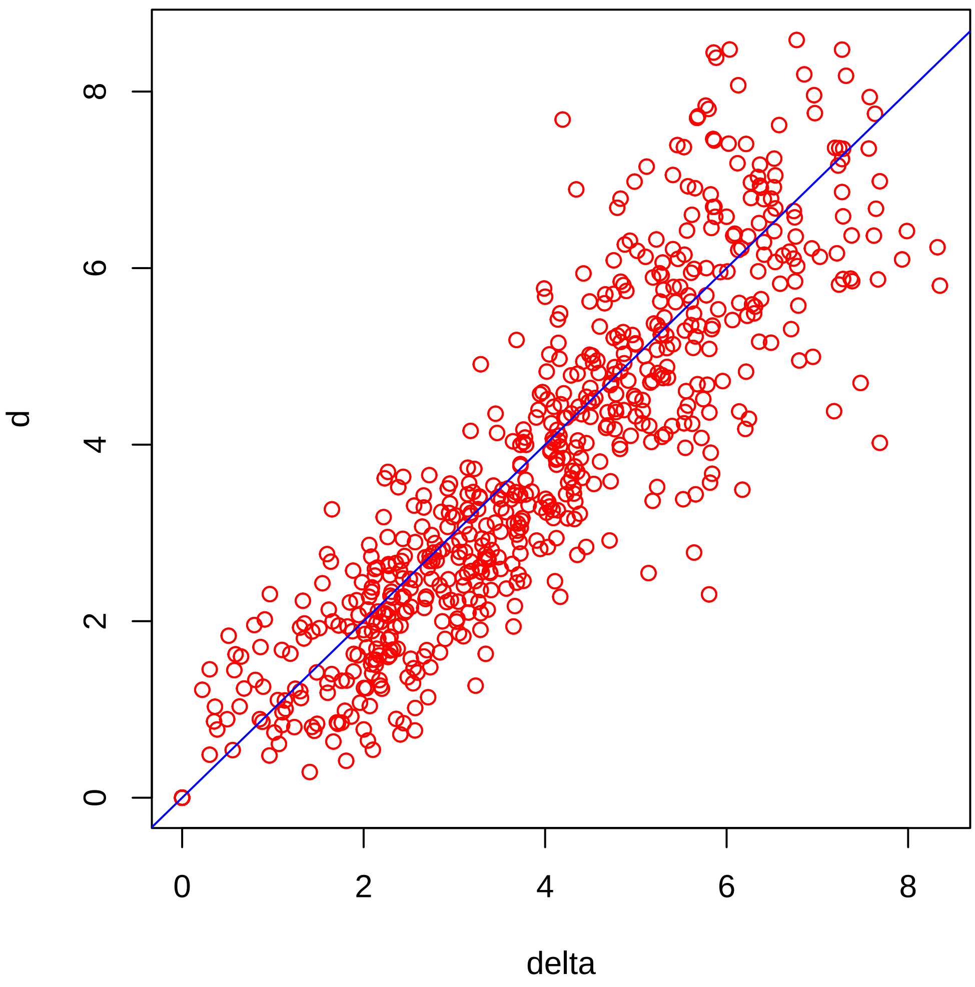

Figure 17: Shepard Plot Least Squares

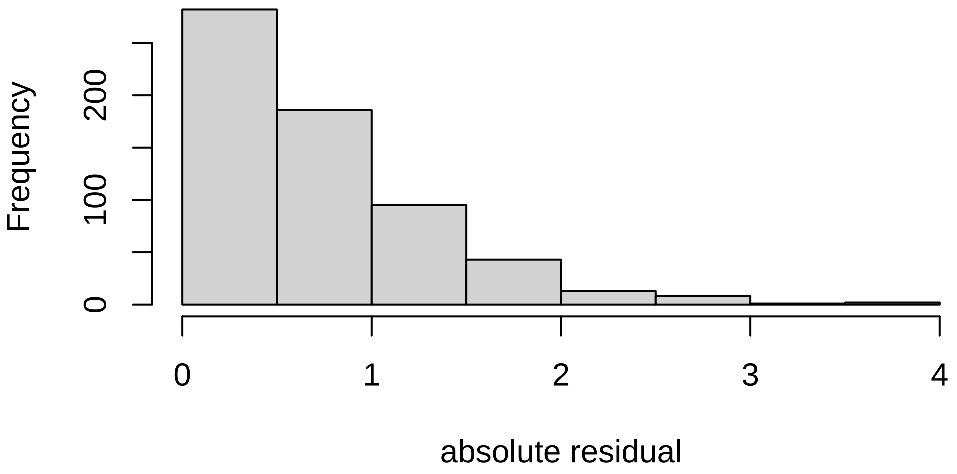

Figure 18: Histogram Least Squares Residuals

7.2.2 Least Absolute Value

Chardonnier with $c = .001$, de facto least absolute value loss.

For least absolute value we use Chardonnier loss with $c = .001.$ We have convergence in 2291 iterations.

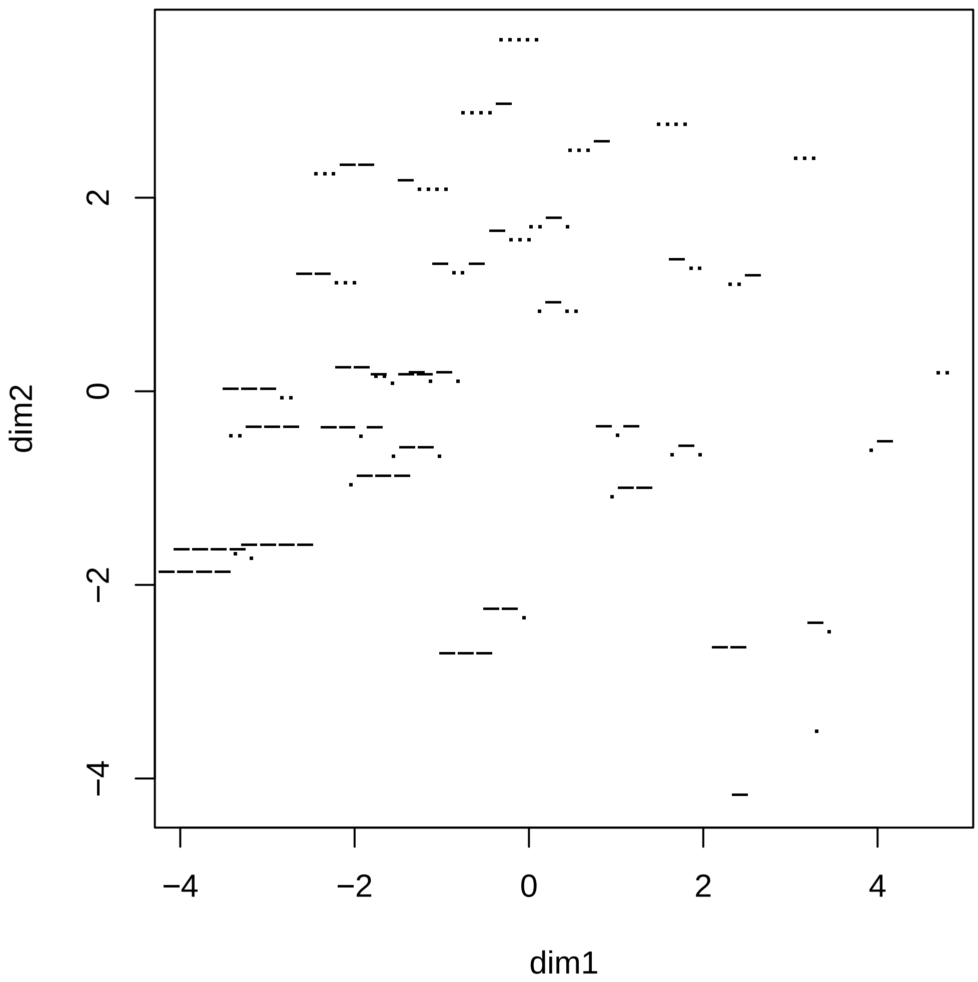

Figure 19: Configuration Least Absolute Value

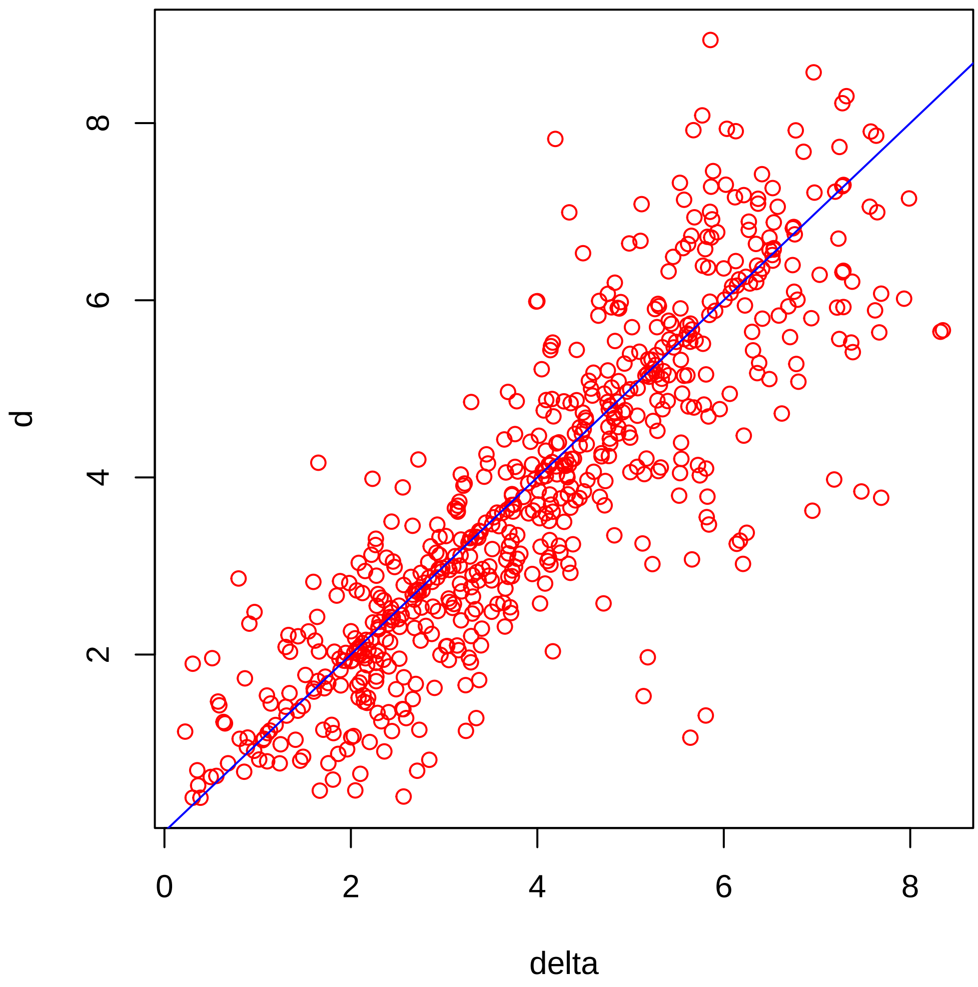

Figure 20: Shepard Plot Least Absolute Value

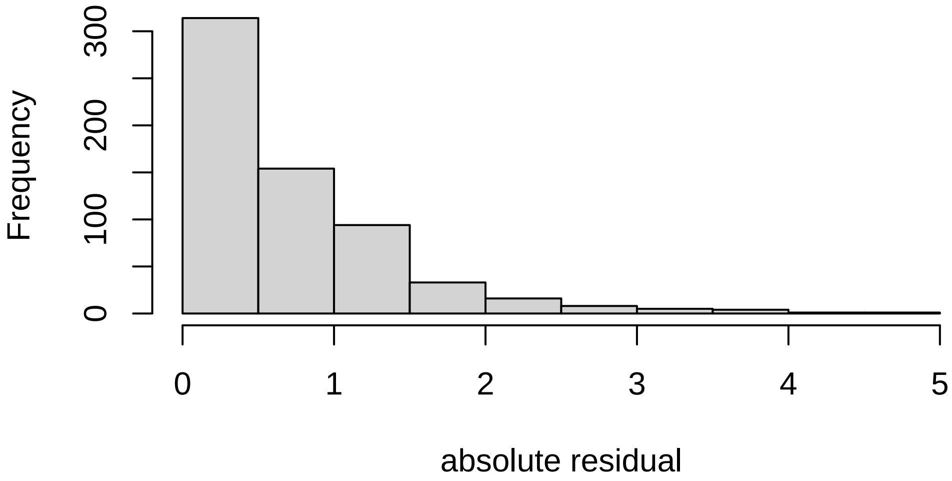

Figure 21: Histogram Least Absolute Value Residuals

7.2.3 Huber

Huber with $c = 1$

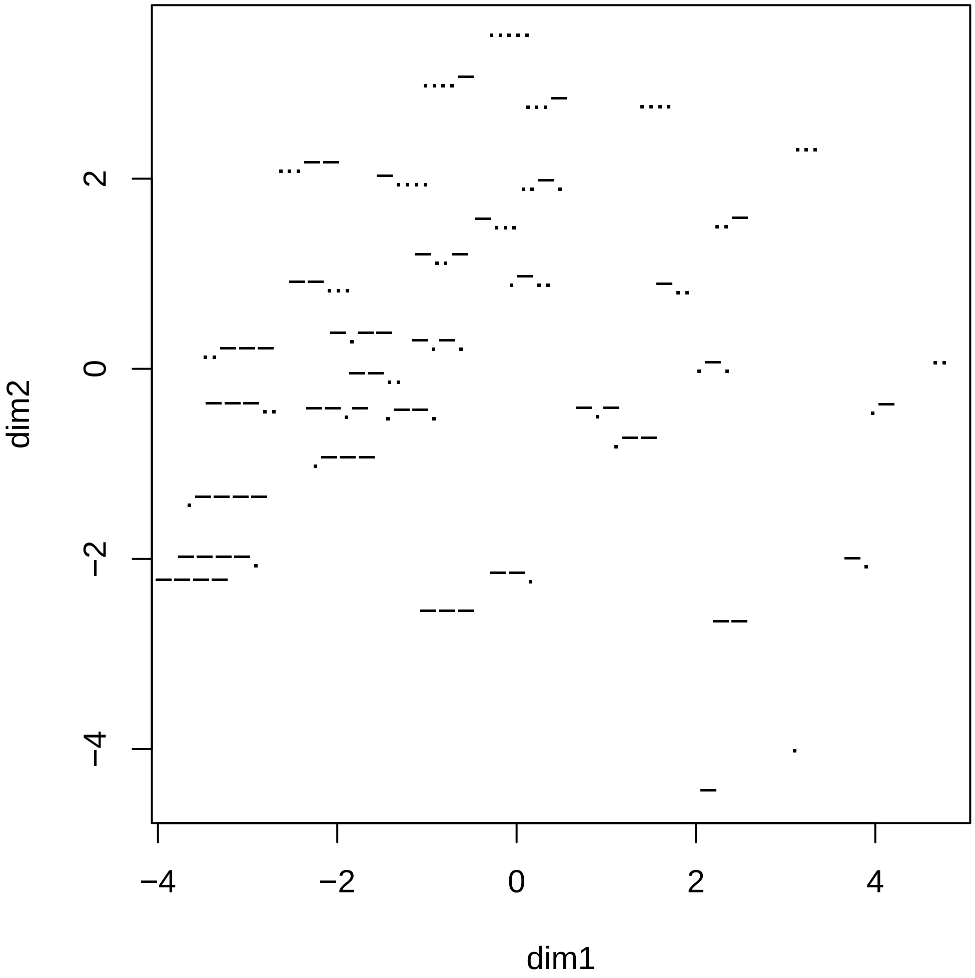

Figure 22: Configuration Huber

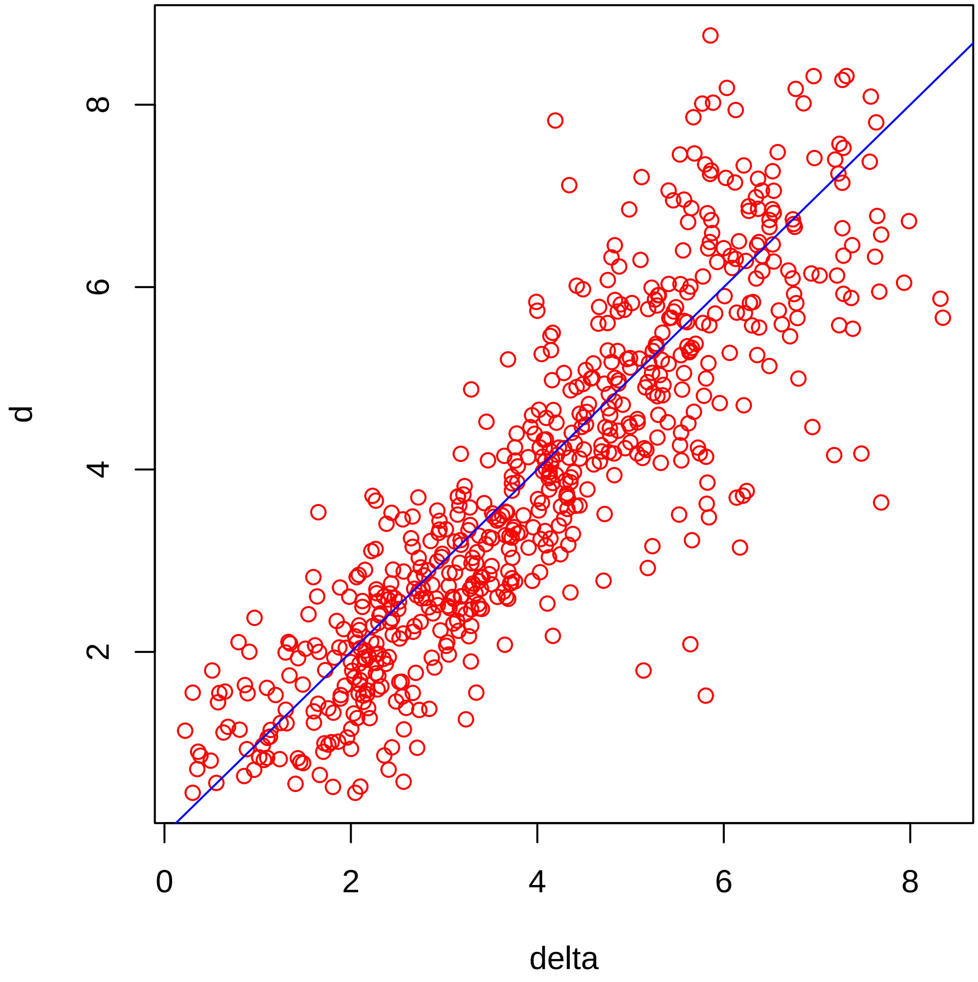

Figure 23: Shepard Plot Huber

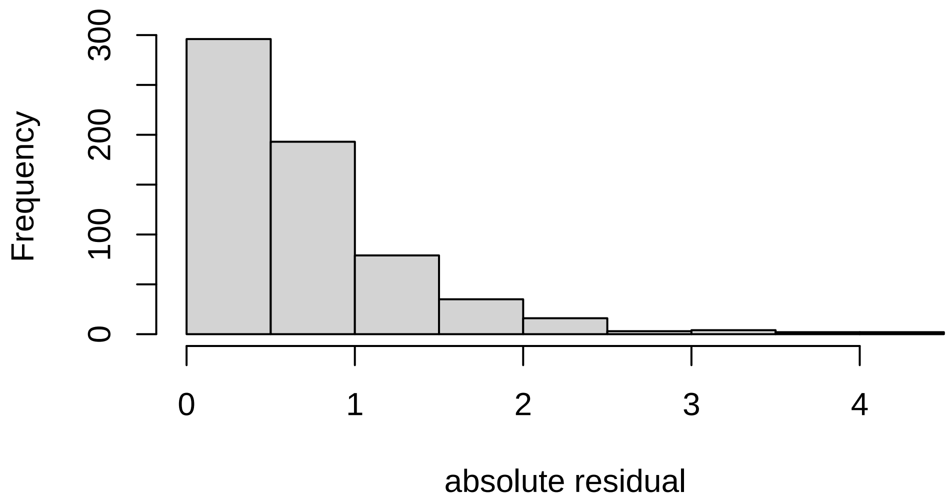

Figure 24: Histogram Huber Residuals

7.2.4 Tukey

Tukey with $c = 1$.

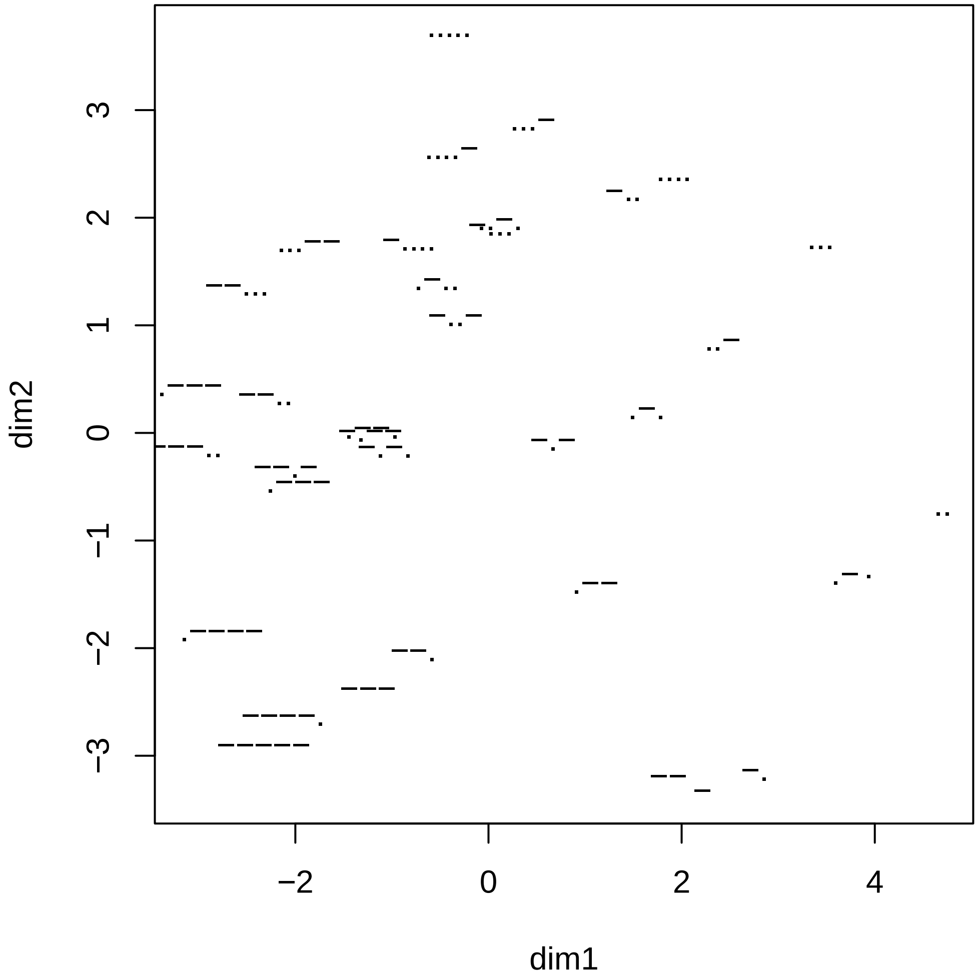

Figure 25: Configuration Median Tukey

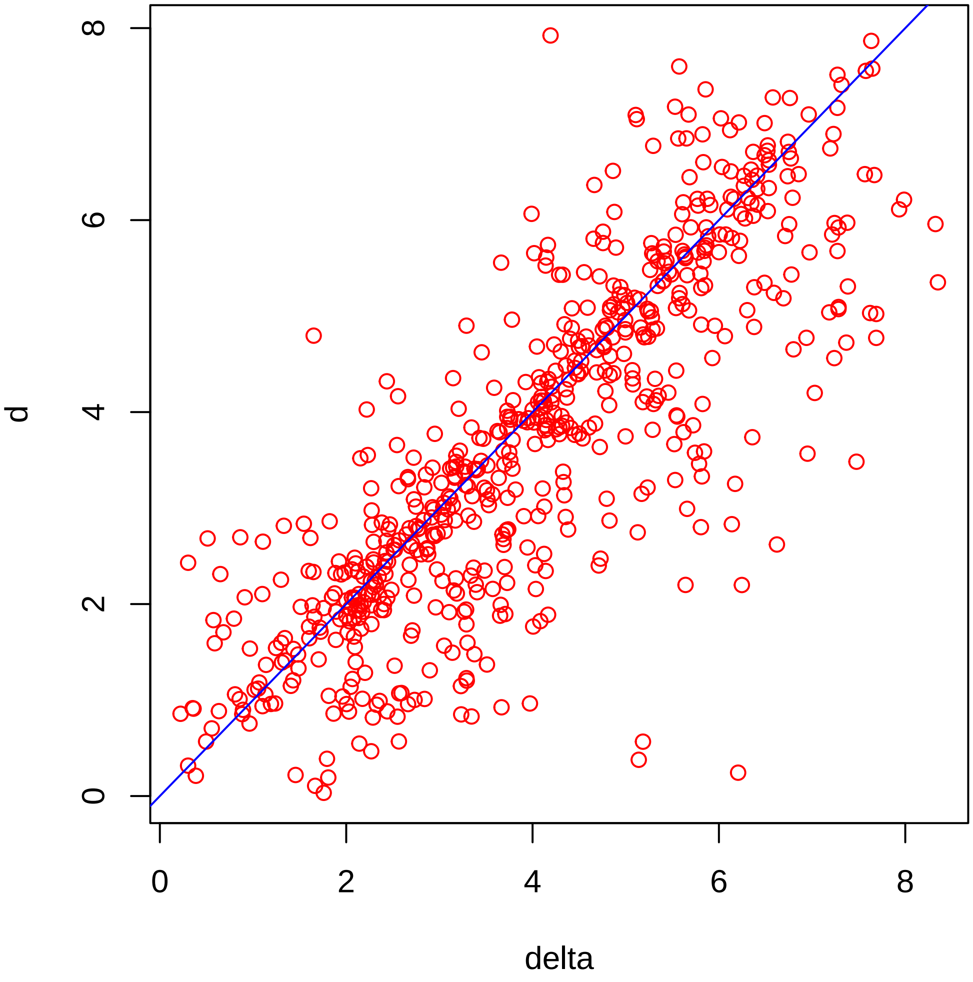

Figure 26: Shepard Plot Median Tukey

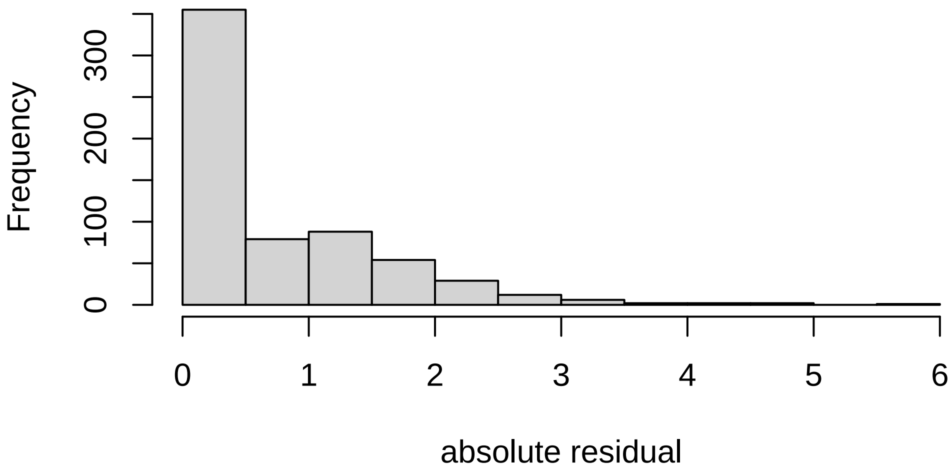

Figure 27: Histogram Tukey Residuals

8 Discussion

8.1 Fixed weights

8.2 Bounding the Second Derivative

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

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

$$f_c''(x) = (x^2 + c^2)^{-\frac{1}{2}} - x^2(x^2 + c^2)^{-\frac{3}{2}} \leq (x^2 + c^2)^{-\frac{1}{2}} \leq c^{-1}.$$

Thus

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

8.3 Residual Choice

In our examples and in our code we use the residuals $\delta_k - d_k(X)$ are arguments of our loss functions. From the statistical point of view we have to remember, however, that most of these loss functions were designed for the robust estimation of a location parameter or a linear regression function. The error distributions were explicitly or implicitly assumed to be symmetric around zero, and defined on the whole real line, which was reflected in the fact that loss functions were even and had infinite support. In MDS, however, distances and dissimilarities are non-negative and reasonable error functions are not symmetric. One could follow the example of Ramsay (1977) and measure residuals as $\log\delta_{ij} - \log d_{ij}(X)$. This does not have any effect on the majorization of the loss functions, but it means that in the smacof step to find $X^{(k+1)}$ we have to minimize

$$\sigma(X) = \sum w_k(X^{(k)})(\log\delta_{ij} - \log d_{ij}(X))^2,$$

which is considerably more complicated (De Leeuw, Groenen, and Mair (2016)).

8.4 Robust Nonmetric MDS

9 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.

```r
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

Aftab, K., and R. Hartley. 2015. "Convergence of Iteratively Re-Weighted Least Squares to Robust m-Estimators." In *2015 IEEE Winter Conference on Applications of Computer Vision,* 480–87. https://doi.org/10.1109/WACV.2015.70.

Andrews, D. F., P. J. Bickel, F. R. Hampel, P. J. Huber, W. H. Rogers, and J. W. Tukey. 1972. *Robust Estimators of Location: Survey and Advances*. Princeton University Press.

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.

Candes, E. J., and T. Tao. 2005. "Decoding by Linear Programming." *IEEE Transactions on Information Theory* 51 (12): 4203–15.

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.

Coleman, D., P. Holland, N. Kaden, V. Klema, and S. C. Peters. 1980. "A System of Subroutines for Iteratively Reweighted Least Squares Computations." *ACM Transactions on Mathematical Software* 6 (3): 327–36.

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., P. Groenen, and P. Mair. 2016. “Minimizing rStress Using Majorization.” 2016. https://jansweb.netlify.app/publication/deleeuw-groenen-mair-e-16-a/deleeuw-groenen-mair-e-16-a.pdf.

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

Donoho, D. L., and M. Elad. 2003. “Optimally Sparse Representation in General (Nonorthogonal) Dictionaries via ℓ_1 Minimization.” *Proceedings of the National Academy of Sciences* 100 (5): 2197–2202.

Groenen, P. J. F., P. Giaquinto, and H. A. L Kiers. 2003. “Weighted Majorization Algorithms for Weighted Least Squares Decomposition Models.” Econometric Institute Report EI 2003-09. Econometric Institute, Erasmus University Rotterdam. https://repub.eur.nl/pub/1700.

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 Statistics 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.

Holland, P. W., and R. E. Welsch. 1977. “Robust Regression Using Iteratively Reweighted Least-Squares.” *Communications in Statistics - Theory and Methods* 6 (9): 813–27. https://doi.org/10.1080/03610927708827533.

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

Hunter, D. R., and R. Li. 2005. “Variable Selection Using MM Algorithms.” *The Annals of Statistics* 33: 1617–42.

Jaakkola, T. S., and M. I. Jordan. 2000. “Bayesian Parameter Estimation via Variational Methods.” *Statistics and Computing* 10: 25–37.

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

Phillips, R. F. 2002. “Least Absolute Deviations Estimation via the EM Algorithm.” *Statistics and Computing* 12: 281–85.

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.

Ramsay, J. O. 1977. "Maximum Likelihood Estimation in Multidimensional Scaling." *Psychometrika* 42: 241–66.

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

Schlossmacher, E. J. 1973. "An Iterative Technique for Absolute Deviations Curve Ftting." *Journal of the American Statistical Association* 68: 857–59.

Van Ruitenburg, J. 2005. "Algorithms for Parameter Estimation in the Rasch Model." Measurement and Research Department Reports 2005-04. Arnhem, Netherlands: CITO. https://www.researchgate.net/publication/355568984_Algorithms_for_parameter_estimation_in_the_Rasch_model_Measurement_and_Research_Report_05-04#fullTextFileContent.

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