Package 'robscale'

March 6, 2026

Type Package

Title Fast Robust Estimation in Very Small Samples

Version 0.1.1

Description High-performance C++ implementation (via 'Rcpp') of the robust location and scale M-estimators described in Rousseeuw & Verboven (2002) <doi:10.1016/S0167-9473(02)00078-6> for very small samples. Provides numerically identical results to the 'revss' package with significantly improved performance through sorting networks and compiled iteration loops.

License MIT + file LICENSE

URL https://github.com/davdittrich/robscale,
https://doi.org/10.5281/zenodo.18828607

BugReports https://github.com/davdittrich/robscale/issues

Encoding UTF-8

Imports Rcpp (>= 1.0.0)

LinkingTo Rcpp

Suggests tinytest, revss, microbenchmark

NeedsCompilation yes

RoxygenNote 7.3.3

Author Dennis Alexis Valin Dittrich [aut, cre, cph] (ORCID: <https://orcid.org/0000-0002-4438-8276>)

Maintainer Dennis Alexis Valin Dittrich <davd@economicscience.net>

Repository CRAN

Date/Publication 2026-03-06 13:30:02 UTC

Contents

adm *Average Distance to the Median*

Description

Compute the mean absolute deviation from the median, scaled by a consistency constant for asymptotic normality under the Gaussian model.

Usage

```
adm(x, center, constant = 1.2533141373155, na.rm = FALSE)
```

Arguments

`x`	A numeric vector.
`center`	Optional numeric scalar giving the central value from which to measure the average absolute distance. Defaults to the median of `x`.
`constant`	Consistency constant for asymptotic normality at the Gaussian. Defaults to $\sqrt{\pi/2} \approx 1.2533$ (Nair, 1947). Set to `1` for the raw (unscaled) mean absolute deviation.
`na.rm`	Logical. If `TRUE`, `NA` values are stripped from `x` before computation. If `FALSE` (the default), the presence of any `NA` raises an error.

Details

The average distance to the median (ADM) is defined as

$$\mathrm{ADM}(x) \;=\; C \cdot \frac{1}{n}\sum_{i=1}^{n} |x_i - \mathrm{med}(x)|$$

where C is the consistency constant and $\mathrm{med}(x)$ is the sample median. When `center` is supplied it replaces the median.

The default constant $C = \sqrt{\pi/2}$ makes the ADM a consistent estimator of the standard deviation under the normal distribution. In large samples the ADM converges to σ at the Gaussian; however, this asymptotic property may **not** hold at the very small sample sizes (n = 3–8) for which this package is primarily intended.

The ADM is *not* robust against outliers in the explosion sense: its explosion breakdown point is $1/n$. However, it is highly resistant to implosion, with an implosion breakdown point of $(n-1)/n$. It therefore serves as the fallback scale estimator in `robScale` when the MAD collapses to zero.

Value

A single numeric value: the scaled mean absolute deviation from the center. Returns `NA` if `x` has length zero after removal of `NA`s.

References

Nair, K. R. (1947) A Note on the Mean Deviation from the Median. *Biometrika*, **34**(3/4), 360–362. doi:10.2307/2332448

Rousseeuw, P. J. and Verboven, S. (2002) Robust estimation in very small samples. *Computational Statistics & Data Analysis*, **40**(4), 741–758. doi:10.1016/S01679473(02)000786

See Also

`mad` for the median absolute deviation from the `median`; `robScale` for the M-estimator of scale that uses the ADM as an implosion fallback.

Examples

```
adm(c(1:9))

x <- c(1, 2, 3, 5, 7, 8)
adm(x)                      # with consistency constant
adm(x, constant = 1)        # raw mean absolute deviation

# Supply a known center
adm(x, center = 4.0)
```

robLoc *Robust M-Estimate of Location*

Description

Compute the robust M-estimate of location for very small samples using the logistic ψ function of Rousseeuw & Verboven (2002).

Usage

```
robLoc(
  x,
  scale = NULL,
  na.rm = FALSE,
  maxit = 80L,
  tol = sqrt(.Machine$double.eps)
)
```

Arguments

`x` A numeric vector.

`scale` Optional numeric scalar giving a known scale. When supplied, the MAD is replaced by this value and the minimum sample size for iteration is lowered from 4 to 3 (see ‘Details’).

`na.rm` Logical. If `TRUE`, `NA` values are stripped from `x` before computation. If `FALSE` (the default), the presence of any `NA` raises an error.

`maxit` Maximum number of Newton–Raphson iterations. Defaults to 80.

`tol` Convergence tolerance. Iteration stops when the absolute Newton step falls below `tol`. Defaults to `sqrt(.Machine$double.eps)`.

Details

The location estimator T_n is the solution to the M-estimating equation

$$\frac{1}{n}\sum_{i=1}^{n}\psi_{\log}\left(\frac{x_i - T_n}{S_n}\right) = 0$$

where S_n is a fixed auxiliary scale estimate and $\psi_{\log}$ is the logistic psi function (Rousseeuw & Verboven 2002, Eq. 23):

$$\psi_{\log}(x) = \frac{e^x - 1}{e^x + 1} = \tanh(x/2)$$

This function is bounded in $(-1, 1)$, smooth (C^∞), and strictly monotone. Boundedness provides robustness against outliers; smoothness avoids the corner artifacts of Huber's ψ at small n.

Iteration scheme. The estimating equation is solved by Newton–Raphson iteration. The derivative of the logistic psi satisfies $\psi'(x) = 1 - \psi^2(x)$, so the Newton step requires no additional transcendental function evaluations beyond those already computed for the numerator. Starting value: $T^{(0)} = \mathrm{med}(x)$. Auxiliary scale: $S = \mathrm{MAD}(x)$ unless `scale` is supplied.

Decoupled estimation. Location and scale are estimated separately: `robLoc` holds the auxiliary scale fixed at $\mathrm{MAD}(x)$ throughout iteration, following the decoupled approach of Rousseeuw & Verboven (2002, Sec. 4.1). This avoids the positive-feedback instability of simultaneous location–scale iteration (Huber's Proposal 2) in small samples.

Fallback. When the sample is too small for reliable iteration the function returns the `median` directly:

- $n < 4$ when `scale` is unknown (the MAD is unreliable at $n = 3$);
- $n < 3$ when `scale` is known.

Value

A single numeric value: the robust M-estimate of location. Returns `NA` if `x` has length zero (after removal of `NA`s when `na.rm = TRUE`).

References

Rousseeuw, P. J. and Verboven, S. (2002) Robust estimation in very small samples. *Computational Statistics & Data Analysis*, **40**(4), 741–758. doi:10.1016/S01679473(02)000786

See Also

`median` for the starting value; `mad` for the auxiliary scale; `robScale` for the companion scale estimator.

Examples

```
robLoc(c(1:9))

x <- c(1, 2, 3, 5, 7, 8)
robLoc(x)

# Known scale lowers the minimum sample size to 3
robLoc(c(1, 2, 3), scale = 1.5)

# Outlier resistance
x_clean <- c(2.0, 3.1, 2.7, 2.9, 3.3)
x_dirty <- replace(x_clean, 5, 100)
c(robLoc(x_clean), robLoc(x_dirty))   # barely moves
c(mean(x_clean), mean(x_dirty))       # destroyed
```

`robScale` *Robust M-Estimate of Scale*

Description

Compute the robust M-estimate of scale for very small samples using the ρ function of Rousseeuw & Verboven (2002).

Usage

```
robScale(
  x,
  loc = NULL,
  implbound = 0.0001,
  na.rm = FALSE,
  maxit = 80L,
  tol = sqrt(.Machine$double.eps)
)
```

Arguments

`x` A numeric vector.

`loc` Optional numeric scalar giving a known location. When supplied, the observations are centered at `loc` and the minimum sample size for iteration is lowered from 4 to 3 (see ‘Details’).

`implbound` Implosion bound: the smallest value the MAD is allowed to take before it is considered to have “imploded” (collapsed to zero). Defaults to `1e-4`.

`na.rm` Logical. If `TRUE`, `NA` values are stripped from `x` before computation. If `FALSE` (the default), the presence of any `NA` raises an error.

`maxit` Maximum number of multiplicative iterations. Defaults to 80.

`tol` Convergence tolerance. Iteration stops when the multiplicative update factor satisfies $|v - 1| \leq \mathrm{tol}$. Defaults to `sqrt(.Machine$double.eps)`.

Details

The scale estimator S_n solves the M-estimating equation

$$\frac{1}{n}\sum_{i=1}^{n}\rho\left(\frac{x_i - T_n}{S_n}\right) = \beta$$

where T_n is fixed at the sample median, $\beta = 0.5$, and ρ is a smooth rho function defined as the square of the logistic psi (Rousseeuw & Verboven, 2002, Sec. 4.2):

$$\rho_{\log}(x) = \psi_{\log}^2\left(\frac{x}{c}\right)$$

with the tuning constant $c = 0.37394112142347236$ chosen so that

$$\int \rho(u)\, d\Phi(u) = 0.5$$

yielding a 50% breakdown point.

Iteration scheme. The equation is solved by multiplicative iteration (Rousseeuw & Verboven, 2002, Eq. 27):

$$S^{(k+1)} = S^{(k)} \cdot \sqrt{2 \cdot \frac{1}{n}\sum \psi_{\log}^2\left(\frac{x_i - T}{c \cdot S^{(k)}}\right)}$$

Starting value: $S^{(0)} = \mathrm{MAD}(x)$. The logistic psi values are computed via the algebraic identity $\psi_{\log}(x) = \tanh(x/2)$.

Decoupled estimation. Scale is estimated with location held fixed at $\mathrm{med}(x)$, following the decoupled approach of Rousseeuw & Verboven (2002, Sec. 4.2). This avoids the positive-feedback instability of Huber's Proposal 2 in small samples.

Known location. When `loc` is supplied, the observations are centered as $x_i - \mu$ and the initial scale is set to $1.4826 \cdot \mathrm{med}(|x_i - \mu|)$ rather than the MAD. This lowers the minimum sample size from 4 to 3 (Rousseeuw & Verboven, 2002, Sec. 5).

Fallback. When n is below the minimum for iteration:

- if $\mathrm{MAD}(x) \leq$ `implbound` (implosion), the function returns `adm(x)`;
- otherwise, it returns $\mathrm{MAD}(x)$.

Value

A single numeric value: the robust M-estimate of scale. Returns `NA` if `x` has length zero (after removal of `NA`s when `na.rm = TRUE`).

References

Rousseeuw, P. J. and Verboven, S. (2002) Robust estimation in very small samples. *Computational Statistics & Data Analysis*, **40**(4), 741–758. doi:10.1016/S01679473(02)000786

See Also

`adm` for the implosion fallback; `mad` for the starting value and classical alternative; `robLoc` for the companion location estimator.

Examples

```
robScale(c(1:9))

x <- c(1, 2, 3, 5, 7, 8)
robScale(x)
robScale(x, loc = 5)           # known location

# Outlier resistance
x_clean <- c(2.0, 3.1, 2.7, 2.9, 3.3)
x_dirty <- replace(x_clean, 5, 100)
c(robScale(x_clean), robScale(x_dirty))   # barely moves
c(sd(x_clean), sd(x_dirty))               # destroyed

# MAD implosion: identical values cause MAD = 0
robScale(c(5, 5, 5, 5, 6))     # falls back to adm()
```

Index