# Package: smoothmest (via r-universe)

September 18, 2024

**Title** Smoothed M-Estimators for 1-Dimensional Location

**Version** 0.1-3

**Date** 2022-04-27

**Author** Christian Hennig <christian.hennig@unibo.it>

**Depends** R (>= 2.0), MASS

**Description** Some M-estimators for 1-dimensional location (Bisquare, ML for the Cauchy distribution, and the estimators from application of the smoothing principle introduced in Hampel, Hennig and Ronchetti (2011) to the above, the Huber M-estimator, and the median, main function is smoothm), and Pitman estimator.

**Maintainer** Christian Hennig <christian.hennig@unibo.it>

**License** GPL

**URL** https://www.unibo.it/sitoweb/christian.hennig/en

**NeedsCompilation** no

**Date/Publication** 2022-04-27 22:10:05 UTC

**Repository** https://chrhennig.r-universe.dev

**RemoteUrl** https://github.com/cran/smoothmest

**RemoteRef** HEAD

**RemoteSha** f5ec2dd159e41a6b6eac48ac2e06ce87bc25aa1c

## Contents

---

ddoublex *The double exponential (Laplace) distribution*

---

### Description

Density for and random values from double exponential (Laplace) distribution with density `exp(-abs(x-mu)/lambda)/(2*l` for which the median is the ML estimator.

### Usage

```
ddoublex(x, mu=0, lambda=1)
rdoublex(n,mu=0,lambda=1)
```

### Arguments

| | |
|---|---|
| `x` | numeric vector. |
| `mu` | numeric. Distribution median. |
| `lambda` | numeric. Scale parameter. |
| `n` | integer. Number of random values to be generated. |

### Details

**ddoublex:** density.

**rdoublex:** random number generation.

### Value

`ddoublex` gives out a vector of density values.

`rdoublex` gives out a vector of random numbers generated by the double exponential distribution.

### Author(s)

Christian Hennig `<chrish@stats.ucl.ac.uk>` http://www.homepages.ucl.ac.uk/~ucakche/

### References

Huber, P. J. and Ronchetti, E. (2009) Robust Statistics (2nd ed.). Wiley, New York.

### Examples

```
set.seed(123456)
ddoublex(1:5,lambda=5)
rdoublex(5,mu=10,lambda=5)
```

dhuber *Huber's least favourable distribution*

## Description

Density for and random values from Huber's least favourable distribution, see Huber and Ronchetti (2009).

## Usage

```
dhuber(x, k=0.862, mu=0, sigma=1)
edhuber(x, k=0.862, mu=0, sigma=1)
rhuber(n,k=0.862, mu=0, sigma=1)
```

## Arguments

| | |
|---|---|
| x | numeric vector. |
| k | numeric. Borderline value of central Gaussian part of the distribution. The default values refers to a 20% contamination neighborhood of the Gaussian distribution. |
| mu | numeric. distribution mean. |
| sigma | numeric. Distribution scale (sigma=1 defines the distribution in standard form, with standard Gaussian centre). |
| n | integer. Number of random values to be generated. |

## Details

**dhuber:** density.

**edhuber:** density, and computes the contamination proportion corresponding to k.

**rhuber:** random number generation.

## Value

dhuber gives out a vector of density values.

edhuber gives out a list with components val (density values) and eps (contamination proportion).

rhuber gives out a vector of random numbers generated by Huber's least favourable distribution.

## Author(s)

Christian Hennig <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche/

## References

Huber, P. J. and Ronchetti, E. (2009) Robust Statistics (2nd ed.). Wiley, New York.

### Examples

```
set.seed(123456)
edhuber(1:5,k=1.5)
rhuber(5)
```

---

pdens *Auxiliary functions for pitman*

---

### Description

Auxiliary functions for `pitman`.

### Usage

```
pdens(z, x, dfunction, ...)
sdens(z, x, dfunction, ...)
dens(x, dfunction, ...)
```

### Arguments

| | |
|---|---|
| `z` | numeric vector. |
| `x` | numeric vector. |
| `dfunction` | a density function defining the distribution for which the Pitman estimator is computed. |
| `...` | further arguments to be passed on to the density function `dfunction`. |

### Details

**dens** product of density values at `x`.

**pdens** vector of `z*dens(x-z)`.

**sdens** vector of `dens(x-z)`.

### Value

Numeric value (`dens`) or vector.

### Author(s)

Christian Hennig `<chrish@stats.ucl.ac.uk>` `http://www.homepages.ucl.ac.uk/~ucakche/`

### References

Pitman, E.J. (1939) The estimation of the location and scale parameters of a continuous population of any given form. Biometrika 30, 391-421.

### See Also

pitman

### Examples

```
dens(1:5,dcauchy)
pdens(1:5,0,dcauchy)
sdens(1:5,0:2,dcauchy)
```

---

## pitman — *Pitman location estimator*

---

### Description

Pitman estimator of one-dimensional location, optimal with scale assumed to be known. Calculated by brute force (using `integrate`).

### Usage

```
pitman(y, d=ddoublex, lower=-Inf, upper=Inf, s=mad(y), ...)
```

### Arguments

| | |
|---|---|
| `y` | numeric vector. Data set. |
| `d` | a density function defining the distribution for which the Pitman estimator is computed. |
| `lower` | numeric. Lower bound for the involved integrals (should be `-Inf` unless there are numerical problems). |
| `upper` | numeric. Lower bound for the involved integrals (should be `Inf` unless there are numerical problems). |
| `s` | numeric. Estimated or assumed scale/standard deviation. |
| `...` | further arguments to be passed on to the density function `d`. |

### Value

The estimated value.

### Author(s)

Christian Hennig <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche/

### References

Pitman, E.J. (1939) The estimation of the location and scale parameters of a continuous population of any given form. Biometrika 30, 391-421.

### See Also

smoothm

### Examples

```
set.seed(10001)
y <- rdoublex(7)
pitman(y,ddoublex)
pitman(y,dcauchy)
pitman(y,dnorm)
```

---

## smoothm *Smoothed and unsmoothed 1-d location M-estimators*

---

### Description

smoothm is an interface for all the smoothed M-estimators introduced in Hampel, Hennig and Ronchetti (2011) for one-dimensional location, the Huber- and Bisquare-M-estimator and the ML-estimator of the Cauchy distribution, calling all the other functions documented on this page.

### Usage

```
smoothm(y, method="smhuber",
  k=0.862, sn=sqrt(2.046/length(y)),
  tol=1e-06,  s=mad(y), init="median")

sehuber(y, k = 0.862, tol = 1e-06, s=mad(y), init="median")

smhuber(y, k = 0.862, sn=sqrt(2.046/length(y)), tol = 1e-06, s=mad(y),
  smmed=FALSE, init="median")

mbisquare(y, k=4.685, tol = 1e-06, s=mad(y), init="median")

smbisquare(y, k=4.685, tol = 1e-06, sn=sqrt(1.0526/length(y)),
  s=mad(y), init="median")

mlcauchy(y, tol = 1e-06, s=mad(y))

smcauchy(y, tol = 1e-06, sn=sqrt(2/length(y)), s=mad(y))
```

### Arguments

| | |
|---|---|
| y | numeric vector. Data set. |
| method | one of "huber", "smhuber", "bisquare", "smbisquare", "cauchy", "smcauchy", "smmed". See details. |

`k` numeric. Tuning constant. This is used for `method` one of `"huber"`, `"smhuber"`, `"bisquare"`, `"smbisquare"` in `smoothm` and the corresponding functions. Tuning constants are defined for the Huber- and Bisquare M-estimator as in Maronna et al. (2006). The default values refer to a Huber M-estimator which is optimal under 20% contamination (0.862) and to a Bisquare M-estimator with 95% efficiency under the Gaussian model (4.685).

`sn` numeric. This is used for `method` one of `"smhuber"`, `"smbisquare"`, `"smcauchy"`, `"smmed"`. This is the smoothing standard error $\sigma_n$ in Hampel et al. (2011) depending on the sample size and the asymptotic variance of the base estimator. The default value of `smoothm` and `smhuber` is based on a Huber estimator with `k=0.862` under Huber's least favourable distribution for which it is ML. The default value of `smbisquare` is based on the Bisquare estimator with `k=4.685` under the Gaussian distribution. The default value of `smcauchy` is based on the Cauchy ML estimator under the Cauchy distribution. A value that can be used for the smoothed median is `sqrt(1.056/length(y))`, which is based on the median under the double exponential (Laplace) distribution with 1.4826 MAD=1. Note that the distributional "assumptions" for these choices are by no means critical; they should work well under many other distributions as well.

`tol` numeric. Stopping criterion for algorithms (absolute difference between two successive values).

`s` numeric. Estimated or assumed scale/standard deviation.

`init` `"median"` or `"mean"`. Initial estimator for iteration. Ignored if `method=="cauchy"` or `"smcauchy"`.

`smmed` logical. If `FALSE`, the smoothed Huber estimator is computed, otherwise the smoothed median by `smhuber`.

## Details

The following estimators can be computed (some computational details are given in Hampel et al. 2011):

**Huber estimator.** `method="huber"` and function `sehuber` compute the standard Huber estimator (Huber and Ronchetti 2009). The only differences from huber are that `s` and `init` can be specified and that the default `k` is different.

**Smoothed Huber estimator.** `method="smhuber"` and function `smhuber` compute the smoothed Huber estimator (Hampel et al. 2011).

**Bisquare estimator.** `method="bisquare"` and function `bisquare` compute the bisquare M-estimator (Maronna et al. 2006). This uses `psi.bisquare`.

**Smoothed bisquare estimator.** `method="smbisquare"` and function `smbisquare` compute the smoothed bisquare M-estimator (Hampel et al. 2011). This uses `psi.bisquare`

**ML estimator for Cauchy distribution.** `method="cauchy"` and function `mlcauchy` compute the ML-estimator for the Cauchy distribution.

**Smoothed ML estimator for Cauchy distribution.** `method="smcauchy"` and function `smcauchy` compute the smoothed ML-estimator for the Cauchy distribution (Hampel et al. 2011).

**Smoothed median.** `method="smmed"` and function `smhuber` with `median=TRUE` compute the smoothed median (Hampel et al. 2011).

## Value

A list with components

| | |
|---|---|
| mu | the location estimator. |
| method | see above. |
| k | see above. |
| sn | see above. |
| tol | see above. |
| s | see above. |

## Author(s)

Christian Hennig <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche/

## References

Hampel, F., Hennig, C. and Ronchetti, E. (2011) A smoothing principle for the Huber and other location M-estimators. Computational Statistics and Data Analysis 55, 324-337.

Huber, P. J. and Ronchetti, E. (2009) Robust Statistics (2nd ed.). Wiley, New York.

Maronna, A.R., Martin, D.R., Yohai, V.J. (2006). Robust Statistics: Theory and Methods. Wiley, New York

## See Also

pitman, huber, rlm

## Examples

```
library(MASS)
set.seed(10001)
y <- rdoublex(7)
median(y)
huber(y)$mu
smoothm(y)$mu
smoothm(y,method="huber")$mu
smoothm(y,method="bisquare",k=4.685)$mu
smoothm(y,method="smbisquare",k=4.685,sn=sqrt(1.0526/7))$mu
smoothm(y,method="cauchy")$mu
smoothm(y,method="smcauchy",sn=sqrt(2/7))$mu
smoothm(y,method="smmed",sn=sqrt(1.0526/7))$mu
smoothm(y,method="smmed",sn=sqrt(1.0526/7),init="mean")$mu
```

smpsi *Auxiliary functions for smoothm*

## Description

Psi-functions, derivatives and further auxiliary functions used for computing the estimators in smoothm.

## Usage

```
psicauchy(x)
psidcauchy(x)
likcauchy(x,mu)
flikcauchy(y,x,mu,sn)
smtfcauchy(x,mu,sn)
smcipsi(y, x, sn=sqrt(2/length(x)))
smcipsid(y, x, sn=sqrt(2/length(x)))
smcpsi(x, sn=sqrt(2/length(x)))
smcpsid(x, sn=sqrt(2/length(x)))
smbpsi(y, x, k=4.685, sn=sqrt(2/length(x)))
smbpsid(y, x, k=4.685, sn=sqrt(2/length(x)))
smbpsii(x, k=4.685, sn=sqrt(2/length(x)))
smbpsidi(x, k=4.685, sn=sqrt(2/length(x)))
smpsi(x,k=0.862,sn=sqrt(2/length(x)))
smpmed(x,sn=sqrt(1/5))
```

## Arguments

| | |
|---|---|
| x | numeric vector. |
| mu | numeric. |
| y | numeric vector. |
| sn | numeric. Smoothing constant. See smoothm. |
| k | numeric. Tuning constant. See smoothm. |

## Details

**psicauchy** psi-function for Cauchy ML-estimator at x.

**psidcauchy** derivative of psicauchy at x.

**likcauchy** Cauchy likelihood of data x for mode parameter mu.

**flikcauchy** vector of Gaussian density at y with mean 0 and st. dev. sn times Cauchy log-likelihood of x with mode parameter mu+y.

**smtfcauchy** integral of flikcauchy with y running from -Inf to Inf.

**smcipsi** psicauchy(x-y)*dnorm(y,sd=sn).

**smcipsid** derivative of smcipsi w.r.t. x.

**smcpsi** psi-function for smoothed Cauchy ML-estimator. Integral of `smpcipsi` with `y` running from `-Inf` to `Inf`.

**smcpsid** integral of `smpcipsid` with `y` running from `-Inf` to `Inf`.

**smbpsi** `(x-y)*psi.bisquare(x-y,c=k)*dnorm(y,sd=sn)`.

**smbpsid** `psi.bisquare(x-y,c=k,deriv=1)*dnorm(y,sd=sn)`.

**smbpsii** psi-function for smoothed bisquare M-estimator. Integral of `smbpsi` with `y` running from `-Inf` to `Inf`.

**smbpsidi** integral of `smbpsid` with `y` running from `-Inf` to `Inf`.

**smpsi** psi-function for smoothed Huber-estimator at `x`.

**smpmed** psi-function for smoothed median at `x`.

## Value

A numeric vector.

## Author(s)

Christian Hennig <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche/

## References

Hampel, F., Hennig, C. and Ronchetti, E. (2011) A smoothing principle for the Huber and other location M-estimators. Computational Statistics and Data Analysis 55, 324-337.

Huber, P. J. and Ronchetti, E. (2009) Robust Statistics (2nd ed.). Wiley, New York.

Maronna, A.R., Martin, D.R., Yohai, V.J. (2006). Robust Statistics: Theory and Methods. Wiley, New York

## See Also

`smoothm`, `psi.huber`, `psi.bisquare`

## Examples

```
psicauchy(1:5)
psidcauchy(1:5)
likcauchy(1:5,0)
flikcauchy(3,1:5,0,1)
smtfcauchy(1:5,0,1)
smcipsi(1,1:3)
smcipsid(1,1:3)
smcpsi(1:5)
smcpsid(1:5)
smbpsi(1,1:5)
smbpsid(0:4,1:5)
smbpsii(1:5)
smbpsidi(1:5)
smpsi(1:5)
smpmed(1:5)
```

# Index