

# Package ‘cellWise’

November 26, 2020

**Type** Package

**Version** 2.2.2

**Date** 2020-11-26

**Title** Analyzing Data with Cellwise Outliers

**Depends** R (>= 3.2.0)

**Suggests** knitr, robustHD, MASS, ellipse, markdown, rospca, GSE

**Imports** reshape2, scales, ggplot2, matrixStats, gridExtra, robustbase,  
rrcov, svd, stats, Rcpp (>= 0.12.10.14)

**LinkingTo** Rcpp, RcppArmadillo (>= 0.7.600.1.0)

**Description** Tools for detecting cellwise outliers and robust methods to analyze data which may contain them. Contains the implementation of the algorithms described in Rousseeuw and Van den Bossche (2018) <doi:10.1080/00401706.2017.1340909> (open access) Hubert et al. (2019) <doi:10.1080/00401706.2018.1562989> (open access), Raymaekers and Rousseeuw (2019) <doi:10.1080/00401706.2019.1677270> (open access), Raymaekers and Rousseeuw (2020) <arXiv:2005.07946> (open access), Raymaekers and Rousseeuw (2020) <arXiv:1912.12446> (open access).  
Examples can be found in the vignettes:  
``DDC\_examples``, ``MacroPCA\_examples``, ``wrap\_examples`` and ``transfo\_examples``.

**License** GPL (>= 2)

**LazyLoad** yes

**Author** Jakob Raymaekers [aut, cre],  
Peter Rousseeuw [aut],  
Wannes Van den Bossche [ctb],  
Mia Hubert [ctb]

**Maintainer** Jakob Raymaekers <jakob.raymaekers@kuleuven.be>

**VignetteBuilder** knitr

**RoxygenNote** 6.1.1

**NeedsCompilation** yes

**Repository** CRAN

**Date/Publication** 2020-11-26 15:50:12 UTC

## R topics documented:

cellHandler	2
cellMap	4
checkDataSet	5
DDC	7
DDCpredict	11
DI	13
dog_walker	15
dposs	15
estLocScale	16
glass	17
ICPCA	18
MacroPCA	20
MacroPCApredict	22
mortality	24
outlierMap	25
philips	26
transfo	27
truncPC	29
wrap	30
<b>Index</b>	<b>33</b>

---

cellHandler	<i>cellHandler algorithm</i>
-------------	------------------------------

---

### Description

This function flags cellwise outliers in  $X$  and imputes them, if robust estimates of the center  $\mu$  and scatter matrix  $\Sigma$  are given. When the latter are not known, as is typically the case, one can use the function [DDC](#) which only requires the data matrix  $X$ . Alternatively, the unknown center  $\mu$  and scatter matrix  $\Sigma$  can be estimated robustly from  $X$  by the function [DI](#).

### Usage

```
cellHandler(X, mu, Sigma, quant = 0.99)
```

### Arguments

$X$	$X$ is the input data, and must be an $n$ by $d$ matrix or a data frame.
$\mu$	An estimate of the center of the data
$\Sigma$	An estimate of the covariance matrix of the data
quant	Cutoff used in the detection of cellwise outliers. Defaults to 0.99

**Value**

A list with components:

- `Ximp`  
The imputed data matrix.
- `indcells`  
Indices of the cells which were flagged in the analysis.
- `indNAs`  
Indices of the NAs in the data.
- `Zres`  
Matrix with standardized cellwise residuals of the flagged cells. Contains zeroes in the unflagged cells.
- `Zres_denom`  
Denominator of the standardized cellwise residuals.
- `cellPaths`  
Matrix with the same dimensions as `X`, in which each row contains the path of least angle regression through the cells of that row, i.e. the order of the coordinates in the path (1=first, 2=second,...)

**Author(s)**

J. Raymaekers and P.J. Rousseeuw

**References**

J. Raymaekers and P.J. Rousseeuw (2020). Handling cellwise outliers by sparse regression and robust covariance. *Arxiv: 1912.12446*. ([link to open access pdf](#))

**See Also**

[DI](#)

**Examples**

```
mu <- rep(0, 3)
Sigma <- diag(3) * 0.1 + 0.9
X <- rbind(c(0.5, 1.0, 5.0), c(-3.0, 0.0, 1.0))
n <- nrow(X); d <- ncol(X)
out <- cellHandler(X, mu, Sigma)
Xres <- X - out$Ximp # unstandardized residual
mean(abs(as.vector(Xres - out$Zres*out$Zres_denom))) # 0
W <- matrix(rep(0,n*d),nrow=n) # weight matrix
W[out$Zres != 0] <- 1 # 1 indicates cells that were flagged
```

---

 cellMap

*Draw a cellmap*


---

### Description

This function draws a cellmap, possibly of a subset of rows and columns of the data, and possibly combining cells into blocks. A cellmap shows which cells are missing and which ones are outlying, marking them in red for unusually large cell values and in blue for unusually low cell values. When cells are combined into blocks, the final color is the average of the colors in the individual cells.

### Usage

```
cellMap(D, R, indcells = NULL, indrows = NULL,
        standOD=NULL, showVals=NULL, rowlabels="",
        columnlabels="", mTitle="", rowtitle="",
        columntitle="", showrows=NULL, showcolumns=NULL,
        nrowinblock=1, ncolumnsinblock=1, autolabel=TRUE,
        columnangle=90, sizetitles=1.1, adjustrowlabels=1,
        adjustcolumnlabels=1, colContrast=1, outlyingGrad=TRUE,
        darkestColor = sqrt(qchisq(0.999,1)),
        drawCircles = TRUE)
```

### Arguments

D	The data matrix (required input argument).
R	Matrix of standardized residuals of the cells (required input argument)
indcells	Indices of outlying cells. Defaults to NULL, which indicates the cells for which $ R  > \sqrt{qchisq(0.99, 1)}$ .
indrows	Indices of outlying rows. By default no rows are indicated.
standOD	Standardized Orthogonal Distance of each row. Defaults to NULL, then no rows are indicated.
showVals	Takes the values "D", "R" or NULL and determines whether or not to show the entries of the data matrix (D) or the residuals (R) in the cellmap. Defaults to NULL, then no values are shown.
rowlabels	Labels of the rows.
columnlabels	Labels of the columns.
mTitle	Main title of the cellMap.
rowtitle	Title for the rows.
columntitle	Title for the columns.
showrows	Indices of the rows to be shown. Defaults to NULL which means all rows are shown.
showcolumns	Indices of the columns to be shown. Defaults to NULL which means all columns are shown.

nrowsinblock	How many rows are combined in a block. Defaults to 1.
ncolumnsinblock	How many columns are combined in a block. Defaults to 1.
autolabel	Automatically combines labels of cells in blocks. If FALSE, you must provide the final columnlabels and/or rowlabels. Defaults to TRUE.
columnangle	Angle of the column labels. Defaults to 90.
sizetitles	Size of row title and column title. Defaults to 1.1.
adjustrowlabels	Adjust row labels: 0=left, 0.5=centered, 1=right. Defaults to 1.
adjustcolumnlabels	Adjust column labels: 0=left, 0.5=centered, 1=right. Defaults to 1.
colContrast	Parameter regulating the contrast of colors, should be in [1, 5]. Defaults to 1.
outlyingGrad	If TRUE, the color is gradually adjusted in function of the outlyingness. Defaults to TRUE.
darkestColor	Standardized residuals bigger than this will get the darkest color.
drawCircles	Whether or not to draw black circles indicating the outlying rows.

**Author(s)**

Rousseeuw P.J., Van den Bossche W.

**References**

Rousseeuw, P.J., Van den Bossche W. (2018). Detecting Deviating Data Cells. *Technometrics*, **60**(2), 135-145. ([link to open access pdf](#))

**See Also**

[DDC](#)

**Examples**

```
# For examples of the cellmap, we refer to the vignette:
vignette("DDC_examples")
```

---

checkDataSet

*Clean the dataset*

---

**Description**

This function checks the dataset X, and sets aside certain columns and rows that do not satisfy the conditions. It is used by the [DDC](#) and [MacroPCA](#) functions but can be used by itself, to clean a dataset for a different type of analysis.

**Usage**

```
checkDataSet(X, fracNA = 0.5, numDiscrete = 3, precScale = 1e-12, silent = FALSE,
cleanNAfirst = "automatic")
```

**Arguments**

<code>X</code>	<code>X</code> is the input data, and must be an $n$ by $d$ matrix or data frame.
<code>fracNA</code>	Only retain columns and rows with fewer NAs than this fraction. Defaults to 0.5.
<code>numDiscrete</code>	A column that takes on <code>numDiscrete</code> or fewer values will be considered discrete and not retained in the cleaned data. Defaults to 3.
<code>precScale</code>	Only consider columns whose scale is larger than <code>precScale</code> . Here scale is measured by the median absolute deviation. Defaults to $1e - 12$ .
<code>silent</code>	Whether or not the function progress messages should be printed. Defaults to FALSE.
<code>cleanNAfirst</code>	If "columns", first columns then rows are checked for NAs. If "rows", first rows then columns are checked for NAs. "automatic" checks columns first if $d \geq 5n$ and rows first otherwise. Defaults to "automatic".

**Value**

A list with components:

- `colInAnalysis`  
Column indices of the columns used in the analysis.
- `rowInAnalysis`  
Row indices of the rows used in the analysis.
- `namesNotNumeric`  
Names of the variables which are not numeric.
- `namesCaseNumber`  
The name of the variable(s) which contained the case numbers and was therefore removed.
- `namesNAcol`  
Names of the columns left out due to too many NA's.
- `namesNArow`  
Names of the rows left out due to too many NA's.
- `namesDiscrete`  
Names of the discrete variables.
- `namesZeroScale`  
Names of the variables with zero scale.
- `remX`  
Remaining (cleaned) data after `checkDataSet`.

**Author(s)**

Rousseeuw P.J., Van den Bossche W.

## References

Rousseeuw, P.J., Van den Bossche W. (2018). Detecting Deviating Data Cells. *Technometrics*, **60**(2), 135-145. ([link to open access pdf](#))

## See Also

[DDC](#), [MacroPCA](#), [transfo](#), [wrap](#)

## Examples

```
library(MASS)
set.seed(12345)
n <- 100; d = 10
A <- matrix(0.9, d, d); diag(A) = 1
x <- mvrnorm(n, rep(0,d), A)
x[sample(1:(n * d), 100, FALSE)] <- NA
x <- cbind(1:n, x)
checkedx <- checkDataSet(x)

# For more examples, we refer to the vignette:
vignette("DDC_examples")
```

---

DDC

*Detect Deviating Cells*

---

## Description

This function aims to detect cellwise outliers in the data. These are entries in the data matrix which are substantially higher or lower than what could be expected based on the other cells in its column as well as the other cells in its row, taking the relations between the columns into account. Note that this function first calls [checkDataSet](#) and analyzes the remaining cleaned data.

## Usage

```
DDC(X, DDCpars = list())
```

## Arguments

X	X is the input data, and must be an $n$ by $d$ matrix or a data frame.
DDCpars	A list of available options: <ul style="list-style-type: none"><li>• <code>fracNA</code> Only consider columns and rows with fewer NAs (missing values) than this fraction (percentage). Defaults to 0.5.</li><li>• <code>numDiscrete</code> A column that takes on <code>numDiscrete</code> or fewer values will be considered discrete and not used in the analysis. Defaults to 3.</li></ul>

- `precScale`  
Only consider columns whose scale is larger than `precScale`. Here scale is measured by the median absolute deviation. Defaults to  $1e - 12$ .
- `cleanNAfirst`  
If "columns", first columns then rows are checked for NAs. If "rows", first rows then columns are checked for NAs. "automatic" checks columns first if  $d \geq 5n$  and rows first otherwise. Defaults to "automatic".
- `tolProb`  
Tolerance probability, with default 0.99, which determines the cutoff values for flagging outliers in several steps of the algorithm.
- `corrlim`  
When trying to estimate  $z_{ij}$  from other variables  $h$ , we will only use variables  $h$  with  $|\rho_{j,h}| \geq \text{corrlim}$ . Variables  $j$  without any correlated variables  $h$  satisfying this are considered standalone, and treated on their own. Defaults to 0.5.
- `combinRule`  
The operation to combine estimates of  $z_{ij}$  coming from other variables  $h$ : can be "mean", "median", "wmean" (weighted mean) or "wmedian" (weighted median). Defaults to wmean.
- `returnBigXimp`  
If TRUE, the imputed data matrix `Ximp` in the output will include the rows and columns that were not part of the analysis (and can still contain NAs). Defaults to FALSE.
- `silent`  
If TRUE, statements tracking the algorithm's progress will not be printed. Defaults to FALSE.
- `nLocScale`  
When estimating location or scale from more than `nLocScale` data values, the computation is based on a random sample of size `nLocScale` to save time. When `nLocScale = 0` all values are used. Defaults to 25000.
- `fastDDC`  
Whether to use the `fastDDC` option or not. The `fastDDC` algorithm uses approximations to allow to deal with high dimensions. Defaults to TRUE for  $d > 750$  and FALSE otherwise.
- `standType`  
The location and scale estimators used for robust standardization. Should be one of "1stepM", "mcd" or "wrap". See [estLocScale](#) for more info. Only used when `fastDDC = FALSE`. Defaults to "1stepM".
- `corrType`  
The correlation estimator used to find the neighboring variables. Must be one of "wrap" (wrapping correlation), "rank" (Spearman correlation) or "gkwlS" (Gnanadesikan-Kettenring correlation followed by weighting). Only used when `fastDDC = FALSE`. Defaults to "gkwlS".
- `transFun`  
The transformation function used to compute the robust correlations when `fastDDC = TRUE`. Can be "wrap" or "rank". Defaults to "wrap".



- `nbngbrs`  
When `fastDDC = TRUE`, each column is predicted from at most `nbngbrs` columns correlated to it. Defaults to 100.

**Value**

A list with components:

- `DDCpars`  
The list of options used.
- `colInAnalysis`  
The column indices of the columns used in the analysis.
- `rowInAnalysis`  
The row indices of the rows used in the analysis.
- `namesNotNumeric`  
The names of the variables which are not numeric.
- `namesCaseNumber`  
The name of the variable(s) which contained the case numbers and was therefore removed.
- `namesNAcol`  
Names of the columns left out due to too many NA's.
- `namesNArow`  
Names of the rows left out due to too many NA's.
- `namesDiscrete`  
Names of the discrete variables.
- `namesZeroScale`  
Names of the variables with zero scale.
- `remX`  
Cleaned data after `checkDataSet`.
- `locX`  
Estimated location of  $X$ .
- `scaleX`  
Estimated scales of  $X$ .
- `Z`  
Standardized `remX`.
- `nbngbrs`  
Number of neighbors used in estimation.
- `ngbrs`  
Indicates neighbors of each column, i.e. the columns most correlated with it.
- `robcors`  
Robust correlations.
- `robslopes`  
Robust slopes.

- `deshrinkage`  
The deshrinkage factor used for every connected (i.e. non-standalone) column of  $X$ .
- `Xest`  
Predicted  $X$ .
- `scalestres`  
Scale estimate of the residuals  $X - X_{est}$ .
- `stdResid`  
Residuals of original  $X$  minus the estimated  $X_{est}$ , standardized by column.
- `indcells`  
Indices of the cells which were flagged in the analysis.
- `Ti`  
Outlyingness (test) value of each row.
- `medTi`  
Median of the  $T_i$  values.
- `madTi`  
Mad of the  $T_i$  values.
- `indrows`  
Indices of the rows which were flagged in the analysis.
- `indNAs`  
Indices of all NA cells.
- `indall`  
Indices of all cells which were flagged in the analysis plus all cells in flagged rows plus the indices of the NA cells.
- `Ximp`  
Imputed  $X$ .

**Author(s)**

Raymaekers J., Rousseeuw P.J., Van den Bossche W.

**References**

Rousseeuw, P.J., Van den Bossche W. (2018). Detecting Deviating Data Cells. *Technometrics*, **60**(2), 135-145. ([link to open access pdf](#))

Raymaekers, J., Rousseeuw P.J. (2019). Fast robust correlation for high dimensional data. *Technometrics*, published online. ([link to open access pdf](#))

**See Also**

[checkDataSet, cellMap](#)

**Examples**

```

library(MASS); set.seed(12345)
n <- 50; d <- 20
A <- matrix(0.9, d, d); diag(A) = 1
x <- mvrnorm(n, rep(0,d), A)
x[sample(1:(n * d), 50, FALSE)] <- NA
x[sample(1:(n * d), 50, FALSE)] <- 10
x[sample(1:(n * d), 50, FALSE)] <- -10
x <- cbind(1:n, x)
DDCx <- DDC(x)
cellMap(DDCx$remX, DDCx$stdResid,
columnlabels = 1:d, rowlabels = 1:n)

# For more examples, we refer to the vignette:
vignette("DDC_examples")

```

DDCpredict

*DDCpredict***Description**

Based on a [DDC](#) fit on an initial (training) data set  $X$ , this function analyzes a new (test) data set  $X_{\text{new}}$ .

**Usage**

```
DDCpredict(Xnew, InitialDDC, DDCpars = NULL)
```

**Arguments**

<code>Xnew</code>	The new data (test data), which must be a matrix or a data frame. It must always be provided.
<code>InitialDDC</code>	The output of the <a href="#">DDC</a> function on the initial (training) dataset. Must be provided.
<code>DDCpars</code>	The input options to be used for the prediction. By default the options of <code>InitialDDC</code> are used.

**Value**

A list with components:

<code>DDCpars</code>	the options used in the call, see <a href="#">DDC</a> .
<code>locX</code>	the locations of the columns, from <code>InitialDDC</code> .
<code>scaleX</code>	the scales of the columns, from <code>InitialDDC</code> .
<code>Z</code>	$X_{\text{new}}$ standardized by <code>locX</code> and <code>scaleX</code> .
<code>nbngbrs</code>	predictions use a combination of <code>nbngbrs</code> columns.
<code>ngbrs</code>	for each column, the list of its neighbors, from <code>InitialDDC</code> .

robcors	for each column, the correlations with its neighbors, from InitialDDC.
robslopes	slopes to predict each column by its neighbors, from InitialDDC.
deshrinkage	for each connected column, its deshrinkage factor used in InitialDDC.
Xest	predicted values for every cell of Xnew.
scalestres	scale estimate of the residuals (Xnew - Xest), from InitialDDC.
stdResid	columnwise standardized residuals of Xnew.
indcells	positions of cellwise outliers in Xnew.
Ti	outlyingness of rows in Xnew.
medTi	median of the Ti in InitialDDC.
madTi	mad of the Ti in InitialDDC.
indrows	row numbers of the outlying rows in Xnew.
indNAs	positions of the NA's in Xnew.
indall	positions of NA's and outlying cells in Xnew.
Ximp	Xnew where all cells in indall are imputed by their prediction.

### Author(s)

Rousseeuw P.J., Van den Bossche W.

### References

Hubert, M., Rousseeuw, P.J., Van den Bossche W. (2019). MacroPCA: An all-in-one PCA method allowing for missing values as well as cellwise and rowwise outliers. *Technometrics*, **61**(4), 459-473. ([link to open access pdf](#))

### See Also

[checkDataSet](#), [cellMap](#), [DDC](#)

### Examples

```
library(MASS)
set.seed(12345)
n <- 100; d <- 10
A <- matrix(0.9, d, d); diag(A) = 1
x <- mvrnorm(n, rep(0,d), A)
x[sample(1:(n * d), 50, FALSE)] <- NA
x[sample(1:(n * d), 50, FALSE)] <- 10
x <- cbind(1:n, x)
DDCx <- DDC(x)
xnew <- mvrnorm(50, rep(0,d), A)
xnew[sample(1:(50 * d), 50, FALSE)] <- 10
predict.out <- DDCpredict(xnew, DDCx)
cellMap(xnew, predict.out$stdResid,
columnlabels = 1:d, rowlabels = 1:50)

# For more examples, we refer to the vignette:
vignette("DDC_examples")
```

**Description**

The Detection-Imputation algorithm computes cellwise robust estimates of the center and covariance matrix of a data set  $X$ . The algorithm alternates between the detection of cellwise outliers and their imputation combined with re-estimation of the center and covariance matrix. By default, it starts by calling `checkDataSet` to clean the data.

**Usage**

```
DI(X, initEst = "DDCWcov", crit = 0.01, maxits = 10, quant = 0.99,
maxCol = 0.25, checkPars = list())
```

**Arguments**

<code>X</code>	<code>X</code> is the input data, and must be an $n$ by $d$ matrix or a data frame.
<code>initEst</code>	An initial estimator for the center and covariance matrix. Should be one of "DDCWcov" or "TSGS", where the latter refers to the function <code>TSGS</code> . The default option "DDCWcov" uses the proposal of Raymaekers and Rousseeuw (2020) which is much faster for increasing dimension.
<code>crit</code>	The algorithm converges when the subsequent estimates of the center and covariance matrix do not differ more than <code>crit</code> in squared Euclidean norm.
<code>maxits</code>	Maximum number of DI-iterations.
<code>quant</code>	The cutoff used to detect cellwise outliers.
<code>maxCol</code>	The maximum number of cellwise outliers allowed in a column.
<code>checkPars</code>	Optional list of parameters used in the call to <code>checkDataSet</code> . The options are: <ul style="list-style-type: none"> <li>• <code>coreOnly</code> If TRUE, skip the execution of <code>checkDataset</code>. Defaults to FALSE</li> <li>• <code>numDiscrete</code> A column that takes on <code>numDiscrete</code> or fewer values will be considered discrete and not retained in the cleaned data. Defaults to 5.</li> <li>• <code>fracNA</code> Only retain columns and rows with fewer NAs than this fraction. Defaults to 0.15.</li> <li>• <code>precScale</code> Only consider columns whose scale is larger than <code>precScale</code>. Here scale is measured by the median absolute deviation. Defaults to <math>1e - 12</math>.</li> <li>• <code>silent</code> Whether or not the function progress messages should be printed. Defaults to FALSE.</li> </ul>

**Value**

A list with components:

- `center`  
The final estimate of the center of the data.
- `cov`  
The final estimate of the covariance matrix.
- `nbits`  
Number of DI-iterations executed to reach convergence.
- `Ximp`  
The imputed data.
- `indcells`  
Indices of the cells which were flagged in the analysis.
- `indNAs`  
Indices of the NAs in the data.
- `Zres`  
Matrix with standardized cellwise residuals of the flagged cells. Contains zeroes in the unflagged cells.
- `Zres_denom`  
Denominator of the standardized cellwise residuals.
- `cellPaths`  
Matrix with the same dimensions as `X`, in which each row contains the path of least angle regression through the cells of that row, i.e. the order of the coordinates in the path (1=first, 2=second,...)
- `checkDataSet_out`  
Output of the call to `checkDataSet` which is used to clean the data.

**Author(s)**

J. Raymaekers and P.J. Rousseeuw

**References**

J. Raymaekers and P.J. Rousseeuw (2020). Handling cellwise outliers by sparse regression and robust covariance. *Arxiv: 1912.12446*. ([link to open access pdf](#))

**See Also**

[cellHandler](#)

**Examples**

```
mu <- rep(0, 3)
Sigma <- diag(3) * 0.1 + 0.9
X <- MASS::mvrnorm(100, mu, Sigma)
DI.out <- DI(X)
DI.out$cov
```

---

dog\_walker

*Dog walker dataset*

---

**Description**

A dataset containing the image sequence of a video. The sequence consists of 54 frames of 144 by 180 pixels pixels in Red/Geen/Blue (RGB) format.

**Usage**

```
data("dog_walker")
```

**Format**

An array of dimensions  $54 \times 144 \times 180 \times 3$ .

**Source**

<http://www.wisdom.weizmann.ac.il/~vision/SpaceTimeActions.html>

**Examples**

```
data(dog_walker)
# For more examples, we refer to the vignette:
vignette("Wrap_examples")
```

---

dposs

*DPOSS dataset*

---

**Description**

This is a random subset of 20'000 stars from the Digitized Palomar Sky Survey (DPOSS) described by Odewahn et al. (1998).

**Usage**

```
data("dposs")
```

**Format**

A matrix of dimensions  $20000 \times 21$ .

**References**

Odewahn, S., S. Djorgovski, R. Brunner, and R. Gal (1998). Data From the Digitized Palomar Sky Survey. Technical report, California Institute of Technology.

**Examples**

```
data(dposs)
# For more examples, we refer to the vignette:
vignette("MacroPCA_examples")
```

---

estLocScale

*Estimate robust location and scale*


---

**Description**

Estimate a robust location estimate and scale estimate of every column in  $X$ .

**Usage**

```
estLocScale(X, type = "wrap", precScale = 1e-12,
center = TRUE, alpha = 0.5, nLocScale = 25000, silent = FALSE)
```

**Arguments**

$X$	The input data. It must be an $n$ by $d$ matrix or a data frame.
type	The type of estimators used. One of: <ul style="list-style-type: none"> <li>• "1stepM": The location is the 1-step M-estimator with the biweight psi function. The scale estimator is the 1-step M-estimator using a Huber rho function with <math>b = 2.5</math>.</li> <li>• "mcd": the location is the weighted univariate MCD estimator with cutoff <math>\sqrt{(qchisq(0.975, 1))}</math>. The scale is the corresponding weighted univariate MCD estimator, with a correction factor to make it approximately unbiased at gaussian data.</li> <li>• "wrap": Starting from the initial estimates corresponding to option "mcd", the location is the 1-step M-estimator with the wrapping psi function with <math>b = 1.5</math> and <math>c = 4</math>. The scale estimator is the same as in option "mcd".</li> </ul> Defaults to "wrap".
precScale	The precision scale used throughout the algorithm. Defaults to $1e - 12$ .
center	Whether or not the data has to be centered before calculating the scale. Not in use for type = "mcd". Defaults to TRUE.
alpha	The value of $\alpha$ in the univariate mcd, must be between 0.5 and 1. The subsetsize is $h = \lceil \alpha n \rceil$ . Only used for type = "mcd". Defaults to $\alpha = 0.5$ .
nLocScale	If $nLocScale < n$ , nLocScale observations are sampled to compute the location and scale. This speeds up the computation if $n$ is very large. When $nLocScale = 0$ all observations are used. Defaults to $nLocScale = 25000$ .
silent	Whether or not a warning message should be printed when very small scales are found. Defaults to FALSE.



**Value**

A list with components:

- `loc`  
A vector with the estimated locations.
- `scale`  
A vector with the estimated scales.

**Author(s)**

Raymaekers, J. and Rousseeuw P.J.

**References**

Raymaekers, J., Rousseeuw P.J. (2019). Fast robust correlation for high dimensional data. *Technometrics*, published online. ([link to open access pdf](#))

**See Also**

[wrap](#)

**Examples**

```
library(MASS)
set.seed(12345)
n = 100; d = 10
X = mvrnorm(n, rep(0, 10), diag(10))
locScale = estLocScale(X)
# For more examples, we refer to the vignette:
vignette("wrap_examples")
```

---

glass

*The glass dataset*

---

**Description**

A dataset containing spectra with  $d = 750$  wavelengths collected on  $n = 180$  archeological glass samples.

**Usage**

```
data("glass")
```

**Format**

A data frame with 180 observations of 750 wavelengths.

**Source**

Lemberge, P., De Raedt, I., Janssens, K.H., Wei, F., and Van Espen, P.J. (2000). Quantitative Z-analysis of 16th-17th century archaeological glass vessels using PLS regression of EPXMA and  $\mu$ -XRF data. *Journal of Chemometrics*, **14**, 751–763.

**Examples**

```
data(glass)
```

---

 ICPCA

---

*Iterative Classical PCA*


---

**Description**

This function carries out classical PCA when the data may contain missing values, by an iterative algorithm. It is based on a Matlab function from the Missing Data Imputation Toolbox v1.0 by A. Folch-Fortuny, F. Arteaga and A. Ferrer.

**Usage**

```
ICPCA(X, k, scale = FALSE, maxiter = 20, tol = 0.005,  
      tolProb = 0.99, distprob = 0.99)
```

**Arguments**

X	the input data, which must be a matrix or a data frame. It may contain NA's. It must always be provided.
k	the desired number of principal components
scale	a value indicating whether and how the original variables should be scaled. If scale=FALSE (default) or scale=NULL no scaling is performed (and a vector of 1s is returned in the \$scaleX slot). If scale=TRUE the variables are scaled to have a standard deviation of 1. Alternatively scale can be a function like mad, or a vector of length equal to the number of columns of x. The resulting scale estimates are returned in the \$scaleX slot of the output.
maxiter	maximum number of iterations. Default is 20.
tol	tolerance for iterations. Default is 0.005.
tolProb	tolerance probability for residuals. Defaults to 0.99.
distprob	probability determining the cutoff values for orthogonal and score distances. Default is 0.99.

**Value**

A list with components:

scaleX	the scales of the columns of X.
k	the number of principal components.
loadings	the columns are the k loading vectors.
eigenvalues	the k eigenvalues.
center	vector with the fitted center.
covmatrix	estimated covariance matrix.
It	number of iteration steps.
diff	convergence criterion.
X.NAimp	data with all NA's imputed.
scores	scores of X.NAimp.
OD	orthogonal distances of the rows of X.NAimp.
cutoffOD	cutoff value for the OD.
SD	score distances of the rows of X.NAimp.
cutoffSD	cutoff value for the SD.
indrows	row numbers of rowwise outliers.
residScale	scale of the residuals.
stdResid	standardized residuals. Note that these are NA for all missing values of X.
indcells	indices of cellwise outliers.

**Author(s)**

Wannes Van Den Bossche

**References**

Folch-Fortuny, A., Arteaga, F., Ferrer, A. (2016). Missing Data Imputation Toolbox for MATLAB. *Chemometrics and Intelligent Laboratory Systems*, **154**, 93-100.

**Examples**

```
library(MASS)
set.seed(12345)
n <- 100; d <- 10
A <- diag(d) * 0.1 + 0.9
x <- mvrnorm(n, rep(0,d), A)
x[sample(1:(n * d), 100, FALSE)] <- NA
ICPCA.out <- ICPCA(x, k = 2)
plot(ICPCA.out$scores)
```

MacroPCA

*MacroPCA***Description**

This function performs the MacroPCA algorithm, which can deal with Missing values and Cellwise and Rowwise Outliers. Note that this function first calls `checkDataSet` and analyzes the remaining cleaned data.

**Usage**

```
MacroPCA(X, k = 0, MacroPCApars = NULL)
```

**Arguments**

- |              |  |
|--------------|--|
| X            | X is the input data, and must be an $n$ by $d$ matrix or a data frame.   |
| k            | k is the desired number of principal components. If $k = 0$ or $k = \text{NULL}$ , the algorithm will compute the percentage of explained variability for $k$ upto $k_{\text{max}}$ and show a scree plot, and suggest to choose a value of $k$ such that the cumulative percentage of explained variability is at least 80 %.   |
| MacroPCApars | <p>A list of available options detailed below. If <code>MacroPCApars = NULL</code> the defaults below are used.</p> <ul style="list-style-type: none"> <li>• <code>DDCpars</code><br/>A list with parameters for the first step of the MacroPCA algorithm (for the complete list see the function <code>DDC</code>). Default is <code>NULL</code>.</li> <li>• <code>kmax</code><br/>The maximal number of principal components to compute. Default is <code>kmax = 10</code>. If <math>k</math> is provided <code>kmax</code> does not need to be specified, unless <math>k</math> is larger than 10 in which case you need to set <code>kmax</code> high enough.</li> <li>• <code>alpha</code><br/>This is the coverage, i.e. the fraction of rows the algorithm should give full weight. Alpha should be between 0.50 and 1, the default is 0.50.</li> <li>• <code>scale</code><br/>A value indicating whether and how the original variables should be scaled. If <code>scale = FALSE</code> or <code>scale = NULL</code> no scaling is performed (and a vector of 1s is returned in the <code>\$scaleX</code> slot). If <code>scale = TRUE</code> (default) the data are scaled by a 1-step M-estimator of scale with the Tukey biweight weight function to have a robust scale of 1. Alternatively <code>scale</code> can be a vector of length equal to the number of columns of <math>x</math>. The resulting scale estimates are returned in the <code>\$scaleX</code> slot of the MacroPCA output.</li> <li>• <code>maxdir</code><br/>The maximal number of random directions to use for computing the outlyingness of the data points. Default is <code>maxdir = 250</code>. If the number <math>n</math> of observations is small all <math>n * (n - 1) / 2</math> pairs of observations are used.</li> </ul> |

- `distprob`  
The quantile determining the cutoff values for orthogonal and score distances. Default is 0.99.
- `silent`  
If TRUE, statements tracking the algorithm's progress will not be printed. Defaults to FALSE.
- `maxiter`  
Maximum number of iterations. Default is 20.
- `tol`  
Tolerance for iterations. Default is 0.005.
- `bigOutput`  
whether to compute and return NAimp, Cellimp and Fullimp. Defaults to TRUE.

### Value

A list with components:

<code>MacroPCApars</code>	the options used in the call.
<code>remX</code>	Cleaned data after <code>checkDataSet</code> .
<code>DDC</code>	results of the first step of MacroPCA. These are needed to run <code>MacroPCApredict</code> on new data.
<code>scaleX</code>	the scales of the columns of X.
<code>k</code>	the number of principal components.
<code>loadings</code>	the columns are the k loading vectors.
<code>eigenvalues</code>	the k eigenvalues.
<code>center</code>	vector with the fitted center.
<code>alpha</code>	alpha from the input.
<code>h</code>	h (computed from alpha).
<code>It</code>	number of iteration steps.
<code>diff</code>	convergence criterion.
<code>X.NAimp</code>	data with all NA's imputed by MacroPCA.
<code>scores</code>	scores of X.NAimp.
<code>OD</code>	orthogonal distances of the rows of X.NAimp.
<code>cutoffOD</code>	cutoff value for the OD.
<code>SD</code>	score distances of the rows of X.NAimp.
<code>cutoffSD</code>	cutoff value for the SD.
<code>indrows</code>	row numbers of rowwise outliers.
<code>residScale</code>	scale of the residuals.
<code>stdResid</code>	standardized residuals. Note that these are NA for all missing values of X.
<code>indcells</code>	indices of cellwise outliers.
<code>NAimp</code>	various results for the NA-imputed data.
<code>Cellimp</code>	various results for the cell-imputed data.
<code>Fullimp</code>	various result for the fully imputed data.

**Author(s)**

Rousseeuw P.J., Van den Bossche W.

**References**

Hubert, M., Rousseeuw, P.J., Van den Bossche W. (2019). MacroPCA: An all-in-one PCA method allowing for missing values as well as cellwise and rowwise outliers. *Technometrics*, **61**(4), 459-473. ([link to open access pdf](#))

**See Also**

[checkDataSet](#), [cellMap](#), [DDC](#)

**Examples**

```
library(MASS)
set.seed(12345)
n <- 50; d <- 10
A <- matrix(0.9, d, d); diag(A) = 1
x <- mvrnorm(n, rep(0,d), A)
x[sample(1:(n * d), 50, FALSE)] <- NA
x[sample(1:(n * d), 50, FALSE)] <- 10
x <- cbind(1:n, x)
MacroPCA.out <- MacroPCA(x, 2)
cellMap(MacroPCA.out$remX, MacroPCA.out$stdResid,
columnlabels = 1:d, rowlabels = 1:n)
# For more examples, we refer to the vignette:
vignette("MacroPCA_examples")
```

---

MacroPCApredict

*MacroPCApredict*

---

**Description**

Based on a [MacroPCA](#) fit of an initial (training) data set X, this function analyzes a new (test) data set Xnew.

**Usage**

```
MacroPCApredict(Xnew, InitialMacroPCA, MacroPCApars = NULL)
```

**Arguments**

Xnew	The new data (test data), which must be a matrix or a data frame. It must always be provided.
InitialMacroPCA	The output of the MacroPCA function on the initial (training) dataset. Must be provided.
MacroPCApars	The input options to be used for the prediction. By default the options of Initial-MacroPCA are used. For the complete list of options see the function <a href="#">MacroPCA</a> .

**Value**

A list with components:

MacroPCApars	the options used in the call.
scaleX	the scales of the columns of $X$ .
k	the number of principal components.
loadings	the columns are the $k$ loading vectors.
eigenvalues	the $k$ eigenvalues.
center	vector with the fitted center.
It	number of iteration steps.
diff	convergence criterion.
X.NAimp	$X_{new}$ with all NA's imputed by MacroPCA.
scores	scores of $X.NAimp$ .
OD	orthogonal distances of the rows of $X.NAimp$ .
cutoffOD	cutoff value for the OD.
SD	score distances of the rows of $X.NAimp$ .
cutoffSD	cutoff value for the SD.
indrows	row numbers of rowwise outliers.
residScale	scale of the residuals.
stdResid	standardized residuals. Note that these are NA for all missing values of $X_{new}$ .
indcells	indices of cellwise outliers.
NAimp	various results for the NA-imputed data.
Cellimp	various results for the cell-imputed data.
Fullimp	various result for the fully imputed data.
DDC	result of DDCpredict which is the first step of MacroPCApredict. See the function <a href="#">DDCpredict</a> .

**Author(s)**

Rousseeuw P.J., Van den Bossche W.

**References**

Hubert, M., Rousseeuw, P.J., Van den Bossche W. (2019). MacroPCA: An all-in-one PCA method allowing for missing values as well as cellwise and rowwise outliers. *Technometrics*, **61**(4), 459-473. ([link to open access pdf](#))

**See Also**

[checkDataSet](#), [cellMap](#), [DDC](#), [DDCpredict](#), [MacroPCA](#)

### Examples

```
library(MASS)
set.seed(12345)
n <- 50; d <- 10
A <- matrix(0.9, d, d); diag(A) = 1
x <- mvrnorm(n, rep(0,d), A)
x[sample(1:(n * d), 50, FALSE)] <- NA
x[sample(1:(n * d), 50, FALSE)] <- 10
x <- cbind(1:n, x)
MacroPCA.out <- MacroPCA(x, 2)
xnew <- mvrnorm(n, rep(0,d), A)
xnew[sample(1:(n * d), 50, FALSE)] <- 10
predict.out <- MacroPCApredict(xnew, MacroPCA.out)
cellMap(xnew, predict.out$stdResid,
columnlabels = 1:d, rowlabels = 1:n)
```

---

mortality

*The mortality dataset*

---

### Description

This dataset contains the mortality by age for males in France, from 1816 to 2013 as obtained from the Human Mortality Database.

### Usage

```
data("mortality")
```

### Format

A data frame with 198 calendar years (rows) and 91 age brackets (columns).

### Source

Human Mortality Database. University of California, Berkeley (USA), and Max Planck Institute for Demographic Research (Germany). Available at <https://www.mortality.org> (data downloaded in November 2015).

### References

Hyndman, R.J., and Shang, H.L. (2010), Rainbow plots, bagplots, and boxplots for functional data, *Journal of Computational and Graphical Statistics*, **19**, 29–45.

### Examples

```
data(mortality)
```



---

outlierMap	<i>Plot the outlier map.</i>
------------	------------------------------

---

### Description

The outlier map is a diagnostic plot for the output of [MacroPCA](#).

### Usage

```
outlierMap(res,title="Robust PCA",col="black", pch=16,labelOut=TRUE,id=3,
xlim = NULL, ylim = NULL, cex = 1, cex.main=1.2, cex.lab=NULL, cex.axis=NULL)
```

### Arguments

res	A list containing the orthogonal distances (OD), the score distances (SD) and their respective cut-offs (cutoffOD and cutoffSD). Can be the output of <a href="#">MacroPCA</a> , <a href="#">robpca</a> , <a href="#">rospca</a> .
title	Title of the plot, default is "Robust PCA".
col	Colour of the points in the plot, this can be a single colour for all points or a vector or list specifying the colour for each point. The default is "black".
pch	Plotting characters or symbol used in the plot, see points for more details. The default is 16 which corresponds to filled circles.
labelOut	Logical indicating if outliers should be labelled on the plot, default is TRUE.
id	Number of OD outliers and number of SD outliers to label on the plot, default is 3.
xlim	Optional argument to set the limits of the x-axis.
ylim	Optional argument to set the limits of the y-axis.
cex	Optional argument determining the size of the plotted points. See <a href="#">plot.default</a> for details.
cex.main	Optional argument determining the size of the main title. See <a href="#">plot.default</a> for details.
cex.lab	Optional argument determining the size of the labels. See <a href="#">plot.default</a> for details.
cex.axis	Optional argument determining the size of the axes. See <a href="#">plot.default</a> for details.

### Details

The outlier map contains the score distances on the x-axis and the orthogonal distances on the y-axis. To detect outliers, cut-offs for both distances are shown, see Hubert et al. (2005).

### Author(s)

P.J. Rousseeuw

**References**

Hubert, M., Rousseeuw, P. J., and Vanden Branden, K. (2005). ROBPCA: A New Approach to Robust Principal Component Analysis. *Technometrics*, **47**, 64-79.

**See Also**

[MacroPCA](#)

**Examples**

```
# empty for now
```

---

philips

*The philips dataset*

---

**Description**

A dataset containing measurements of  $d = 9$  characteristics of  $n = 677$  diaphragm parts, used in the production of TV sets.

**Usage**

```
data("philips")
```

**Format**

A matrix with 677 rows and 9 columns.

**Source**

The data were provided in 1997 by Gertjan Otten and permission to analyze them was given by Herman Veraa and Frans Van Dommelen at Philips Mecoma in The Netherlands.

**References**

Rousseeuw, P.J., and Van Driessen, K. (1999). A fast algorithm for the Minimum Covariance Determinant estimator. *Technometrics*, **41**, 212-223.

**Examples**

```
data(philips)
```

---

transfo	<i>Robustly fit the Box-Cox or Yeo-Johnson transformation</i>
---------	---

---

### Description

This function uses reweighted maximum likelihood to robustly fit the Box-Cox or Yeo-Johnson transformation to each variable in a dataset. Note that this function first calls [checkDataSet](#) to ensure that the variables to be transformed are not too discrete.

### Usage

```
transfo(X, type = "YJ", robust = TRUE, lambdarange = NULL,
        prestandardize = TRUE, prescaleBC = F, scalefac = 1,
        quant = 0.99, nbsteps = 2, checkPars = list())
```

### Arguments

X	A data matrix of dimensions n x d. Its columns are the variables to be transformed.
type	The type of transformation to be fit. Should be one of: <ul style="list-style-type: none"> <li>• "BC": Box-Cox power transformation. Only works for strictly positive variables. If this type is given but a variable is not strictly positive, the function stops with a message about that variable.</li> <li>• "YJ" Yeo-Johnson power transformation. The data may have positive as well as negative values.</li> <li>• "bestObj" for strictly positive variables both BC and YJ are run, and the solution with lowest objective is kept. On the other variables YJ is run.</li> </ul>
robust	if TRUE the Reweighted Maximum Likelihood method is used, which first computes a robust initial estimate of the transformation parameter lambda. If FALSE the classical ML method is used.
lambdarange	range of lambda values that will be optimized over. If NULL, the range goes from -4 to 6.
prestandardize	whether to standardize the variables <b>before</b> the power transformation. For BC the variable is divided by its median. For YJ and robust = TRUE this subtracts its median and divides by its mad (median absolute deviation). For YJ and robust = F this subtracts the mean and divides by the standard deviation.
prescaleBC	for BC only. This standardizes the logarithm of the original variable by subtracting its median and dividing by its mad, after which the exponential function turns the result into a positive variable again.
scalefac	when YJ is fit and prestandardize = TRUE, the standardized data is multiplied by scalefac. When BC is fit and prescaleBC = TRUE the same happens to the standardized log of the original variable.
quant	quantile for determining the weights in the reweighting step (ignored when robust=FALSE).

nbsteps	number of reweighting steps (ignored when robust=FALSE).
checkPars	Optional list of parameters used in the call to <code>checkDataSet</code> . The options are: <ul style="list-style-type: none"> <li>• <code>coreOnly</code> If TRUE, skip the execution of <code>checkDataset</code>. Defaults to FALSE</li> <li>• <code>numDiscrete</code> A column that takes on <code>numDiscrete</code> or fewer values will be considered discrete and not retained in the cleaned data. Defaults to 5.</li> <li>• <code>precScale</code> Only consider columns whose scale is larger than <code>precScale</code>. Here scale is measured by the median absolute deviation. Defaults to <math>1e - 12</math>.</li> <li>• <code>silent</code> Whether or not the function progress messages should be printed. Defaults to FALSE.</li> </ul>

### Value

A list with components:

- `lambdahats`  
the estimated transformation parameter for each column of X.
- `Xt`  
A matrix in which each column is the transformed version of the corresponding column of X.
- `muhat`  
The estimated location of each column of `Xt`.
- `sigmahat`  
The estimated scale of each column of `Xt`.
- `Zt`  
`Xt` poststandardized by the centers in `muhat` and the scales in `sigmahat`. Is always provided.
- `weights`  
The final weights from the reweighting.
- `ttypes`  
The type of transform used in each column.
- `objective`  
Value of the (reweighted) maximum likelihood objective function.

### Author(s)

J. Raymaekers and P.J. Rousseeuw

### References

J. Raymaekers and P.J. Rousseeuw (2020). Transforming variables to central normality. *Arxiv: 2005.07946*. ([link to open access pdf](#))

## Examples

```
# find Box-Cox transformation parameter for lognormal data:
set.seed(123)
x <- exp(rnorm(1000))
transfo.out <- transfo(x, type = "BC")
# estimated parameter:
transfo.out$lambdahat
# value of the objective function:
transfo.out$objective
# the transformed variable:
transfo.out$Xt
# the poststandardized transformed variable:
transfo.out$Zt
# the type of transformation used:
transfo.out$types
# qqplot of the poststandardized transformed variable:
qqnorm(transfo.out$Zt); abline(0,1)

# For more examples, we refer to the vignette:
vignette("transfo_examples")
```

---

truncPC

*Classical Principal Components by truncated SVD.*


---

## Description

Similar usage to `classPC` of `robustbase` except for the new argument `ncomp` which is the desired number of components. Only this many PC's are computed in order to save computation time. Makes use of `propack.svd` of package `svd`.

## Usage

```
truncPC(X, ncomp = NULL, scale = FALSE, center = TRUE,
        signflip = TRUE, via.svd = NULL, scores = FALSE)
```

## Arguments

<code>X</code>	a numeric matrix.
<code>ncomp</code>	the desired number of components (if not specified, all components are computed).
<code>scale</code>	logical, or numeric vector for scaling the columns.
<code>center</code>	logical or numeric vector for centering the matrix.
<code>signflip</code>	logical indicating if the signs of the loadings should be flipped such that the absolutely largest value is always positive.
<code>via.svd</code>	dummy argument for compatibility with <code>classPC</code> calls, will be ignored.
<code>scores</code>	logical indicating whether or not scores should be returned.

**Value**

A list with components:

rank	the (numerical) matrix rank of $X$ , i.e. an integer number between 0 and $\min(\dim(x))$ .
eigenvalues	the $k$ eigenvalues, proportional to the variances, where $k$ is the rank above.
loadings	the loadings, a $d \times k$ matrix.
scores	if the scores argument was TRUE, the $n \times k$ matrix of scores.
center	a vector of means, unless the center argument was FALSE.
scale	a vector of column scales, unless the scale argument was false.

**Author(s)**

P.J. Rousseeuw

**See Also**

[classPC](#)

**Examples**

```
library(MASS)
set.seed(12345)
n <- 100; d <- 10
A <- diag(d) * 0.1 + 0.9
x <- mvrnorm(n, rep(0,d), A)
truncPCA.out <- truncPC(x, ncomp = 2, scores = TRUE)
plot(truncPCA.out$scores)
```

---

wrap

*Wrap the data.*

---

**Description**

Transforms multivariate data  $X$  using the wrapping function with  $b = 1.5$  and  $c = 4$ . By default, it starts by calling [checkDataSet](#) to clean the data and [estLocScale](#) to estimate the location and scale of the variables in the cleaned data. Alternatively, it works with user-provided vectors of location and scale given by `locX` and `scaleX`.

**Usage**

```
wrap(X, locX = NULL, scaleX = NULL, precScale = 1e-12,
     imputeNA = TRUE, checkPars = list())
```

**Arguments**

<code>X</code>	the input data. It must be an $n$ by $d$ matrix or a data frame.
<code>locX</code>	The location estimates of the columns of the input data <code>X</code> . Must be a vector of length $d$ .
<code>scaleX</code>	The scale estimates of the columns of the input data <code>X</code> . Must be a vector of length $d$ .
<code>precScale</code>	The precision scale used throughout the algorithm. Defaults to $1e - 12$
<code>imputeNA</code>	Whether or not to impute the NAs with the location estimate of the corresponding variable. Defaults to TRUE.
<code>checkPars</code>	Optional list of parameters used in the call to <code>checkDataSet</code> . The options are: <ul style="list-style-type: none"><li>• <code>coreOnly</code> If TRUE, skip the execution of <code>checkDataset</code>. Defaults to FALSE</li><li>• <code>numDiscrete</code> A column that takes on <code>numDiscrete</code> or fewer values will be considered discrete and not retained in the cleaned data. Defaults to 5.</li><li>• <code>precScale</code> Only consider columns whose scale is larger than <code>precScale</code>. Here scale is measured by the median absolute deviation. Defaults to <math>1e - 12</math>.</li><li>• <code>silent</code> Whether or not the function progress messages should be printed. Defaults to FALSE.</li></ul>

**Value**

A list with components:

- `Xw`  
The wrapped data.
- `colInWrap`  
The column numbers of the variables which were wrapped. Variables which were filtered out by `checkDataSet` (because of a (near) zero scale for example), will not appear in this output.
- `loc`  
The location estimates for all variables used for wrapping.
- `scale`  
The scale estimates for all variables used for wrapping.

**Author(s)**

Raymaekers, J. and Rousseeuw P.J.

**References**

Raymaekers, J., Rousseeuw P.J. (2019). Fast robust correlation for high dimensional data. *Technometrics*, published online. ([link to open access pdf](#))

**See Also**[estLocScale](#)**Examples**

```
library(MASS)
set.seed(12345)
n <- 100; d <- 10
X <- mvrnorm(n, rep(0, 10), diag(10))
locScale <- estLocScale(X)
Xw <- wrap(X, locScale$loc, locScale$scale)$Xw
# For more examples, we refer to the vignette:
vignette("wrap_examples")
```



# Index

cellHandler, [2](#), [14](#)  
cellMap, [4](#), [10](#), [12](#), [22](#), [23](#)  
checkDataSet, [5](#), [7](#), [10](#), [12–14](#), [20](#), [22](#), [23](#), [27](#),  
[28](#), [30](#), [31](#)  
classPC, [29](#), [30](#)  
  
DDC, [2](#), [5](#), [7](#), [7](#), [11](#), [12](#), [20](#), [22](#), [23](#)  
DDCpredict, [11](#), [23](#)  
DI, [2](#), [3](#), [13](#)  
dog\_walker, [15](#)  
dposs, [15](#)  
  
estLocScale, [8](#), [16](#), [30](#), [32](#)  
  
glass, [17](#)  
  
ICPCA, [18](#)  
  
MacroPCA, [5](#), [7](#), [20](#), [22](#), [23](#), [25](#), [26](#)  
MacroPCApredict, [22](#)  
mortality, [24](#)  
  
outlierMap, [25](#)  
  
philips, [26](#)  
plot.default, [25](#)  
  
robpca, [25](#)  
rospca, [25](#)  
  
transfo, [7](#), [27](#)  
truncPC, [29](#)  
TSGS, [13](#)  
  
wrap, [7](#), [17](#), [30](#)