

Package ‘miscTools’

January 20, 2026

Version 0.6-30
Date 2026-01-19
Title Miscellaneous Tools and Utilities
Author Arne Henningsen [aut, cre],
Ott Toomet [aut]
Maintainer Arne Henningsen <arne.henningsen@gmail.com>
Depends R (>= 2.14.0)
Imports digest
Suggests Ecdat (>= 0.1-5)
Description Miscellaneous small tools and utilities.
Many of them facilitate the work with matrices,
e.g. inserting rows or columns, creating symmetric matrices,
or checking for semidefiniteness.
Other tools facilitate the work with regression models,
e.g. extracting the standard errors,
obtaining the number of (estimated) parameters,
or calculating R-squared values.
License GPL (>= 2)
URL <https://github.com/arne-henningsen/miscTools>
BugReports <https://github.com/arne-henningsen/miscTools/issues>
NeedsCompilation no
Repository CRAN
Date/Publication 2026-01-20 06:10:14 UTC

Contents

coefTable	2
colMedians	3
compPlot	4
ddnorm	4

histDens	5
insertCol	6
insertRow	7
isSemidefinite	8
margEff	10
nObs	10
nParam	11
quasiconcavity	12
rowMedians	13
rSquared	14
stdEr	14
sumKeepAttr	15
summarizeDF	16
symMatrix	17
triang	18
vecli	18
vecli2m	19
veclipos	20
Index	21

coefTable	<i>Coefficient Table</i>
-----------	--------------------------

Description

Generate Table for Coefficients, Std. Errors, t-values and P-values.

Usage

```
coefTable( coef, stdErr, df = NULL )
```

Arguments

- coef vector that contains the coefficients.
- stdErr vector that contains the standard errors of the coefficients.
- df degrees of freedom of the t-test used to calculate P-values.

Value

a matrix with 4 columns: coefficients, standard errors, t-values and P-values. If argument df is not provided, the last column (P-values) is filled with NAs.

Author(s)

Arne Henningsen

Examples

```
coefTable( rnorm( 10 ), 0.5 * abs( rnorm( 10 ) ), 20 )
```

colMedians	<i>Medians of Columns</i>
------------	---------------------------

Description

Compute the sample medians of the columns (non-rows) of a data.frame or array.

Usage

```
colMedians( x, na.rm = FALSE )
```

Arguments

- x a data.frame or array.
- na.rm a logical value indicating whether NA values should be stripped before the computation proceeds.

Value

A vector or array of the medians of each column (non-row) of x with dimension `dim( x )[-1]`.

Author(s)

Arne Henningsen

See Also

[rowMedians](#), [median](#), [colMeans](#).

Examples

```
data( "Electricity", package = "Ecdat" )
colMedians( Electricity )

a4 <- array( 1:120, dim = c(5,4,3,2),
  dimnames = list( c("a","b","c","d","e"), c("A","B","C","D"),
    c("x","y","z"), c("Y","Z") ) )
colMedians( a4 )
median( a4[ , "B", "x", "Z" ] ) # equal to
colMedians( a4 )[ "B", "x", "Z" ]
```

compPlot

Scatterplot to Compare two Variables

Description

Plot a scatterplot to compare two variables.

Usage

```
compPlot( x, y, lim = NULL, ... )
```

Arguments

x	values of the first variable (on the X axis).
y	values of the second variable (on the Y axis).
lim	optional vector of two elements specifying the limits of both axes).
...	further arguments are passed to plot .

Author(s)

Arne Henningsen

Examples

```
set.seed( 123 )
x <- runif( 25 )
y <- 2 + 3 * x + rnorm( 25 )
ols <- lm( y ~ x )

compPlot( y, fitted( ols ) )
compPlot( y, fitted( ols ), lim = c( 0, 10 ) )
compPlot( y, fitted( ols ), pch = 20 )
compPlot( y, fitted( ols ), xlab = "observed", ylab = "fitted" )
compPlot( y, fitted( ols ), log = "xy" )
```

ddnorm

Derivative of the Normal Distribution's Density Function

Description

This function returns the derivative(s) of the density function of the normal (Gaussian) distribution with respect to the quantile, evaluated at the quantile(s), mean(s), and standard deviation(s) specified by arguments x, mean, and sd, respectively.

Usage

```
ddnorm( x, mean = 0, sd = 1 )
```

Arguments

x	quantile or vector of quantiles.
mean	mean or vector of means.
sd	standard deviation or vector of standard deviations.

Value

numeric value(s): derivative(s) of the density function of the normal distribution with respect to the quantile

Author(s)

Arne Henningsen

See Also

[dnorm](#)

Examples

```
ddnorm( c( -1, 0, 1 ) )
```

histDens	<i>Histogram with a Density Line</i>
----------	--------------------------------------

Description

Plot a histogram and add a kernel density line.

Usage

```
histDens( x, breaks = "Sturges", ... )
```

Arguments

x	values of the variable.
breaks	passed to hist .
...	further arguments are passed to hist .

Author(s)

Arne Henningsen

Examples

```
set.seed( 123 )  
x <- rnorm( 100 )  
histDens( x )  
histDens( x, 20 )  
histDens( x, 20, main = "My Title" )
```

insertCol*Insert Column into a Matrix*

Description

Insert a new column into a matrix.

Usage

```
insertCol( m, c, v = NA, cName = "" )
```

Arguments

m	matrix.
c	column number where the new column should be inserted.
v	optional values of the new column.
cName	optional character string: the name of the new column.

Value

a matrix with one more column than the provided matrix **m**.

Author(s)

Arne Henningsen

See Also

[insertRow](#).

Examples

```
m <- matrix( 1:4, 2 )  
insertCol( m, 2, 5:6 )
```

insertRow	<i>Insert Row into a Matrix</i>
-----------	---------------------------------

Description

Insert a new row into a matrix.

Usage

```
insertRow( m, r, v = NA, rName = "" )
```

Arguments

m	matrix.
r	row number where the new row should be inserted.
v	optional values for the new row.
rName	optional character string: the name of the new row.

Value

a matrix with one more row than the provided matrix m.

Author(s)

Arne Henningsen

See Also

[insertCol](#).

Examples

```
m <- matrix( 1:4, 2 )  
insertRow( m, 2, 5:6 )
```

isSemidefinite	<i>Positive or Negative Semidefiniteness</i>
----------------	--

Description

Check whether a symmetric matrix is positive or negative semidefinite.

Usage

```
isSemidefinite( m, ... )

## Default S3 method:
isSemidefinite( m, ... )

## S3 method for class 'matrix'
isSemidefinite( m, positive = TRUE,
  tol = 100 * .Machine$double.eps,
  method = ifelse( nrow( m ) < 13, "det", "eigen" ), ... )

## S3 method for class 'list'
isSemidefinite( m, ... )

semidefiniteness( m, ... )
```

Arguments

<code>m</code>	a symmetric quadratic matrix or a list containing symmetric quadratic matrices.
<code>positive</code>	logical. Check for positive semidefiniteness (if TRUE, default) or for negative semidefiniteness (if FALSE).
<code>tol</code>	tolerance level (values between $-tol$ and tol are considered to be zero).
<code>method</code>	method to test for semidefiniteness, either checking the signs of the principal minors (if "det", default for matrices with up to 12 rows/columns) or checking the signs of the eigenvalues (if "eigen", default for matrices with 13 or more rows/columns).
<code>...</code>	further arguments of <code>isSemidefinite.list</code> are passed to <code>isSemidefinite.matrix</code> ; further arguments of <code>semidefiniteness</code> are passed to <code>isSemidefinite</code> ; further arguments of other functions are currently ignored.

Details

Function `semidefiniteness()` passes all its arguments to `isSemidefinite()`. It is only kept for backward-compatibility and may be removed in the future.

If argument `positive` is set to FALSE, `isSemidefinite()` checks for negative semidefiniteness by checking for positive semidefiniteness of the negative of argument `m`, i.e. $-m$.

If method "det" is used (default for matrices with up to 12 rows/columns), `isSemidefinite()` checks whether all principal minors (not only the leading principal minors) of the matrix `m` (or

of the matrix $-m$ if argument `positive` is `FALSE`) are larger than $-\text{tol}$. Due to rounding errors, which are unavoidable on digital computers, the calculated determinants of singular (sub-)matrices (which should theoretically be zero) can considerably deviate from zero. In order to reduce the probability of incorrect results due to rounding errors, `isSemidefinite()` does not calculate the determinants of (sub-)matrices with reciprocal condition numbers smaller than argument `tol` but sets the corresponding principal minors to (exactly) zero. The number of principal minors of an $N \times N$ matrix is $\sum_{k=1}^N (N \text{ choose } k)$, which gets very large for large matrices. Therefore, it is not recommended to use method `"det"` for matrices with, say, more than 12 rows/columns.

If method `"eigen"` (default for matrices with 13 or more rows/columns) is used, `isSemidefinite()` checks whether all eigenvalues of the matrix m (or of the matrix $-m$ if argument `positive` is `FALSE`) are larger than $-\text{tol}$. In case of a singular or nearly singular matrix, some eigenvalues that theoretically should be zero can considerably deviate from zero due to rounding errors, which are unavoidable on digital computers. `isSemidefinite()` uses the following procedure to reduce the probability of incorrectly returning `FALSE` due to rounding errors in the calculation of eigenvalues of singular or nearly singular matrices: if the reciprocal condition number of an $N \times N$ matrix is smaller than argument `tol` and not all of the eigenvalues of this matrix are larger than $-\text{tol}$, `isSemidefinite()` checks whether all $(N \text{ choose } (N - k)) (N - k) \times (N - k)$ submatrices are positive semidefinite, where k with $0 < k < N$ is the number of eigenvalues in the interval $-\text{tol}$ and `tol`. If necessary, this procedure is done recursively.

Please note that a matrix can be neither positive semidefinite nor negative semidefinite.

Value

`isSemidefinite()` and `semidefiniteness()` return a logical value (if argument m is a matrix) or a logical vector (if argument m is a list) indicating whether the matrix (or each of the matrices) is positive/negative (depending on argument `positive`) semidefinite.

Author(s)

Arne Henningsen

References

Chiang, A.C. (1984): *Fundamental Methods of Mathematical Economics*, 3rd ed., McGraw-Hill.
 Gantmacher, F.R. (1959): *The Theory of Matrices*, Chelsea Publishing.

Examples

```
# a positive semidefinite matrix
isSemidefinite( matrix( 1, 3, 3 ))

# a negative semidefinite matrix
isSemidefinite( matrix(-1, 3, 3 ), positive = FALSE )

# a matrix that is positive and negative semidefinite
isSemidefinite( matrix( 0, 3, 3 ))
isSemidefinite( matrix( 0, 3, 3 ), positive = FALSE )

# a matrix that is neither positive nor negative semidefinite
```

```

isSemidefinite( symMatrix( 1:6 ) )
isSemidefinite( symMatrix( 1:6 ), positive = FALSE )

# checking a list of matrices
ml <- list( matrix( 1, 3, 3 ), matrix(-1, 3, 3 ), matrix( 0, 3, 3 ) )
isSemidefinite( ml )
isSemidefinite( ml, positive = FALSE )

```

margEff	<i>Method for Returning Marginal Effects</i>
---------	--

Description

Currently, this package just defines the generic function `margEff` so that it can be used to define `margEff` methods for objects of specific classes in other packages.

Usage

```
margEff( object, ... )
```

Arguments

<code>object</code>	an object of which marginal effects should be calculated.
<code>...</code>	further arguments for methods

Author(s)

Arne Henningsen

nObs	<i>Return number of observations for statistical models</i>
------	---

Description

Returns number of observations for statistical models. The default method assumes presence of a component `param$nObs` in `x`.

Usage

```

nObs(x, ...)
## Default S3 method:
nObs(x, ...)
## S3 method for class 'lm'
nObs(x, ...)

```

Arguments

`x` a statistical model, such as created by [lm](#)
`...` further arguments for methods

Details

This is a generic function. The default method returns the component `x$param$nObs`. The `lm`-method is based on qr-decomposition, in the same way as the does [summary.lm](#).

Value

numeric, number of observations

Author(s)

Ott Toomet, <otoomet@econ.au.dk>

Examples

```
# Construct a simple OLS regression:
x1 <- runif(100)
x2 <- runif(100)
y <- 3 + 4*x1 + 5*x2 + rnorm(100)
m <- lm(y~x1+x2) # estimate it
nObs(m)
```

nParam	<i>Number of model parameters</i>
--------	-----------------------------------

Description

This function returns the number of model parameters. The default method returns the component `x$param$nParam`.

Usage

```
nParam(x, free=FALSE, ...)
## Default S3 method:
nParam(x, ...)
## S3 method for class 'lm'
nParam(x, ...)
```

Arguments

`x` a statistical model
`free` logical, whether to report only the free parameters or the total number of parameters (default)
`...` other arguments for methods

Details

Free parameters are the parameters with no equality restrictions. Some parameters may be restricted (e.g. sum of two probabilities may be restricted to equal unity). In this case the total number of parameters may depend on the normalisation.

Value

Number of parameters in the model

Author(s)

Ott Toomet, <otoomet@econ.au.dk>

See Also

[nObs](#) for number of observations

Examples

```
# Construct a simple OLS regression:
x1 <- runif(100)
x2 <- runif(100)
y <- 3 + 4*x1 + 5*x2 + rnorm(100)
m <- lm(y~x1+x2) # estimate it
summary(m)
nParam(m) # you get 3
```

quasiconcavity

Test for quasiconcavity / quasiconvexity

Description

Test whether a function is quasiconcave or quasiconvex. The bordered Hessian of this function is checked by `quasiconcavity()` or `quasiconvexity()`.

Usage

```
quasiconcavity( m, tol = .Machine$double.eps )
quasiconvexity( m, tol = .Machine$double.eps )
```

Arguments

`m` a bordered Hessian matrix or a list containing bordered Hessian matrices
`tol` tolerance level (values between $-tol$ and tol are considered to be zero).

Value

logical or a logical vector (if `m` is a list).

Author(s)

Arne Henningsen

ReferencesChiang, A.C. (1984) *Fundamental Methods of Mathematical Economics*, 3rd ed., McGraw-Hill.**Examples**

```

quasiconcavity( matrix( 0, 3, 3 ) )

quasiconvexity( matrix( 0, 3, 3 ) )

m <- list()
m[[1]] <- matrix( c( 0,-1,-1, -1,-2,3, -1,3,5 ), 3, 3 )
m[[2]] <- matrix( c( 0,1,-1, 1,-2,3, -1,3,5 ), 3, 3 )

quasiconcavity( m )

quasiconvexity( m )

```

rowMedians

Medians of Rows

Description

Compute the sample medians of the rows of a data.frame or matrix.

Usage

```
rowMedians( x, na.rm = FALSE )
```

Arguments

x	a data.frame or matrix.
na.rm	a logical value indicating whether NA values should be stripped before the computation proceeds.

Value

A vector of the medians of each row of x.

Author(s)

Arne Henningsen

See Also

[colMedians](#), [median](#), [colMeans](#).

Examples

```
m <- matrix( 1:12, nrow = 4 )
rowMedians( m )
```

rSquared	<i>Calculate R squared value</i>
----------	----------------------------------

Description

Calculate R squared value.

Usage

```
rSquared( y, resid )
```

Arguments

y	vector of endogenous variables
resid	vector of residuals

Author(s)

Arne Henningsen

Examples

```
data( "Electricity", package = "Ecdat" )
reg <- lm( cost ~ q + pl + pk + pf, Electricity )
rSquared( Electricity$cost, reg$residuals )
summary( reg )$r.squared # returns the same value
```

stdEr	<i>Standard deviations</i>
-------	----------------------------

Description

Extract standard deviations from estimated models.

Usage

```
stdEr(x, ...)
## Default S3 method:
stdEr(x, ...)
## S3 method for class 'lm'
stdEr(x, ...)
```

Arguments

`x` a statistical model, such as created by [lm](#)
`...` further arguments for methods

Details

`stdEr` is a generic function with methods for objects of "lm" class. The default method returns the square root of the diagonal of the variance-covariance matrix.

Value

numeric, the estimated standard errors of the coefficients.

Author(s)

Ott Toomet <otoomet@ut.ee>

See Also

[vcov](#), [summary](#).

Examples

```
data(cars)
lmRes <- lm(dist ~ speed, data=cars)
stdEr( lmRes )
```

sumKeepAttr

Sum of an Array While Keeping its Attributes

Description

This function returns the sum of an numeric array (e.g. vector or matrix) while keeping its attributes.

Usage

```
sumKeepAttr( x, keepNames = FALSE, na.rm = FALSE )
```

Arguments

`x` an numeric array (e.g. vector or matrix).
`keepNames` logical. Should the name(s) of the element(s) of `x` be assigned to the returned sum? (only relevant if `x` has only one element).
`na.rm` logical. Passed to [sum](#). Should missing values be removed?

Value

the sum (see [sum](#)).

Author(s)

Arne Henningsen

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

```
a <- 1:10
attr( a, "min" ) <- 1
attr( a, "max" ) <- 10
sum(a)
sumKeepAttr(a)
```

summarizedF

*Summarize a data.rframe***Description**

This function summarizes each variable that is in a data.frame. It can be used, e.g., in an R script to write summary information about a data.frame into a text file that is in a version control system so that one can see in the version control system whether one or more variables in the data frame have changed.

Usage

```
summarizedF( dat, printValues = TRUE, maxLevel = 20, file = NULL, ... )
```

Arguments

<code>dat</code>	a data.frame.
<code>printValues</code>	logical. If FALSE only MD5 checksums are returned, which could be desirable if the data frame contains confidential data that should not be included in the output.
<code>maxLevel</code>	integer. If the number of unique values in a variable is less than or equal to the number specified in this argument (and argument <code>printValues</code> is TRUE), a frequency table is included in the output.
<code>file</code>	a character string or a writable connection naming the file to write to.
<code>...</code>	further arguments forwarded to <code>sink()</code> if argument <code>file</code> is not NULL.

Author(s)

Arne Henningsen

`symMatrix`*Symmetric Matrix*

Description

Create a Symmetric Matrix.

Usage

```
symMatrix( data = NA, nrow = NULL, byrow = FALSE,  
           upper = FALSE )
```

Arguments

<code>data</code>	an optional data vector.
<code>nrow</code>	the desired number of rows and columns.
<code>byrow</code>	logical. If 'FALSE' (the default) the matrix is filled by columns, otherwise the matrix is filled by rows.
<code>upper</code>	logical. If 'FALSE' (the default) the lower triangular part of the matrix (including the diagonal) is filled, otherwise the upper triangular part of the matrix is filled.

Value

a symmetric matrix.

Author(s)

Arne Henningsen

See Also

[matrix](#), [lower.tri](#).

Examples

```
# fill the lower triangular part by columns  
symMatrix( 1:10, 4 )  
# fill the upper triangular part by columns  
symMatrix( 1:10, 4, upper = TRUE )  
# fill the lower triangular part by rows  
symMatrix( 1:10, 4, byrow = FALSE )
```

triang	<i>Upper triangular matrix from a vector</i>
--------	--

Description

Creates an upper triangular square matrix from a vector.

Usage

```
triang( v, n )
```

Arguments

v	vector
n	desired dimension of the returned square matrix

Note

If the vector has less elements than the upper triangular matrix, the last elements are set to zero.

Author(s)

Arne Henningsen

See Also

[veclipos](#).

Examples

```
v <- c( 1:5 )
triang( v, 3 )
```

vecli	<i>Vector of linear independent values</i>
-------	--

Description

Returns a vector containing the linear independent elements of a symmetric matrix (of full rank).

Usage

```
vecli( m )
```

Arguments

m	symmetric matrix
---	------------------

Author(s)

Arne Henningsen

See Also[veclipos](#).**Examples**

```
# a symmetric n x n matrix
m <- cbind(c(11,12,13),c(12,22,23),c(13,23,33))
vecli(m) # returns: 11 12 13 22 23 33
```

vecli2m*Convert vector of linear independent values into a Matrix*

Description

Converts a vector into a symmetric matrix that the original vector contains the linear independent values of the returned symmetric matrix.

Usage

```
vecli2m( v )
```

Arguments

v a vector.

Author(s)

Arne Henningsen

See Also[vecli](#), [veclipos](#).**Examples**

```
v <- c( 11, 12, 13, 22, 23, 33 )
vecli2m( v )
```

`veclipos`*Position in a vector of linear independent values*

Description

Returns the position of the [i,j]th element of a symmetric $n \times n$ matrix that this element has in a vector of the linear independent values of the matrix.

Usage

```
veclipos( i, j, n )
```

Arguments

<code>i</code>	row of the element in the matrix.
<code>j</code>	column of the element in the matrix.
<code>n</code>	dimension of the matrix.

Note

A symmetric $n \times n$ matrix has $n*(n+1)/2$ independent values.
The function is: $n*(n-1)/2 - ((n-\min(i,j))*(n-\min(i,j)+1)/2) + \max(i,j)$

Author(s)

Arne Henningsen

See Also

[vecli](#), [vecli2m](#).

Examples

```
veclipos( 1, 2, 3 ) # returns: 2
```

Index

- * **array**
 - colMedians, [3](#)
 - insertCol, [6](#)
 - insertRow, [7](#)
 - isSemidefinite, [8](#)
 - quasiconcavity, [12](#)
 - rowMedians, [13](#)
 - rSquared, [14](#)
 - symMatrix, [17](#)
 - triang, [18](#)
 - vecli, [18](#)
 - vecli2m, [19](#)
 - veclipos, [20](#)
- * **methods**
 - ddnorm, [4](#)
 - margEff, [10](#)
 - nObs, [10](#)
 - nParam, [11](#)
 - stdEr, [14](#)
 - sumKeepAttr, [15](#)
 - summarizeDF, [16](#)
- * **models**
 - coefTable, [2](#)
 - compPlot, [4](#)
 - histDens, [5](#)
- * **multivariate**
 - rSquared, [14](#)
- * **univar**
 - rSquared, [14](#)
- coefTable, [2](#)
- colMeans, [3](#), [13](#)
- colMedians, [3](#), [13](#)
- compPlot, [4](#)
- ddnorm, [4](#)
- dnorm, [5](#)
- hist, [5](#)
- histDens, [5](#)
- insertCol, [6](#), [7](#)
- insertRow, [6](#), [7](#)
- isSemidefinite, [8](#)
- lm, [11](#), [15](#)
- lower.tri, [17](#)
- margEff, [10](#)
- matrix, [17](#)
- median, [3](#), [13](#)
- nObs, [10](#), [12](#)
- nParam, [11](#)
- plot, [4](#)
- quasiconcavity, [12](#)
- quasiconvexity (quasiconcavity), [12](#)
- rowMedians, [3](#), [13](#)
- rSquared, [14](#)
- semidefiniteness (isSemidefinite), [8](#)
- stdEr, [14](#)
- sum, [15](#), [16](#)
- sumKeepAttr, [15](#)
- summarizeDF, [16](#)
- summary, [15](#)
- summary.lm, [11](#)
- symMatrix, [17](#)
- triang, [18](#)
- vcov, [15](#)
- vecli, [18](#), [19](#), [20](#)
- vecli2m, [19](#), [20](#)
- veclipos, [18](#), [19](#), [20](#)