Package 'Rfast'

July 22, 2017

Type Package

Title Fast R Functions

Version 1.8.2

Date 2017-07-21

Author Manos Papadakis, Michail Tsagris, Marios Dimitriadis, Ioannis Tsamardinos, Matteo Fasiolo, Giorgos Borboudakis, John Burkardt and Changliang Zou

Maintainer Manos Papadakis <papadakm95@gmail.com>

Depends R (>= 3.2.2), Rcpp (>= 0.12.3), RcppZiggurat

LinkingTo Rcpp (>= 0.12.3), RcppArmadillo

SystemRequirements C++11

URL https://rfast.eu

Description A collection of fast (utility) functions for data analysis. Column- and row-wise means, medians, variances, minimums, maximums, many t, F and G-square tests, many regressions (normal, logistic, Poisson), are some of the many fast functions.

License GPL (>= 2.0)

LazyData TRUE

NeedsCompilation yes

Repository CRAN

Date/Publication 2017-07-21 22:33:27 UTC

R topics documented:

`Rfast-package` *Really fast R functions*

Description

A collection of Rfast functions for data analysis. Note 1: The vast majority of the functions accept matrices only, not data.frames. Note 2: Do not have matrices or vectors with have missing data (i.e NAs). We do no check about them and C++ internally transforms them into zeros (0), so you may get wrong results. Note 3: In general, make sure you give the correct input, in order to get the correct output. We do no checks and this is one of the many reasons we are fast.

Details

Package:	Rfast
Type:	Package
Version:	1.8.2
Date:	2017-07-21
License:	GPL-2

Maintainers

Manos Papadakis <papadakm95@gmail.com>

Note

Acknowledgments:

We would like to acknowledge Professor Kurt Hornik, Doctor Uwe Ligges (and the rest of R core team) for their invaluable help with this R package. Erich Studerus for his invaluable comments and Neal Fultz for his suggestions. Vassilis Vasdekis for his invaluable help with the random effects models. Marios Dimitriadis work was funded by the Special Account for Research Funds of the University of Crete, Department of Computer Science. Phillip Sai is greatly acknowledged for his help with the Floyd-Warshal algorithm.

Author(s)

Manos Papadakis <papadakm95@gmail.com>, Michail Tsagris <mtsagris@yahoo.gr>, Marios Dimitriadis <kmdimitriadis@gmail.com>, Ioannis Tsamardinos <tsamard@csd.uoc.gr> , Matteo Fasiolo <matteo.fasiolo@gmail.com>, Giorgos Borboudakis <borbudak@gmail.com> and John Burkardt <jburkardt@fsu.edu>

`All k possible combinations from n elements`

All k possible combinations from n elements

Description

All k possible combinations from n elements.

Usage

```
comb_n(n, k, trans = FALSE)
```

Arguments

`n`	A stirctly (greater than zero) positive integer number or a vector with numbers.
`k`	A positive integer number at most equal to n or at most equal to the length of n, if n is a vector.
`trans`	A boolean value for returning the matrix transposed.

Value

A matrix with k columns and rows equal to the number of possible unique combinations of n with k elements.

Author(s)

Manos Papadakis and Marios Dimitriadis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com> and Marios Dimitriadis <kmdimitriadis@gmail.com>.

References

Nijenhuis A. and Wilf H.S. (1978). Combinatorial Algorithms for Computers and Calculators. Academic Press, NY.

See Also

`nth`, `colMaxs`, `colMins`, `colrange`

Examples

```
system.time( comb_n(20, 4) )
system.time( combn(20, 4) )
x <- rnorm(5)
comb_n(x, 3)
```

`Analysis of covariance`
Analysis of covariance

Description

Analysis of covariance

Usage

```
ancova1(y, ina, x, logged = FALSE)
```

Arguments

`y`	A numerical vector with the data, the response variable.
`ina`	A numerical vector with 1s, 2s, 3s and so one indicating the two groups. Be careful, the function is desinged to accept numbers greater than zero. Alternatively it can a factor variable.
`x`	A numerical vector whose length is equal to the number of rows of y. This is the covariate.
`logged`	Should the p-values be returned (FALSE) or their logarithm (TRUE)?

Details

Analysis of covariance is performed. No interaction between the factor and the covariate is tested. Only the main effects. The design need not be balanced. The values of ina need not have the same frequency. The sums of squares have been adjusted to accept balanced and unbalanced designs.

Value

A matrix with the test statistic and the p-value for the factor variable and the covariate.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

D.C. Montgomery (2001). Design and analysis of experiments (5th Edition). New York: John Wiley \& Sons

See Also

`ancovas`, `ftests`, `ttests`, `anova1`

Examples

```
y <- rnorm(90)
ina <- factor( rbinom(90, 2, 0.5) )
x <- rnorm(90)
system.time( a <- ancova1(y, ina, x) )
m1 <- lm(y ~ factor(ina) + x)
m2 <- lm(y ~ x + factor(ina))
anova(m1)
anova(m2)
```

`Analysis of variance with a count variable`

Analysis of variance with a count variable

Description

Analysis of variance with a count variable.

Usage

```
poisson.anova(y, ina, logged = FALSE)
geom.anova(y, ina, type = 1, logged = FALSE)
```

Arguments

`y`	A numerical vector with discrete valued data, i.e. counts.
`ina`	A numerical vector with discrete numbers starting from 1, i.e. 1, 2, 3, 4,... or a factor variable. This is suppose to be a categorical predictor. If you supply a continuous valued vector the function will obviously provide wrong results.
`type`	This rgument is for the geometric distribution. Type 1 refers to the case where the minimum is zero and type 2 for the case of the minimum being 1.
`logged`	Should the p-values be returned (FALSE) or their logarithm (TRUE)?

Details

This is the analysis of variance with Poisson or geometric distributed data. What we do is a log-likelihood ratio test. However, this is exactly the same as Poisson regression with a single predictor variable who happens to be categorical. Needless to say that this is faster function than the glm command in R. For the same purpose with a Bernoulli variable use `g2Test`.

Value

A vector with two values, the difference in the deviances and the relevant p-value.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

See Also

logistic.cat1, g2Test, poisson.anovas, anova, poisson_only,poisson.mle

Examples

```
y <- rpois(500, 10)
ina <- as.factor( rbinom(500, 4, 0.5) )
a1 <- poisson.anova(y, ina)
a2 <- glm(y ~ ina, poisson)
a1
anova(a2, test = "Chisq")
y <- rgeom(500, 0.7)
geom.anova(y, ina)
```

Angular central Gaussian random values simulation

Angular central Gaussian random values simulation

Description

Angular central Gaussian random values simulation.

Usage

```
racg(n, sigma)
```

Arguments

n	The sample size, a numerical value.
sigma	The covariance matrix in R^d.

Details

The algorithm uses univariate normal random values and transforms them to multivariate via a spectral decomposition. The vectors are then scaled to have unit length.

Value

A matrix with the simulated data.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>

References

Tyler D. E. (1987). Statistical analysis for the angular central Gaussian distribution on the sphere. Biometrika 74(3): 579-589.

See Also

acg.mle, rmvnorm, rmvlaplace, rmvt

Examples

```
s <- cov( iris[, 1:4] )
x <- racg(100, s)
acg.mle(x)
vmf.mle(x)  ## the concentration parameter, kappa, is very low, close to zero, as expected.
```

BIC (using partial correlation) forward regression

BIC (using partial correlation) forward regression

Description

BIC (using partial correlation) forward regression.

Usage

```
bic.corfsreg(y, x, tol = 2)
```

Arguments

y	A numerical vector.
x	A matrix with data, the predictor variables.
tol	If the BIC difference between two successive models is less than the tolerance value, the variable will not enter the model.

Details

The forward regression tries one by one the variables using the F-test, basically partial F-test every time for the latest variable. This is the same as testing the significance of the coefficient of this latest enetered variable. Alternatively the correlation can be used and this case the partial correlation coefficient. There is a direct relationship between the t-test statistic and the partial correlation coefficient. Now, instead of having to calculate the test statistic, we calculate the partial correlation coefficient. The largest partial correlation indicates the candidate variable to enter the model. If the BIC of the regression model with that variable included, reduces, less than "tol" from the previous model without this variable, the variable enters.

Value

A matrix with two columns, the index of the selected variable(s) and the BIC of each model. The first line is always 0 and the BIC of the model with no predictor variables.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Draper, N.R. and Smith H. (1988). Applied regression analysis. New York, Wiley, 3rd edition.

See Also

cor.fsreg, score.glms, univglms, logistic_only,poisson_only, regression

Examples

```
## 200 variables, hence 200 univariate regressions are to be fitted
x <- matrix( rnorm(200 * 200), ncol = 200 )
y <- rnorm(200)
system.time( a1 <- bic.corfsreg(y, x) )
system.time( a2 <- cor.fsreg(y, x) )
```

`Binary search algorithm`

Binary search algorithm

Description

Search a value in an ordered vector.

Usage

```
binary_search(x, v, index=FALSE)
```

Arguments

x	A vector with the data.
v	A value to check if exists in the vector x.
index	A boolean value for choose to return the position inside the vector.

Details

The functions is written in C++ in order to be as fast as possible.

Value

Search if the v exists in x. Then returns TRUE/FALSE if the value is been found.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

is_element

Examples

```
x <- sort(rnorm(1000))
v <- x[50]
b <- binary_search(x,v)
b1 <- binary_search(x,v,TRUE)
```

Binomial coefficient and its logarithm

Binomial coefficient and its logarithm

Description

Binomial coefficient and its logarithm.

Usage

```
Lchoose(x, k)
Choose(x, k)
```

Arguments

x	A vector with integer values numbers.
k	A positive non zero at most equal to x.

Details

The binomial coefficient or its logarithm are evaluated.

Value

A vector with the answers.

Author(s)

Manos Papadakis

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>

See Also

`comb_n`, `Lbeta`, `Lgamma`

Examples

```
x <- sample(20:30, 100, replace = TRUE)
Choose(x, 4)
Lchoose(x, 4)
```

`Check if any column or row is fill with zeros`

Check if any column or row is fill with zeros

Description

Check if any column or row is fill with zeros.

Usage

```
colrow.zero(x)
```

Arguments

`x`	A vector with data.

Details

Check all the column if any has all its elements zeros. If found, return \"TRUE\". Otherwise continues with rows. If columns and rows hasn't any zero vector then return \"FALSE\". Even if it returns \"FALSE\" that doesn't mean the determinant can't be zero. It might be but if check before and found any zero vector then for sure the determinant it'll be zero.

Value

A boolean value, \"TRUE\" if any column OR row is zero. \"FALSE\" otherwise.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

rowMins, rowFalse, nth, colrange, colMedians, colVars, sort_mat, rowTrue

Examples

```
x <- matrix(runif(10*10),10,10)
colrow.zero(x)
```

Check Namespace and Rd files

Check Namespace and Rd files

Description

Check Namespace/Rd and examples files.

Usage

```
checkNamespace(path.namespace,path.rfolder)
checkAliases(path.man,path.rfolder)
checkExamples(path.man,dont.read = "",print.errors = FALSE)
```

Arguments

path.namespace	An full path to the \"NAMESPACE\" file.
path.rfolder	An full path to the directory that contains the \"R\" files.
path.man	An full path to the directory that contains the \"Rd\" files.
dont.read	A character vector with the name of the files that you wish not to read. By default it's empty \"\".
print.errors	A boolean value (TRUE/FALSE) for printing the errors, if exists, for every file. By default it's \"FALSE\".

Details

For function \"checkNamespace\": reads from the NAMESPACE file all the export R functions, reads from file R all the R functions and check if all the functions are export.

For function \"checkAliases\": reads from the man directory all the Rd files, then reads from each file the aliases and check if: 1) All the R files has man file or an alias. 2) All aliases belongs to functions. 3) If there are dublicated aliases.

For function \"checkExamples\": reads from the man directory all the Rd files, then read from each file the examples and then run each of them. If you want to print the errors then set \"print.errors=TRUE\" and then you will see all the errors for every file. For succeed run of your code you should first run \"library(PACKAGE_NAME)\".

Value

For function \"checkNamespace\": a vector with the names of missing R files.

For function \"checkAliases\": a list with 3 fields.

`Missing Man files`
: A vector with the names of the missing Rd files.

`Missing R files`
: A vector with the names of the missing R files.

`Duplicate alias`
: A vector with the names of the dublicate aliases.

For function \"checkExamples\": a character vector with the names of the Rd files that produced an error.

Author(s)

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

`read.directory`, `AddToNamespace`, `sourceR`, `sourceRd`, `read.examples`

Examples

```
# for example: path.namespace="C:\some_file\NAMESPACE"
# for example: path.rfolder="C:\some_file\R\"
# for example: path.man="C:\some_file\man\"
# system.time( a<-checkNamespace(path.namespace,path.rfolder) )
# system.time( b<-checkAliases(path.man,path.rfolder) )
# system.time( b<-checkExamples(path.man) )
```

Check whether a square matrix is symmetric

Check whether a square matrix is symmetric

Description

Check whether a square matrix is symmetric.

Usage

```
is.symmetric(x)
```

Arguments

`x`	A square matrix with data.

Details

Instead of going through the whole matrix, the function will stop if the first disagreement is met.

Value

A boolean value, TRUE of FALSE.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

colVars, cor, cov

Examples

```
x <-matrix( rnorm( 100 * 400), ncol = 400 )
s1 <- cor(x)
is.symmetric(s1)
x <- x[1:100, ]
is.symmetric(x)
```

Cholesky decomposition of a square matrix

Cholesky decomposition of a square matrix

Description

Cholesky decomposition of a square matrix.

Usage

```
cholesky(x,parallel = FALSE)
```

Arguments

x	A square positive definite matrix.
parallel	A boolean value for parallel version.

Details

The Cholesky decomposition of a square positive definite matrix is computed.

Value

An upper triangular matrix.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>

See Also

`is.symmetric`

Examples

```
x = matrix(rnorm(1000 * 50), ncol = 50)
s = cov(x)
system.time(a1 <- cholesky(s))
system.time(a2 <- chol(s))
all.equal(a1[upper.tri(a1)], a2[upper.tri(a2)])
```

`Circular or angular regression`

Circular or angular regression

Description

Regression with circular dependent variable and Euclidean or categorical independent variables.

Usage

```
spml.reg(y, x, tol = 1e-07, seb = FALSE)
```

Arguments

`y`	The dependent variable, a numerical vector, it can be in radians or degrees.
`x`	The independent variable(s). Can be Euclidean or categorical (factor variables).
`tol`	The tolerance value to terminatate the Newton-Raphson algorithm.
`seb`	Do you want the standard error of the estimates to be returned? TRUE or FALSE.

Details

The Newton-Raphson algorithm is fitted in this regression as described in Presnell et al. (1998).

Value

A list including:

iters	The number of iterations required until convergence of the EM algorithm.
be	The regression coefficients.
seb	The standard errors of the coefficients.
loglik	The value of the maximised log-likelihood.
seb	The covariance matrix of the beta values.

Author(s)

Michail Tsagris and Manos Papadakis

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>

References

Presnell Brett, Morrison Scott P. and Littell Ramon C. (1998). Projected multivariate linear models for directional data. Journal of the American Statistical Association, 93(443): 1068-1077.

See Also

spml.mle, iag.mle, acg.mle

Examples

```
x <- rnorm(100)
z <- cbind(3 + 2 * x, 1 -3 * x)
y <- cbind( rnorm(100,z[ ,1], 1), rnorm(100, z[ ,2], 1) )
y <- y / sqrt( rowsums(y^2) )
y <- atan( y[, 2] / y[, 1] ) + pi * I(y[, 1] < 0)
spml.reg(y, x)
```

Column and row-wise Any

Column and row-wise Any

Description

Column and row-wise Any of a matrix.

Usage

```
colAny(x)
rowAny(x)
```

Arguments

x	A logical matrix with the data.

Details

The functions is written in C++ in order to be as fast as possible.

Value

A vector where item \"i\" is true if found Any true in column/row \"i\". Otherwise false.

Author(s)

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

`med`, `colMedians`, `colMeans` (buit-in R function)

Examples

```
x <- matrix(as.logical(rbinom(100*100,1,0.5)),100,100)
system.time( a<-colAny(x) )
system.time( b<-apply(x,2,any) )
all.equal(a,b)

system.time( a<-rowAny(x) )
system.time( b<-apply(x,1,any) )
all.equal(a,b)
```

Column and row-wise means of a matrix

Column and row-wise means of a matrix

Description

Column and row-wise means of a matrix.

Usage

```
colmeans(x)
rowmeans(x)
colhameans(x)
rowhameans(x)
```

Arguments

x	A numerical matrix with data.

Value

A vector with the column or row arithmetic or harmonic means.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

colsums, rowsums, colMins, colMedians, colMads

Examples

```
x <- matrix(rpois(100 * 100, 10),ncol = 100)
x1 <- colmeans(x)
x2 <- colMeans(x)
all.equal(x1,x2)

x1 <- rowmeans(x)
x2 <- rowMeans(x)
all.equal(x1,x2)
system.time( colhameans(x) )
system.time( rowhameans(x) )
```

Column and row-wise medians

Column and row-wise medians

Description

Column and row-wise medians of a matrix.

Usage

```
colMedians(x)
rowMedians(x)
```

Arguments

x	A matrix with the data.

Details

The functions is written in C++ in order to be as fast as possible.

Value

A vector with the column medians.

Author(s)

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

`med`, `colVars`, `colMeans` (buit-in R function)

Examples

```
x <- matrix( rnorm(100 * 100), ncol = 100 )
a <- apply(x, 2, median)
b1 <- colMedians(x)
all.equal(as.vector(a), b1)
```

`Column and row-wise nth`

Column and row-wise nth

Description

Column and row-wise nth of a matrix.

Usage

```
colnth(x,elems)
rownth(x,elems)
```

Arguments

`x`	A matrix with the data.
`elems`	Am integer vector with the kth smallest number to be returned for each column/row.

Details

The functions is written in C++ in order to be as fast as possible.

Value

A vector with the column/row nth.

Author(s)

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

`med`, `colMedians`, `colMeans` (buit-in R function)

Examples

```
x <- matrix( rnorm(100 * 100), ncol = 100 )
elems <- rpois(100,10)
system.time( colnth(x, elems) )
elems <- rpois(100, 10)
system.time( rownth(x,elems) )
```

Column and row-wise order

Column and row-wise order

Description

Column and row-wise order.

Usage

```
colOrder(x,stable=FALSE)
rowOrder(x,stable=FALSE)
```

Arguments

x	A matrix with numbers.
stable	A boolean value for using a stable sorting algorithm.

Details

The function applies \"order\" in a column or row-wise fashion.

Value

A matrix with integer numbers. The result is the same as apply(x, 2, order) or apply(x, 1, order).

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

colsums, coldiffs, colMedians, colprods

Examples

```
x <- matrix( runif(10 * 10), ncol = 10 )
colOrder(x)
apply(x, 2, order)
rowOrder(x)
t(apply(x, 1, order))
```

`Column and row-wise products` *Column and row-wise products*

Description

Column and row-wise products.

Usage

```
colprods(x)
rowprods(x)
```

Arguments

`x`	A matrix with numbers.

Details

The product of the numbers in a matrix is returned either column-wise or row-wise.

Value

A vector with the column or the row products.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

`colsums`, `coldiffs`, `colMedians`

Examples

```
x <- matrix( runif(100 * 10), ncol = 10 )
colprods(x)
rowprods(x)
```

Column and row-wise range of values of a matrix

Column and row-wise range of values of a matrix.

Description

Column and row-wise range of values of a matrix.

Usage

```
colrange(x, cont = TRUE)
rowrange(x, cont = TRUE)
```

Arguments

x	A numerical matrix with data.
cont	If the data are continuous, leave this TRUE and it will return the range of values for each variable (column). If the data are integers, categorical, or if you want to find out the number of unique numbers in each column set this to FALSE.

Value

A vector with the relevant values.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

colMins, colMaxs, rowMins, rowMaxs, nth, colMedians, colVars, sort_mat

Examples

```
x <- matrix( rnorm(100 * 100), ncol = 100 )

a1 <- colrange(x)
a2 <- apply(x, 2, function(x) diff( range(x)) )
all.equal(a1, a2)

a1 <- rowrange(x)
a2 <- apply(x, 1, function(x) diff( range(x)) )
all.equal(a1, a2)
```

`Column and row-wise Shuffle`
Column and row-wise Shuffle

Description

Column and row-wise shuffle of a matrix.

Usage

```
colShuffle(x)
rowShuffle(x)
```

Arguments

x	A matrix with the data.

Details

The functions is written in C++ in order to be as fast as possible.

Value

A vector with the column/row Shuffle.

Author(s)

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

`med`, `colVars`, `colMeans` (buit-in R function)

Examples

```
x <- matrix( rnorm(100 * 100), ncol = 100 )
system.time( colShuffle(x) )
system.time( rowShuffle(x) )
```

Column and row-wise sums of a matrix

Column and row-wise sums of a matrix

Description

Column and row-wise sums of a matrix.

Usage

```
colsums(x)

rowsums(x)
```

Arguments

x	A numerical matrix with data.

Value

A vector with sums.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

colMedians, colmeans, colVars

Examples

```
x <- matrix(rpois(500 * 100, 10),ncol = 100)
x1 <- colsums(x)
x2 <- colSums(x)
all.equal(x1,x2)

x1 <- rowsums(x)
x2 <- rowSums(x)
all.equal(x1,x2)
```

`Column and row-wise tabulate`
Column and row-wise tabulate

Description

Column and row-wise tabulate of a matrix.

Usage

```
colTabulate(x, max_number = max(x))
rowTabulate(x, max_number = max(x))
```

Arguments

`x`	An integer matrix with the data. The numbers must start from 1, i.e. 1, 2, 3, 4,... No zeros are allowed. Anything else may cause a crash.
`max_number`	The maximum value of vector x. If you know which is the max number use this argument for faster results or by default max(x).

Details

The functions is written in C++ in order to be as fast as possible.

Value

A matrix where in each column the command "tabulate" has been performed. The number of rows of the returned matrix will be equal to the max_number if given. Otherwise, the functions will find this number.

Author(s)

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

`colShuffle`, `colVars`, `colmeans`

Examples

```
x <- matrix( rbinom(100 * 100, 4, 0.5), ncol = 100 )
system.time( colTabulate(x) )
x <- t(x)
system.time( rowTabulate(x) )
```

`Column and row-wise variances and standard deviations`

Column and row-wise variances and standard deviations of a matrix

Description

Column and row-wise variances and standard deviations of a matrix

Usage

```
colVars(x, suma = NULL, std = FALSE)
rowVars(x, suma = NULL, std = FALSE)
groupcolVars(x, ina, std = FALSE)
```

Arguments

`x`	A matrix with the data.
`suma`	If you already have the column sums vector supply it, otherwise leave it NULL.
`std`	A boolean variable specyfying whether you want the variances (FALSE) or the standard deviations (TRUE) of each column.
`ina`	A numerical vector specifying the groups. If you have numerical values, do not put zeros, but 1, 2, 3 and so on.

Details

We found this in stackoverflow and was created by David Arenburg. We then modified the function to match the sums type formula of the variance, which is faster.

Value

A vector with the colum variances or standard deviations.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

See Also

`colmeans`, `colMedians`, `colrange`

Examples

```
x <- matrix( rnorm(100 * 100), ncol = 100 )
a2 <- colVars(x)
groupcolVars( as.matrix(iris[, 1:4]), as.numeric(iris[, 5]) )
```

`Column and rows-wise mean absolute deviations` *Column and row-wise mean absolute deviations*

Description

Column and row-wise mean absolute deviations.

Usage

```
colMads(x)
rowMads(x)
```

Arguments

x	A matrix with the data.

Details

The functions is written in C++ in order to be as fast as possible.

Value

A vector with the column-wise mean absolute deviations.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

See Also

`colMedians`, `rowMedians`, `colVars`, `colmeans`, `colMeans` (buit-in R function)

Examples

```
x <- matrix( rnorm(100 * 100), ncol = 100 )
system.time( a <- colMads(x) )
```

Column-wise differences
Column-wise differences

Description

Column-wise differences.

Usage

```
coldiffs(x)
```

Arguments

x	A matrix with numbers.

Details

This function simply does this function x[, -1] - x[, -k], where k is the last column of the matrix x. But it does it a lot faster. That is, 2nd column - 1st column, 3rd column - 2nd column, and so on.

Value

A matrix with one column less containing the differences between the successive columns.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

Dist, dista, colmeans

Examples

```
x <- matrix( rnorm(50 * 10), ncol = 10 )
coldiffs(x)
```

`Column-wise minimum and maximum` *Column-wise minimum and maximum of a matrix*

Description

Column-wise minimum and maximum of a matrix.

Usage

```
colMins(x, value = FALSE)

colMaxs(x, value = FALSE)

colMinsMaxs(x)
```

Arguments

`x`	A numerical matrix with data.
`value`	If the value is FALSE it returns the indices of the minimum/maximum, otherwise it returns the minimum and maximum values.

Value

A vector with the relevant values.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

`rowMins`, `rowMaxs`, `nth`, `colrange`, `colMedians`, `colVars`, `sort_mat`

Examples

```
x <- matrix( rnorm(100 * 200), ncol = 200 )

s1 <- colMins(x)
s2 <- apply(x, 2, min)

s1 <- colMaxs(x)
s2 <- apply(x, 2, max)

s1 <- colMinsMaxs(x)
s2 <- c(apply(x, 2, min), apply(x, 2, max))
```

Column-wise true value

Column-wise true value of a matrix

Description

Column-wise true value of a matrix.

Usage

```
colTrue(x)
colFalse(x)
colTrueFalse(x)
```

Arguments

x	A logical matrix with data.

Value

An integer vector where item \"i\" is the number of the true/false values of \"i\" column.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

rowMins, rowFalse, nth, colrange, colMedians, colVars, sort_mat, rowTrue

Examples

```
x <- matrix(as.logical(rbinom(100*100,1,0.5)),100,100)

s1 <- colTrue(x)

s1 <- colFalse(x)

s1 <- colTrueFalse(x)
```

`Correlation based forward regression`
Correlation based forward regression.

Description

Correlation based forward regression.

Usage

```
cor.fsreg(y, x, threshold = 0.05, tolb = 2, tolr = 0.02, stopping = "BIC")
```

Arguments

`y`	A numerical vector.
`x`	A matrix with data, the predictor variables.
`threshold`	The significance level, set to 0.05 by default. Bear in mind that the logarithm of it is used, as the logarithm of the p-values is calculated at every point. This will avoid numerical overflows and small p-values, less than the machine epsilon, being returned as zero.
`tolb`	If we see only the significane of the variables, many may enter the linear regression model. For this reason, we also use the BIC as a way to validate the inclusion of a candidate variable. If the BIC difference between two successive models is less than the tolerance value, the variable will not enter the model, even if it statistically significant. Set it to 0 if you do not want this extra check.
`tolr`	This is an alternative to the BIC change and it uses the adjusted coefficient of determination. If the increase in the adjusted R^2 is more than the tolr continue.
`stopping`	This refers to the type of extra checking to do. If you want the BIC check, set it to \"BIC\". If you want the adjusted R^2 check set this to \"ar2\". Or, if you want both of them to take place, both of these criteria to be satisfied make this \"BICR2\".

Details

The forward regression tries one by one the variables using the F-test, basically partial F-test every time for the latest variable. This is the same as testing the significance of the coefficient of this latest enetered variable. Alternatively the correlation can be used and this case the partial correlation coefficient. There is a direct relationship between the t-test statistic and the partial correlation coefficient. Now, instead of having to calculate the test statistic, we calculate the partial correlation coefficient. Using Fisher's z-transform we get the variance imediately. The partial correlation coefficient, using Fisher's z-transform, and the partial F-test (or the coefficient's t-test statistic) are not identical. They will be identical for large sample sizes though.

Value

A matrix with three columns, the index of the selected variables, the logged p-value and the the test statistic value and the BIC or adjusted R^2 of each model. In the case of stopping="BICR2" both of these criteria will be returned.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Draper, N.R. and Smith H. (1988). Applied regression analysis. New York, Wiley, 3rd edition.

See Also

score.glms, univglms, logistic_only, poisson_only, regression

Examples

```
## 200 variables, hence 200 univariate regressions are to be fitted
x <- matrix( rnorm(200 * 200), ncol = 200 )
y <- rnorm(200)
system.time( cor.fsreg(y, x) )
```

`Correlation between pairs of variables`

Correlation between pairs of variables

Description

Correlations between pairs of variables.

Usage

```
corpairs(x, y, rho = NULL, logged = FALSE)
```

Arguments

x	A matrix with real valued data.
y	A matrix with real valued data whose dimensions match those of x.
rho	This can be a vector of assumed correlations (equal to the number of variables or the columns of x or y) to be tested. If this is not the case, leave it NULL and only the correlations will be returned.
logged	Should the p-values be returned (FALSE) or their logarithm (TRUE)? This is taken into account only if "rho" is a vector.

Details

The paired correlations are calculated. For each column of the matrices x and y the correlation between them is calculated.

Value

A vector of correlations in the case of "rho" being NULL, or a matrix with two extra columns, the test statistic and the (logged) p-value.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Lambert Diane (1992). Zero-Inflated Poisson Regression, with an Application to Defects in Manufacturing. Technometrics. 34(1):1-14.

Johnson Norman L., Kotz Samuel and Kemp Adrienne W. (1992). Univariate Discrete Distributions (2nd ed.). Wiley

Cohen, A. Clifford (1960). Estimating parameters in a conditional Poisson distribution. Biometrics. 16:203-211.

Johnson, Norman L. Kemp, Adrianne W. Kotz, Samuel (2005). Univariate Discrete Distributions (third edition). Hoboken, NJ: Wiley-Interscience.

See Also

correls, allbetas, mvbetas

Examples

```
x <- matrix( rnorm(100 * 100), ncol = 100 )
y <- matrix( rnorm(100 * 100), ncol = 100 )
system.time( corpairs(x, y) )
a <- corpairs(x, y)
```

Correlations *Correlation between a vector and a set of variables*

Description

Correlation between a vector and a set of variables.

Usage

```
correls(y, x, type = "pearson", a = 0.05, rho = 0)
groupcorrels(y, x, type = "pearson", ina)
```

Arguments

y	A numerical vector.
x	A matrix with the data.
type	The type of correlation you want. "pearson" and "spearman" are the two supported types for the "correls" because their standard error is easily calculated. For the "groupcorrels" you can also put "kendall" because no hypothesis test is performed in that function.
a	The significance level used for the confidence intervals.
rho	The value of the hypothesised correlation to be used in the hypothesis testing.
ina	A factor variable or a numeric variable idicating the group of each observation.

Details

The functions uses the built-in function "cor" which is very fast and then includes confidence intervals and produces a p-value for the hypothesis test.

Value

For the "correls" a matrix with 5 column; the correlation, the p-value for the hypothesis test that each of them is eaqual to "rho", the test statistic and the $a/2%$ lower and upper confidence limits.

For the "groupcorrels" a matrix with rows equal to the number of groups and columns equal to the number of columns of x. The matrix contains the correlations only, no statistical hypothesis test is performed.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

See Also

allbetas, univglms

Examples

```
x <- matrix( rnorm(60 * 100), ncol = 100 )
y <- rnorm(60)
r <- cor(y, x)  ## correlation of y with each of the xs
a <- allbetas(y, x)  ## the coefficients of each simple linear regression of y with x
b <- correls(y, x)
ina <- rep(1:2, each = 30)
b2 <- groupcorrels(y, x, ina = ina)
```

`Count the frequency of a value`
Count the frequency of a value

Description

Count the frequency of a value.

Usage

```
count_value(x, value)
```

Arguments

`x`	A vector with the data (numeric or character).
`value`	The value to check its frequency in the vector \"x\".

Details

The functions is written in C++ in order to be as fast as possible. The \"x\" and \"value\" must have the same type. The type can be numeric or character.

Value

The frequency of a value in a vector in linear time.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

`med`, `binary_search`, `Order`, `nth`

Examples

```
x <- rnorm(100)
value <- x[50]
system.time( b <- count_value(x,value) )
x <- sample(letters,replace=TRUE)
value <- "r"
system.time( g <- count_value(x,value) )
```

Covariance and correlation matrix

Fast covariance and correlation matrix calculation

Description

Fast covariance and correlation matrix calculation.

Usage

```
cova(x)

cora(x)
```

Arguments

x	A matrix with data. It has to be matrix, if it is data.frame for example the function does not turn it into a matrix.

Details

The calculations take place faster than the built-in functions cor as the number of variables increases. For a few tens of variables. This is true if the number of variables is high, say from 500 and above. The "cova" on the other hand is always faster.

Value

The covariance or correlation matrix.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

See Also

colVars, cor, cov

Examples

```
x <-matrix( rnorm( 1000 * 100), ncol = 100 )
s1 <- cov(x)
s2 <- cova(x)
all.equal(s1, s2)
```

`data.frame.to_matrix` *Convert a dataframe to matrix*

Description

Convert a dataframe to matrix.

Usage

```
data.frame.to_matrix(x)
```

Arguments

`x`	A Numeric matrix with data.

Details

This functions converts a dataframe to matrix. Even if there are factors, the function converts them into numerical values. Attributes are not allowed for now.

Value

A matrix wich has the numrical values from the dataframe.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>

See Also

`Match`, `is.symmetric`, `permutation`

Examples

```
data.frame.to_matrix(iris)
```

Density of the multivariate normal and t distributions

Density of the multivariate normal and t distributions

Description

Density of the multivariate normal and t distributions.

Usage

```
dmvnorm(x, mu, sigma, logged = FALSE)
dmvt(x, mu, sigma, nu, logged = FALSE)
```

Arguments

x	A numerical matrix with the data. The rows correspond to observations and the columns to variables.
mu	The mean vector.
sigma	The covariance matrix.
nu	The degrees of freedom for the multivariate t distribution.
logged	Should the logarithm of the density be returned (TRUE) or not (FALSE)?

Details

The (log) density of the multivariate normal distribution is calculated for given mean vector and covariance matrix.

Value

A numerical vector with the density values calculated at each vector (row of the matrix x).

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

See Also

rmvnorm, rmvt, mvnorm.mle, iag.mle

Examples

```
x <- matrix(rnorm(100 * 20), ncol = 20)
mu <- colmeans(x)
s <- cova(x)
a1 <- dmvnorm(x, mu, s)
a2 <- dmvt(x, mu, s, 1)
```

`Design Matrix` *Design Matrix*

Description

Design Matrix.

Usage

```
design_matrix(x, ones = TRUE)
```

Arguments

`x`	A character vector or a factor type vector or a dataframe. Do not supply a numerical vector.
`ones`	A boolean variable specifying whether to include the ones in the design matrix or not. The default value is TRUE.

Details

This function implements the R's "model.matrix" function and is used only when the x is a factor/charactervector or Dataframe.

Value

Returns the same matrix with model.matrix.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>

See Also

`model.matrix`

Examples

```
a <- design_matrix( iris[, 5] )
b <- model.matrix( ~ iris[,5] )  ## R's built-in funciton
all.equal(as.vector(a),as.vector(b)) ## true
```

Diagonal Matrix *Diagonal Matrix*

Description

Fill the diagonal of a matrix or create a diagonal and initialize it with a specific value.

Usage

```
Diag.fill(x,v=0)
Diag.matrix(len,v=0)
```

Arguments

x	A matrix with data.
len	Number of columns or rows.
v	Value to initialize the diagonal of a matrix.

Value

Diag.fill returns a diagonal matrix where all the elements in the diagonal are equal to \"v\". Diag.matrix returns a diagonal matrix where has dimension \"len,len\" and all the elements in the diagonal are equal to \"v\".

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

rowMins, colFalse, nth, rowrange, rowMedians, rowVars, sort_mat, colTrue

Examples

```
x <- matrix( as.logical(rbinom(100*100,1,0.5),100,100))

f <- Diag.fill(x,1)
all(diag(f)==1)
f <- Diag.matrix(100,1) ##equals to diag(1,100,100)
```

`Distance between vectors and a matrix`
Distance between vectors and a matrix

Description

Distance between vectors and a matrix.

Usage

```
dista(xnew, x, type = "euclidean", trans = TRUE)
```

Arguments

`xnew`	A matrix with some data or a vector.
`x`	A matrix with the data, where rows denotes observations (vectors) and the columns contain the variables.
`type`	This can be either \"euclidean\" or \"manhattan\".
`trans`	Do you wan the returned matrix to be transposed? TRUE or FALSE.

Details

The target of this function is to calculate the distances between xnew and x without having to calculate the whole distance matrix of xnew and x. The latter does extra calculaitons, which can be avoided.

Value

A matrix with the distances of each xnew from each vector of x. The number of rows of the xnew and and the number of columns of xnew are the dimensions of this matrix.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

See Also

`mahala`, `Dist`, `total.dist`, `total.dista`

Examples

```
xnew <- as.matrix( iris[1:10, 1:4] )
x <- as.matrix( iris[-c(1:10), 1:4] )
a <- dista(xnew, x)
b <- as.matrix( dist( rbind(xnew, x) ) )
b <- b[ 1:10, -c(1:10) ]
sum( abs(a - b) )

## see the time
x <- matrix( rnorm(1000 * 4), ncol = 4 )
system.time( dista(xnew, x) )
system.time( as.matrix( dist( rbind(xnew, x) ) ) )
```

Distance matrix *Distance matrix*

Description

Distance matrix.

Usage

```
Dist(x, method = "euclidean", square = FALSE, p = 0)
vecdist(x)
```

Arguments

x	A matrix with data. The distances will be calculated between pairs of rows. In the case of **vecdist** this is a vector.
method	This is either "euclidean", "mahnatatan", "canberra1", "canberra2", "minimum", "maximum", "minkowski", "bhattacharyya", "hellinger", "total_variation" or "kullback_leibler" or "jensen_shannon". The last two options are basically the same.
square	If you choose "euclidean" or "hellinger" as the method, then you can have the optino to return the squared Euclidean distances by setting this argument to TRUE.
p	This is for the the Mikowski, the power of the metric.

Details

The distance matrix is computer with an extra argument for the Euclidean distances.

Value

A square matrix with the pairwise distances.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>

References

Mardia K. V., Kent J. T. and Bibby J. M. (1979). Multivariate Analysis. Academic Press.

See Also

`dista`, `colMedians`

Examples

```
x <- matrix(rnorm(50 * 10), ncol = 10)
a1 <- Dist(x)
a2 <- as.matrix( dist(x) )
```

`Eigenvalues in high dimensional principal component analysis`

Eigenvalues in high dimensional principal component analysis

Description

Eigenvalues in high dimensional (n«p) principal component analysis.

Usage

```
hd.eigen(x, center = TRUE, scale = FALSE)
```

Arguments

`x`	A numerical matrix with data where the rows are the observations and the columns are the variables.
`center`	Do you want your data centered? TRUE or FALSE.
`scale`	Do you want each of your variables scaled, i.e. to have unit variance? TRUE or FALSE.

Details

When n«p, at most the first n eigenvalues are non zero. Hence, there is no need to calculate the other p-n zero eigenvalues. When center is TRUE, the eigenvalues of the covariance matrix are calculated. When both the center and scale is TRUE the eigenvalues of the correlation matrix are calculated.

Value

A vector with the first n eigenvalues and eigenvectors.

Author(s)

Michail Tsagris and Giorgos Borboudakis

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>.

See Also

prcomp, eigen, rmdp

Examples

```
x <- matrix( rnorm(40 * 200), ncol = 200)
a <- hd.eigen(x)
b <- prcomp(x)
b$sdev^2
```

Energy distance between two matrices

Energy distance between two matrices

Description

Energy distance between two matrices.

Usage

```
edist(x, y)
```

Arguments

x	A matrix with numbers.
y	A second matrix with data. The number of comlumns of this matrix must be the same with the matrix x. The number of rows can be different.

Details

This calculates the energy distance between two matrices. It will work even for tens of thousands of rows, it will just take some time. See the references for more information.

Value

A numerical value, the energy distance.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

References

Szekely G. J. and Rizzo M. L. (2004) Testing for Equal Distributions in High Dimension, InterStat, November (5).

Szekely G. J. (2000) Technical Report 03-05, E-statistics: Energy of Statistical Samples, Department of Mathematics and Statistics, Bowling Green State University.

Sejdinovic D., Sriperumbudur B., Gretton A. and Fukumizu, K. (2013). Equivalence of distance-based and RKHS-based statistics in hypothesis testing. The Annals of Statistics, 41(5), 2263-2291.

See Also

`total.dist`, `total.dista`, `Dist`, `dista`

Examples

```
x <- matrix( rnorm(50 * 10), ncol = 10 )
y <- matrix( rnorm(20 * 10), ncol = 10 )
edist(x, y)
edist(y, x)
```

Equality of objects *Equality of objects*

Description

Equality of objects.

Usage

```
all_equals(x,y,round_digits = FALSE,without_attr=FALSE,fast_result=FALSE)
```

Arguments

`x`	A Matrix, List, Dataframe or Vector.
`y`	A Matrix, List, Dataframe or Vector.
`round_digits`	The digit for rounding numbers.
`without_attr`	A boolean value (TRUE/FALSE) for deleting attributes. Be carefull although because some atributes are very important for you item.
`fast_result`	A boolean value (TRUE/FALSE) for using just identical.But you can combine only with round_digits argument.

Value

A boolean (TRUE/FALSE) value which represents if the items x and y are equal.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

Match, mvbetas, correls, univglms, colsums, colVars

Examples

```
x <- matrix( rnorm(100 * 100), ncol = 100 )
y <- matrix( rnorm(100 * 100), ncol = 100 )
all_equals(x,y)
all_equals(x, x)
```

Find element *Find element*

Description

Search a value in an unordered vector.

Usage

```
is_element(x, key)
```

Arguments

x	A vector or matrix with the data.
key	A value to check if exists in the vector x.

Details

Find if the key exists in the vector and return returns TRUE/FALSE if the value is been found. If the vector is unordered it is fast but if the vector is ordered then use binary_search. The functions is written in C++ in order to be as fast as possible.

Value

TRUE/FALSE if the value is been found.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

`binary_search` (buit-in R function)

Examples

```
x <- rnorm(500)
key <- x[50]
b <- is_element(x, key)
```

`Find the given value in a hash table`

Find the given value in a hash table

Description

Find the given value in a hash table or list.

Usage

```
hash.find(x,key)
```

Arguments

`x`	A hash table or list.
`key`	The key for searching the table.

Details

This function search the given key.

Value

If the given key exists return its value else returns 0.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>

See Also

`hash.list`

Examples

```
x <- hash.list(letters,c(1:26))
value <- hash.find(x,"a")
x[["a"]]==value
```

`Fitted probabilities of the Terry-Bradley model`

Fitted probabilities of the Terry-Bradley model

Description

Fitted probabilities of the Terry-Bradley model.

Usage

```
btmprobs(x, tol = 1e-09)
```

Arguments

`x`	A numerical square, usually not symmetric, matrix with discrete valued data. Each entry is a frequency, to give an example, the number of wins. x[i, j] is the number of wins of home team i against guest team j. x[j, i] is the number of wins of home team j against guest team i.
`tol`	The tolerance level to terminate the iterative algorithm.

Details

It fits a Bradley-Terry model to the given matrix and returns the fitted probabilities only.

Value

A list including:

`iters`	The numbetr of iterations required.
`probs`	A vector with probabilities which sum to 1. This is the probability of win for each item (or team in our hypothetical example).

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Bradley R.A. and Terry M.E. (1952). Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons. Biometrika, 39(3/4):324-345.

Huang Tzu-Kuo, Ruby C. Weng and Chih-Jen Lin (2006). Generalized Bradley-Terry models and multi-class probability estimates. Journal of Machine Learning Research, 7:85-115.

Agresti A. (2002). Categorical Data Analysis (2nd ed). New York: Wiley.

See Also

g2tests, poisson.anova, anova, poisson_only, poisson.mle

Examples

```
x <- matrix( rpois(10 * 10, 10), ncol = 10) ## not the best example though
btmprobs(x)
```

`Fitting a Dirichlet distribution via Newton-Rapshon`

Fitting a Dirichlet distribution via Newton-Rapshon

Description

Fitting a Dirichlet distribution via Newton-Rapshon.

Usage

```
diri.nr2(x, tol = 1e-07)
```

Arguments

`x`	A matrix containing the compositional data. Zeros are not allowed.
`tol`	The tolerance level idicating no further increase in the log-likelihood.

Details

Maximum likelihood estimation of the parameters of a Dirichlet distribution is performed via Newton-Raphson. Initial values suggested by Minka (2012) are used.

Value

A list including:

`iters`	The number of iterations required.
`loglik`	The value of the log-likelihood.
`param`	The estimated parameters.

Author(s)

Michail Tsagris and Manos Papadakis

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>

References

Minka Thomas (2012). Estimating a Dirichlet distribution. Technical report.

Ng Kai Wang, Guo-Liang Tian, and Man-Lai Tang (2011). Dirichlet and related distributions: Theory, methods and applications. John Wiley & Sons.

See Also

beta.mle

Examples

```
x <- matrix( rgamma(100 * 4, c(5, 6, 7, 8), 1), ncol = 4)
x <- x / rowsums(x)
diri.nr2(x)
```

Floyd-Warshall algorithm

Floyd-Warshall algorithm for shortest paths in a directed graph

---

Description

Floyd-Warshall algorithm for shortest paths in a directed graph.

Usage

```
floyd(x)
```

Arguments

x	The adjacency matrix of a directed graph. A positive number (including) in x[i, j] indicates that there is an arrow from i to j and it also shows the cost of going from i to j. Hence, the algorithm will find not only the shortest path but also the with the smallest cost. A value of NA means that there is no path. Put positive number only, as negative will cause problems.

Details

The Floyd-Warshall algorithm is designed to find the shortest path (if it exists) between two nodes in a graph.

Value

A matrix, say z, with 0 and positive numbers. The elements denote the length of the shortest path between each pair of points. If z[i, j] is zero it means that there is no cost from i to j. If z[i, j] has a positive value it means that the length of going from i to j is equal to that value.

Author(s)

John Burkardt (C++ code)

Ported into R and documentation: Manos Papadakis <papadakm95@gmail.com>.

References

Floyd, Robert W. (1962). Algorithm 97: Shortest Path. Communications of the ACM. 5(6): 345.

Warshall, Stephen (1962). A theorem on Boolean matrices. Journal of the ACM. 9 (1): 11-12.

https://en.wikipedia.org/wiki/Floyd

See Also

`sort_mat`

Examples

```
x <- matrix(NA, 10, 10)
x[sample(1:100, 10)] <- rpois(10, 3)
floyd(x)
```

Forward selection with generalised linear regression models

Variable selection in generalised linear regression models with forward selection

---

Description

Variable selection in generalised linear regression models with forward selection

Usage

```
fs.reg(y, ds, sig = 0.05, tol = 2, type = "logistic")
```

Arguments

`y`	The dependent variable. This can either be a binary numeric (0, 1) or a vector with integers (numeric or integer class), count data. The first case is for the binary logistic regression and the second for the Poisson regression.
`ds`	The dataset; provide a matrix where columns denote the variables and the rows the observations. The variables must be continuous, no categorical variables are accepted.

sig	Significance level for assessing the p-values significance. Default value is 0.05.
tol	The difference bewtween two successive values of the stopping rule. By default this is is set to 2. If for example, the BIC difference between two succesive models is less than 2, the process stops and the last variable, even though significant does not enter the model.
type	If you have a binary dependent variable, put "logistic" or "quasibinomial". If you have percentages, values between 0 and 1, including 0 and or 1, use "quasibinomial" as well. If you have count data put "poisson".

Details

The classical forward regression is implemented. The difference is that we have an extra step of check. Even if a variable is significant, the BIC of the model (with that variable) is calculated. If the decrease from the previous BIC (of the model without this variable) is less thatn a prespecified by the user value (default is 2) the variable wil enter. This way, we guard somehow against over-fitting.

Value

A matrix with for columns, the selected variables, the logarithm of their p-value, their test statistic and the BIC of the model with these variables included. If no variable is selected, the matrix is empty.

Author(s)

Marios Dimitriadis

Documentation: Marios Dimitriadis <kmdimitriadis@gmail.com>.

See Also

cor.fsreg, logistic_only, poisson_only, glm_logistic, glm_poisson

Examples

```
set.seed(123)

#simulate a dataset with continuous data
x <- matrix(rnorm(100 * 50), ncol = 50)
y <- rpois(100, 10)
a <- fs.reg(y, x, sig = 0.05, tol = 2, type = "poisson")

x <- matrix(rnorm(100 * 50), ncol = 50)
y <- rbinom(100, 1, 0.5)
b <- fs.reg(y, x, sig = 0.05, tol = 2, type = "logistic")
```

G-square test of conditional indepdence

G-square test of conditional indepdence

Description

G-square test of conditional indepdence with and without permutations.

Usage

```
g2Test(data, x, y, cs, dc)

g2Test_perm(data, x, y, cs, dc, nperm)
```

Arguments

data	A numerical matrix with the data. **The minimum must be 0, otherwise the function can crash or will produce wrong results**.
x	A number between 1 and the number of columns of data. This indicates which variable to take.
y	A number between 1 and the number of columns of data (other than x). This indicates the other variable whose independence with x is to be tested.
cs	A vector with the indices of the variables to condition upon. It must be non zero and between 1 and the number of variables. If you want unconditional independence test see g2Test_univariate and g2Test_univariate_perm. If there is an overlap between x, y and cs you will get 0 as the value of the test statistic.
dc	A numerical value equal to the number of variables (or columns of the data matrix) indicating the number of distinct, unique values (or levels) of each variable. Make sure you give the correct numbers here, otherwise the degrees of freedom will be wrong.
nperm	The number of permutations. The permutations test is slower than without permutations and should be used with small sample sizes or when the contigency tables have zeros. When there are few variables, R's "chisq.test" function is faster, but as the number of variables increase the time difference with R's procedure becomes larger and larger.

Details

The functions calculates the test statistic of the G^2 test of conditional independence between x and y conditional on a set of variable(s) cs.

Value

A list including:

`statistic`	The G^2 test statistic.
`df`	The degrees of freedom of the test statistic.
`x`	The row or variable of the data.
`y`	The column or variable of the data.

Author(s)

Giorgos Borboudakis. The permutation version used a C++ code by John Burkardt.

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

References

Tsamardinos, I., & Borboudakis, G. (2010). Permutation testing improves Bayesian network learning. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases (pp. 322-337). Springer Berlin Heidelberg

See Also

`g2Test_univariate`, `g2Test_univariate_perm`, `correls`, `univglms`

Examples

```
nvalues <- 3
nvars <- 10
nsamples <- 5000
data <- matrix( sample( 0:(nvalues - 1), nvars * nsamples, replace = TRUE ), nsamples, nvars )
dc <- rep(nvalues, nvars)

g2Test( data, 1, 2, 3, c(3, 3, 3) )
g2Test_perm( data, 1, 2, 3, c(3, 3, 3), 1000 )
```

Hash - Pair function *Hash - Pair function*

Description

Hash - Pair function.

Usage

```
hash.list(key,x)
```

Arguments

key	The keys of the given values.
x	The values.

Details

This function pairs each item of of key and value make a unique hash table.

Value

Returns the hash-list table.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>

See Also

hash.find

Examples

```
x <- hash.list(letters,c(1:26))
x[["a"]]==1
```

Hash object to a list object

Hash object to a list object

Description

Hash object to a list object.

Usage

```
hash2list(x, sorting = FALSE)
```

Arguments

x	A hash table with two parts, the keys (number(s) as string) and the key values (a single number).
sorting	This is if you you want the numbers in the keys sorted. The default value is FALSE.

Details

For every key, there is a key value. This function creates a list and puts every pair of keys and value in a component of a list.

Value

A list whose length is equal to the size of the hash table.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

hash.list, hash.find

Examples

```
x=list("1 2 4 3"=2.56,"2.34 1.05"=2)
hash2list(x)
hash2list(x,TRUE)
```

High dimensional MCD based detection of outliers

High dimensional MCD based detection of outliers

Description

High dimensional MCD based detection of outliers.

Usage

```
rmdp(y, alpha = 0.05, itertime = 100)
```

Arguments

y	A matrix with numerical data with more columns (p) than rows (n), i.e. n<p.
alpha	The significance level, i.e. used to decide whether an observation is said to be considered a possible outlier. The default value is 0.05.
itertime	The number of iterations the algorithm will be ran. The higher the sample size, the larger this number must be. With 50 observations in R^1000 maybe this has to be 1000 in order to produce stable results.

Details

High dimensional outliers (n«p) are detected using a properly constructed MCD. The variances of the variables are used and the determinant is simply their product.

Value

A list including: runtime = runtime, dis = dis, wei = wei

`runtime`	The duration of the process.
`dis`	The final estimated Mahalanobis type normalised distances.
`wei`	A bollean variable vector specifying whether an observation is "clean" (TRUE) or a possible outlier (FALSE).
`cova`	The estimated covatriance matrix.

Author(s)

Initial R code: Changliang Zou <nk.chlzou@gmail.com> R code modifications: Michail Tsagris <mtsagris@yahoo.gr> C++ implementation: Manos Papadakis <papadakm95@gmail.com> Documentation: Michail Tsagris <mtsagris@yahoo.gr> and Changliang Zhou <nk.chlzou@gmail.com>

References

Ro K., Zou C., Wang Z. and Yin G. (2015). Outlier detection for high-dimensional data. Biometrika, 102(3):589-599.

See Also

`colmeans`, `colVars`, `colMedians`

Examples

```
x <- matrix(rnorm(50 * 400), ncol = 400)
a <- rmdp(x, itertime = 500)
```

`Index of the columns of a data.frame which are factor variables`

Index of the columns of a data.frame which are factor variables

---

Description

Index of the columns of a data.frame which are factor variables.

Usage

```
which_isFactor(x)
```

Arguments

`x`	A data.frame where some columns are expected to be factor variables.

Details

The function is written in C++ and this is why it is very fast.

Value

A vector with the column indices which are factor variables. If there are no factor variables it will return an empty vector.

Author(s)

Manos Papadakis <papadakm95@gmail.com>

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

nth, Match

Examples

```
which_isFactor(iris)
```

Insert new function names in the NAMESPACE file

Insert new function names in the NAMESPACE file

Description

Insert new function names in the NAMESPACE file.

Usage

```
AddToNamespace(path.namespace,path.rfolder)
```

Arguments

path.namespace	An full path to the NAMESPACE file.
path.rfolder	An full path to the directory the new files to be added are stored.

Details

Reads the files that are exported in NAMESPACE and the files inside rfolder (where R files are) and insert every file that is not exported. To work properly must each R file to have the same name with the exported function. Also every file must have only one function.

Value

Returns the file that added in the export or empty character vector if all the files was inserted.

Author(s)

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

colShuffle, colVars, colmeans, read.directory

Examples

```
# for example: path.namespace="C:\some_file\NAMESPACE" where is NAMESPACE file
# path.rfolder="C:\some_file\R\" where is R files are
# system.time( a<-AddToNamespace(path.namespace,path.rfolder) )
# if(length(a)==0){
# print("all the files are inserted")
# }else{
# print("The new files that inserted are: \n")
# a
#  }
```

---

k nearest neighbours algorithm (k-NN)

k nearest neighbours algorithm (k-NN)

Description

k nearest neighbours algorithm (k-NN).

Usage

```
knn(xnew, y, x, k, dist.type = "euclidean", type = "C",
method = "average", freq.option = 0)
```

Arguments

xnew	The new data, new predictor variable values. A matrix with numerical data.
y	A vector with the response variable, whose values for the new data we wish to predict. This can be numerical data, or discrete, 0, 1, ... The latter is for classification.
x	The dataset. A matrix with numerical data.
k	The number of nearest neighbours to use. The number can either be a single value or a vector with multiple values.
dist.type	The type of distance to be used. Either \"euclidean\" or \"manhattan\".
type	If your response variable \"y\" is numerical data, then this should be \"R\" (regression). If \"y\" is in general categorical, factor or discrete set this argument to \"C\" (classification).
method	In case you have regression (type = \"R\") you want a way to summarise the prediction. If you want to take the average of the reponses of the k closest observations, type \"average\". For the median, type \"median\" and for the harmonic mean, type \"harmonic\".

`freq.option` If classification (type = \"C\") and ties occur in the prediction, more than one class has the same number of k nearest neighbours, in which case there are two strategies available: Option 0 selects the first most frequent encountered. Option 1 randomly selects the most frequent value, in the case that there are duplicates.

Details

The concept behind k-NN is simple. Suppose we have a matrix with predictor variables and a vector with the response variable (numerical or categorical). When a new vector with observations (predictor variables) is available, its corresponding response value, numerical or category is to be predicted. Instead of using a model, parametric or not, one can use this ad hoc algorithm.

The k smallest distances between the new predictor variables and the existing ones are calculated. In the case of regression, the average, median or harmonic mean of the corresponding respone values of these closest predictor values are calculated. In the case of classification, i.e. categorical response value, a voting rule is applied. The most frequent group (response value) is where the new observation is to be allocated.

Value

A matrix whose number of columns is equal to the size of k. If in the input you provided there is just one value of k, then a matrix with one column is returned containing the predicted values. If more than one value was supplied, the matrix will contain the predicted values for every value of k.

Author(s)

Marios Dimitriadis

R implementation and documentation: Marios Dimitriadis <kmdimitriadis@gmail.com>

References

Cover TM and Hart PE (1967). Nearest neighbor pattern classification. IEEE Transactions on Information Theory. 13(1):21-27.

Friedman J., Hastie T. and Tibshirani R. (2017). The elements of statistical learning. New York: Springer.

http://web.stanford.edu/~hastie/ElemStatLearn/printings/ESLII_print12.pdf

http://statlink.tripod.com/id3.html

See Also

`logistic_only`, `fs.reg`, `cor.fsreg`

Examples

```
# Simulate a dataset with continuous data
x <- as.matrix(iris[, 1:4])
y <- as.numeric(iris[, 5])
id <- sample(1:150, 120)
mod <- knn(x[-id, ], y[id], x[id, ], k = c(4, 5, 6), type = "C")
mod # Predicted values of y for 3 values of k.
```

```
table(mod[, 1], y[-id] )  # Confusion matrix for k = 4
table(mod[, 2], y[-id] )  # Confusion matrix for k = 5
table(mod[, 3], y[-id] )  # Confusion matrix for k = 6
```

`Linear models for large scale data`
Linear models for large scale data

Description

Linear models for large scale data.

Usage

```
lmfit(x, y, w = NULL)
```

Arguments

`x`	The design matrix with the data, where each column refers to a different sample of subjects. You must supply the design matrix, with the column of 1s. This function is the analogue of lm.fit and .lm.fit.
`y`	A numerical vector or a numerical matrix.
`w`	An optional numerical vector with weights. Note that if you supply this, the function does not make them sum to 1. So, you should do it.

Details

We have simply exploitted R's powerful function and managed to do better than .lm.fit which is a really powerful function as well. This is a bare bones function as it returns only two things, the coefficients and the residuals. .lm.fit returns more and lm.fit even more and finally lm returns too much. The motivatrion came form this site https://m-clark.github.io/docs/fastr.html . We changed the function a bit.

Value

A list including:

`be`	The beta coefficients.
`residuals`	The residuals of the linear model(s).

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Draper, N.R. and Smith H. (1988). Applied regression analysis. New York, Wiley, 3rd edition.

See Also

regression, allbetas, correls, mvbetas, cor.fsreg

Examples

```
n <- 1000  ;  p <- 5
X <- matrix(rnorm(n * p), n, p)
y <- rnorm(1000)
a1 <- .lm.fit(X, y)
a2 <- lmfit(X, y)
```

Logistic and Poisson regression models

Logistic and Poisson regression models

Description

Logistic and Poisson regression models.

Usage

```
glm_logistic(x, y, full = FALSE,tol = 1e-09)
glm_poisson(x, y, full = FALSE,tol = 1e-09)
```

Arguments

x	A matrix with the data, where the rows denote the samples (and the two groups) and the columns are the variables. This can be a matrix or a data.frame (with factors).
y	The dependent variable; a numerical vector with two values (0 and 1) for the logistic regression or integer values, 0, 1, 2,... for the Poisson regression.
full	If this is FALSE, the coefficients and the deviance will be returned only. If this is TRUE, more information is returned.
tol	The tolerance value to terminate the Newton-Raphson algorithm.

Details

The function is written in C++ and this is why it is very fast.

Value

When full is FALSE a list including:

be	The regression coefficients.
devi	The deviance of the model.

When full is TRUE a list including:

info	The regression coefficients, their standard error, their Wald test statistic and their p-value.
devi	The deviance.

Author(s)

Manos Papadakis <papadakm95@gmail.com>

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

McCullagh, Peter, and John A. Nelder. Generalized linear models. CRC press, USA, 2nd edition, 1989.

See Also

poisson_only, logistic_only, univglms, regression

Examples

```
x <- matrix(rnorm(100 * 3), ncol = 3)
y <- rbinom(100, 1, 0.6)   ## binary logistic regression
a1 <- glm_logistic(x, y, full = TRUE)
a2 <- glm(y ~ x, binomial)

x <- matrix(rnorm(100 * 3), ncol = 3)
y <- rpois(100, 10)   ## binary logistic regression
b1 <- glm_poisson(x, y, full = TRUE)
b2 <- glm(y ~ x, poisson)
```

`Logistic or Poisson regression with a single categorical predictor`

Logistic or Poisson regression with a single categorical predictor

---

Description

Logistic or Poisson regression with a single categorical predictor.

Usage

```
logistic.cat1(y, x, logged = FALSE)
poisson.cat1(y, x, logged = FALSE)
```

Arguments

y	A numerical vector with values 0 or 1.
x	A numerical vector with discrete numbers or a factor variable. This is suppose to be a categorical predictor. If you supply a continuous valued vector the function will obviously provide wrong results. **Note:** For the \"binomial.anova\" if this is a numerical vector it must contain strictly positive numbers, i.e. 1, 2, 3, 4, ..., no zeros are allowed.
logged	Should the p-values be returned (FALSE) or their logarithm (TRUE)?

Details

There is a closed form solution for the logistic regression in the case of a single predictor variable. See the references for more information.

Value

info	A matrix similar to the one produced by the glm command. The estimates, their standard error, the Wald value and the relevant p-value.
devs	For the logistic regression case a vector with the null and the residual deviances, their difference and the significance of this difference.
res	For the Poisson regression case a vector with the log likelihood ratio test statistic value and its significance.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Stan Lipovetsky (2015). Analytical closed-form solution for binary logit regression by categorical predictors. Journal of Applied Statistics, 42(1): 37–49.

See Also

poisson.anova, poisson.anovas, anova, logistic_only, poisson_only

Examples

```
y <- rbinom(20000, 1, 0.6)
x <- as.factor( rbinom(20000, 3, 0.5) )
system.time( a1 <- logistic.cat1(y, x) )
system.time( a2 <- glm(y ~ x, binomial) )
a1  ;  a2

y <- rpois(20000, 10)
x <- as.factor( rbinom(20000, 3, 0.5) )
system.time( a1 <- poisson.cat1(y, x) )
system.time( a2 <- glm(y ~ x, poisson) )
a1   ;  a2
```

`Lower and Upper triangular of a matrix/vector`

Lower and Upper triangular of a matrix/vector

Description

Lower/upper triangular matrix/vector.

Usage

```
lower_tri(x)
upper_tri(x)
```

Arguments

x	A matrix with data.

Value

Get a lower/upper trinagular matrix or a vector with the values of a lower/upper triangular.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

`rowMins`, `colFalse`, `nth`, `rowrange`, `rowMedians`, `rowVars`, `sort_mat`, `colTrue`

Examples

```
x <- matrix(runif(10*10),10,10)

all.equal(lower_tri(c(10,10)),lower.tri(x))

all.equal(lower_tri(x),x[lower.tri(x)])

#all.equal(upper_tri(c(10,10)),upper.tri(x))

#all.equal(upper_tri(x),x[upper.tri(x)])
```

Mahalanobis distance *Mahalanobis distance*

Description

Mahalanobis distance.

Usage

```
mahala(x, mu, sigma, ischol)
```

Arguments

x	A matrix with the data, where rows denotes observations (vectors) and the columns contain the variables.
mu	The mean vector.
sigma	The covariance or any square symmetric matrix.
ischol	A boolean variable set to true if the Cholesky decomposition of the covariance matrix is supplied in the argument \"sigma\".

Value

A vector with the Mahalanobis distances.

Author(s)

Matteo Fasiolo <matteo.fasiolo@gmail.com>,

C++ and R implementation and documentation: Matteo Fasiolo <matteo.fasiolo@gmail.com>.

See Also

dista, colmeans

Examples

```
x <- matrix( rnorm(100 * 50), ncol = 50 )
m <- colmeans(x)
s <- cov(x)
a1 <- mahala(x, m, s)
```

`Many 2 sample proportions tests`
Many 2 sample proportions tests

Description

It performs very many 2 sample proportions tests.

Usage

```
proptests(x1, x2, n1, n2)
```

Arguments

`x1`	A vector with the successes of the one group.
`x2`	A vector with the successes of the one group.
`n1`	A vector with the number of trials of the one group.
`n2`	A vector with the number of trials of the one group.

Details

The 2-sample proportions test is performed for each pair of proportions of teh two groups.

Value

A matrix with the proportions of each group (two columns), the test statistic and the p-value of each test.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

B. L. Welch (1951). On the comparison of several mean values: an alternative approach. Biometrika, 38(3/4), 330-336.

See Also

ttests, ftests, colVars

Examples

```
## 10000 variables, hence 10000 t-tests will be performed
set.seed(12345)
x1 <- rpois(1000, 5)
x2 <- rpois(1000, 5)
n1 <- rpois(1000, 40)
n2 <- rpois(1000, 40)
a <- proptests(x1, x2, n1, n2)
hist(a[, 4])

x1 <- rbinom(1000, 500, 0.6)
x2 <- rbinom(1000, 500, 0.6)
b <- proptests(x1, x2, 500, 500)
hist(b[, 4])
```

Many 2 sample tests *Many 2 sample tests tests*

Description

It performs very many 2 sample tests.

Usage

```
ttests(x, y = NULL, ina, paired = FALSE, logged = FALSE)
mcnemars(x, y = NULL, ina, logged = FALSE)
var2tests(x, y = NULL, ina, alternative = "unequal", logged = FALSE)
```

Arguments

x	A matrix with the data, where the rows denote the samples and the columns are the variables.
y	A second matrix with the data of the second group. If this is NULL (default value) then the argument ina must be supplied. Notice that when you supply the two matrices the procedure is two times faster.
ina	A numerical vector with 1s and 2s indicating the two groups. Be careful, the function is designed to accept only these two numbers. In addition, if your "y" is NULL, you must specify "ina".
alternative	The type of hypothesis to be checked, "equal", "greater", "less".
paired	If the groups are not independent paired t-tests should be performed and this must be TRUE, otherwise, leave it FALSE. In this case, the two groups must have equal smaple sizes, otherwise no test will be performed.
logged	Should the p-values be returned (FALSE) or their logarithm (TRUE)?

Details

For the ttests, if the groups are independent, the Welch's t-test (without assuming equal variances) is performed. Otherwise many paired t-tests are performed. The McNemar's test requires a number of observations, at least 30 would be good in order for the test to have some power and be size corect.

Value

A matrix with the test statistic, the degrees of freedom (if the groups are independent) and the p-value (or their logarithm) of each test.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

B. L. Welch (1951). On the comparison of several mean values: an alternative approach. Biometrika, 38(3/4), 330-336. McNemar Q. (1947). Note on the sampling error of the difference between correlated proportions or percentages. Psychometrika. 12(2):153-157.

See Also

ftests, anovas, ttest

Examples

```
## 1000 variables, hence 1000 t-tests will be performed
x = matrix( rnorm(100 * 100), ncol = 100 )
## 100 observations in total
ina = rbinom(100, 1, 0.6) + 1   ## independent samples t-test
system.time( ttests(x, ina = ina) )
x1 = x[ina == 1, ]
x2 = x[ina == 2, ]
system.time( ttests(x1, x2) )
```

Many analysis of variance tests with a discrete variable

Many analysis of variance tests with a discrete variable

Description

Many analysis of variance tests with a discrete variable.

Usage

```
poisson.anovas(y, ina, logged = FALSE)
geom.anovas(y, ina, type = 1, logged = FALSE)
```

Arguments

y	A numerical matrix with discrete valued data, i.e. counts for the case of the Poisson, or with 0s and 1s for the case of the Bernoulli distribution. Each column represents a variable.
ina	A numerical vector with discrete numbers starting from 1, i.e. 1, 2, 3, 4,... or a factor variable. This is suppose to be a categorical predictor. If you supply a continuous valued vector the function will obviously provide wrong results.
type	This rgument is for the geometric distribution. Type 1 refers to the case where the minimum is zero and type 2 for the case of the minimum being 1.
logged	Should the p-values be returned (FALSE) or their logarithm (TRUE)?

Details

This is the analysis of variance with Poisson distributed data. What we do is many log-likelihood ratio tests.

Value

A matrix with two values, the difference in the deviances (test statistic) and the relevant p-value.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

See Also

g2tests, poisson.anova, anova, poisson_only, poisson.mle

Examples

```
ina <- as.factor( rbinom(1000, 3, 0.5) )
## Poisson example
y <- matrix( rpois(1000 * 100,10), ncol= 100 )
system.time(a1 <- poisson.anovas(y, ina) )
```

`Many ANCOVAs` *Many ANCOVAs*

Description

Many ANCOVAs.

Usage

```
ancovas(y, ina, x, logged = FALSE)
```

Arguments

`y`	A matrix with the data, where the rows denote the observations and the columns are the variables.
`ina`	A numerical vector with 1s, 2s, 3s and so one indicating the two groups. Be careful, the function is desinged to accept numbers greater than zero. Alternatively it can a factor variable.
`x`	A numerical vector whose length is equal to the number of rows of y. This is the covariate.
`logged`	Should the p-values be returned (FALSE) or their logarithm (TRUE)?

Details

Many Analysis of covariance tests are performed. No interaction between the factor and the covariate is tested. Only the main effects. The design need not be balanced. The values of ina need not have the same frequency. The sums of squares have been adjusted to accept balanced and unbalanced designs.

Value

A matrix with the test statistic and the p-value for the factor variable and the covariate.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

D.C. Montgomery (2001). Design and analysis of experiments (5th Edition). New York: John Wiley \& Sons

See Also

`ftests`, `ttests`, `anovas`

Examples

```
## 100 variables, hence 100 F-tests will be performed
y <- matrix( rnorm(90 * 100), ncol = 100 )
ina <- factor( rbinom(90, 2, 0.5) )
x <- rnorm(90)
system.time( a <- ancovas(y, ina, x) )
m1 <- lm(y[, 15] ~ factor(ina) + x)
m2 <- lm(y[, 15] ~ x + factor(ina))
anova(m1)
anova(m2)
a[15, ]  ## the same with the m2 model, but not the m1
hist(a[, 3])
hist(a[, 4])
```

`Many are aunder the curve values`

Many are aunder the curve values

Description

Many are aunder the curve values.

Usage

```
colaucs(group, preds)
```

Arguments

`group`	A numeric vector with two values, one of which must be strictly 1.
`preds`	A numerical matrix with scores, probabilities or any other measure.

Details

The AUCs are calculated column-wise.

Value

A vector with length equal to the number of columns of the "preds" argument. The AUC vlaues for each column.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

See Also

ttests, ttest, ftests

Examples

```
## 200 variables, hence 200 F-tests will be performed
x <- matrix( rnorm(100 * 200), ncol = 200 )
ina <- rbinom(100, 1, 0.6)
system.time( colaucs(ina, x) )
a <- colaucs(ina, x)
```

`Many G-square tests of indepedence`

Many G-square tests of indepedence

Description

Many G-square tests of indepdence with and without permutations.

Usage

```
g2tests(data, x, y, dc)

g2tests_perm(data, x, y, dc, nperm)
```

Arguments

data	A numerical matrix with the data. **The minimum must be 0, otherwise the function can crash or will produce wrong results**.
x	An integer number or a vector of integer numbers showing the other variable(s) to be used for the G^2 test of independence.
y	An integer number showing which column of data to be used.
dc	A numerical value equal to the number of variables (or columns of the data matrix) indicating the number of distinct, unique values (or levels) of each variable. Make sure you give the correct numbers here, otherwise the degrees of freedom will be wrong.
nperm	The number of permutations. The permutations test is slower than without permutations and should be used with small sample sizes or when the contigency tables have zeros. When there are few variables, R's \"chisq.test\" function is faster, but as the number of variables increase the time difference with R's procedure becomes larger and larger.

Details

The function does all the pairwise G^2 test of independence and gives the position inside the matrix. The user must build the associations matrix now, similarly to the correlation matrix. See the examples of how to do that. The p-value is not returned, we leavve this to the user. See the examples of how to obtain it.

Value

A list including:

`statistic`	The G^2 test statistic for each pair of variables.
`pvalue`	This is returned when you have selected the permutation based G^2 test.
`x`	The row or variable of the data.
`y`	The column or variable of the data.
`df`	The degrees of freedom of each test.

Author(s)

Giorgos Borboudakis. The permutation version used a C++ code by John Burkardt.

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

References

Tsagris M. (2017). Conditional independence test for categorical data using Poisson log-linear model. Journal of Data Science, 15(2):347-356.

Tsamardinos, I., & Borboudakis, G. (2010). Permutation testing improves Bayesian network learning. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases (pp. 322-337). Springer Berlin Heidelberg.

See Also

`g2Test`, `g2Test_perm`, `correls`, `univglms`

Examples

```
nvalues <- 3
nvars <- 10
nsamples <- 2000
data <- matrix( sample( 0:(nvalues - 1), nvars * nsamples, replace = TRUE ), nsamples, nvars )
dc <- rep(nvalues, nvars)
a <- g2tests(data = data, x = 2:9, y = 1, dc = dc)
pval <- pchisq(a$statistic, a$df, lower.tail = FALSE)  ## p-value
b <- g2tests_perm(data = data, x = 2:9, y = 1, dc = dc, nperm = 1000)
```

`Many moment and maximum likelihood estimations of variance components`

Many moment and maximum likelihood estimations of variance components

---

Description

Many moment and maximum likelihood estimations of variance components.

Usage

```
colvarcomps.mom(x, id)
colvarcomps.mle(x, id, ranef = FALSE)
```

Arguments

x	A matrix with the data, where each column refers to a different sample of subjects.
id	A numerical vector indicating the subject. You must put consecutive numbers and no zero values. Alternatively this can be a factor variable.
ranef	Do you also want the random effects to be returned? TRUE or FALSE.

Details

Note that this formula works for **balanced designs only**, i.e. for each subject the same number of measurements have been taken.

The variance components, the variance of the between measurements and the variance of the within are estimated using moment estimators. The "colvarcomps.mom" is the moment analogue of a random effects model which uses likelihood estimation ("colvarcomps.mle"). It is much faster, but can give negative variance of the random effects, in which case it becomes zero.

The maximum likelihood version is a bit slower (try youselves to see the difference), but statistically speaking is to be preferred when small samples are available. The reason why it is only a little bit slower and not a lot slower as one would imagine is because we are using a closed formula to calculate the two variance components (Demidenko, 2013, pg. 67-69). Yes, there are closed formulas for linear mixed models.

Value

For the "colvarcomps.mom": A matrix with 5 columns, The MSE, the estimate of the between variance, the variance components ratio and a 95% confidence for the ratio.

For the "colvarcomps.mle": **If ranef = FALSE** a list with a single component called "info". That is a matrix with 3 columns, The MSE, the estimate of the between variance and the log-likelihood value. **If ranef = TRUE** a list including "info" and an extra component called "ranef" containing the random effects. It is a matrix with the same number of columns as the data. Each column contains the randome effects of each variable.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

D.C. Montgomery (2001). Design and analysis of experiments (5th Edition). New York: John Wiley \& Sons.

Charles S. Davis (2002). Statistical methods for the analysis of repeated measures. New York: Springer-Verlag.

Demidenko E. (2013). Mixed Models: Thoery and Applications with R 2nd Edition). New Jersey: John Wiley \& Sons (Excellent book).

See Also

varcomps.mle, colrint.regbx

Examples

```
## example taken from Montgomery, page 514-517.
y <- c(98, 97, 99, 96, 91, 90, 93, 92,
96, 95, 97, 95, 95, 96, 99, 98)
y <- matrix(y)
id <- rep(1:4, each = 4)


x <- rmvnorm(100, numeric(100), diag(rexp(100)) )
id <- rep(1:25, each = 4)
n <- 25  ;  d <- 4
a <- colvarcomps.mom(x, id)
mean(a[, 4]<0 & a[, 5]>0)
b <- colvarcomps.mle(x, id)
```

Many multi-sample tests

Many multi-sample tests

Description

Many multi-sample tests.

Usage

```
ftests(x, ina, logged = FALSE)
anovas(x, ina, logged = FALSE)
vartests(x, ina, type = "levene", logged = FALSE)
block.anovas(x, treat, block, logged = FALSE)
```

Arguments

x	A matrix with the data, where the rows denote the observations (and the two groups) and the columns are the variables.
ina	A numerical vector with 1s, 2s, 3s and so one indicating the two groups. Be careful, the function is desinged to accept numbers greater than zero. Alternatively it can be a factor variable.

`type`	This is for the variances test and can be either "levene" or "bf" corresponding to Levene's or Brown-Forsythe's testing procedure.
`treat`	In the case of the blocking ANOVA this argument plays the role of the "ina" argument.
`block`	This item, in the blocking ANOVA denotes the subjects which are the same. Similarly to "ina" a numeric vector with 1s, 2s, 3s and so on.
`logged`	Should the p-values be returned (FALSE) or their logarithm (TRUE)?

Details

The Welch's F-test (without assuming equal variances) is performed with the "ftests" function. The "anovas" function perform the classical (Fisher's) one-way analysis of variance (ANOVA) which assumes equal variance across the groups.

The "vartests" perform hypothesis test for the equality of the variances in two ways, either via the Levene or via the Brown-Forshythe procedure. Levene's test employs the means, whereas the Brown-Forsythe procedure employs the medians and is therefore more robust to outliers. The "var2tests" implement the classical F test.

The "block.anova" is the ANOVA with blocking, randomised complete block design (RCBD). In this case, for every combination of the block and treatment values, there is only one observation. The mathematics are the same as in the case of two way ANOVA, but the assumptions different and the testing procedure also different. In addition, no interaction is present.

Value

A matrix with the test statistic and the p-value of each test.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

B.L. Welch (1951). On the comparison of several mean values: an alternative approach. Biometrika, 38(3/4), 330-336.

D.C. Montgomery (2001). Design and analysis of experiments (5th Edition). New York: John Wiley \& Sons

See Also

`ttests`

Examples

```
## 1000 variables, hence 1000 F-tests will be performed
x <- matrix( rnorm(300 * 500), ncol = 500 )
## 300 observations in total
ina <- rbinom(300, 3, 0.6) + 1
a1 <- ftests(x, ina)
a2 <- anovas(x, ina)
a3 <- vartests(x, ina)
```

`Many multivariate simple linear regressions coefficients`
Many multivariate simple linear regressions coefficients

Description

Many multivariate simple linear regressions coefficients.

Usage

```
mvbetas(y, x, pvalue = FALSE)
```

Arguments

y	A matrix with the data, where rows denotes the observations and the columns contain the dependent variables.
x	A numerical vector with one continuous independent variable only.
pvalue	If you want a hypothesis test that each slope (beta coefficient) is equal to zero set this equal to TRUE. It will also produce all the correlations between y and x.

Details

It is a function somehow opposite to the `allbetas`. Instead of having one y and many xs we have many ys and one x.

Value

A matrix with the constant (alpha) and the slope (beta) for each simple linear regression. If the p-value is set to TRUE, the correlation of each y with the x is calculated along with the relevant p-value.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

See Also

allbetas, correls, univglms

Examples

```
y <- matrix( rnorm(100 * 100), ncol = 100 )
x <- rnorm(100)
a <- mvbetas(y, x, pvalue = FALSE)
b <- matrix(nrow = 100, ncol = 2)
z <- cbind(1, x)

system.time( a <- mvbetas(y, x) )
system.time(  for (i in 1:100) b[i, ] = coef( lm.fit( z, y[, i] ) )  )
```

Many non parametric multi-sample tests

Many multi-sample tests

Description

Many multi-sample tests.

Usage

```
kruskaltests(x, ina, logged = FALSE)
cqtests(x, treat, block, logged = FALSE)
```

Arguments

x	A matrix with the data, where the rows denote the samples (and the two groups) and the columns are the variables.
ina	A numerical vector with 1s, 2s, 3s and so one indicating the two groups. Be careful, the function is desinged to accept numbers greater than zero.
treat	In the case of the Cochran's Q test, this argument plays the role of the "ina" argument.
block	This item denotes the subjects which are the same. Similarly to "ina" a numeric vector with 1s, 2s, 3s and so on.
logged	Should the p-values be returned (FALSE) or their logarithm (TRUE)?

Details

The "kruskaltests" performs the Kruskal-Wallis non parametric alternative to analysis of variance test. The "cqtests" performs the Cocrhan's Q test for the equality of more than two groups whose values are strictly binary (0 or 1). This is a generalisation of the McNemar's test in the multi-sample case.

Value

A matrix with the test statistic and the p-value of each test.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

See Also

block.anovas, ftests

Examples

```
x <- matrix( rexp(300 * 500), ncol = 500 )
## 300 observations in total
ina <- rbinom(300, 3, 0.6) + 1
system.time( kruskaltests(x, ina) )
x <- matrix( rbinom(300 * 500, 1, 0.6), ncol = 500 )
treat <- rep(1:3, each = 100)
block <- rep(1:3, 100)
system.time( cqtests(x, treat, block) )
```

Many odds ratio tests *Many odds ratio tests*

Description

It performs very many odds ratio tests.

Usage

```
odds(x, y = NULL, ina, logged = FALSE)
```

Arguments

x	A matrix with the data, where the rows denote the samples and the columns are the variables. They must be 0s and 1s only.
y	A second matrix with the data of the second group. If this is NULL (default value) then the argument ina must be supplied. Notice that when you supply the two matrices the procedure is two times faster. They must be 0s and 1s only.
ina	A numerical vector with 1s and 2s indicating the two groups. Be careful, the function is designed to accept only these two numbers. In addition, if your "y" is NULL, you must specify "ina".
logged	Should the p-values be returned (FALSE) or their logarithm (TRUE)?

Details

Many odds ratio tests are performed.

Value

A matrix with the test statistic and the p-value (or their logarithm) of each test.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Mosteller Frederick (1968). Association and Estimation in Contingency Tables. Journal of the American Statistical Association. 63(321):1-28.

Edwards A.W.F. (1963). The measure of association in a 2x2 table. Journal of the Royal Statistical Society, Series A. 126(1):109-114.

See Also

`odds.ratio`, `g2Test_univariate`

Examples

```
x <- matrix(rbinom(100 * 500, 1, 0.5), ncol = 500)
ina <- rep(1:2, each = 50)
a <- odds(x,ina=ina)
```

Many one sample tests *Many one sample tests*

Description

Many one sample tests.

Usage

```
proptest(x, n, p, alternative = "unequal", logged = FALSE)
ttest(x, m, alternative = "unequal", logged = FALSE, conf = NULL)
vartest(x, sigma, alternative = "unequal", logged = FALSE, conf = NULL)
```

Arguments

x	A matrix with numerical data. Each column of the matrix corresponds to a sample, or a group. In the case of the "proptest" this is a vector integers ranging from 0 up to n. It is the number of "successes".
n	This is for the "proptest" only and is a vector with integer numbers specifying the number of tries for the proptest. Its size is equal to the size of x.
p	A vector with the assumed probabilities of success in the "proptest". Its size is equal to the number of colums of the matrix x.
m	A vector with the assumed means. Its size is equal to the number of colums of the matrix x.
sigma	A vector with assumed variances. Its size is equal to the number of colums of the matrix x.
alternative	The type of hypothesis to be checked. Equal to ("unequal"), grater than("greater") or less than ("less") the assumed parameter.
logged	Should the p-values be returned (FALSE) or their logarithm (TRUE)?
conf	If you want confidence intervals to be returned specify the confidence level, otherwise leave it NULL.

Details

Despite the functions having been written in R, they are very fast.

Value

For all tests except for the "sftests" a matrix with two colums, the test statistic and the p-value respectively.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

See Also

ftests, ttests

Examples

```
R <- 100
## protest
x <- rbinom(R, 50, 0.6)
n <- rep(50, R)
p <- rep(0.6, R)
a1 <- proptest(x, n, p, "unequal", logged = FALSE)
sum( a1[, 2] < 0.05 ) / R
```

```
## vartest
x <- matrix( rnorm(100 * 1000), ncol = R )
a2 <- vartest(x, rep(1, R) )
sum( a2[, 2] < 0.05 )

## ttest
a4 <- ttest(x, numeric(R) )
sum(a4[, 2] < 0.05) / R
```

`Many random intercepts LMMs for balanced data with a single identical covariate.`
Many random intercepts LMMs for balanced data with a single identical covariate

---

Description

Many random intercepts LMMs for balanced data with a single identical covariate.

Usage

```
colrint.regbx(y, x, id)
```

Arguments

`y`	A numerical matrix with the data. The subject values.
`x`	A numerical vector with the same length as the number of rows of y indicating the fixed predictor variable. Its values are the same for all levels of y. An example of this x is time which is the same for all subjects.
`id`	A numerical variable with 1, 2, ... indicating the subject.

Details

This is a special case of a balanced random intercepts model with a compound symmetric covariance matrix and one single covariate which is constant for all replicates. An example, is time, which is the same for all subjects. Maximum likelihood estimation has been performed. In this case the mathematics exist in a closed formula (Demidenko, 2013, pg. 67-69).

This is the generalistion of `rint.regbx` to matrices. Assume you have many observations, gene expressions over time for example, and you want to calculate the random effects or something else for each expression. Instead of using a "for" loop with `rint.regbx` function we have used amtrix operations to make it even faster.

Value

A list including:

`info`	A matrix with the random intercepts variance (between), the variance of the errors (within), the log-likelihood, the deviance (twice the log-likelihood) and the BIC. In the case of "rint.reg" it also includes the number of iterations required by the generalised least squares.

be — The estimated regression coefficients, which in the case of "rint.regbx" are simply two: the constant and the slope (time effect).

ranef — A matrix with random intercepts effects. Each row corresponds to a column in y. Instead of having a matrix with the same number of columns as y we return a transposed matrix.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Eugene Demidenko (2013). Mixed Models: Theory and Applications with R, 2nd Edition. New Jersey: Wiley \& Sons (excellent book).

See Also

colvarcomps.mle, rint.regbx, rm.lines, varcomps.mom, rint.reg

Examples

```
y <- matrix( rnorm(100 * 100), ncol = 100)
id <- rep(1:20, each = 5)
x <- rep(1:10, 10)
system.time( a<- colrint.regbx(y, x, id) )
```

Many regression based tests for single sample repeated measures

Many regression based tests for single sample repeated measures

---

Description

Many regression based tests for single sample repeated measures.

Usage

```
rm.lines(y, x, logged = FALSE)
rm.anovas(y, x, logged = FALSE)
```

Arguments

y	A matrix with the data, where each column refers to a different sample of subjects. For example, the first column is the repeated measurements of a sample of subjects, the second column contains repeated measurements of a second sample of subjects and so on. Within each column, the measurements of each subjects are stacked one upon the other. Say for examples there are n subjects and each of them has been measured d times (in time or at different experimental conditions). We put these in a matrix with just one column. The first d rows are the measurements of subject 1, the next d rows are the measurements of subject 2 and so on.
x	A numerical vector with time (usually) or the the predictor variable. For example the temperature, or the pressure. See the details for more information. Its length is equal to the time points for example, i.e. it must not have the same length as the number of rows of y. For the "rm.lines" this is a continuous variable. For the "rm.anovas" this is treated as a categorical variable, indicating say the type of experimental condition, but no difference between the points is important. Hence, for this function only, x can also be a facto variable.
logged	Should the p-values be returned (FALSE) or their logarithm (TRUE)?

Details

In order to see whether the repeated measurements are associated with a single covariate, e.g. time we perform many regressions and each time calculate the slope. For each subject, its regression slope with the covariate is calculated. In the end a t-test for the hypothesis that the average slopes is zero is performed. The regression slopes ignore that the measurements are not independent, but note that the slopes are independent, because they come from different subjects. This is a simple, summary statistics based approach found in Davis (2002), yet it can provide satisfactory results.

The second approach ("rm.anovas") found in Davis (2002) is the usual repeated measures ANOVA. In this case, suppose you have taken measurements on one or more variables from the same group of people. See the example below on how to put such data.

Value

A matrix with the test statistic (t-test) and its associated p-value.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Charles S. Davis (2002). Statistical methods for the analysis of repeated measures. Springer-Verlag, New York.

See Also

rint.regbx, rint.reg, varcomps.mle

Examples

```
y <- c(74.5,81.5,83.6,68.6,73.1,79.4,
75.5,84.6,70.6,87.3,73.0,75.0,
68.9,71.6,55.9,61.9,60.5,61.8,
57.0,61.3,54.1,59.2,56.6,58.8,
78.3,84.9,64.0,62.2,60.1,78.7,
54.0,62.8,63.0,58.0,56.0,51.5,
72.5,68.3,67.8,71.5,65.0,67.7,
80.8,89.9,83.2,83.0,85.7,79.6)
y <- as.matrix(y)
### the first 6 measurements are from subject 1, measurments 7-12 are from subject 2,
## measurements 13-18 are from subject 3 and so on.
x <- c(-10, 25, 37, 50, 65, 80) ## all subjects were measured at the same time points
rm.lines(y, x) ## Is linear trend between the measurements and the temperature?
rm.anovas(y, x)  ## Tests whether the means of the individuals are the same
## the temperature is treated as categorical variable here.

## fake example
y <- matrix( rnorm(10 * 4), ncol = 4 )
## the y matrix contains 4 repeated measurements for each of the 10 persons.
x <- 1:4
## we stack the measurements of each subject, one under the other in a matrix form.
y1 <- matrix( t(y) )
rm.anovas(y1, x)  ## perform the test
z <- matrix( rnorm(20 * 8), ncol = 2) ## same example, but with 2 sets of measurements.
rm.anovas(z, x)
```

Many score based GLM regressions

Many score based GLM regressions

Description

Many score based GLM regressions.

Usage

```
score.glms(y, x, oiko = NULL, logged = FALSE )
score.multinomregs(y, x, logged = FALSE )
score.negbinregs(y, x, logged = FALSE )
```

Arguments

y	A vector with either discrete or binary data for the Poisson or negative binomial and binary logistic regression respectively. Otherwise it is a vector with discrete values or factor values for the multinomial regression. If the vector is binary and choose multinomial regression the function checks and transfers to the binary logistic regression.
x	A matrix with data, the predictor variables.
oiko	This can be either \"poisson\" or \"binomial\". If you are not sure leave it NULL and the function will check internally.
logged	A boolean variable; it will return the logarithm of the pvalue if set to TRUE.

Details

Instead of maximising the log-likelihood via the Newton-Raphson algorithm in order to perform the hypothesis testing that $\beta_i = 0$ we use the score test. This is dramatcially faster as no model needs to be fitted. The first derivative (score) of the log-likelihood is known and in closed form and under the null hypothesis the fitted values are all equal to the mean of the response variable y. The variance of the score is also known in closed form. The test is not the same as the likelihood ratio test. It is size correct nonetheless but it is a bit less efficient and less powerful. For big sample sizes though (5000 or more) the results are the same. It is also much faster then the classical log-likelihood ratio test.

Value

A matrix with two columns, the test statistic and its associated p-value. For the Poisson and logistic regression the p-value is derived via the t distribution, whereas for the multinomial regressions via the χ^2 distribution.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Campbell, M.J. (2001). Statistics at Square Two: Understand Modern Statistical Applications in Medicine, pg. 112. London, BMJ Books.

Draper, N.R. and Smith H. (1988). Applied regression analysis. New York, Wiley, 3rd edition.

McCullagh, Peter, and John A. Nelder. Generalized linear models. CRC press, USA, 2nd edition, 1989.

Agresti Alan (1996). An introduction to categorical data analysis. New York: Wiley.

Joseph M.H. (2011). Negative Binomial Regression. Cambridge University Press, 2nd edition.

See Also

univglms, logistic_only, poisson_only, regression

Examples

```
x <- matrix( rnorm(500 * 500), ncol = 500 )
y <- rbinom(500, 1, 0.6)   ## binary logistic regression
a1 <- univglms(y, x)
system.time( score.glms(y, x) )
a2 <- score.glms(y, x)
cor(a1, a2)
mean(a1 - a2)
```

Many score based regression models

Many score based regression models.

Description

Many score based regression models.

Usage

```
score.weibregs(y, x, logged = FALSE)
score.betaregs(y, x, logged = FALSE)
score.gammaregs(y, x, logged = FALSE)
score.expregs(y, x, logged = FALSE)
score.invgaussregs(y, x, logged = FALSE)
score.ztpregs(y, x, logged = FALSE)
score.geomregs(y, x, logged = FALSE)
```

Arguments

y	A vector with data. For the Weibull, gamma and exponential regressions they must be strictly positive data, lifetimes or durations for example. For the beta regression they must be numbers between 0 and 1. For the zero truncated Poisson regression (score.ztpregs) they must be integer valued data strictly greater than 0.
x	A matrix with data, the predictor variables.
logged	A boolean variable; it will return the logarithm of the pvalue if set to TRUE.

Details

Instead of maximising the log-likelihood via the Newton-Raphson algorithm in order to perform the hypothesis testing that $\beta_i = 0$ we use the score test. This is dramatcially faster as no model need to be fitted. The first derivative of the log-likelihood is known in closed form and under the null hypothesis the fitted values are all equal to the mean of the response variable y. The test is not the same as the likelihood ratio test. It is size correct nonetheless but it is a bit less efficient and less powerful. For big sample sizes though (5000 or more) the results are the same. You cna try for yourselves and see that even with 500 the results are pretty close. The score test is also very faster then the classical likelihood ratio test.

What we have seen via simulation studies is that it is size correct to large sample sizes, at elast a few thousands.

Value

A matrix with two columns, the test statistic and its associated p-value.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

McCullagh, Peter, and John A. Nelder. Generalized linear models. CRC press, USA, 2nd edition, 1989.

Campbell, M.J. (2001). Statistics at Square Two: Understand Modern Statistical Applications in Medicine, pg. 112. London, BMJ Books.

See Also

`score.glms univglms`, `logistic_only`, `poisson_only`, `regression`

Examples

```
x <- matrix( rnorm(500 * 200), ncol = 200 )
y <- rweibull(500, 2, 3)
a <- score.weibregs(y, x)
hist(a[, 2])
sum(a[, 2] < 0.05) / 200
```

`Many Shapiro-Francia normality tests`

Many Shapiro-Francia normality tests

Description

Many Shapiro-Francia normality tests.

Usage

```
sftests(x, logged = FALSE)
sftest(x, logged = FALSE)
```

Arguments

x	A matrix with the data, where the rows denote the observations and the columns are the variables. In the case of a single sample, then this must be a vector and "sftest" is to be used.
logged	Should the p-values be returned (FALSE) or their logarithm (TRUE)?

Details

The Shapiro-Francia univariate normality test is performed for each column (variable) of the matrix x.

Value

A matrix with the squared correlation between the ordered values and the standard normal ordered statistics, the test statistic and the p-value of each test. If the "sftest" has been used, the output is a vector with these three elements.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Royston J. P. (1983). A simple method for evaluating the Shapiro-Francia W' test of non-normality. The Statistician, 32(3): 297-300.

Mbah A. K. & Paothong A. (2015). Shapiro-Francia test compared to other normality test using expected p-value. Journal of Statistical Computation and Simulation, 85(15): 3002-3016.

See Also

ttests, ttest, ftests

Examples

```
x <- matrix( rnorm(200 * 100), ncol = 100 )
system.time( sftests(x) )
a <- sftests(x)
hist(a[, 3])
x <- rnorm(100)
sftest(x)
```

`Many simple linear regressions coefficients`
Simple linear regressions coefficients

Description

Simple linear regressions coefficients.

Usage

```
allbetas(y, x, pvalue = FALSE, logged = FALSE)
```

Arguments

`y`	A numerical vector with the response variable. If the y contains proportions or percentages, i.e. values between 0 and 1, the logit transformation is applied first and the transformed data are used.
`x`	A matrix with the data, where rows denotes the observations and the columns contain the independent variables.
`pvalue`	If you want a hypothesis test that each slope (beta coefficient) is equal to zero set this equal to TRUE. It will also produce all the correlations between y and x.
`logged`	A boolean variable; it will return the logarithm of the pvalue if set to TRUE.

Value

A matrix with the constant (alpha) and the slope (beta) for each simple linear regression. If the p-value is set to TRUE, the correlation of each y with the x is calculated along with the relevant test statistic and its associated p-value.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

See Also

`mvbetas`, `correls`, `univglms`, `colsums`, `colVars`

Examples

```
x <- matrix( rnorm(100 * 500), ncol = 500 )
y <- rnorm(100)
r <- cor(y, x)  ## correlation of y with each of the xs
a <- allbetas(y, x)  ## the coefficients of each simple linear regression of y with x
```

`Many tests for the dispersion parameter in Poisson distribution`
Many tests for the dispersion parameter in Poisson distribution

Description

Many tests for the dispersion parameter in Poisson distribution.

Usage

```
colpoisdisp.tests(y, alternative = "either", logged = FALSE)
colpois.tests(y, logged = FALSE)
```

Arguments

`y`	A numerical matrix with count data, 0, 1,...
`alternative`	Do you want to test specifically for either over or underspirsion ("either"), overdispersion ("over") or undersispersion ("under")?
`logged`	Set to TRUE if you want the logarithm of the p-value.

Value

A matrix with two columns, the test statistic and the (logged) p-value.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Yang Zhao, James W. Hardin, and Cheryl L. Addy. (2009). A score test for overdispersion in Poisson regression based on the generalized Poisson-2 model. Journal of statistical planning and inference 139(4):1514-1521.

Dimitris Karlis and Evdokia Xekalaki (2000). A Simulation Comparison of Several Procedures for Testing the Poisson Assumption. Journal of the Royal Statistical Society. Series D (The Statistician), 49(3): 355-382.

Bohning, D., Dietz, E., Schaub, R., Schlattmann, P. and Lindsay, B. (1994) The distribution of the likelihood ratio for mixtures of densities from the one-parameter exponential family. Annals of the Institute of Statistical Mathematics, 46(): 373-388.

See Also

`poisson.mle`, `negbin.mle`, `poisson.anova`, `poisson.anovas`, `poisson_only`

Examples

```
y <- matrix(rnbinom(100* 100, 10, 0.6), ncol = 100)
a1 <- colpoisdisp.tests(y, "over")
b1 <- colpois.tests(y)

y <- matrix(rpois(100* 100, 10), ncol = 100)
a2 <- colpoisdisp.tests(y, "either")
b2 <- colpois.tests(y)
```

`Many two-way ANOVAs` *Many two-way ANOVAs*

Description

Many two-way ANOVAs.

Usage

```
twoway.anovas(y, x1, x2, interact = FALSE, logged = FALSE)
```

Arguments

`y`	A matrix with the data, where the rows denote the observations (and the two groups) and the columns are the variables.
`x1`	A numerical vector with 1s, 2s, 3s and so one indicating the two groups. Alternatively it can be a factor variable. This is the one factor.
`x2`	A numerical vector with 1s, 2s, 3s and so one indicating the two groups. Alternatively it can be a factor variable. This is the other factor.
`interact`	A boolean variable specifying whether you want to test for interaction.
`logged`	Should the p-values be returned (FALSE) or their logarithm (TRUE)?

Details

The classical two-way ANOVA design is performed. Note that the design must be balanced. For every combination of values of the two factors, x1 and x2 the same number of observations must exist. If that's not the case, regression models must be used.

Value

A matrix with the test statistic and the p-value of each test.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

D.C. Montgomery (2001). Design and analysis of experiments (5th Edition). New York: John Wiley \& Sons

See Also

ancovas, ftests, ttests

Examples

```
y <- as.matrix( rnorm(125) )
x1 <- rep(1:5, 25)
x2 <- rep(1:5, each = 25)
x1 <- factor(x1)
x2 <- factor(x2)
anova( lm(y ~ x1 + x2) )
twoway.anovas(y, x1, x2)
anova( lm(y ~ x1*x2) )
twoway.anovas(y, x1, x2, interact = TRUE)
y <- matrix(rnorm(125 * 1000), ncol = 1000)
system.time( a1 <- twoway.anovas(y, x1, x2) )
system.time( a2 <- twoway.anovas(y, x1, x2, interact = TRUE) )
```

Many univariate generalised linear models

Many univariate generalised linear regressions

Description

It performs very many univariate generalised linear regressions.

Usage

```
univglms(y, x, oiko = NULL, logged = FALSE)
```

Arguments

y	The dependent variable. It can be a factor or a numerical variable with two values only (binary logistic regression), a discrete valued vector (count data) corresponding to a poisson regression or a numerical vector with continuous values (normal regression). If it contains percentages or proportions (values between 0 and 1) they are transformed in R using $logy/log(1-y)$ and linear regression is applied.
x	A matrix with the data, where the rows denote the samples (and the two groups) and the columns are the variables. Currently only continuous variables are allowed. You are advised to standardise the data before hand to avoid numerical overflow or similar issues. If you see NaN in the outcome, this is the case.

`oiko` This can be either "normal", "poisson" or "binomial". If you are not sure leave it NULL and the function will check internally. However, you might have discrete data (e.g. years of age) and want to perform many simple linear regressions. In this case you should specify the family.

`logged` A boolean variable; it will return the logarithm of the pvalue if set to TRUE.

Details

If you specify no family of distributions the function internally checkes the type of your data and decides on the type of regression to perform. The function is written in C++ and this is why it is very fast. It can accept thousands of predictor variables. It is usefull for univariate screening. We provide no p-value correction (such as fdr or q-values); this is up to the user.

Value

A matrix with the test statistic and the p-value for each predictor variable.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Draper, N.R. and Smith H. (1988). Applied regression analysis. New York, Wiley, 3rd edition.

McCullagh, Peter, and John A. Nelder. Generalized linear models. CRC press, USA, 2nd edition, 1989.

See Also

`logistic_only`, `poisson_only`, `allbetas`, `correls`, `regression`

Examples

```
x = matrix( rnorm(100 * 200), ncol = 200 )
## 100 observations in total
y = rbinom(100, 1, 0.6)   ## binary logistic regression
system.time( univglms(y, x) )
a1 = univglms(y, x)
a2 <- numeric(200)
system.time( for (i in 1:200) a2[i] = glm(y ~ x[, i], binomial)$deviance )

a2 = glm(y ~ 1, binomial)$null.dev - a2
```

Many univariate simple binary logistic regressions
Many univariate simple binary logistic regressions

Description

It performs very many univariate simple binary logistic regressions.

Usage

```
logistic_only(x, y, tol = 1e-09, b_values = FALSE)
```

Arguments

x	A matrix with the data, where the rows denote the samples (and the two groups) and the columns are the variables. Currently only continuous variables are allowed.
y	The dependent variable; a numerical vector with two values (0 and 1).
tol	The tolerance value to terminate the Newton-Raphson algorithm.
b_values	Do you want the values of the coefficients returned? If yes, set this to TRUE.

Details

The function is written in C++ and this is why it is very fast. It can accept thousands of predictor variables. It is usefull for univariate screening. We provide no p-value correction (such as fdr or q-values); this is up to the user.

Value

A vector with the deviance of each simple binayr logistic regression model for each predictor variable.

Author(s)

Manos Papadakis <papadakm95@gmail.com>

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

McCullagh, Peter, and John A. Nelder. Generalized linear models. CRC press, USA, 2nd edition, 1989.

See Also

allbetas, correls, poisson_only, regression

Examples

```
## 300 variables, hence 300 univariate regressions are to be fitted
x = matrix( rnorm(100 * 300), ncol = 300 )

## 100 observations in total
y = rbinom(100, 1, 0.6)   ## binary logistic regression
system.time( logistic_only(x, y) )
a1 = logistic_only(x, y)

a2 <- numeric(300)
system.time( for (i in 1:300) a2[i] = glm(y ~ x[, i], binomial)$deviance )
a2 = as.vector(a2)
all.equal(a1, a2)
```

`Many univariate simple linear regressions`
Many univariate simple linear regressions

Description

It performs very many univariate simple linear regressions with or without categorical variables.

Usage

```
regression(x, y)
```

Arguments

x	A data.frame with the data, where the rows denote the samples (and the two groups) and the columns are the variables. Currently only continuous variables are allowed. A data frame is expected if you have categorical predictor variables. If you only have continuous predictor variables you should the function `allbetas` instead as it is faster.
y	The dependent variable; a numerical vector.

Details

The function is written in C++ and this is why it is very fast. It can accept thousands of predictor variables. It is usefull for univariate screening. We provide no p-value correction (such as fdr or q-values); this is up to the user. The user must take the F test statistics along with the numertor degrees and calculate the relevant p-values. We provide the degrees of freedom of the numerator in the F test. The degrees of freedom in the denominator are always n - df of the numerator.

Value

A matrix with two rows. The first row is the F statistic and the second row is the degrees of freedom of the numerator in the F test. That is the number of estimated parameters. For example if some variables are continuous, the number of parameters is 1. In the case of a categorical variable with D distinct values (or levels) the number of parameters is D-1.

Author(s)

Manos Papadakis <papadakm95@gmail.com>

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Draper, N.R. and Smith H. (1988). Applied regression analysis. New York, Wiley, 3rd edition.

McCullagh, Peter, and John A. Nelder. Generalized linear models. CRC press, USA, 2nd edition, 1989.

See Also

univglms, allbetas, correls, univglms, mvbetas

Examples

```
## 500 variables, hence 500 univariate regressions are to be fitted
x = matrix( rnorm(100 * 200), ncol = 200 )
x = as.data.frame(x)
y = rnorm(100)
system.time( a1 <- regression(x, y) )

x = matrix( rnorm(150 * 200), ncol = 200 )
x = as.data.frame(x)
y = rnorm(150)
for (i in 1:200) x[, i] <- iris[, 5]
system.time( a <- regression(x, y) )
a[, 1:5]
summary(lm(y ~ x[, 1]) )  ## check the F-test
```

`Many univariate simple poisson regressions`

Many univariate simple poisson regressions

Description

It performs very many univariate simple poisson regressions.

Usage

```
poisson_only(x, y, tol = 1e-09, b_values = FALSE)
```

Arguments

x	A matrix with the data, where the rows denote the samples (and the two groups) and the columns are the variables. Currently only continuous variables are allowed.
y	The dependent variable; a numerical vector with many discrete values (count data).
tol	The tolerance value to terminate the Newton-Raphson algorithm.
b_values	Do you want the values of the coefficients returned? If yes, set this to TRUE.

Details

The function is written in C++ and this is why it is very fast. It can accept thousands of predictor variables. It is usefull for univariate screening. We provide no p-value correction (such as fdr or q-values); this is up to the user.

Value

A vector with the deviance of each simple poisson regression model for each predictor variable.

Author(s)

Manos Papadakis <papadakm95@gmail.com>

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

McCullagh, Peter, and John A. Nelder. Generalized linear models. CRC press, USA, 2nd edition, 1989.

See Also

univglms, logistic_only, allbetas, regression

Examples

```
## 200 variables, hence 200 univariate regressions are to be fitted
x = matrix( rnorm(100 * 200), ncol = 200 )

y = rpois(100, 10)
system.time( poisson_only(x, y) )

b1 = poisson_only(x, y)

b2 = numeric(500)
```

```
system.time( for (i in 1:200) b2[i] = glm(y ~ x[, i], poisson)$deviance )

all.equal(b1, b2)
```

Match *Match*

Description

Return the positions of its first argument that matches in its second.

Usage

```
Match(x,key)
```

Arguments

x	A numeric vector.
key	The value/vector for searching in vector x.

Details

This function implements the R's \"match\" function.

Value

Returns the position/positions of the given key/keys in the x vector.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>

See Also

match

Examples

```
y <- rnorm(100)
a <- Match(y,50)
b <-50
all.equal(as.vector(a),as.vector(b))
```

`Matrix with all pairs of t-tests`
Matrix with all pairs of t-tests

Description

Matrix with all pairs of t-tests.

Usage

```
allttests(x, y = NULL, ina, logged = FALSE)
ttests.pairs(x, logged = FALSE)
```

Arguments

`x`	A numerical matrix with the data.
`y`	For the case of \"all.tests\", if you have the second group or sample provide it here, otherwise leave it NULL. For the case of \"ttests.pairs\" this is not required.
`ina`	If you have the data in one matric then provide this indicator variable separating the samples. This numerical vector must contain 1s and 2s only as values. For the case of \"ttests.pairs\" this is not required.
`logged`	Should the p-values be returned (FALSE) or their logarithm (TRUE)?

Details

The function does all the pairwise t-tests assuming unequal variances (Welch's t-test). The \"all.ttests\" does all the pairs formed by \"cutting\" the matrices x and y in two and everything between them. The \"ttests.pairs\" accepts a matrix x and does all the pairs of t-tests. This is similar to the correlation matrix style.

Value

A list including:

`stat`	A matrix with t-test statistic for each pair of variables.
`pvalue`	A matrix with the corresponding p-values.
`dof`	A matrix with the relevant degrees of freedom.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

See Also

`ttests`, `ftests`, `ttest`, `g2Test_univariate`

Examples

```
x <- as.matrix( iris[1:100, 1:4] )
ina <- as.numeric(iris[1:100, 5])
a <- allttests(x, ina = ina)
b <- ttests.pairs(x)  ## less tests
```

`Matrix with G-square tests of indepedence`

Matrix with G-square tests of indepdence

Description

Matrix with G-square tests of indepdence with and without permutations.

Usage

```
g2Test_univariate(data, dc)

g2Test_univariate_perm(data, dc, nperm)
```

Arguments

`data`	A numerical matrix with the data. **The minimum must be 0, otherwise the function can crash or will produce wrong results**.
`dc`	A numerical value equal to the number of variables (or columns of the data matrix) indicating the number of distinct, unique values (or levels) of each variable. Make sure you give the correct numbers here, otherwise the degrees of freedom will be wrong.
`nperm`	The number of permutations. The permutations test is slower than without permutations and should be used with small sample sizes or when the contigency tables have zeros. When there are few variables, R's "chisq.test" function is faster, but as the number of variables increase the time difference with R's procedure becomes larger and larger.

Details

The function does all the pairwise G^2 test of independence and gives the position inside the matrix. The user must build the associations matrix now, similarly to the correlation matrix. See the examples of how to do that. The p-value is not returned, we live this to the user. See the examples of how to obtain it.

Value

A list including:

`statistic`	The G^2 test statistic for each pair of variables.
`pvalue`	This is returned when you have selected the permutation based G^2 test.

x	The row or variable of the data.
y	The column or variable of the data.
df	The degrees of freedom of each test.

Author(s)

Giorgos Borboudakis. The permutation version used a C++ code by John Burkardt.

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

References

Tsagris M. (2017). Conditional independence test for categorical data using Poisson log-linear model. Journal of Data Science, 15(2):347-356.

Tsamardinos, I., & Borboudakis, G. (2010). Permutation testing improves Bayesian network learning. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases (pp. 322-337). Springer Berlin Heidelberg

See Also

g2Test, g2Test_perm, correls, univglms

Examples

```
nvalues <- 3
nvars <- 10
nsamples <- 2000
data <- matrix( sample( 0:(nvalues - 1), nvars * nsamples, replace = TRUE ), nsamples, nvars )
dc <- rep(nvalues, nvars)
system.time( g2Test_univariate(data = data, dc = dc) )
a <- g2Test_univariate(data = data, dc = dc)
pval <- pchisq(a$statistic, a$df, lower.tail = FALSE)

g <- matrix(0, nvars, nvars)
g[ cbind(a$x, a$y) ] <- a$statistic
g <- g + t(g)
diag(g) <- 0
g  ## matrix of G^2 test statistics
```

Median of a vector *Median of a vector*

Description

Median of a vector.

Usage

```
med(x)
```

Arguments

x	A numerical vector.

Details

The function is written in C++ and this is why it is very fast.

Value

The median of the vector of a numbers.

Author(s)

Manos Papadakis <papadakm95@gmail.com>

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

See Also

nth, colMedians

Examples

```
x <- rnorm(1000)
system.time( for (i in 1:100) med(x) )
system.time( for (i in 1:100) median(x) )
```

Minima and maxima of two vectors

Minima and maxima of two vectors

Description

Minima and maxima of two vectors.

Usage

```
Pmax(x, y)
Pmin(x, y)
```

Arguments

x	A numverical vector with numbers.
y	A numverical vector with numbers. The lengths of these two vectorsa must match.

Details

The parallel minima or maxima are returned. This are the same as the base functions pmax and pmin.

Value

A numerical vector with numbers, whose length is equal to the length of the initital vectors containing the maximum or minimum between each pair.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

sort_mat, Sort, colMins

Examples

```
x <- rnorm(10)
y <- rnorm(10)
Pmax(x, y)
pmax(x, y)
Pmin(x, y)
pmin(x, y)
```

---

minimum and maximum *Minimum and maximum of a vector*

Description

Minimum and maximum of a vector.

Usage

```
min_max(x,index=FALSE, percent = FALSE)
```

Arguments

x	A numerical vector with data. NAs are handled naturally.
index	A boolean value for the indices of the minimum and the maximum value.
percent	A boolean value for the percent of the positive and negative numbers.

Value

A vector with the relevant values, min and max.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

rowMins, rowMaxs, nth, colrange, colMedians,sort_mat

Examples

```
x <- rnorm(100 * 500)
s1 <- min_max(x)
s2 <- c(min(x), max(x))
```

MLE for multivariate discrete data

MLE for multivariate discrete data

Description

MLE for multivariate discrete data.

Usage

```
multinom.mle(x)
colpoisson.mle(x)
colgeom.mle(x, type = 1)
```

Arguments

x	A matrix with discrete valued non negative data.
type	This is for the geometric distribution only. Type 1 refers to the case where the minimum is zero and type 2 for the case of the minimum being 1.

Details

For the Poisson and geometric distributions we simply fit independent Poisson and geometric distributions respectively.

Value

A list including:

loglik	A vector with the value of the maximised log-likelihood.
param	A vector of the parameters.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Johnson Norman L., Kotz Samuel and Balakrishnan (1997). Discrete Multivariate Distributions. Wiley

See Also

poisson.mle, zip.mle, ztp.mle, negbin.mle, poisson.nb

Examples

```
x <- t( rmultinom(1000, 20, c(0.4, 0.5, 0.1) ) )
multinom.mle(x)
colpoisson.mle(x)
```

`MLE of (hyper-)spherical distributions`

MLE of (hyper-)spherical distributions

Description

MLE of (hyper-)spherical distributions.

Usage

```
vmf.mle(x, tol = 1e-07)
multivmf.mle(x, ina, tol = 1e-07, ell = FALSE)
acg.mle(x, tol = 1e-07)
iag.mle(x, tol = 1e-07)
```

Arguments

x	A matrix with directional data, i.e. unit vectors.
ina	A numerical vector with discrete numbers starting from 1, i.e. 1, 2, 3, 4,... or a factor variable. Each number denotes a sample or group. If you supply a continuous valued vector the function will obviously provide wrong results.
ell	This is for the multivmf.mle only. Do you want the log-likelihood returned? The default value is TRUE.
tol	The tolerance value at which to terminate the iterations.

Details

For the von Mises-Fisher, the normalised mean is the mean direction. For the concentration parameter, a Newton-Raphson is implemented. For the angular central Gaussian distribution there is a constraint on the estimated covariance matrix; its trace is equal to the number of variables. An iterative algorithm takes place and convergence is guaranteed. Newton-Raphson for the projected normal distribution, on the sphere, is implemented as well. Finally, the von Mises-Fisher distribution for groups of data is also implemented.

Value

For the von Mises-Fisher a list including:

`loglik`	The maximum log-likelihood value.
`mu`	The mean direction.
`kappa`	The concentration parameter.

For the multi von Mises-Fisher a list including:

`loglik`	A vector with the maximum log-likelihood values if ell is set to TRUE. Otherwise NULL is returned.
`mi`	A matrix with the group mean directions.
`ki`	A vector with the group concentration parameters.

For the angular central Gaussian a list including:

`iter`	The number if iterations required by the algorithm to converge to the solution.
`cova`	The estimated covariance matrix.

For the spherical projected normal a list including:

`iters`	The number of iteration required by the Newton-Raphson.
`mesi`	A matrix with two rows. The first row is the mean direction and the second is the mean vector. The first comes from the second by normalising to have unit length.
`param`	A vector with the elements, the norm of mean vector, the log-likelihood and the log-likelihood of the spherical uniform distribution. The third value helps in case you want to do a log-likleihood ratio test for uniformity.

Author(s)

Michail Tsagris R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>

References

Mardia, K. V. and Jupp, P. E. (2000). Directional statistics. Chicester: John Wiley & Sons.

Sra, S. (2012). A short note on parameter approximation for von Mises-Fisher distributions: and a fast implementation of Is(x). Computational Statistics, 27(1): 177–190.

Tyler D. E. (1987). Statistical analysis for the angular central Gaussian distribution on the sphere. Biometrika 74(3): 579-589.

Paine P.J., Preston S.P., Tsagris M and Wood A.T.A. (2017). An Elliptically Symmetric Angular Gaussian Distribution. Statistics and Computing (To appear).

See Also

racg, vm.mle, rvmf

Examples

```
m <- c(0, 0, 0, 0)
s <- cov(iris[, 1:4])
x <- racg(100, s)
mod <- acg.mle(x)
mod
cov2cor(mod$cova)  ## estimated covariance matrix turned into a correlation matrix
cov2cor(s)  ## true covariance matrix turned into a correlation matrix
vmf.mle(x)
x <- rbind( rvmf(100,rnorm(4), 10), rvmf(100,rnorm(4), 20) )
a <- multivmf.mle(x, rep(1:2, each = 100) )
```

MLE of continuous univariate distributions defined on the positive line

MLE of continuous univariate distributions defined on the positive line

---

Description

MLE of continuous univariate distributions defined on the positive line.

Usage

```
gammamle(x, tol = 1e-09)
chisq.mle(x, tol = 1e-09)
weibull.mle(x, tol = 1e-09)
lomax.mle(x, tol = 1e-09)
foldnorm.mle(x, tol = 1e-09)
betaprime.mle(x, tol = 1e-09)
logcauchy.mle(x, tol = 1e-09)
loglogistic.mle(x, tol = 1e-09)
halfnorm.mle(x)
invgauss.mle(x)
lognorm.mle(x)
pareto.mle(x)
expmle(x)
maxboltz.mle(x)
rayleigh.mle(x)
normlog.mle(x)
lindley.mle(x)
```

Arguments

x	A vector with positive valued data (zeros are not allowed).
tol	The tolerance level up to which the maximisation stops; set to 1e-09 by default.

Details

Instead of maximising the log-likelihood via a numerical optimiser we have used a Newton-Raphson algorithm which is faster. See wikipedia for the equations to be solved. For the t distribution we need the degrees of freedom and estimate the location and scatter parameters. If you want to to fit an inverse gamma distribution simply do \"gamma.mle(1/x)\". The log-likelihood and the parameters are for the inverse gamma.

The \"normlog.mle\" is simply the normal distribution where all values are positive. Note, this is not log-normal. It is the normal with a log link. Similarly to the inverse gaussian distribution where the mean is an exponentiated. This comes from the GLM theory.

Value

Usually a list with three elements, but this is not for all cases.

iters	The number of iterations required for the Newton-Raphson to converge.
loglik	The value of the maximised log-likelihood.
param	The vector of the parameters.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Kalimuthu Krishnamoorthy, Meesook Lee and Wang Xiao (2015). Likelihood ratio tests for comparing several gamma distributions. Environmetrics, 26(8):571-583.

N.L. Johnson, S. Kotz \& N. Balakrishnan (1994). Continuous Univariate Distributions, Volume 1 (2nd Edition).

N.L. Johnson, S. Kotz \& N. Balakrishnan (1970). Distributions in statistics: continuous univariate distributions, Volume 2

Tsagris M., Beneki C. and Hassani H. (2014). On the folded normal distribution. Mathematics, 2(1):12-28.

Sharma V. K., Singh S. K., Singh U. \& Agiwal V. (2015). The inverse Lindley distribution: a stress-strength reliability model with application to head and neck cancer data. Journal of Industrial and Production Engineering, 32(3): 162-173.

You can also check the relevant wikipedia pages for these distributions.

See Also

zip.mle, normal.mle, beta.mle

Examples

```
x <- rgamma(1000, 3, 4)
system.time( for (i in 1:50) gammamle(x) )
##system.time( for (i in 1:50) fitdistr(x,"gamma") )

a <- glm(x ~ 1, gaussian(log) )
normlog.mle(x)
```

`MLE of continuous univariate distributions defined on the real line`
MLE of continuous univariate distributions defined on the real line

---

Description

MLE of continuous univariate distributions defined on the real line.

Usage

```
normal.mle(x)
gumbel.mle(x, tol = 1e-09)
cauchy.mle(x, tol = 1e-09)
logistic.mle(x, tol = 1e-07)
ct.mle(x, tol = 1e-09)
tmle(x, v = 5, tol = 1e-08)
wigner.mle(x, tol = 1e-09)
laplace.mle(x)
```

Arguments

`x`	A numerical vector with data.
`v`	The degrees of freedom of the t distribution.
`tol`	The tolerance level up to which the maximisation stops set to 1e-09 by default.

Details

Instead of maximising the log-likelihood via a numerical optimiser we have used a Newton-Raphson algorithm which is faster. See wikipedia for the equation to be solved. For the t distribution we need the degrees of freedom and estimate the location and scatter parameters.

The Cauchy is the t distribution with 1 degree of freedom. If you want to fit such a distribution used the cauchy.mle and not the t.mle with 1 degree of freedom as it's faster. The Laplace distribution is also called double exponential distribution.

Value

Usually a list with three elements, but this is not for all cases.

`iters`	The number of iterations required for the Newton-Raphson to converge.
`loglik`	The value of the maximised log-likelihood.
`param`	The vector of the parameters.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Johnson, Norman L. Kemp, Adrianne W. Kotz, Samuel (2005). Univariate Discrete Distributions (third edition). Hoboken, NJ: Wiley-Interscience.

See Also

`zip.mle`, `gammamle`, `vm.mle`

Examples

```
x <- rt(1000,10)
a <- ct.mle(x)
tmle(x, v = a$nu)
cauchy.mle(x)
normal.mle(x)
logistic.mle(x)

n <- 100
x <- 5 - 0.005 *log( - log (runif(n) ) )
gumbel.mle(x)
```

`MLE of count data (univariate discrete distributions)`

MLE of count data

Description

MLE of count data.

Usage

```
zip.mle(x, tol = 1e-09)
ztp.mle(x, tol = 1e-09)
negbin.mle(x, type = 1, tol = 1e-09)
binom.mle(x, N = NULL, tol = 1e-07)
borel.mle(x)
geom.mle(x, type = 1)
logseries.mle(x, tol = 1e-09)
poisson.mle(x)
```

Arguments

x	A vector with discrete valued data.
type	This argument is for the negative binomial and the geometric distribution. In the negative binomial you can choose which way your prefer. Type 1 is for smal sample sizes, whereas type 2 is for larger ones as is faster. For the geometric it is related to its two forms. Type 1 refers to the case where the minimum is zero and type 2 for the case of the minimum being 1.
N	This is for the binomial distribution only, specifying the total number of successes. If NULL, it is sestimated by the data.
tol	The tolerance level up to which the maximisation stops set to 1e-09 by default.

Details

Instead of maximising the log-likelihood via a numerical optimiser we used a Newton-Raphson algorithm which is faster.

See wikipedia for the equation to be solved in the case of the zero inflated distribution. https://en.wikipedia.org/wiki/Zero-inflated_model. In order to avoid negative values we have used link functions, log for the $lambda$ and logit for the π as suggested by Lambert (1992). As for the zero truncated Poisson see https://en.wikipedia.org/wiki/Zero-truncated_Poisson_distribution.

zip.mle is for the zero inflated Poisson, whereas ztp.mle is for the zero truncated Poisson distribution.

Value

A list including:

mess	This is for the negbin.mle only. If there is no reason to use the negative binomial distribution a message will appear, otherwise this is NULL.
iters	The number of iterations required for the Newton-Raphson to converge.
loglik	The value of the maximised log-likelihood.
prob	The probability parameter of the distribution. In some distributions this argument might have a different name. For example, param in the zero inflated Poisson.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Lambert Diane (1992). Zero-Inflated Poisson Regression, with an Application to Defects in Manufacturing. Technometrics. 34 (1): 1-14

Johnson Norman L., Kotz Samuel and Kemp Adrienne W. (1992). Univariate Discrete Distributions (2nd ed.). Wiley

See Also

poisson_only, colrange

Examples

```
x <- rpois(100, 2)
zip.mle(x)
poisson.mle(x)
## small difference in the two log-likelihoods as expected.

x <- rpois(100, 10)
x[x == 0 ] <- 1
ztp.mle(x)
poisson.mle(x)
## significant difference in the two log-likelihoods.

x <- rnbinom(100, 10, 0.6)
poisson.mle(x)
negbin.mle(x)
```

MLE of distributions defined in the (0, 1) interval

MLE of distributions defined in the (0, 1) interval

Description

MLE of distributions defined in the (0, 1) interval.

Usage

```
beta.mle(x, tol = 1e-09)
ibeta.mle(x, tol = 1e-09)
logitnorm.mle(x)
hsecant01.mle(x, tol = 1e-09)
```

Arguments

x	A numerical vector with proportions, i.e. numbers in (0, 1) (zeros and ones are not allowed).
tol	The tolerance level up to which the maximisation stops.

Details

Maximum likelihood estimation of the parameters of the beta distribution is performed via Newton-Raphson. The distributions and hence the functions does not accept zeros. \"logitnorm.mle\" fits the logistic normal, hence no nwewton-Raphson is required and the \"hypersecant01.mle\" uses the golden ratio search as is it faster than the Newton-Raphson (less calculations)

Value

A list including:

iters	The number of iterations required by the Newton-Raphson.
loglik	The value of the log-likelihood.
param	The estimated parameters. In the case of \"hypersecant01.mle\" this is called \"theta\" as there is only one parameter.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>

See Also

diri.nr2,

Examples

```
x <- rbeta(1000, 1, 4)
system.time( for(i in 1:1000) beta.mle(x) )
beta.mle(x)
ibeta.mle(x)

x <- runif(1000)
hsecant01.mle(x)
logitnorm.mle(x)
ibeta.mle(x)

x <- rbeta(1000, 2, 5)
x[sample(1:1000, 50)] <- 0
ibeta.mle(x)

## Newton Raphson for the beta distribution
## We use the diri.nr2 function though
```

```
# beta.mle <- function (x, tol = 1e-09) {
#   n <- length(x)
#   sly1 <- sum(log(x))
#   sly2 <- sum(log(1 - x))
#   sy <- sum(x)
#   sy2 <- sum(x^2)
#   iniphi <- (sy - sy2)/(sy2 - sy^2/n) * (n - 1)/n
#   a <- sum(x) * iniphi/n
#   b <- iniphi - a
#   phi <- a + b
#   derab <- n * trigamma(phi)
#   dera <- n * digamma(phi) - n * digamma(a) + sly1
#   dera2 <- n * trigamma(phi) - n * trigamma(a)
#   derb <- n * digamma(phi) - n * digamma(b) + sly2
#   derb2 <- n * trigamma(phi) - n * trigamma(b)
#   aold <- c(a, b)
#   anew <- aold - c(derb2 * dera - derab * derb, -derab * dera +
#                      dera2 * derb)/(dera2 * derb2 - derab^2)
#   i <- 2
#   while (sum(abs(anew - aold)) > tol) {
#     i <- i + 1
#     aold <- anew
#     a <- anew[1]
#     b <- anew[2]
#     phi <- a + b
#     derab <- n * trigamma(phi)
#     dera <- n * digamma(phi) - n * digamma(a) + sly1
#     dera2 <- n * trigamma(phi) - n * trigamma(a)
#     derb <- n * digamma(phi) - n * digamma(b) + sly2
#     derb2 <- n * trigamma(phi) - n * trigamma(b)
#     anew <- aold - c(derb2 * dera - derab * derb, -derab *
#                        dera + dera2 * derb)/(dera2 * derb2 - derab^2)
#   }
#   a <- anew[1]
#   b <- anew[2]
#   loglik <- -n * lbeta(a, b) + (a - 1) * sly1 + (b - 1) * sly2
#   names(anew) <- c(\"alpha\", \"beta\")
#   list(iters = i, loglik = loglik, param = anew)
# }
```

MLE of some circular distributions

MLE of some circular distributions

Description

MLE of some circular distributions.

Usage

```
vm.mle(x, tol = 1e-09)
spml.mle(x, tol = 1e-09)
wrapcauchy.mle(x, tol = 1e-09)
```

Arguments

x	A numerical vector with the circular data. They must be expressed in radians.
tol	The tolerance level to stop the iterative process of finding the MLEs.

Details

The parameters of the von Mises, the bivariate angular Gaussian and wrapped Cauchy distributions are estimated. For the Wrapped Cauchy, the iterative procedure described by Kent and Tyler (1988) is used. As for the von Mises distribution, we use a Newton-Raphson to estimate the concentration parameter. The angular Gaussian is described, in the regression setting in Presnell et al. (1998).

Value

A list including:

iters	The iterations required until convergence. This is returned in the wrapped Cauchy distribution only.
loglik	The value of the maximised log-likelihood.
param	A vector consisting of the estimates of the two parameters, the mean direction for both distributions and the concentration parameter kappa and the rho for the von Mises and wrapped Cauchy respectively.
gamma	The norm of the mean vector of the angualr Gaussian distribution.
mu	The mean vector of the angular Gaussian distribution.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>

References

Mardia K. V. and Jupp P. E. (2000). Directional statistics. Chicester: John Wiley \& Sons.

Sra S. (2012). A short note on parameter approximation for von Mises-Fisher distributions: and a fast implementation of Is(x). Computational Statistics, 27(1): 177-190.

Presnell Brett, Morrison Scott P. and Littell Ramon C. (1998). Projected multivariate linear models for directional data. Journal of the American Statistical Association, 93(443): 1068-1077.

Kent J. and Tyler D. (1988). Maximum likelihood estimation for the wrapped Cauchy distribution. Journal of Applied Statistics, 15(2): 247–254.

See Also

vmf.mle, rvonmises, rvmf

Examples

```
y <- rcauchy(100, 3, 1)
x <- y
vm.mle(x)
spml.mle(x)
wrapcauchy.mle(x)
```

MLE of the inverted Dirichlet distribution

MLE of the inverted Dirichlet distribution

Description

MLE of the inverted Dirichlet distribution.

Usage

```
invdir.mle(x, tol = 1e-09)
```

Arguments

x	A matrix with strictly positive data (no zeros are allowed).
tol	The tolerance level up to which the maximisation stops.

Details

Maximum likelihood estimation of the parameters of the inverted is performed via Newton-Raphson. We took the initial values suggested by Bdiri T. and Bouguila N. (2012) and modified them a bit.

Value

A list including:

iters	The number of iterations required by the Newton Raphson.
loglik	The value of the log-likelihood.
param	The estimated parameters.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>

References

Bdiri T. and Bouguila N. (2012). Positive vectors clustering using inverted Dirichlet finite mixture models. Expert Systems with Applications, 39(2): 1869-1882.

See Also

`diri.nr2`, `multinom.mle`

Examples

```
x <- as.matrix(iris[, 1:4])
system.time( for(i in 1:100) invdir.mle(x) )
invdir.mle(x)
```

`MLE of the multivariate normal distribution`

MLE of the multivariate normal distribution

Description

MLE of the multivariate normal distribution.

Usage

```
mvnorm.mle(x)
```

Arguments

`x`	A matrix with numerical data.

Details

The mean vector, covariance matrix and the value of the log-likelihood is calculated.

Value

A list including:

`iters`	The number of iterations required for the Newton-Raphson to converge.
`loglik`	The value of the maximised log-likelihood.
`param`	The vector of the parameters for the zero inflated Poisson.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Johnson Norman L., Kotz Samuel and Balakrishnan (1997). Discrete Multivariate Distributions. Wiley

See Also

multinom.mle, dmvnorm, gaussian.nb

Examples

```
x <- matrix( rnorm(100 * 4), ncol = 4)
mvnorm.mle(x)
```

MLE of the ordinal model without covariates

Natural logarithm of the beta functionMLE of the ordinal model without covariates

---

Description

MLE of the ordinal model without covariates.

Usage

```
ordinal.mle(y, link = "logit")
```

Arguments

y	A numerical vector with values 1, 2, 3,..., not zeros, or an ordered factor.
link	This can either be "logit" or "probit". It is the link function to be used.

Details

Maximum likelihood of the ordinal model (proportional odds) is implemented. See for example the "polr" command in R or the examples.

Value

A list including:

loglik	The log-likelihood of the model.
a	The intercepts (threshold coefficients) of the model.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

References

Agresti, A. (2002) Categorical Data. Second edition. Wiley.

See Also

beta.mle, diri.nr2

Examples

```
y <- factor( rbinom(100,3,0.5), ordered = TRUE )
ordinal.mle(y)
ordinal.mle(y, link = "probit")
```

`MLE of the tobit model`

MLE of the tobit model

Description

MLE of the tobit model.

Usage

```
tobit.mle(y, tol = 1e-09)
```

Arguments

`y`	A vector with positive valued data and zero values. If there are no zero values, a simple normal model is fitted in the end.
`tol`	The tolerance level up to which the maximisation stops; set to 1e-09 by default.

Details

The tobin model is useful for (univariate) positive data with left censoring at zero. There is the assumption of a latent variable. Tthe values of that variable which are positive concide with the observed values. If some values are negative, they are left censored and the observed values are zero. Instead of maximising the log-likelihood via a numerical optimiser we have used a Newton-Raphson algorithm which is faster.

Value

A list with three elements including

`iters`	The number of iterations required for the Newton-Raphson to converge.
`loglik`	The value of the maximised log-likelihood.
`param`	The vector of the parameters.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Tobin James (1958). Estimation of relationships for limited dependent variables. Econometrica. 26(1):24–36.

https://en.wikipedia.org/wiki/Tobit_model

See Also

gammamle, normal.mle

Examples

```
x <- rnorm(500, 3, 5)
x[ x < 0 ] <- 0   ## left censoring. Values below zero become zero
system.time( for (i in 1:100) tobit.mle(x) )
```

Moment and maximum likelihood estimation of variance components

Moment and maximum likelihood estimation of variance components

---

Description

Moment and maximum likelihood estimation of variance components.

Usage

```
rint.mle(x, ina, ranef = FALSE, tol = 1e-09)
varcomps.mom(x, ina)
varcomps.mle(x, ina, tol = 1e-09)
```

Arguments

x	A numerical vector with the data.
ranef	Should the random effects be returned as well? The default value is FALSE.
ina	A numerical vector with 1s, 2s, 3s and so one indicating the two groups. Be careful, the function is desinged to accept numbers greater than zero. Alternatively it can be a factor variable.
tol	The tolerance level to terminate the golden ratio search. the default value is 10^(-9).

Details

Note that this formula works for **balanced designs only**, i.e. for each subject the same number of measurements have been taken.

The variance components, the variance of the between measurements and the variance of the within are estimated using moment estimators. The "colvarcomsp.mom" is the moment analogue of a random effects model which uses likelihood estimation ("colvarcomps.mle"). It is much faster, but can give negative variance of the random effects, in which case it becomes zero.

The maximum likelihood version is a bit slower (try youselves to see the difference), but statistically speaking is to be preferred when small samples are available. The reason why it is only a little bit slower and not a lot slower as one would imagine is because we are using a closed formula to calculate the two variance components (Demidenko, 2013, pg. 67-69). Yes, there are closed formulas for linear mixed models.

Value

For the "varcomps.mom": A vector with 5 elemets, The MSE, the estimate of the between variance, the variance components ratio and a 95% confidence for the ratio.

For the "varcomps.mle": a list with a single component called "info". That is a matrix with 3 columns, The MSE, the estimate of the between variance and the log-likelihood value. **If ranef = TRUE** a list including "info" and an extra component called "ranef" containing the random effects. It is a matrix with the same number of columns as the data. Each column contains the randome effects of each variable.

Author(s)

Michail Tsagris and Manos Papadakis

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

D.C. Montgomery (2001). Design and analysis of experiments (5th Edition). New York: John Wiley \& Sons.

Charles S. Davis (2002). Statistical methods for the analysis of repeated measures. New York: Springer-Verlag.

Demidenko E. (2013). Mixed Models: Theory and Applications with R 2nd Edition). New Jersey: John Wiley \& Sons (Excellent book).

See Also

`colvarcomps.mle`, `rint.reg`, `rint.regbx`

Examples

```
## example from Montgomery, pages 514-517
x <- c(98,97,99,96,91,90,93,92,96,95,97,95,95,96,99,98)
ina <- rep(1:4, each = 4)
```

```
varcomps.mom(x, ina)
varcomps.mle(x, ina)
```

Multi-sample tests for vectors

Multi-sample tests for vectors

Description

Multi-sample tests for vectors.

Usage

```
ftest(x, ina, logged = FALSE)
anova1(x, ina, logged = FALSE)
kruskaltest(x, ina, logged = FALSE)
var2test(x, y, alternative = "unequal", logged = FALSE)
mcnemar(x, y, logged = FALSE)
ttest2(x, y, paired = FALSE, logged = FALSE)
cqtest(x, treat, block, logged = FALSE)
block.anova(x, treat, block, logged = FALSE)
twoway.anova(y, x1, x2, interact = FALSE, logged = FALSE)
```

Arguments

x	A numerical vector with the data.
y	A numerical vector with the data.
ina	A numerical vector with 1s, 2s, 3s and so one indicating the two groups. Be careful, the function is desinged to accept numbers greater than zero. Alternatively it can be a factor variable.
paired	This is for the two sample t-test only and is TRUE or FALSE specifying whether the two samples are paired or not.
alternative	This can either be "unequal", "greater" or "less".
treat	In the case of the blocking ANOVA and Cochran's Q test, this argument plays the role of the "ina" argument.
block	This item (in the blocking ANOVA and Cochran's Q test) denotes the subjects which are the same. Similarly to "ina" a numeric vector with 1s, 2s, 3s and so on.
x1	The first factor in the two way ANOVA.
x2	The second factor in the two way ANOVA. The orderis not important.
interact	Should interaction in the two way ANOVA be included? The default value is FALSE (no interaction).
logged	Should the p-values be returned (FALSE) or their logarithm (TRUE)?

Details

The Welch's F-test (without assuming equal variances) is performed with the "ftest" function. The "anova" function perform the classical (Fisher's) one-way analysis of variance (ANOVA) which assumes equal variance across the groups. The "kruskaltest" performs the Kruskal-Wallis non parametric alternative to analysis of variance test. The "var2tests" implement the classical F test for the equality of two sample variances. The "cqtest" performs the Cocrhan's Q test for the equality of more than two groups whose values are strictly binary (0 or 1). This is a generalisation of the McNemar's test in the multi-sample case. The "block.anova" is the ANOVA with blocking, randomised complete block design (RCBD). In this case, for every combination of the block and treatment values, there is only one observation. The mathematics are the same as in the case of "twoway.anova", but the assumptions different and the testing procedure also different. In addition, no interaction is present.

Value

A vector with the test statistic and the p-value of each test. For the case of t-test, an extra column with the degrees of freedom is given. For the two way ANOVA there can can be either 2 or three F test statistics and hence the same number of p-values.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

B.L. Welch (1951). On the comparison of several mean values: an alternative approach. Biometrika, 38(3/4), 330-336.

D.C. Montgomery (2001). Design and analysis of experiments (5th Edition). New York: John Wiley \& Sons.

McNemar Q. (1947). Note on the sampling error of the difference between correlated proportions or percentages. Psychometrika. 12(2):153-157.

See Also

ttests, ftests

Examples

```
x <- rnorm(300)
ina <- rbinom(300, 3, 0.5) + 1
anova1(x, ina)
ftest(x, ina)
ina <- rbinom(300, 1, 0.5) + 1
x1 <- x[ ina == 1 ]   ; x2 <- x[ ina == 2 ]
ttest2(x1, x2)
var2test(x1, x2)
```

```
## RCBD example 4.1 from Montgomery (2001), page 131-132
x <- c(9.3, 9.4, 9.2, 9.7, 9.4, 9.3, 9.4, 9.6, 9.6, 9.8, 9.5, 10,
10, 9.9, 9.7, 10.2)
tr <- rep(1:4, 4)
bl <- rep(1:4, each = 4)
block.anova(x, tr, bl)
```

`Multivariate Laplace random values simulation`

Multivariate Laplace random values simulation

Description

Multivariate Laplace random values simulation.

Usage

```
rmvlaplace(n, lam, mu, G)
```

Arguments

`n`	The sample size, a numerical value.
`lam`	The the parameter of the exponential distribution, a positive number.
`mu`	The mean vector.
`G`	A $d \times d$ covariance matrix with determinant 1.

Details

The algorithm uses univariate normal random values and transforms them to multivariate via a spectral decomposition.

Value

A matrix with the simulated data.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>

References

Eltoft T., Kim T., and Lee T.W. (2006). On the multivariate laplace distribution. Signal Processing Letters, IEEE, 13(5):300-303.

See Also

`rmvnorm`, `racg`, `rmvt`

Examples

```
m <- colmeans( as.matrix( iris[, 1:4] ) )
s <- cov(iris[,1:4])
s <- s / det(s)^0.25
lam <- 3
x <- rmvlaplace(1000, lam, m, s)
```

`Multivariate normal and t random values simulation`
Multivariate normal and t random values simulation

Description

Multivariate normal and t random values simulation.

Usage

```
rmvnorm(n, mu, sigma)
rmvt(n, mu, sigma, v)
```

Arguments

`n`	The sample size, a numerical value.
`mu`	The mean vector in R^d.
`sigma`	The covariance matrix in R^d.
`v`	The degrees of freedom.

Details

The algorithm uses univariate normal random values and transforms them to multivariate via a spectral decomposition. It is faster than the command "mvrnorm" available from MASS, and it allows for singular covariance matrices.

Value

A matrix with the simulated data.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>

References

Aitchison J. (1986). The statistical analysis of compositional data. Chapman & Hall.

See Also

racg, rmvlaplace, rmvt

Examples

```
x <- as.matrix(iris[, 1:4])
m <- colmeans(x)
s <- cov(x)
y <- rmvnorm(1000, m, s)
colmeans(y)
cov(y)
```

Naive Bayes classifiers

Naive Bayes classifiers

Description

Gaussian, Poisson and multinomial naive Bayes classifiers.

Usage

```
gaussian.nb(xnew = NULL, x, ina)
poisson.nb(xnew, x, ina)
multinom.nb(xnew, x, ina)
```

Arguments

xnew	A numerical matrix with new predictor variables whose group is to be predicted. For the Gaussian naive Bayes, this is set to NUUL, as you might want just the model and not to predict the membership of new observations. For the Gaussian case this contains any numbers, but for the multinomial and Poisson cases, the matrix must contain integer valued numbers only.
x	A numerical matrix with the observed predictor variable values. For the Gaussian case this contains any numbers, but for the multinomial and Poisson cases, the matrix must contain integer valued numbers only.
ina	A numerical vector with strictly positive numbers, i.e. 1,2,3 indicating the groups of the dataset. Alternatively this can be a factor variable.

Value

For the Poisson and Multinomial naive Bayes classifiers the estimated group, a numerical vector with 1, 2, 3 and so on. For the Gaussian naive Bayes classifier a list including:

mu	A matrix with the mean vector of each group based on the dataset.
sigma	A matrix with the variance of each group and variable based on the dataset.
ni	The sample size of each group in the dataset.

est	The estimated group of the xnew observations. It returns a numerical value back regardless of the target variable being numerical as well or factor. Hence, it is suggested that you do \"as.numeric(target)\" in order to see what is the predicted class of the new data.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

See Also

colmeans, colVars

Examples

```
x <- as.matrix(iris[, 1:4])
a <- gaussian.nb(x, x, iris[, 5])
x1 <- matrix( rpois(100 * 4, 5), ncol = 4)
x2 <- matrix( rpois(50 * 4, 10), ncol = 4)
x <- rbind(x1, x2)
ina <- c( rep(1, 100), rep(2, 50) )
poisson.nb(x, x, ina)
multinom.nb(x, x, ina)
```

Natural Logarithm each element of a matrix

Natural Logarithm each element of a matrix

Description

Natural Logarithm each element of a matrix.

Usage

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

Arguments

x	A matrix with data.
na.rm	A boolean value (TRUE/FALSE) for removing NA.

Details

The argument must be a matrix. For vector the time was the same as R's "log" function so we did not add it.

Value

A matrix where each element is the natural logarithm of the given argument.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

Lbeta, Lchoose, Choose

Examples

```
x <-matrix( runif( 100 * 100), ncol = 100 )
a <- log(x)
b <- Log(x)
all.equal(a, b) # true
```

`Natural logarithm of the beta function`

Natural logarithm of the beta function

Description

Natural logarithm of the beta function.

Usage

```
Lbeta(x, y)
```

Arguments

x	A numerical matrix, or a vector or just a number with positive numbers in either case.
y	A numerical matrix, or a vector or just a number with positive numbers in either case. The dimensions of y must match those of x.

Details

The function is faster than R's lbeta when the dimensions of x any are large. If you have only two numbers, then lbeta is faster. But if you have for example two vectors of 1000 values each, Lbeta becomes two times faster than lbeta.

Value

The matrix, vector or number with the resulting values.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

References

Abramowitz, M. and Stegun, I. A. (1972) Handbook of Mathematical Functions. New York: Dover. https://en.wikipedia.org/wiki/Abramowitz_and_Stegun provides links to the full text which is in public domain. Chapter 6: Gamma and Related Functions.

See Also

`Lgamma`, `beta.mle`, `diri.nr2`

Examples

```
x <- rexp(1000)
y <- rexp(1000)
a1 <- Lbeta(x, y)
```

`Natural logarithm of the gamma function and its derivatives`
Natural logarithm of the gamma function and its derivatives.

Description

Natural logarithm of the gamma function and its derivatives.

Usage

```
Lgamma(x)
Digamma(x)
Trigamma(x)
```

Arguments

`x`	A numerical matrix or vector with positive numbers in either case.

Details

We have spotted that the time savings come when there are more than 50 elements, with vector or matrix.

Value

The matrix or the vector with the resulting values.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

References

Abramowitz, M. and Stegun, I. A. (1972) Handbook of Mathematical Functions. New York: Dover. https://en.wikipedia.org/wiki/Abramowitz_and_Stegun provides links to the full text which is in public domain. Chapter 6: Gamma and Related Functions.

See Also

beta.mle, diri.nr2

Examples

```
x <- matrix( rnorm(500 * 500), ncol = 500 )
 a1 <- Lgamma(x)
a2 <- lgamma(x)
all.equal(as.vector(a1), as.vector(a2))

a1 <- Digamma(x)
a2 <- digamma(x)
all.equal(as.vector(a1), as.vector(a2))
```

Norm of a matrix *Norm of a matrix*

Description

Norm of a matrix.

Usage

```
Norm(x, type = "F")
```

Arguments

x	A matrix with numbers.
type	The type of norm to be calculated. The default is "F" standing for Frobenius norm ("f" in R's norm). The other options are "C" standing for the one norm ("o" in R's norm), "R" for the identiy norm ("I" in R's norm) and "M" for the maximum modulus among elements of a matrix ("M" in R's norm)

Value

A number, the norm of the matrix.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

`Dist`, `dista`, `colmeans`

Examples

```
x <- matrix( rnorm(10 * 10), ncol = 10 )
Norm(x, "F")
norm(x, "f")
Norm(x, "M")
norm(x, "M")
```

`Number of equal columns between two matrices`

Number of equal columns between two matrices

Description

Number of equal columns between two matrices.

Usage

```
mat.mat(x, y)
```

Arguments

`x`	A numerical matrix. See details for more information. It must have the same number of rows as y.
`y`	A numerical matrix. See details for more information. It must have the same number of rows as x.

Details

The function takes each column of x and checks the number of times it matches a column of y. In the example below, we take the first 3 columns of iris as the x matrix. The y matrix is the whole of iris. We will see how many times, each column of x appears in the y matrix. The answer is 1 for each column.

Value

A numerical vector of size equal to the number of columns of x.

Author(s)

Manos Papadakis

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

See Also

Match, colmeans, colMedians

Examples

```
x <- as.matrix(iris[, 1:3])
y <- iris
y[, 5] <- as.numeric(y[, 5])
y <- as.matrix(y)
mat.mat(x, y)
```

Odds ratio *Odds ratio*

Description

Odds ratio.

Usage

```
odds.ratio(x, a = 0.05, logged = FALSE)
```

Arguments

x	A 2 x 2 matrix or a vector with 4 elements. In the case of the vector make sure it corresponds to the correct table.
a	The significance level, set to 0.05 by default.
logged	Should the p-values be returned (FALSE) or their logarithm (TRUE)?

Details

The odds ratio and the confidence interval are calculated.

Value

A list including:

res	The estimated odds ratio and the p-value for the null hypothesis test that it is equal to 1.
ci	The (1-a)% confidence interval for the true value of the odds ratio.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Giorgos Athineou <athineou@csd.uoc.gr>

References

Mosteller Frederick (1968). Association and Estimation in Contingency Tables. Journal of the American Statistical Association. 63(321):1-28.

Edwards A.W.F. (1963). The measure of association in a 2x2 table. Journal of the Royal Statistical Society, Series A. 126(1):109-114.

See Also

odds, g2Test

Examples

```
x <- rpois(4, 30)+2
odds.ratio(x)
odds.ratio( matrix(x, ncol = 2) )
```

One sample empirical and exponential empirical likelihood test

One sample exponential empirical likelihood test

Description

One sample exponential empirical likelihood test.

Usage

```
eel.test1(x, mu, tol = 1e-09, logged = FALSE)
el.test1(x, mu, tol = 1e-07, logged = FALSE)
```

Arguments

x	A numerical vector.
mu	The hypothesised mean value.
tol	The tolerance value to stop the iterations of the Newton-Raphson.
logged	Should the logarithm of the p-value be returned? TRUE or FALSE.

Details

Exponential empirical likelihood is a non parametric method. In this case we use it as the non parametric alternative to the t-test. Newton-Raphson is used to maximise the log-likelihood ratio test statistic. In the case of no solution, NULL is returned. Despite the function having beeen written in R, it is pretty fast. As for the empirical likelihood ratio test, there is a condition for the range of possible values of mu. If mu is outside this range it is rejected immediately.

Value

`iters`	The number of iterations required by the Newton-Raphson algorithm. If no covnergence occured this is NULL. This is not returned for the empircial likelihood ratio test.
`info`	A vector with three elements, the value of the λ, the likelihood ratio test statistic and the relevant p-value. If no convergence occured, the value of the λ before is becomes NA, the value of test statistic is 10^5 and the p-value is 0. No convergence can be interpreted as rejection of the hypothesis test.
`p`	The estimated probabilities, one for each observation. If no covnergence occured this is NULL.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Owen A. B. (2001). Empirical likelihood. Chapman and Hall/CRC Press.

See Also

`ftest`, `ttest1`

Examples

```
x <- rnorm(5000)
system.time(a1 <- eel.test1(x, 0) )
system.time(a2 <- el.test1(x, 0) )
```

`One sample t-test for a vector`

One sample t-test for a vector

Description

One sample t-test for a vector.

Usage

```
ttest1(x, m, alternative = "unequal", logged = FALSE, conf = NULL)
```

Arguments

x	A numerical vector with the data.
m	The mean value under the null hypothesis.
alternative	The alternative hypothesis, "unequal", "greater" or "less".
logged	Should the p-values be returned (FALSE) or their logarithm (TRUE)?
conf	If you want a confidence interval supply the confidence level.

Details

The usual one sample t-test is implemented, only faster.

Value

A list including:

res	A two valued vector with the test statistic and its (logged) p-value.
ci	In the case you supplied a number in the input argument "conf" the relevant confidence interval will be returned as well.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>

See Also

ttest, anova1, ttests

Examples

```
x = rnorm(1000)
t.test(x, mu = 0)
ttest1(x, 0, conf = 0.95)
```

`Operations between two matrices` *Operations between two matrices*

Description

Operations between two matrices.

Usage

```
XopY.sum(x, y = NULL, oper = "*")
```

Arguments

x	A numerical matrix.
y	A second numerical matrix whose dimensions must match the ones of x.
oper	The operation to be performed, either \"*\", \"/\", \"+\" or \"-\".

Details

This function simply does operations between two matrices.

Value

Sum of \"x*x\" or \"x*y\".

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

`Dist`, `dista`, `colmeans`

Examples

```
x <- matrix( rnorm(5 * 5), ncol = 5 )
y <- matrix( rnorm(5 * 5), ncol = 5 )
XopY.sum(x, y, oper = "*")
```

Order *Order*

Description

Order function like R's order.

Usage

```
Order(x,stable=FALSE)
```

Arguments

x	A Numeric vector with data.
stable	A boolean value for using a stable sorting algorithm.

Details

This function implements the R's "Order" function. If you want the same results as R's, then set "stable=TRUE" because "stable=FALSE" uses a sorting algorithm that it is not stable like R's sort. But it is faster to use the default. This verion is faster for large data, more than 1000.

Value

Sort the vector and returns the indices of each element that it has before the sorting.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>

See Also

Match

Examples

```
y <- rnorm(100)
b <- Order(y)
a <- order(y)
all.equal(a,b) ## false because it is not stable
a <- Order(y,TRUE)
all.equal(a,b) ## true because it is stable
```

Permutation *Permutation*

Description

Permute the given vector.

Usage

```
permutation(x, all=TRUE)
permutation.next(x, all.next=TRUE)
permutation.prev(x, all.prev=TRUE)
```

Arguments

x	A numeric vector with data.
all	A logical value for returning all or one possible combinations.
all.next	A logical value for returning all the next or one possible combinations, if there are.
all.prev	A logical value for returning all the previous or one possible combinations, if there are.

Details

This function implements "Permutation", which means all the possible combinations. In the permutation.next and permutation.prev if there aren't possible combinations it returns the same vector.

Value

Returns a matrix with all possible combinations of the given vector or a matrix row with one possible combinations.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>

See Also

combn, comb_n

Examples

```
y <- rnorm(5)
b <- permutation(y)
b <- permutation.next(y)
b <- permutation.prev(y)
```

`Prediction with some naive Bayes classifiers`
Prediction with some naive Bayes classifiers

Description

Prediction with some naive Bayes classifiers.

Usage

```
gaussiannb.pred(xnew, m, s, ni)
poissonnb.pred(xnew, m)
multinomnb.pred(xnew, m)
```

Arguments

`xnew`	A numerical matrix with new predictor variables whose group is to be predicted. For the Gaussian case this contains any numbers, but for the multinomial and Poisson cases, the matrix must contain integer valued numbers only.
`m`	A matrix with the group means. Each row corresponds to a group.
`s`	A matrix with the group colum-wise variances. Each row corresponds to a group.
`ni`	A vector with the frequencies of each group.

Value

A numerical vector with 1, 2, ... denoting the predicted group.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

See Also

`gaussian.nb`, `colpoisson.mle` `colVars`

Examples

```
ina <- sample(1:150, 100)
x <- as.matrix(iris[, 1:4])
id <- as.numeric(iris[, 5])
a <- gaussian.nb(xnew = NULL, x[ina, ], id[ina])
est <- gaussiannb.pred(x[-ina, ], a$mu, a$sigma, a$ni)
table(id[-ina], est)
```

Quasi binomial regression for proportions

Quasi binomial regression for proportions

Description

Quasi binomial regression for proportions.

Usage

```
prop.reg(y, x, varb = "quasi", tol = 1e-09)
prop.regs(y, x, varb = "quasi", tol = 1e-09, logged = FALSE)
```

Arguments

y	A numerical vector proportions. 0s and 1s are allowed.
x	For the "prop.reg" a matrix with data, the predictor variables. This can be a matrix or a data frame. For the "prop.regs" this must be a numerical matrix, where each columns denotes a variable.
tol	The tolerance value to terminate the Newton-Raphson algorithm. This is set to 10^{-9} by default.
varb	The type of estimate to be used in order to estimate the covariance matrix of the regression coefficients. There are two options, either "quasi" (default value) or "glm". See the references for more information.
logged	Should the p-values be returned (FALSE) or their logarithm (TRUE)?

Details

We are using the Newton-Raphson, but unlike R's built-in function "glm" we do no checks and no extra calculations, or whatever. Simply the model. The "prop.regs" is to be used for very many univariate regressions. The "x" is a matrix in this case and the significance of each variable (column of the matrix) is tested. The function accepts binary responses as well (0 or 1).

Value

For the "prop.reg" function a list including:

iters	The number of iterations required by the Newton-Raphson.
varb	The covariance matrix of the regression coefficients.
phi	The phi parameter is returned if the input argument "varb" was set to "glm", othwerise this is NULL.
info	A table similar to the one produced by "glm" with the estimated regression coefficients, their standard error, Wald test statistic and p-values.

For the "prop.regs" a two-column matrix with the test statistics (Wald statistic) and the associated p-values (or their loggarithm).

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Papke L. E. & Wooldridge J. (1996). Econometric methods for fractional response variables with an application to 401(K) plan participation rates. Journal of Applied Econometrics, 11(6): 619–632.

McCullagh, Peter, and John A. Nelder. Generalized linear models. CRC press, USA, 2nd edition, 1989.

See Also

anova_propreg univglms, score.glms, logistic_only

Examples

```
y <- rbeta(100, 1, 4)
x <- matrix(rnorm(100 * 3), ncol = 3)
a <- prop.reg(y, x)
y <- rbeta(100, 1, 4)
x <- matrix(rnorm(400 * 100), ncol = 400)
b <- prop.regs(y, x)
mean(b[, 2] < 0.05)
hist(b[, 2])
```

Quasi Poisson regression

Quasi Poisson regression

Description

Quasi Poisson regression.

Usage

```
qpois.reg(x, y, full = FALSE, tol = 1e-09)
```

Arguments

x	The predictor variables. This can be a matrix or a data frame.
y	A numerical vector with positive discrete data.
full	If this is FALSE, the coefficients, the deviance and the estimated phi parameter will be returned only. If this is TRUE, more information is returned.
tol	The tolerance value to terminate the Newton-Raphson algorithm. This is set to 10^{-9} by default.

Details

We are using the Newton-Raphson, but unlike R's built-in function "glm" we do no checks and no extra calculations, or whatever. Simply the model, unless the user requests for the Wald tests of the coefficients.

Value

When full is FALSE a list including:

`be`	The regression coefficients.
`devi`	The deviance of the model.
`varb`	The covariance matrix of the beta coefficients.
`phi`	The phi parameter, the estimate of dispersion.

When full is TRUE, the additional item is:

`info`	The regression coefficients, their standard error, their Wald test statistic and their p-value.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

McCullagh, Peter, and John A. Nelder. Generalized linear models. CRC press, USA, 2nd edition, 1989.

See Also

`prop.reg` `univglms`, `score.glms`, `poisson_only`

Examples

```
y <- rnbinom(100, 10, 0.6)
x <- matrix(rnorm(100*3), ncol = 3)
mod1 <- glm(y ~ x, quasipoisson)
summary(mod1)
qpois.reg(x, y, full = TRUE)
```

`Random intercepts linear mixed models`
Random intercepts linear mixed models

Description

Random intercepts linear mixed models (for balanced data with a single identical covariate).

Usage

```
rint.reg(y, x, id ,tol = 1e-08)
rint.regbx(y, x, id)
```

Arguments

`y`	A numerical vector with the data. The subject values.
`x`	For the case of "rint.reg" this can be a vector or a numerical matrix with data. In the case of "rint.regbx" this is a numerical vector with the same length as y indicating the fixed predictor variable. Its values are the same for all levels of y. An example of this x is time which is the same for all subjects.
`id`	A numerical variable with 1, 2, ... indicating the subject.
`tol`	The tolerance level to terminate the generalised elast squares algorithm.

Details

Random intercepts linear mixed models with compound covariance structure is fitted in both functions. The "rint.reg" allows any numerical matrix, with balanced or unbalanced data. See Demidenko (2013, pg. 65-67) for more information.

The "rint.regbx" is a special case of a balanced random intercepts model with a compound symmetric covariance matrix and one single covariate which is constant for all replicates. An example, is time, which is the same for all subjects. Maximum likelihood estimation has been performed. In this case the mathematics exist in a closed formula (Demidenko, 2013, pg. 67-69).

Value

A list including:

`info`	A vector with the random intercepts variance (between), the variance of the errors (within), the log-likelihood, the deviance (twice the log-likelihood) and the BIC. In the case of "rint.reg" it also includes the number of iterations required by the generalised least squares.
`be`	The estimated regression coefficients, which in the case of "rint.regbx" are simply two: the constant and the slope (time effect).
`ranef`	The random intercepts effects.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Eugene Demidenko (2013). Mixed Models: Theory and Applications with R, 2nd Edition. New Jersey: Wiley \& Sons (excellent book).

See Also

rm.lines, varcomps.mom, colvarcomps.mom

Examples

```
y <- rnorm(100)
x <- rnorm(10)
x <- rep(x, 10)
id <- id <- rep(1:10, each = 10)
system.time( for (i in 1:50) a <- rint.regbx(y, x, id) )
```

Random values simulation from a von Mises distribution

Random values simulation from a von Mises distribution

Description

It generates random vectors following the von Mises distribution. The data can be spherical or hyper-spherical.

Usage

```
rvonmises(n, m, k, rads = TRUE)
```

Arguments

n	The sample size.
m	The mean angle expressed in radians or degrees.
k	The concentration parameter. If k is zero the sample will be generated from the uniform distribution over $(0, 2\pi)$.
rads	If the mean angle is expressed in radians, this should be TRUE and FALSE otherwise. The simulated data will be expressed in radians or degrees depending on what the mean angle is expressed.

Details

The mean direction is transformed to the Euclidean coordinates (i.e. unit vector) and then the fvmf function is employed. It uses a rejection smapling as suggested by Andrew Wood in 1994. I have mentioned the description of the algorithm as I found it in Dhillon and Sra in 2003. Finally, the data are transformed to radians or degrees.

Value

A vector with the simulated data.

Author(s)

Michail Tsagris and Manos Papadakis

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm85@gmail.com>

References

Wood, A. T. (1994). Simulation of the von Mises Fisher distribution. Communications in statistics-simulation and computation, 23(1): 157-164.

Dhillon, I. S., & Sra, S. (2003). Modeling data using directional distributions. Technical Report TR-03-06, Department of Computer Sciences, The University of Texas at Austin. http://citeseerx.ist.psu.edu/viewdoc/download?d

See Also

vm.mle, rvmf

Examples

```
x <- rvonmises(1000, 2, 25, rads = TRUE)
vm.mle(x)
```

Reading the files of a directory

Reading the files of a directory

Description

Reading the files of a directory.

Usage

```
read.directory(path.directory)
read.examples(path.man,dont.read = "")
```

Arguments

path.directory	The full path to the directory. For example: \"C:\Users\username\Documents\R\Rfast_1.8.0\R\"
path.man	The full path to the directory with the Rd files in it. For example: \"C:\Users\username\Documents\R\Rfas
dont.read	A character vector with the name of the files that you wish not to read. By default it's empty \"\".

Details

For function \"read.directory\": Takes as an argument a full path to a directory and returns the names of the files.

For function \"read.examples\": Takes as an argument a full path to the directory of the Rd files and the name of the files that shouldn't read.

Value

For function \"read.directory\": The names of the files.

For function \"read.examples\": a list with 2 fields

examples	A character vector with the examples of each Rd file.
files	A character vector with the name of the file that each examples belongs.

Author(s)

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

AddToNamespace, sourceR, sourceRd, checkRd, checkExamples

Examples

```
# for example: path="C:\some_file\"
# system.time( read.directory(path) )
# system.time( list.dirs(path) )

# for example: path.man="C:\some_file\man\"
# system.time( read.examples(path.man) )
# system.time( read.examples(path.man,dont.read=c("somef_1.Rd",...,"somef_n.Rd") ) )
```

`Replicate columns` *Replicate columns*

Description

Replicate columns.

Usage

```
rep_col(x,ncols)
```

Arguments

`x`	A vector with data.
`ncols`	Number of new columns.

Value

A matrix where each column is equal to \"x\".

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

`rowMins`, `rowFalse`, `nth`, `colrange`, `colMedians`, `colVars`, `sort_mat`, `rowTrue`

Examples

```
x <- runif(10)
all.equal(rep_col(x,10),matrix(x,nrow=length(x),ncol=10))
```

`Round each element of a matrix/vector`
Round each element of a matrix/vector

Description

Round each element of a matrix/vector.

Usage

```
Round(x,digit=0)
```

Arguments

x	A numeric matrix/vector with data or NA. NOT integer values.
digit	An integer value for 0...N-1 where N is the number of the digits. By default is 0.

Details

Round is a very fast C++ implementation. Especially for large data. It handles NA.

Value

A vector/matrix where each element is been rounded in the given digit.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

Lchoose, Log, Choose

Examples

```
x <-matrix( rnorm( 500 * 100), ncol = 100 )
system.time( a <- Round(x,5) )
system.time( b <- round(x,5) )
all.equal(a,b) #true
x <-rnorm( 1000)
system.time( a <- Round(x,5) )
system.time( b <- round(x,5) )
all.equal(a,b) # true
```

Row-wise minimum and maximum

Row-wise minimum and maximum of a matrix.

Description

Row-wise minimum and maximum of a matrix.

Usage

```
rowMins(x, value = FALSE)
rowMaxs(x, value = FALSE)
rowMinsMaxs(x)
```

Arguments

x	A numerical matrix with data.
value	If the value is FALSE it returns the indices of the minimum/maximum, otherwise it returns the minimum and maximum values.

Value

A vector with the relevant values.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

colMins, colMaxs, nth, rowrange colMedians, colVars, sort_mat

Examples

```
x <- matrix( rnorm(500 * 500), ncol = 500 )

system.time( s1 <- rowMins(x) )
system.time( s2 <- apply(x, 1, min) )

system.time( s1 <- rowMaxs(x) )
system.time( s2 <- apply(x, 1, max) )

system.time( s1 <- c(apply(x, 1, min),apply(x, 1, max) ))
system.time( s2 <- rowMinsMaxs(x) )
```

Row-wise true value *Row-wise true value of a matrix*

Description

Row-wise true value of a matrix.

Usage

```
rowTrue(x)
rowFalse(x)
rowTrueFalse(x)
```

Arguments

x	A logical matrix with data.

Value

An integer vector where item \"i\" is the number of the true/false values of \"i\" row.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

rowMins, colFalse, nth, rowrange, rowMedians, rowVars, sort_mat, colTrue

Examples

```
x <- matrix(as.logical(rbinom(100*100,1,0.5)),100,100)

s1 <- rowTrue(x)

s1 <- rowFalse(x)

s1 <- rowTrueFalse(x)
```

Search for variables with zero range in a matrix

Search for variables with zero range in a matrix

Description

Search for variables with zero range in a matrix.

Usage

```
check_data(x, ina = NULL)
```

Arguments

x	A matrix or a data.frame with the data, where rows denotes the observations and the columns contain the dependent variables.
ina	If your data are grouped, for example there is a factor or numerical variable indicating the groups of the data supply it here, otherwise leave it NULL.

Details

The function identifies the variabels with zero range, instead of a zero variance as this is faster. It will work with matrices and data.frames.

Value

A numerical vector of length zero if no zero ranged variable exists, or of length at least one with the index (or indices) of the variable(s) that need attention or need to be removed.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

See Also

colrange, colVars

Examples

```
x <- matrix( rnorm(100 * 100), ncol = 100 )
check_data(x)

## some variables have a constant value
x[, c(1,10, 50, 70)] <- 1
check_data(x)
id <- rep(1:4, each = 25 )
x[1:25, 2] <- 0
check_data(x)  ## did not use the id variable
check_data(x, id)  ## see now
```

Significance testing for the coefficients of Quasi binomial regression for proportions

Significance testing for the coefficients of Quasi binomial regression for proportions

---

Description

Significance testing for the coefficients of Quasi binomial regression for proportions.

Usage

```
anova_propreg(mod, poia = NULL)
```

Arguments

mod	An object as returned by the "prop.reg" function.
poia	If you want to test the significance of a single coefficient this must be a number. In this case, the "prop.reg" function contains this information. If you want more coefficients to be testes simultaneously, e.g. for a categorical predictor, then this must contain the positions of the coefficients. If you want to see if all coefficients are zero, like an overall F-test, leave this NULL.

Details

Even though the name of this function starts with anova it is not an ANOVA type significance testing, but a Wald type.

Value

A vector with three elements, the test statistic value, its associated p-value and the relevant degrees of freedom.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Papke L. E. & Wooldridge J. (1996). Econometric methods for fractional response variables with an application to 401(K) plan participation rates. Journal of Applied Econometrics, 11(6): 619–632.

McCullagh, Peter, and John A. Nelder. Generalized linear models. CRC press, USA, 2nd edition, 1989.

See Also

prop.reg, univglms, score.glms, logistic_only

Examples

```
y <- rbeta(1000, 1, 4)
x <- matrix(rnorm(1000 * 3), ncol = 3)
a <- prop.reg(y, x)
## all coefficients are tested
anova_propreg(a)
## the first predictor variable is tested
anova_propreg(a, 2)
a  ## this information is already included in the model output
## the first and the second predictor variables are tested
anova_propreg(a, 2:3)
```

Simulation of random values from a von Mises-Fisher distribution

Random values simulation from a von Mises-Fisher distribution

---

Description

It generates random vectors following the von Mises-Fisher distribution. The data can be spherical or hyper-spherical.

Usage

```
rvmf(n, mu, k)
```

Arguments

n	The sample size.
mu	The mean direction, a unit vector.
k	The concentration parameter. If k = 0, random values from the spherical uniform will be drwan. Values from a multivariate normal distribution with zero mean vector and the identity matrix as the covariance matrix. Then each vector becomes a unit vector.

Details

It uses a rejection smapling as suggested by Andrew Wood (1994).

Value

A matrix with the simulated data.

Author(s)

Michail Tsagris and Manos Papadakis

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm85@gmail.com>

References

Wood A. T. A. (1994). Simulation of the von Mises Fisher distribution. Communications in statistics-simulation and computation, 23(1): 157–164.

Dhillon I. S. & Sra S. (2003). Modeling data using directional distributions. Technical Report TR-03-06, Department of Computer Sciences, The University of Texas at Austin. http://citeseerx.ist.psu.edu/viewdoc/download?d

See Also

vmf.mle, rvonmises, iag.mle

Examples

```
m <- rnorm(4)
m <- m/sqrt(sum(m^2))
x <- rvmf(1000, m, 25)
m
vmf.mle(x)
```

`Skeleton of the PC algorithm`

The skeleton of a Bayesian network produced by the PC algorithm

Description

The skeleton of a Bayesian network produced by the PC algorithm.

Usage

```
pc.skel(dataset, method = "pearson", alpha = 0.05, R = 1)
```

Arguments

`dataset`	A numerical matrix with the variables. If you have a data.frame (i.e. categorical data) turn them into a matrix using `data.frame.to_matrix`. Note, that for the categorical case data, the numbers must start from 0. No missing data are allowed.
`method`	If you have continuous data, you can choose either "pearson" or "spearman". If you have categorical data though, this must be "cat". In this case, make sure the minimum value of each variable is zero. The `g2Test` and the relevant functions work that way.
`alpha`	The significance level (suitable values in (0, 1)) for assessing the p-values. Default (preferred) value is 0.01.
`R`	The number of permutations to be conducted. The p-values are assessed via permutations. Use the default value if you want no permutation based assessment.

Details

The PC algorithm as proposed by Spirtes et al. (2000) is implemented. The variables must be either continuous or categorical, only. The skeleton of the PC algorithm is order independent, since we are using the third heuristic (Spirte et al., 2000, pg. 90). At every stage of the algorithm use the pairs which are least statistically associated. The conditioning set consists of variables which are most statistically associated with each other of the pair of variables.

For example, for the pair (X, Y) there can be two conditioning sets for example (Z1, Z2) and (W1, W2). All p-values and test statistics and degrees of freedom have been computed at the first step of the algorithm. Take the p-values between (Z1, Z2) and (X, Y) and between (Z1, Z2) and (X, Y). The conditioning set with the minimum p-value is used first. If the minimum p-values are the same, use the second lowest p-value. If the unlikely, but not impossible, event of all p-values being the same, the test statistic divided by the degrees of freedom is used as a means of choosing which conditioning set is to be used first.

If two or more p-values are below the machine epsilon (.Machine$double.eps which is equal to 2.220446e-16), all of them are set to 0. To make the comparison or the ordering feasible we use the logarithm of p-value. Hence, the logarithm of the p-values is always calculated and used.

In the case of the G^2 test of independence (for categorical data) with no permutations, we have incorporated a rule of thumb. If the number of samples is at least 5 times the number of the parameters to be estimated, the test is performed, otherwise, independence is not rejected according to Tsamardinos et al. (2006). We have modified it so that it calculates the p-value using permutations.

Value

A list including:

`stat`	The test statistics of the univariate associations.
`pvalue`	The logarithm of the p-values of the univariate associations.
`runtime`	The amount of time it took to run the algorithm.
`kappa`	The maximum value of k, the maximum cardinality of the conditioning set at which the algorithm stopped.
`n.tests`	The number of tests conducted during each k.
`G`	The adjancency matrix. A value of 1 in G[i, j] appears in G[j, i] also, indicating that i and j have an edge between them.
`sepset`	A list with the separating sets for every value of k.

Author(s)

Marios Dimitriadis

R implementation and documentation: Marios Dimitriadis <kmdimitriadis@gmail.com>

References

Spirtes P., Glymour C. and Scheines R. (2001). Causation, Prediction, and Search. The MIT Press, Cambridge, MA, USA, 3nd edition.

Tsamardinos I., Borboudakis G. (2010) Permutation Testing Improves Bayesian Network Learning. In Machine Learning and Knowledge Discovery in Databases. ECML PKDD 2010. 322-337.

Tsamardinos I., Brown E.L. and Aliferis F.C. (2006). The max-min hill-climbing Bayesian network structure learning algorithm. Machine learning 65(1):31-78.

See Also

`g2Test`, `g2Test_univariate`, `cora`, `correls`

Examples

```
# simulate a dataset with continuous data
dataset <- matrix(rnorm(1000 * 50, 1, 100), nrow = 1000)
a <- pc.skel(dataset, method = "pearson", alpha = 0.01)
```

Sort a vector coresponding to another
Sort

Description

Sort a vector coresponding to another.

Usage

```
sort_cor_vectors(x,y)
sort_index(x, descending = FALSE)
```

Arguments

x	A numeric vector.
y	A numeric vector to help sorting the x.
descending	A logical value for choosing asceding or descending order.

Details

This function implements the R's "sort" function using the C++'s function sort.

Value

Returns the first argument but sorted acording to the second vector.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>

See Also

sort_mat, sort_unique

Examples

```
y <- rnorm(100)
x <- rnorm(100)
b <- sort_cor_vectors(x,y)
b <- sort_index(x)
```

`Sort and unique numbers` *Sort and unique*

Description

Sort and unique numbers.

Usage

```
sort_unique(x)
sort_unique.length(x)
```

Arguments

`x`	A numeric vector.

Details

The "sort_unique" function implements R's "unique" function using C++'s function. The "sort_unique.length" returns the length of the unique numbers.

Value

Returns the discrete values but sorted or their length (depending on the function you do).

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>

See Also

`sort_mat`, `sort_index`, `sort_cor_vectors`

Examples

```
y <- rnorm(100)
a <- sort_unique(y)
b <- sort.int(unique(y))
all.equal(as.vector(a),as.vector(b))
x <- rpois(10000,10)
sort_unique.length(x)
length(sort_unique(x))
```

Sorting a vector *Sorting a vector*

Description

Fast sorting a vector.

Usage

```
Sort(x, descending = FALSE, partial = NULL, stable = FALSE)
```

Arguments

x	A numerical/character vector with data .
descending	A boolean value (TRUE/FALSE) for sorting the vector in descending order. By default sorts the vector in ascending.
partial	An index number for sorting partial the vector. Not character vector.
stable	A boolean value (TRUE/FALSE) for choosing a stable sort algorithm. Stable means that discriminates on the same elements. Not character vector.

Details

This function uses the sorting algorithm from C++. The implementation is very fast and highly optimised. Especially for large data.

Value

The sorted vector.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

nth, colnth, rownth, sort_cor_vectors,sort_index, sort_unique, Round

Examples

```
x <- rnorm(1000)
system.time( s1 <- Sort(x) )
system.time( s2 <- sort(x) )
all.equal(s1,s2) #true

system.time( s1 <- Sort(x,partial=100) )
system.time( s2 <- sort(x,partial=100) )
```

```
all.equal(s1,s2) #true

system.time( s1 <- Sort(x,partial=c(10,100)) )
system.time( s2 <- sort(x,partial=c(10,100)) )
all.equal(s1,s2) #true

system.time( s1 <- Sort(x,stable=TRUE) )
system.time( s2 <- sort(x) )
all.equal(s1,s2) #true

x <- as.character(x)
system.time( s1 <- Sort(x) )
system.time( s2 <- sort(x) )
all.equal(s1,s2) #true
```

`Sorting of the columns-rows of a matrix`

Sorting of the columns-rows of a matrix

Description

Fast sorting of the columns-rows of a matrix.

Usage

```
sort_mat(x, by.row = FALSE, descending = FALSE, stable = FALSE)
```

Arguments

`x`	A numerical matrix with data.
`by.row`	If you want to sort the rows of the matrix set this to TRUE.
`descending`	If you want the sorting in descending order, set this to TRUE.
`stable`	If you the stable version, so that the results are the same as R's (in the case of ties) set this to TRUE. If this is TRUE, the algorithm is a bit slower.

Value

The matrix with its columns-rows (or rows) independently sorted.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

`nth`, `colMaxs`, `colMins`, `colrange`, `sort_cor_vectors`,`sort_index`, `sort_unique`

Examples

```
x <- matrix( rnorm(100 * 500), ncol = 500 )
system.time( s1 <- sort_mat(x) )
system.time( s2 <- apply(x, 2, sort)  )
all.equal(as.vector(s1), as.vector(s2))
```

Source many R files *Source many R files*

Description

Source many R/Rd files.

Usage

```
sourceR(path,local=FALSE,encode = "UTF-8",print.errors=FALSE)
sourceRd(path,print.errors=FALSE)
```

Arguments

path	An full path to the directory where R file are.
local	TRUE, FALSE or an environment, determining where the parsed expressions are evaluated. FALSE (the default) corresponds to the user's workspace (the global environment) and TRUE to the environment from which source is called.
encode	Character vector. The encoding(s) to be assumed when file is a character string: see file. A possible value is \"unknown\" when the encoding is guessed: see the \"Encodings\" section.
print.errors	A boolean value (TRUE/FALSE) for printing the errors, if exists, for every file.

Details

Reads many R files and source them.

Value

Returns the files that had produced errors during source.

Author(s)

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

read.directory, AddToNamespace

Examples

```
# for example: path="C:\some_file\R\" where is R files are
# system.time( a<-sourceR(path) )
# for example: path="C:\some_file\man\" where is Rd files are
# system.time( a<-sourceRd(path) )
```

`Spatial median for Euclidean data`

Spatial median for Euclidean data

Description

Spatial median for Euclidean data.

Usage

```
spat.med(x, tol = 1e-09)
```

Arguments

`x`	A matrix with Euclidean data, continuous variables.
`tol`	A tolerance level to terminate the process. This is set to 1e-09 by default.

Details

The spatial median, using a fixed point iterative algorithm, for Euclidean data is calculated. It is a robust location estimate.

Value

A vector with the spatial median.

Author(s)

Manos Papadakis and Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>

References

Jyrki Mottonen, Klaus Nordhausen and Hannu Oja (2010). Asymptotic theory of the spatial median. In Nonparametrics and Robustness in Modern Statistical Inference and Time Series Analysis: A Festschrift in honor of Professor Jana Jureckova.

T. Karkkaminen and S. Ayramo (2005). On computation of spatial median for robust data mining. Evolutionary and Deterministic Methods for Design, Optimization and Control with Applications to Industrial and Societal Problems, EUROGEN 2005, R. Schilling, W.Haase, J. Periaux, H. Baier, G. Bugeda (Eds) FLM, Munich. http://users.jyu.fi/~samiayr/pdf/ayramo_eurogen05.pdf

See Also

colMedians

Examples

```
spat.med( as.matrix( iris[, 1:4] ) )
colMeans( as.matrix(iris[, 1:4]) )
colMedians( as.matrix(iris[, 1:4]) )
```

Spherical and hyperspherical median

Fast calculation of the spherical and hyperspherical median

Description

It calculates, very fast, the (hyper-)spherical median of a sample.

Usage

```
mediandir(x)
```

Arguments

x	The data, a numeric matrix with unit vectors.

Details

The "mediandir" employes a fixed poit iterative algorithm stemming from the first derivative (Cabrera and Watson, 1990) to find the median direction as described in Fisher (1985) and Fisher, Lewis and Embleton (1987).

Value

The median direction.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr>

References

Fisher N. I. (1985). Spherical medians. Journal of the Royal Statistical Society. Series B, 47(2): 342-348.

Fisher N. I., Lewis T. and Embleton B. J. (1987). Statistical analysis of spherical data. Cambridge university press.

Cabrera J. and Watson G. S. (1990). On a spherical median related distribution. Communications in Statistics-Theory and Methods, 19(6): 1973-1986.

See Also

vmf.mle

Examples

```
m <- rnorm(3)
m <- m / sqrt( sum(m^2) )
x <- rvmf(100, m, 10)
mediandir(x)
```

Standardisation *Standardisation*

Description

Standardisation.

Usage

```
standardise(x, center = TRUE, scale = TRUE)
```

Arguments

x	A matrix with data. It has to be matrix, if it is data.frame for example the function does not turn it into a matrix.
center	Should the data be centred as well? TRUE or FALSE.
scale	Should the columns have unit variance, yes (TRUE) or no (FALSE)?

Details

Similar to R's built in functions \"scale\" there is the option for centering or scaling only or both (default).

Value

A matrix with the standardised data.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

See Also

colVars, colmeans, colMads

Examples

```
x <-matrix( rnorm( 100 * 100), ncol = 100 )
a1 <- scale(x)[1:100, ]
a2 <- standardise(x)
all.equal(as.vector(a1), as.vector(a2))
```

Sum of a matrix *Sum of a matrix*

Description

Sum of a matrix.

Usage

```
matrix.sum(x)
```

Arguments

x	A matrix with data.

Value

The sum of the matrix \"x\".

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

rowMins, rowFalse, nth, colrange, colMedians, colVars, sort_mat, rowTrue

Examples

```
x <- matrix(runif(100*100),100,100)
f <- matrix.sum(x)
f==sum(x)
```

`Sum of all pairwise distances in a distance matrix`
Sum of all pairwise distances in a distance matrix

Description

Sum of all pairwise distances in a distance matrix.

Usage

```
total.dist(x)
total.dista(x, y)
```

Arguments

x	A matrix with numbers.
y	A second matrix with data. The number of comlumns of this matrix must be the same with the matrix x. The number of rows can be different.

Details

In order to do the total.dist one would have to calcualte the distance matrix and sum it. We do this internally in C++ without creating the matrix. For the total.dista it is the same thing.

Value

A numerical value, the sum of the distances.

Author(s)

Manos Papadakis

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

Dist, dista

Examples

```
x <- matrix( rnorm(50 * 10), ncol = 10 )
total.dist(x)
y <- matrix( rnorm(40 * 10), ncol = 10)
total.dista(x, y)
total.dista(y, x)
```

Sums of a vector for each level of a grouping variable
Sums of a vector for each level of a grouping variable

Description

Sums of a vector for each level of a grouping variable.

Usage

```
group.sum(x, ina)
```

Arguments

x	A numerical vector whose sums are to be calculated for each value of "ina".
ina	A numerical vector or a factor variable. For evey distinct value of "ina" the sum of "x" will be calculated. Note that negative values are not allowed as this can cause R to run forever.

Details

This is the rowsum, but only for vectors. No names are returned, simply a matrix with one column, just like rowsum. Note that this command works only for vectors.

Value

A matrix with one column, the sum of x for each distinct value of ina.

Author(s)

We found the C++ code (written by Francois Romain) in http://stackoverflow.com/questions/16975034/rcpp-equivalent-for-rowsum

Manos Papadakis then added it in Rfast.

R Documentation: Michail Tsagris <mtsagris@yahoo.gr>.

See Also

colmeans, colVars, Var, med

Examples

```
x <- rnorm(1000)
ina <- sample(1:5, 1000, replace = TRUE)
a1 <- group.sum(x, ina)
a2 <- rowsum(x, ina)
```

`Table Creation - Frequency of each value`
Table Creation - Frequency of each value

Description

Table Creation - Frequency of each value.

Usage

```
Table(x,as.vector = TRUE)
```

Arguments

`x`	A vector with numeric/character data.
`as.vector`	A boolean value for return vector.

Details

Like R's \"table\".

Value

if is.vector is \"TRUE\" then return a vector with names the discrete values of \"x\" and values there frequencies. Otherwise, a list with 2 fields:

`freqs`	The frequency of each value from vector \"x\".
`values`	The discrete values from vector \"x\" sorted.

Author(s)

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

`colShuffle`, `colVars`, `colmeans`, `read.directory`

Examples

```
x<-runif(10)
y1<-Table(x)
y2<-as.vector(table(x)) # Neads a lot of time.
all.equal(y1,y2)
y1<-Table(x,FALSE)
all.equal(as.character(y1$values),names(y2))
all.equal(y1$freqs,y2)
```

`Tests for the dispersion parameter in Poisson distribution`
Tests for the dispersion parameter in Poisson distribution

Description

Tests for the dispersion parameter in Poisson distribution.

Usage

```
poisdisp.test(y, alternative = "either", logged = FALSE)
pois.test(y, logged = FALSE)
```

Arguments

`y`	A numerical vector with count data, 0, 1,...
`alternative`	Do you want to test specifically for either over or underspirsion ("either"), overdispersion ("over") or undersispersion ("under")?
`logged`	Set to TRUE if you want the logarithm of the p-value.

Value

A vector with two elements, the test statistic and the (logged) p-value.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Yang Zhao, James W. Hardin, and Cheryl L. Addy. (2009). A score test for overdispersion in Poisson regression based on the generalized Poisson-2 model. Journal of statistical planning and inference 139(4): 1514-1521.

Dimitris Karlis and Evdokia Xekalaki (2000). A Simulation Comparison of Several Procedures for Testing the Poisson Assumption. Journal of the Royal Statistical Society. Series D (The Statistician), 49(3): 355-382.

Bohning, D., Dietz, E., Schaub, R., Schlattmann, P. and Lindsay, B. (1994) The distribution of the likelihood ratio for mixtures of densities from the one-parameter exponential family. Annals of the Institute of Statistical Mathematics, 46(): 373-388.

See Also

`poisson.mle`, `negbin.mle`, `poisson.anova`, `poisson.anovas`, `poisson_only`

Examples

```
y <- rnbinom(1000, 10, 0.6)
poisdisp.test(y, "either")
poisdisp.test(y, "over")
pois.test(y)

y <- rpois(1000, 10)
poisdisp.test(y, "either")
poisdisp.test(y, "over")
pois.test(y)
```

`The nth smallest value of a vector`

The nth smallest value of a vector

Description

The nth smallest value of a vector.

Usage

```
nth(x, k)
```

Arguments

x	A numerical vector.
k	The kth smallest number to be returned.

Details

The function is written in C++ and this is why it is very fast. This is called (and used) by `colMedians`.

Value

The desired value.

Author(s)

Manos Papadakis <papadakm95@gmail.com>

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

See Also

`med`, `colMedians`

Examples

```
x <- rnorm(1000)
nth(x, 500)
sort(x)[500]
```

Two sample exponential empirical likelihood test

Two sample exponential empirical likelihood test

Description

Two sample exponential empirical likelihood test.

Usage

```
eel.test2(x, y, tol = 1e-09, logged = FALSE)
```

Arguments

x	A numerical vector.
y	Another numerical vector.
tol	The tolerance value to stop the iterations of the Newton-Raphson.
logged	Should the logarithm of the p-value be returned? TRUE or FALSE.

Details

Exponential empirical likelihood is a non parametric method. In this case we use it as the non parametric alternative to the t-test. Newton-Raphson is used to maximise the log-likelihood ratio test statistic. In the case of no solution, NULL is returned.

Value

iters	The number of iterations required by the Newton-Raphson algorithm. If no covnergence occured this is NULL.
info	A vector with three elements, the value of the λ, the likelihood ratio test statistic and the relevant p-value. If no convergence occured, the value of the λ before is becomes NA, the value of test statistic is 10^5 and the p-value is 0. No convergence can be interpreted as rejection of the hypothesis test.
p1	The estimated probabilities, one for each observation for the first sample. If no covnergence occured this is NULL.
p2	The estimated probabilities, one for each observation for the second sample. If no covnergence occured this is NULL.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

References

Owen A. B. (2001). Empirical likelihood. Chapman and Hall/CRC Press.

See Also

ftests, ttests,, ttest

Examples

```
x <- rnorm(100)
y <- rnorm(200)
system.time( eel.test2(x, y) )
x <- rnorm(1000)
system.time( eel.test2(x, y) )
x <- rnorm(500)
y <- rexp(1000)
system.time( eel.test2(x, y) )
```

Variance of a vector *Variance (and standard deviation) of a vector*

Description

Variance (and standard deviation) of a vector.

Usage

```
Var(x, std = FALSE)
```

Arguments

x	A vector with data.
std	If you want the standard deviation set this to TRUE, otherwise leave it FALSE.

Details

This is a faster calculaiton of the usual variance of a matrix.

Value

The variance of the vector.

Author(s)

Michail Tsagris

R implementation and documentation: Michail Tsagris <mtsagris@yahoo.gr> and Manos Papadakis <papadakm95@gmail.com>.

See Also

colVars, cova

Examples

```
x <- rnorm(1000)
system.time( for (i in 1:100) Var(x) )
#system.time( for (i in 1:100) var(x) )
```

Vector allocation in a symmetric matrix

Vector allocation in a symmetric matrix

Description

Vector allocation in a symmetric matrix.

Usage

```
squareform(x)
```

Arguments

x	An numverical vector whose size must be the one that matches the dimensions of the final matrix. See examples.

Details

The functions is written in C++ in order to be as fast as possible.

Value

A symmetric matrix. The vector is allocated in the upper and in the lower part of the matrix. The diagonal is filled with zeros.

Author(s)

R implementation and documentation: Manos Papadakis <papadakm95@gmail.com>.

See Also

colShuffle, colVars, colmeans

Examples

```
x <- rnorm(1)
squareform(x)  ## OK
x <- rnorm(3)
squareform(x)  ## OK
x <- rnorm(4)
squareform(x)  ## not OK
```

Index