Package: minerva (via r-universe)

August 23, 2024

Version 1.5.10

Date 2021-06-19

Title Maximal Information-Based Nonparametric Exploration for Variable Analysis

Depends R (>= 3.3.0)

Imports parallel, Rcpp, stats, graphics

LinkingTo Rcpp, RcppArmadillo

Suggests testthat

Description Wrapper for 'minepy' implementation of Maximal Information-based Nonparametric Exploration statistics (MIC and MINE family). Detailed information of the ANSI C implementation of 'minepy' can be found at <http://minepy.readthedocs.io/en/latest>.

URL https://www.r-project.org, http://minepy.readthedocs.io/en/latest, http://www.exploredata.net

License GPL-3

Author Michele Filosi [aut, cre], Roberto Visintainer [aut], Davide Albanese [aut], Samantha Riccadonna [ctb], Giuseppe Jurman [ctb], Cesare Furlanello [ctb]

Maintainer Michele Filosi <michele.filosi@gmail.com>

Date/Publication 2013-01-07 17:53:58

RoxygenNote 7.1.1

Encoding UTF-8

LazyData true

Repository https://filosi.r-universe.dev

RemoteUrl https://github.com/filosi/minerva

RemoteRef HEAD

RemoteSha 2cf860713d0386cb8f1db361296d79b9f1084c40

Contents

`minerva-package` *The minerva package*

Description

Maximal Information-Based Nonparametric Exploration R Package for Variable Analysis. The package provides the `mine` function allowing the computation of Maximal Information-based Nonparametric Exploration statistics, firstly introduced in D. Reshef et al. (2011) *Detecting novel associations in large datasets*. Science 334, 6062 (`http://www.exploredata.net`). In particular, the package is an R wrapper for the C engine *cmine* (`http://minepy.readthedocs.io/en/latest/`).

Details

Summary:

Package:	minerva
Version:	1.4.3
Date:	2014-10-08
Depends:	R >= (2.14.0)
Enshances:	parallel
URL:	http://www.r-project.org, http://minepy.readthedocs.io/en/latest/, http://www.exploredata.net
License:	GPL-3

Index:

`Spellman`	Yeast Gene Expression Dataset
`mine`	MINE-family statistics
`minerva-package`	The minerva package

Author(s)

Michele Filosi [aut, cre], Roberto Visintainer [aut], Davide Albanese [aut], Samantha Riccadonna [ctb], Giuseppe Jurman [ctb], Cesare Furlanello [ctb]

Maintainer: Michele Filosi <filosi@fbk.eu>

References

D. Reshef, Y. Reshef, H. Finucane, S. Grossman, G. McVean, P. Turnbaugh, E. Lander, M. Mitzenmacher, P. Sabeti. (2011) *Detecting novel associations in large datasets*. Science 334, 6062 (http://www.exploredata.net).

D. Albanese, M. Filosi, R. Visintainer, S. Riccadonna, G. Jurman, C. Furlanello. *cmine, minerva & minepy: a C engine for the MINE suite an its R and Python wrappers.* http://minepy.readthedocs.io/en/latest/

minepy. Maximal Information-based Nonparametric Exploration in C and Python. (http://minepy.sourceforge.net)

`cstats` — *Compute statistics (MIC and normalized TIC) between each pair of the two collections of variables (convenience function). If n and m are the number of variables in X and Y respectively, then the statistic between the (row) i (for X) and j (for Y) is stored in* `mic[i, j]` *and* `tic[i, j]`.

Description

Compute statistics (MIC and normalized TIC) between each pair of the two collections of variables (convenience function). If n and m are the number of variables in X and Y respectively, then the statistic between the (row) i (for X) and j (for Y) is stored in `mic[i, j]` and `tic[i, j]`.

Usage

```
cstats(x, y, alpha = 0.6, C = 15, est = "mic_approx")
```

Arguments

`x`	Numeric Matrix of m-by-n with n variables and m samples.
`y`	Numeric Matrix of m-by-p with p variables and m samples.
`alpha`	number (0, 1.0] or >=4 if alpha is in (0,1] then B will be max(n^alpha, 4) where n is the number of samples. If alpha is >=4 then alpha defines directly the B parameter. If alpha is higher than the number of samples (n) it will be limited to be n, so B = min(alpha, n).
`C`	number (> 0) determines how many more clumps there will be than columns in every partition. Default value is 15, meaning that when trying to draw x grid lines on the x-axis, the algorithm will start with at most 15*x clumps.
`est`	string ("mic_approx", "mic_e") estimator. With est="mic_approx" the original MINE statistics will be computed, with est="mic_e" the equicharacteristic matrix is is evaluated and MIC_e and TIC_e are returned.

Value

list of two elements: MIC: the MIC statistic matrix (n x p). TIC: the normalized TIC statistic matrix (n x p).

Examples

```
x <- matrix(rnorm(2560), ncol=8, nrow=320)
y <- matrix(rnorm(1280), ncol=4, nrow=320)

mictic <- cstats(x, y, alpha=9, C=5, est="mic_e")
head(mictic)
```

---

mictools *Function that implements the* mictools *pipeline. In particular it computes the null and observed distribution of the* tic_e *measure*

Description

Function that implements the mictools pipeline. In particular it computes the null and observed distribution of the tic_e measure

Usage

```
mictools(x, alpha = 9, C = 5, seed = 0, nperm = 2e+05, p.adjust.method = "BH")
```

Arguments

x	a numeric matrix with N samples on the rows and M variables on the columns (NxM).
alpha	float (0, 1.0] or >=4 if alpha is in (0,1] then B will be max(n^alpha, 4) where n is the number of samples. If alpha is >=4 then alpha defines directly the B parameter. If alpha is higher than the number of samples (n) it will be limited to be n, so B = min(alpha, n) Default value is 0.6 (see Details).
C	a positive integer number, the C parameter of the mine statistic. See mine function for further details.
seed	seed for random number generation reproducibility
nperm	integer, number of permutation to perform
p.adjust.method	method for pvalue adjustment, see p.adjust for available methods.

Details

This is a function to implement the ‘mictools‘ pipeline. Differently from the python pipeline available on github we consider a data matrix of NxM with N samples by rows and M variables by columns as standard for R.

Value

A list of 5 named elements containing the following information of the computed statistic:

tic This is a vector with the null distribution of tic_e values based on the permutation.

nulldist Null distribution of the `tic_e` measure. It is a `data.frame` of 4 columns containing the histogram of the distribution of `tic_e` for each bin delimited by `BinStart` and `BinEnd`, the count for each bin `NullCount` and the cumulative distribution of the right tail area `NullCumSum`

obstic `data.frame` with the observed `tic_e` values, the indexes of the variables between the tic is computed. If the input matrix has column names then the names are reported in the dataframe, otherwise "Var<i>" is added for each variable.

obsdists `data.frame` similar to `nulldist` but with observed values of `tic_e`

pval data.frame with the pvalue computed for each comparison. The adjusted pvalue is also reported based on the method chosen with the parameter `p.adjust.method`

References

D. Albanese, S. Riccadonna, C. Donati, P. Franceschi (2018) _A practical tool for Maximal Information Coefficient Analysis_ GigaScience, 7, 4, doi:10.1093/gigascience/giy032

See Also

`p.adjust`, `hist`, `mine`

Examples

```
data(Spellman)
Spellman <- as.matrix(Spellman)
spellress <- mictools(Spellman[, 10:20], nperm=1000)

## Use a different pvalue correction method
spellressb <- mictools(Spellman[,10:20], nperm=1000, seed=1234, p.adjust.method="bonferroni")

## Distribution of tic_e null
hist(spellress$tic, breaks=100, main="Tic_e null distribution")
barplot(spellress$nulldist$NullCount)

## Distribution of the observed tic
hist(spellress$obstic$TIC)
barplot(spellress$obsdist$Count)

## Distribution of empirical pvalues
hist(spellress$pval$pval, breaks=50)
```

`mictools_null` *This set of functions are helper function to compute null distribution of the* `tic_e` *and* `tic_e` *observed distribution from a matrix*

Description

This set of functions are helper function to compute null distribution of the `tic_e` and `tic_e` observed distribution from a matrix

Usage

```
mictools_null(x, alpha = 9, C = 5, nperm = 200000L, seed = 0L)
```

Arguments

`x`	matrix N x M with M variables and N samples
`alpha`	numeric value representing parameter for the mine statistic see `mine`
`C`	c parameter for the mine statistic see `mine`
`nperm`	numper of permutation
`seed`	integer to set the starting seed for random number generation (for reproducibility).

Value

It returns a vector of `nperm` `tic_e` values.

Functions

- `mictools_null`: compute the `tic_e` null distribution

See Also

`mictools`

`mic_strength` *Compute the association strengh*

Description

This function uses the null distribution of the `tic_e` computed with the function `mictools`. Based on the available pvalue and the permutation null distribution it identifies reliable association between variables.

Usage

```
mic_strength(x, pval, alpha = NULL, C = 5, pthr = 0.05, pval.col = NULL)
```

Arguments

x	a numeric matrix with N samples on the rows and M variables on the columns (NxM).
pval	a data.frame with pvalues for each pair of association of the x input matrix. It should contain two colums with the indices of the computed association according to the x input matrix
alpha	float (0, 1.0] or >=4 if alpha is in (0,1] then B will be max(n^alpha, 4) where n is the number of samples. If alpha is >=4 then alpha defines directly the B parameter. If alpha is higher than the number of samples (n) it will be limited to be n, so B = min(alpha, n) Default value is 0.6 (see Details).
C	a positive integer number, the C parameter of the mine statistic. See mine function for further details.
pthr	threshold on pvalue for measure to consider for computing mic_e
pval.col	an integer or character or vector relative to the columns of pval dataframe respectively for pvalue, association between variable 1, variable 2 in the x input matrix. See Details for further information.

Details

The method implemented here is a wrapper for the original method published by Albanese et al. (2018). The python version is available at https://github.com/minepy/mictools.

This function should be called after the estimation of the null distribution of tic_e scores based on permutations of the input data.

The mic association is computed only for the variables for which the pvalue in the pval data.frame is less then the threshold set with the pthr input parameter. We assume the first column of the pval data.frame contains the pvalue, this value can be changed using the pval.col[1] parameter.

The pval.col parameter, by default takes the first three columns in the pval data.frame, in particular the first column containing the pvalues of the association between variable in column pval.col[2] and pval.col[3]. If a character vector is provided names in pval.col are matched with the names in pval data.frame. If NULL is passed it is assumed the first column contains pvalue, while the 2 and 3 the index or name of the variable in x. If one value is passed it refers to the pvalue column and the consecutive two columns are assume to contain variable indexes.

Value

A dataframe with the tic_e Pvalue, the mic value and the column identifier regarding the input matrix x of the variables of which the association is computed.

See Also

mine, mictools, p.adjust

Examples

```
data(Spellman)
mydata <- as.matrix(Spellman[, 10:20])
ticenull <- mictools(mydata, nperm=1000)

## Use the nominal pvalue:
ms <- mic_strength(mydata, pval=ticenull$pval, alpha=NULL, pval.col = c(1, 4,5))

## Use the adjusted pvalue:
ms <- mic_strength(mydata, pval=ticenull$pval, alpha=NULL, pval.col = c(6, 4,5))

ms

## Not run:
## Use qvalue
require(qvalue)
qobj <- qvalue(ticenull$pval$pval)
ticenull$pval$qvalue <- qobj$qvalue
ms <- mic_strength(mydata, pval=ticenull$pval, alpha=NULL, pval.col = c("qvalue", "Var1", "Var2"))

## Get the data from mictools repository

lnf <- "https://raw.githubusercontent.com/minepy/mictools/master/examples/datasaurus.txt"
datasaurus <- read.table(lnf, header=TRUE, row.names = 1, stringsAsFactors = FALSE)
datasaurus <- t(datasaurus)
ticenull <- mictools(datasaurus, nperm=200000)
micres <- mic_strength(mydata, ticenull$pval, pval.col=c(6, 4, 5))

## Plot distribution of pvalues
hist(ticenull$pval, breaks=50, freq=FALSE)

## Plot distribution of tic_e values
hist(ticenull$tic)

## Correct pvalues using qvalue package
require(qvalue)
require(ggplot2)
qobj <- qvalue(ticenull$pval$pval)
ticenull$pval$qvalue <- qobj$qvalue
micres <- mic_strength(datasaurus, ticenull$pval, pval.col=c("qvalue", "Var1", "Var2"))

hist(qobj$qvalue)

df <- data.frame(pi0.labmda=qobj$pi0.lambda, lambda=qobj$lambda, pi0.smooth=qobj$pi0.smooth)
gp0 <- ggplot(df, aes(lambda, pi0.labmda)) + geom_point()
gp0 <- gp0 + geom_line(aes(lambda, pi0.smooth))
gp0 <- gp0 + geom_hline(yintercept = qobj$pi0, linetype="dashed", col="red")

## End(Not run)
```

`mine` *MINE family statistics Maximal Information-Based Nonparametric Exploration (MINE) statistics.* `mine` *computes the MINE family measures between two variables.*

Description

MINE family statistics Maximal Information-Based Nonparametric Exploration (MINE) statistics. `mine` computes the MINE family measures between two variables.

Usage

```
mine(
  x,
  y = NULL,
  master = NULL,
  alpha = 0.6,
  C = 15,
  n.cores = 1,
  var.thr = 1e-05,
  eps = NULL,
  est = "mic_approx",
  na.rm = FALSE,
  use = "all.obs",
  normalization = FALSE,
  ...
)
```

Arguments

`x`	a numeric vector (of size *n*), matrix or data frame (which is coerced to matrix).
`y`	NULL (default) or a numeric vector of size *n* (*i.e.*, with compatible dimensions to x).
`master`	an optional vector of indices (numeric or character) to be given when y is not set, otherwise master is ignored. It can be either one column index to be used as reference for the comparison (versus all other columns) or a vector of column indices to be used for computing all mutual statistics.
`alpha`	float (0, 1.0] or >=4 if alpha is in (0,1] then B will be max(n^alpha, 4) where n is the number of samples. If alpha is >=4 then alpha defines directly the B parameter. If alpha is higher than the number of samples (n) it will be limited to be n, so B = min(alpha, n) Default value is 0.6 (see Details).
`C`	an optional number determining the starting point of the *X-by-Y* search-grid. When trying to partition the *x*-axis into *X* columns, the algorithm will start with at most C*X clumps*. Default value is 15 (see Details).

n.cores	ooptional number of cores to be used in the computations, when master is specified. It requires the **parallel** package, which provides support for parallel computing, released with R >= 2.14.0. Defaults is 1 (*i.e.*, not performing parallel computing).
var.thr	minimum value allowed for the variance of the input variables, since mine can not be computed in case of variance close to 0. Default value is 1e-5. Information about failed check are reported in *var_thr.log* file.
eps	integer in [0,1]. If 'NULL' (default) it is set to 1-MIC. It can be set to zero for noiseless functions, but the default choice is the most appropriate parametrization for general cases (as stated in Reshef et al. SOM). It provides robustness.
est	Default value is "mic_approx". With est="mic_approx" the original MINE statistics will be computed, with est="mic_e" the equicharacteristic matrix is is evaluated and the mic() and tic() methods will return MIC_e and TIC_e values respectively.
na.rm	boolean. This variable is passed directly to the cor-based functions. See cor for further details.
use	Default value is "all.obs". This variable is passed directly to the cor-based functions. See cor for further details.
normalization	logical whether to use normalization when computing tic measure. Ignored for other measures. Default to FALSE.
...	currently ignored

Details

mine is an R wrapper for the C engine *cmine* (http://minepy.readthedocs.io/en/latest/), an implementation of Maximal Information-Based Nonparametric Exploration (MINE) statistics. The MINE statistics were firstly detailed in D. Reshef et al. (2011) *Detecting novel associations in large datasets*. Science 334, 6062 (http://www.exploredata.net).

Here we recall the main concepts of the MINE family statistics. Let $D = (x, y)$ be the set of n ordered pairs of elements of x and y. The data space is partitioned in an *X-by-Y* grid, grouping the x and y values in X and Y bins respectively.

The **Maximal Information Coefficient (MIC)** is defined as

$$\text{MIC}(D) = \max_{XY<B(n)} M(D)_{X,Y} = \max_{XY<B(n)} \frac{I^*(D,X,Y)}{log(\min X, Y)},$$

where $B(n) = n^\alpha$ is the search-grid size, $I^*(D,X,Y)$ is the maximum mutual information over all grids *X-by-Y*, of the distribution induced by D on a grid having X and Y bins (where the probability mass on a cell of the grid is the fraction of points of D falling in that cell). The other statistics of the MINE family are derived from the mutual information matrix achieved by an *X-by-Y* grid on D.

The **Maximum Asymmetry Score (MAS)** is defined as

$$\text{MAS}(D) = \max_{XY<B(n)} |M(D)_{X,Y} - M(D)_{Y,X}|.$$

The **Maximum Edge Value (MEV)** is defined as

$$\text{MEV}(D) = \max_{XY<B(n)} \{M(D)_{X,Y} : X = 2 \; or \; Y = 2\}.$$

The **Minimum Cell Number (MCN)** is defined as

$$\text{MCN}(D, \epsilon) = \min_{XY<B(n)} \{\log(XY) : M(D)_{X,Y} \geq (1 - \epsilon)MIC(D)\}.$$

More details are provided in the supplementary material (SOM) of the original paper.

The MINE statistics can be computed for two numeric vectors `x` and `y`. Otherwise a matrix (or data frame) can be provided and two options are available according to the value of `master`. If `master` is a column identifier, then the MINE statistics are computed for the *master* variable versus the other matrix columns. If `master` is a set of column identifiers, then all mutual MINE statistics are computed among the column subset. `master`, `alpha`, and `C` refers respectively to the *style*, *exp*, and *c* parameters of the original *java* code. In the original article, the authors state that the default value $\alpha = 0.6$ (which is the exponent of the search-grid size $B(n) = n^{\alpha}$) has been empirically chosen. It is worthwhile noting that `alpha` and `C` are defined to obtain an heuristic approximation in a reasonable amount of time. In case of small sample size (*n*) it is preferable to increase `alpha` to 1 to obtain a solution closer to the theoretical one.

Value

The Maximal Information-Based Nonparametric Exploration (MINE) statistics provide quantitative evaluations of different aspects of the relationship between two variables. In particular `mine` returns a list of 5 statistics:

`MIC` **Maximal Information Coefficient.**
It is related to the relationship strenght and it can be interpreted as a correlation measure. It is symmetric and it ranges in [0,1], where it tends to 0 for statistically independent data and it approaches 1 in probability for noiseless functional relationships (more details can ben found in the original paper).

`MAS` **Maximum Asymmetry Score.**
It captures the deviation from monotonicity. Note that MAS $<$ MIC.
Note: it can be useful for detecting periodic relationships (unknown frequencies).

`MEV` **Maximum Edge Value.**
It measures the closeness to being a function. Note that MEV $\leq$ MIC.

`MCN` **Minimum Cell Number.**
It is a complexity measure.

`MIC-R2` It is the difference between the MIC value and the Pearson correlation coefficient.

When computing `mine` between two numeric vectors `x` and `y`, the output is a list of 5 numeric values. When `master` is provided, `mine` returns a list of 5 matrices having `ncol` equal to *m*. In particular, if `master` is a single value, then `mine` returns a list of 5 matrices having 1 column, whose rows correspond to the MINE measures between the *master* column versus all. Instead if `master` is a vector of *m* indices, then `mine` output is a list of 5 *m-by-m* matrices, whose element *i,j* corresponds to the MINE statistics computed between the *i* and *j* columns of `x`.

Author(s)

Michele Filosi and Roberto Visintainer

References

D. Reshef, Y. Reshef, H. Finucane, S. Grossman, G. McVean, P. Turnbaugh, E. Lander, M. Mitzenmacher, P. Sabeti. (2011) *Detecting novel associations in large datasets*. Science 334, 6062
http://www.exploredata.net
(SOM: Supplementary Online Material at https://science.sciencemag.org/content/suppl/2011/12/14/334.6062.1518.DC1)

D. Albanese, M. Filosi, R. Visintainer, S. Riccadonna, G. Jurman, C. Furlanello. *minerva and minepy: a C engine for the MINE suite and its R, Python and MATLAB wrappers*. Bioinformatics (2013) 29(3): 407-408, doi:10.1093/bioinformatics/bts707.

minepy. Maximal Information-based Nonparametric Exploration in C and Python.
http://minepy.sourceforge.net

Examples

```
A <- matrix(runif(50),nrow=5)
mine(x=A, master=1)
mine(x=A, master=c(1,3,5,7,8:10))

x <- runif(10); y <- 3*x+2; plot(x,y,type="l")
mine(x,y)
# MIC = 1
# MAS = 0
# MEV = 1
# MCN = 2
# MIC-R2 = 0

set.seed(100); x <- runif(10); y <- 3*x+2+rnorm(10,mean=2,sd=5); plot(x,y)
mine(x,y)
# rounded values of MINE statistics
# MIC = 0.61
# MAS = 0
# MEV = 0.61
# MCN = 2
# MIC-R2 = 0.13

t <-seq(-2*pi,2*pi,0.2); y1 <- sin(2*t); plot(t,y1,type="l")
mine(t,y1)
# rounded values of MINE statistics
# MIC = 0.66
# MAS = 0.37
# MEV = 0.66
# MCN = 3.58
# MIC-R2 = 0.62

y2 <- sin(4*t); plot(t,y2,type="l")
```

```
mine(t,y2)
# rounded values of MINE statistics
# MIC = 0.32
# MAS = 0.18
# MEV = 0.32
# MCN = 3.58
# MIC-R2 = 0.31

# Note that for small n it is better to increase alpha
mine(t,y1,alpha=1)
# rounded values of MINE statistics
# MIC = 1
# MAS = 0.59
# MEV = 1
# MCN = 5.67
# MIC-R2 = 0.96

mine(t,y2,alpha=1)
# rounded values of MINE statistics
# MIC = 1
# MAS = 0.59
# MEV = 1
# MCN = 5
# MIC-R2 = 0.99

# Some examples from SOM
x <- runif(n=1000, min=0, max=1)

# Linear relationship
y1 <- x; plot(x,y1,type="l"); mine(x,y1)
# MIC = 1
# MAS = 0
# MEV = 1
# MCN = 4
# MIC-R2 = 0

# Parabolic relationship
y2 <- 4*(x-0.5)^2; plot(sort(x),y2[order(x)],type="l"); mine(x,y2)
# rounded values of MINE statistics
# MIC = 1
# MAS = 0.68
# MEV = 1
# MCN = 5.5
# MIC-R2 = 1

# Sinusoidal relationship (varying frequency)
y3 <- sin(6*pi*x*(1+x)); plot(sort(x),y3[order(x)],type="l"); mine(x,y3)
# rounded values of MINE statistics
# MIC = 1
# MAS = 0.85
# MEV = 1
# MCN = 4.6
# MIC-R2 = 0.96
```

```
# Circle relationship
t <- seq(from=0,to=2*pi,length.out=1000)
x4 <- cos(t); y4 <- sin(t); plot(x4, y4, type="l",asp=1)
mine(x4,y4)
# rounded values of MINE statistics
# MIC = 0.68
# MAS = 0.01
# MEV = 0.32
# MCN = 5.98
# MIC-R2 = 0.68

data(Spellman)
res <- mine(Spellman,master=1,n.cores=1)

## Not run: ## example of multicore computation
res <- mine(Spellman,master=1,n.cores=parallel::detectCores()-1)
## End(Not run)
```

`mine_stat` *This is an helper function to compute one* `mine` *statistic. It take two vectors of the same dimension as an input.*

Description

This is an helper function to compute one `mine` statistic. It take two vectors of the same dimension as an input.

Usage

```
mine_stat(
  x,
  y,
  alpha = 0.6,
  C = 15,
  est = "mic_approx",
  measure = "mic",
  eps = NA_real_,
  p = -1,
  norm = FALSE
)
```

Arguments

`x`	Numeric Vector of size `n`
`y`	Numeric Vector of size `n`
`alpha`	numeric value representing parameter for the mine statistic see `mine`
`C`	c parameter for the mine statistic see `mine`

est	character estimation parameter for the mine statistic. Possible values are "mic_approx" or "mic_e"
measure	integer indicating which measure to return available measures are: mic, mas, mev, mcn, tic, gmic. The string could be also uppercase. For measure mic-r2 see details.
eps	eps value for MCN statistic should be in (0,1). If NA (default) is passed then the normal MCN statistic is returned.
p	probability for the generalized mic
norm	boolean if require normalization between 0 and 1 for the tic statistic

Details

This is a wrapper function to compute the mine statistic between two variables. for more details on the available measure and the meaning of the other parameters see also the documentation for the mine function.

For measure mic-r2 use the Pearson R coefficient score cor and the measure mic. See the example below.

See Also

mine

Examples

```
x <- runif(10); y <- 3*x+2;
mine_stat(x,y, measure="mic")

## Measure mic-r2
x <- matrix(rnorm(20), ncol=2, nrow=10)
mmic <- mine_stat(x[,1], x[,2], measure="mic")
r2 <- cor(x[,1], x[,2])

mmic - r2**2
```

---

pstats *Compute pairwise statistics (MIC and normalized TIC) between variables (convenience function).*

Description

For each statistic, the upper triangle of the matrix is stored by row (condensed matrix). If m is the number of variables, then for i < j < m, the statistic between (col) i and j is stored in k = m*i - i*(i+1)/2 - i - 1 + j. The length of the vectors is n = m*(m-1)/2.

Usage

```
pstats(x, alpha = 0.6, C = 15, est = "mic_approx")
```

Arguments

x	Numeric Matrix of m-by-n with n variables and m samples.
alpha	number (0, 1.0] or >=4 if alpha is in (0,1] then B will be max(n^alpha, 4) where n is the number of samples. If alpha is >=4 then alpha defines directly the B parameter. If alpha is higher than the number of samples (n) it will be limited to be n, so B = min(alpha, n).
C	number (> 0) determines how many more clumps there will be than columns in every partition. Default value is 15, meaning that when trying to draw x grid lines on the x-axis, the algorithm will start with at most 15*x clumps.
est	string ("mic_approx", "mic_e") estimator. With est="mic_approx" the original MINE statistics will be computed, with est="mic_e" the equicharacteristic matrix is is evaluated and MIC_e and TIC_e are returned.

Value

A matrix of (n x (n-1)/2) rows and 4 columns. The first and second column are the indexes relative to the columns in the input matrix x for which the statistic is computed for. Column 3 contains the MIC statistic, while column 4 contains the normalized TIC statistic.

Examples

```
## Create a matrix of random numbers
## 10 variables x 100 samples
x <- matrix(rnorm(1000), ncol=10)
res <- pstats(x)

head(res)
```

Spellman *CDC15 Yeast Gene Expression Dataset*

Description

The Spellman dataset provides the gene expression data measured (on a custom platform) in *Saccharomyces cerevisiae* cell cultures that have been synchronized at different points of the cell cycle by using a temperature-sensitive mutation (*cdc15-2*), which arrestes cells late in mitosis at the restrictive temperature (it can cause heat-shock).

Usage

```
Spellman
```

Format

23 rows x 4382 columns: 4381 transcripts (columns 2:4382) measured at 23 timepoints (column 1).

Source

The original data were published by Spellman and colleagues in Mol. Biol. Cell (1998) as the Botstein dataset. Here we include the version of the dataset as processed by Reshef and colleagues for the MINE statistics original article published in Science (2011) (details are provided in the supplementary material).

References

D. Reshef, Y. Reshef, H. Finucane, S. Grossman, G. McVean, P. Turnbaugh, E. Lander, M. Mitzenmacher, P. Sabeti. (2011) *Detecting novel associations in large datasets*. Science 334, 6062 (http://www.exploredata.net).

P. T. Spellman, G. Sherlock, M. Q. Zhang, V. R. Iyer, K. Anders, M. B. Eisen, P. O. Brown, D. Botstein, B. Futcher. (1998) *Comprehensive Identification of Cell Cycle–regulated Genes of the Yeast Saccharomyces cerevisiae by Microarray Hybridization*. Mol. Biol. Cell, 9:12 3273–3297.

Index