Package 'minerva'

December 14, 2018

Version 1.5

Date 2018-11-30

Title Maximal Information-Based Nonparametric Exploration for Variable Analysis

Depends R (>= 2.14.0)

Imports parallel, Rcpp, stats

LinkingTo Rcpp

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>

Repository CRAN

Date/Publication 2018-12-14 12:20:12 UTC

RoxygenNote 6.1.1

Encoding UTF-8

NeedsCompilation yes

R topics documented:

`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)

`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", ...)
```

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`	an optional number of cells allowed in the *X-by-Y* search-grid. 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.
...	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

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

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

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

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 http://www.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_compute` *Function to compute one statistic at time*

Description

Function to compute one statistic at time

Usage

```
mine_compute(x, y, alpha = 0.6, C = 15, est = "mic_approx",
  measure = 1L, eps = 0, p = -1, norm = FALSE)
```

Arguments

`x`	Numeric Vector
`y`	Numeric Vector
`alpha`	alpha parameter for the mine statistic
`C`	c parameter for the mine statistic
`est`	estimation parameter for the mine statistic
`measure`	character which measure to return
`eps`	eps value for MCN statistic
`p`	probability for the generalized mic
`norm`	boolean if require normalization between 0 and 1 for the tic statistic

`mine_compute_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

```
mine_compute_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`	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).
`C`	float (> 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).

`mine_compute_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

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

Arguments

`x`	Numeric matrix of m-by-n of n variables and m samples
`alpha`	alpha parameter for the mine statistic
`C`	c parameter for the mine statistic
`est`	estimation parameter for the mine statistic

Value

Matrix (n x (n-1)/2) by 4. The first and second column indicate the indexes relative of the columns in the inut matrix the statistic is computed for. Column 3 contains the MIC statistic, while column 4 contains the normalized TIC statistic.

`mine_stat` *Function to compute one measure at time*

Description

Function to compute one measure at time

Usage

```
mine_stat(x, y, alpha = 0.6, C = 15, est = "mic_approx",
  measure = "mic", eps = NULL, p = -1, norm = 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).
alpha	an optional number of cells allowed in the *X-by-Y* search-grid. 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).
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.
measure	character indicating the measure to extract. Default value "mic". Availiable values are: "mic", "mas", "mev", "mcn", "tic", "gmic".
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.
p	probability for the generalized 'mic'
norm	boolean if require normalization between 0 and 1 for the 'tic' statistic

Examples

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

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