Package ‘statcomp’

September 1, 2016

Title Statistical Complexity and Information Measures for Time Series Analysis

Version 0.0.1.1000

Author Sebastian Sippel [aut, cre], Holger Lange [aut], Fabian Gans [aut]

Maintainer Sebastian Sippel <ssippel@bgc-jena.mpg.de>

Description An implementation of local and global statistical complexity measures (aka Information Theory Quantifiers, ITQ) for time series analysis based on ordinal statistics (Bandt and Pompe (2002) <DOI:10.1103/PhysRevLett.88.174102>). Several distance measures that operate on ordinal pattern distributions, auxiliary functions for ordinal pattern analysis, and generating functions for stochastic and deterministic-chaotic processes for ITQ testing are provided.

Depends R (>= 2.7.0)

License GPL-2

LazyData true

Imports stats, zoo

NeedsCompilation yes

Repository CRAN

Date/Publication 2016-09-01 18:30:15

R topics documented:

`adjust_pattern` *A function to create new pattern-coding schemes for the Fisher Information.*

Description

Adjusts and reorders a pattern ordering matrix.

Usage

```
adjust_pattern(pattern_matrix, adjustment)
```

Arguments

`pattern_matrix`	A numeric matrix that specifies the pattern to be transformed into the position vector. ATTENTION: Pattern should be in the ranks permutation notation, otherwise does not really make sense.
`adjustment`	A character vector, either adjustment = "jumps" or adjustment = "bitflips" that denotes the sorting type

Details

This function reorders permutations based on "jumps" or based on "bitflips".

Value

A numeric matrix that contains the permutation matrix.

Author(s)

Sebastian Sippel

References

Sebastian Sippel, 2014. Master Thesis. University of Bayreuth.

`fis` *A (low-level) function to compute the Fisher-information*

Description

The function computes the Fisher information, i.e. a local information measure based on two different discretizations.

Usage

```
fis(opd, discretization)
```

Arguments

`opd`	A numeric vector that details an ordinal pattern distribution in a user-specified permutation coding scheme.
`discretization`	The discretization scheme to use, either 'Olivares.2012' or 'Ferri.2009'

Details

The Fisher information is a local information and complexity measure, computed based on the ordinal pattern distribution. The Fisher information is based on local gradients, hence it is sensitive to the permutation coding scheme. Options for discretization: 'Olivares.2012' or 'Ferri.2009', following Fisher Information discretization schemes in the respective publications.

Value

The normalized Fisher information measure in the range [0, 1].

Author(s)

Sebastian Sippel

References

Olivares, F., Plastino, A. and Rosso, O.A., 2012. Ambiguities in Bandt-Pompe's methodology for local entropic quantifiers. Physica A: Statistical Mechanics and its Applications, 391(8), pp.2518-2526. Ferri, G.L., Pennini, F. and Plastino, A., 2009. LMC-complexity and various chaotic regimes. Physics Letters A, 373(26), pp.2210-2214.

Examples

```
x = arima.sim(model=list(ar = 0.3), n = 10^4)
opd = ordinal_pattern_distribution(x = x, ndemb = 6)
fis(opd = opd)
```

---

generate_lehmerperm_matrix

A function to generate the Lehmer permutation ordering.

Description

Generates all permutations of a given embedding dimension, ordered according to the Lehmer coding scheme.

Usage

```
generate_lehmerperm_matrix(ndemb)
```

Arguments

ndemb	The embedding dimension.

Details

This function converts ranks to indices and back.

Value

A numeric matrix that contains the Lehmer permutation pattern.

Author(s)

Sebastian Sippel

References

http://www.keithschwarz.com/interesting/code/?dir=factoradic-permutation

`global_complexity` *A function to compute global information and complexity measures for time series*

Description

This is a high-level function that calculates global complexity measures directly from a given time series or ordinal pattern distribution.

Usage

```
global_complexity(x = NA, opd = NA, ndemb)
```

Arguments

`x`	(OPTIONAL) If opd is not specified, a time series vector x must be specified
`opd`	A numeric vector that details an ordinal pattern distribution in a user-specified permutation coding scheme.
`ndemb`	(OPTIONAL) If x is given, the embedding dimension (ndemb) is required.

Details

This function calculates the following global measures of complexity and information:

- Permutation Entropy (PE, cf. Bandt and Pompe, 2002)
- Permutation Statistical complexity (MPR complexity, cf. Martin, Plastino and Rosso, 2006)
- Number of "forbiden patterns" (cf. Amigo 2010)

Value

A named vector containing the three global complexity measures.

Author(s)

Sebastian Sippel

References

Bandt, C. and Pompe, B., 2002. Permutation entropy: a natural complexity measure for time series. Physical review letters, 88(17), p.174102. Martin, M.T., Plastino, A. and Rosso, O.A., 2006. Generalized statistical complexity measures: Geometrical and analytical properties. Physica A: Statistical Mechanics and its Applications, 369(2), pp.439-462. Amigo, J., 2010. Permutation complexity in dynamical systems: ordinal patterns, permutation entropy and all that. Springer Science & Business Media.

Examples

```
x = arima.sim(model=list(ar = 0.3), n = 10^4)
global_complexity(x = x, ndemb = 6)
# or:
opd = ordinal_pattern_distribution(x = x, ndemb = 6)
global_complexity(opd = opd, ndemb = 6)
```

---

hellinger_distance — *Distance measure between ordinal pattern distributions: Hellinger distance*

Description

Compute the Hellinger Distance

Usage

```
hellinger_distance(p, q)
```

Arguments

p	An ordinal pattern distribution
q	A second ordinal pattern distribution to compare against p.

Details

This function returns a distance measure.

Value

A vector of length 1.

Author(s)

Sebastian Sippel

References

none

Examples

```
p = ordinal_pattern_distribution(rnorm(10000), ndemb = 5)
q = ordinal_pattern_distribution(arima.sim(model=list(ar=0.9), n= 10000), ndemb = 5)
hellinger_distance(p=p, q = q)
```

henon_map *A function to generate a time series from the Henon Map*

Description

Generates a time series from the Henon map

Usage

```
henon_map(N, a, b, startx="rand", starty="rand", disregard_N=0)
```

Arguments

N	length of the time series that is to be generated
a	Henon map parameter a
b	Henon map parameter b
startx	start value in x direction. Default is to random.
starty	start value in y direction. Default is to random.
disregard_N	Number of values at the beginning of the series to disregard

Value

A vector of length N

Author(s)

Sebastian Sippel

References

Schuster, H.G., 1988. Deterministic chaos. An Introduction.

Examples

```
henon_map(N = 10^4, a=1.4, b=0.3)
```

`jensen_shannon_divergence` *Generalized disequilibrium measure for ordinal pattern distributions based on the Jensen-Shannon Divergence*

Description

Computes a normalized form of the Jensen-Shannon Divergence

Usage

```
jensen_shannon_divergence(p, q="unif")
```

Arguments

p	An ordinal pattern distribution
q	A second ordinal pattern distribution to compare against p, or a character vector q="unif" (comparison of p to uniform distribution)

Details

This function returns a distance measure.

Value

A vector of length 1.

Author(s)

Sebastian Sippel

References

Martin, M.T., Plastino, A. and Rosso, O.A., 2006. Generalized statistical complexity measures: Geometrical and analytical properties. Physica A: Statistical Mechanics and its Applications, 369(2), pp.439-462.

Examples

```
p = ordinal_pattern_distribution(rnorm(10000), ndemb = 5)
q = ordinal_pattern_distribution(arima.sim(model=list(ar=0.9), n= 10000), ndemb = 5)
jensen_shannon_divergence(p = p, q = q)
```

limit_curves — *Limit curves in the Entropy-Complexity plane*

Description

Compute the limit curves in the Entropy Complexity plane

Usage

```
limit_curves(ndemb, fun = "min")
```

Arguments

`ndemb`	Embedding dimension
`fun`	Whether the upper (max) or lower (min) limit curve should be computed

Details

This function returns the respective limit curve.

Value

A list with two entries

Author(s)

Sebastian Sippel

References

none

logistic_map — *A function to generate a time series from the logistic map*

Description

Generates a time series from the logistic map

Usage

```
logistic_map(N, r, start="rand", disregard_N=0)
```

Arguments

`N`	length of the time series that is to be generated
`r`	logistic map parameter, must be in the range [0,4]
`start`	start value. Default is to random.
`disregard_N`	Number of values at the beginning of the series to disregard

Value

A vector of length N

Author(s)

Sebastian Sippel

References

May, R.M., 1976. Simple mathematical models with very complicated dynamics. Nature, 261(5560), pp.459-467.

Examples

```
logistic_map(N = 10^4, r=4)
```

`maxd3` *Maximum curve of time-causal entropy-complexity plane at ndemb=3*

Description

Maximum curve of time-causal entropy-complexity plane at ndemb=3

Usage

```
maxd3
```

Format

A data frame with 494 rows and 2 columns:

x x-values of minimum curve if ndemb==3

y y-values of minimum curve if ndemb==3 ...

Source

Computed based on Martin, M.T., Plastino, A. and Rosso, O.A., 2006. Generalized statistical complexity measures: Geometrical and analytical properties. Physica A: Statistical Mechanics and its Applications, 369(2), pp.439-462.

maxd4 *Maximum curve of time-causal entropy-complexity plane at ndemb=4*

Description

Maximum curve of time-causal entropy-complexity plane at ndemb=4

Usage

```
maxd4
```

Format

A data frame with 2139 rows and 2 columns:

x x-values of minimum curve if ndemb==4

y y-values of minimum curve if ndemb==4 ...

Source

Computed based on Martin, M.T., Plastino, A. and Rosso, O.A., 2006. Generalized statistical complexity measures: Geometrical and analytical properties. Physica A: Statistical Mechanics and its Applications, 369(2), pp.439-462.

maxd5 *Maximum curve of time-causal entropy-complexity plane at ndemb=5*

Description

Maximum curve of time-causal entropy-complexity plane at ndemb=5

Usage

```
maxd5
```

Format

A data frame with 4151 rows and 2 columns:

x x-values of minimum curve if ndemb==5

y y-values of minimum curve if ndemb==5 ...

Source

Computed based on Martin, M.T., Plastino, A. and Rosso, O.A., 2006. Generalized statistical complexity measures: Geometrical and analytical properties. Physica A: Statistical Mechanics and its Applications, 369(2), pp.439-462.

`maxd6` *Maximum curve of time-causal entropy-complexity plane at ndemb=6*

Description

Maximum curve of time-causal entropy-complexity plane at ndemb=6

Usage

```
maxd6
```

Format

A data frame with 3438 rows and 2 columns:

x x-values of minimum curve if ndemb==6

y y-values of minimum curve if ndemb==6 ...

Source

Computed based on Martin, M.T., Plastino, A. and Rosso, O.A., 2006. Generalized statistical complexity measures: Geometrical and analytical properties. Physica A: Statistical Mechanics and its Applications, 369(2), pp.439-462.

`mind3` *Minimum curve of time-causal entropy-complexity plane at ndemb=3*

Description

Minimum curve of time-causal entropy-complexity plane at ndemb=3

Usage

```
mind3
```

Format

A data frame with 500 rows and 2 columns:

x x-values of minimum curve if ndemb==3

y y-values of minimum curve if ndemb==3 ...

Source

Computed based on Martin, M.T., Plastino, A. and Rosso, O.A., 2006. Generalized statistical complexity measures: Geometrical and analytical properties. Physica A: Statistical Mechanics and its Applications, 369(2), pp.439-462.

mind4 — *Minimum curve of time-causal entropy-complexity plane at ndemb=4*

Description

Minimum curve of time-causal entropy-complexity plane at ndemb=4

Usage

```
mind4
```

Format

A data frame with 500 rows and 2 columns:

x x-values of minimum curve if ndemb==4

y y-values of minimum curve if ndemb==4 ...

Source

Computed based on Martin, M.T., Plastino, A. and Rosso, O.A., 2006. Generalized statistical complexity measures: Geometrical and analytical properties. Physica A: Statistical Mechanics and its Applications, 369(2), pp.439-462.

mind5 — *Minimum curve of time-causal entropy-complexity plane at ndemb=5*

Description

Minimum curve of time-causal entropy-complexity plane at ndemb=5

Usage

```
mind5
```

Format

A data frame with 500 rows and 2 columns:

x x-values of minimum curve if ndemb==5

y y-values of minimum curve if ndemb==5 ...

Source

Computed based on Martin, M.T., Plastino, A. and Rosso, O.A., 2006. Generalized statistical complexity measures: Geometrical and analytical properties. Physica A: Statistical Mechanics and its Applications, 369(2), pp.439-462.

mind6 — *Minimum curve of time-causal entropy-complexity plane at ndemb=6*

Description

Minimum curve of time-causal entropy-complexity plane at ndemb=6

Usage

```
mind6
```

Format

A data frame with 500 rows and 2 columns:

x x-values of minimum curve if ndemb==6

y y-values of minimum curve if ndemb==6 ...

Source

Computed based on Martin, M.T., Plastino, A. and Rosso, O.A., 2006. Generalized statistical complexity measures: Geometrical and analytical properties. Physica A: Statistical Mechanics and its Applications, 369(2), pp.439-462.

MPR_complexity — *A function to compute the MPR-complexity*

Description

The function computes the MPR complexity, i.e. a generalized (global) complexity measure based on the Jenson-Shannon divergence.

Usage

```
MPR_complexity(opd)
```

Arguments

`opd`	A numeric vector that details an ordinal pattern distribution.

Details

Generalized complexity measures combine an information measure (i.e. entropy) with the distance of the distribution from the uniform distribution ("disequilibrium"). As a global measure, MPR-complexity is insensitive to the permutation coding scheme.

Value

The normalized MPR complexity measure in the range [0, 1].

Author(s)

Sebastian Sippel

References

Martin, M.T., Plastino, A. and Rosso, O.A., 2006. Generalized statistical complexity measures: Geometrical and analytical properties. Physica A: Statistical Mechanics and its Applications, 369(2), pp.439-462.

Examples

```
x = arima.sim(model=list(ar = 0.3), n = 10^4)
opd = ordinal_pattern_distribution(x = x, ndemb = 6)
MPR_complexity(opd)
```

---

nbitflips — *A function to compute bitflip statistics and time series*

Description

Computation of bitflip statistics of a time series

Usage

```
nbitflips(x, ndemb)
```

Arguments

`x`	A numeric vector (e.g. a time series), from which the ordinal pattern distribution is to be calculated
`ndemb`	Embedding dimension of the ordinal patterns (i.e. sliding window size) for which bitflips are to be calculated. Should be chosen such as length(x) » ndemb

Details

This function returns a histogram and time series of the number of bitflips occurring in the associated ordinal patterns. NA values are allowed, and any pattern that contains at least one NA value will be ignored. WARNING: Can be slow with very long time series (n > 10^7).

Value

A list with two entries is returned.

Author(s)

Sebastian Sippel

References

Sippel, S., 2014. Evaluating the carbon dynamics of biogeochemical models using statistical complexity measures. Master Thesis, University of Bayreuth.

Examples

```
x = arima.sim(model=list(ar = 0.3), n = 10^4)
nbitflips(x = x, ndemb = 6)
```

---

ordinal_pattern_distribution

A function to compute ordinal pattern statistics

Description

Computation of the ordinal patterns of a time series (see e.g. Bandt and Pompe 2002)

Usage

```
ordinal_pattern_distribution(x, ndemb)
```

Arguments

`x`	A numeric vector (e.g. a time series), from which the ordinal pattern distribution is to be calculated
`ndemb`	Embedding dimension of the ordinal patterns (i.e. sliding window size). Should be chosen such as length(x) » ndemb

Details

This function returns the distribution of ordinal patterns using the Keller coding scheme, detailed in Physica A 356 (2005) 114-120. NA values are allowed, and any pattern that contains at least one NA value will be ignored. (Fast) C routines are used for computing ordinal patterns.

Value

A character vector of length factorial(ndemb) is returned.

Author(s)

Sebastian Sippel

References

Bandt, C. and Pompe, B., 2002. Permutation entropy: a natural complexity measure for time series. Physical review letters, 88(17), p.174102.

Examples

```
x = arima.sim(model=list(ar = 0.3), n = 10^4)
ordinal_pattern_distribution(x = x, ndemb = 6)
```

---

ordinal_pattern_time_series

A function to compute time series of ordinal patterns

Description

Computation of the ordinal patterns of a time series (see e.g. Bandt and Pompe 2002)

Usage

```
ordinal_pattern_time_series(x, ndemb)
```

Arguments

x	A numeric vector (e.g. a time series), from which the ordinal pattern time series is to be calculated
ndemb	Embedding dimension of the ordinal patterns (i.e. sliding window size). Should be chosen such as length(x) » ndemb

Details

This function returns the distribution of ordinal patterns using the Keller coding scheme, detailed in Physica A 356 (2005) 114-120. NA values are allowed, and any pattern that contains at least one NA value will be ignored. (Fast) C routines are used for computing ordinal patterns.

Value

A character vector of length(x) is returned.

Author(s)

Sebastian Sippel

References

Bandt, C. and Pompe, B., 2002. Permutation entropy: a natural complexity measure for time series. Physical review letters, 88(17), p.174102.

Examples

```
x = arima.sim(model=list(ar = 0.3), n = 10^4)
ordinal_pattern_time_series(x = x, ndemb = 6)
```

---

permutation_entropy *A function to compute the permutation entropy*

Description

Computation of the permutation entropy of a time series based on its ordinal pattern distribution (see Bandt and Pompe 2002). Permutation entropy is a global information measure, hence insensitive to the permutation ordering scheme.

Usage

```
permutation_entropy(opd)
```

Arguments

opd	A numeric vector that details an ordinal pattern distribution.

Details

This function calculates the permutation entropy as described in Bandt and Pompe 2002.

Value

The normalized permutation entropy as a numeric value in the range [0,1].

Author(s)

Sebastian Sippel

References

Bandt, C. and Pompe, B., 2002. Permutation entropy: a natural complexity measure for time series. Physical review letters, 88(17), p.174102.

Examples

```
x = arima.sim(model=list(ar = 0.3), n = 10^4)
opd = ordinal_pattern_distribution(x = x, ndemb = 6)
permutation_entropy(opd)
```

`powernoise` *A function to generate k-noise*

Description

Generates samples of power law noise.

Usage

```
powernoise(k, N)
```

Arguments

k	Power law scaling exponent
N	number of samples to generate

Details

Generates samples of power law noise. The power spectrum of the signal scales as f^(-k). The R function uses fft(), similarly to the knoise_fft Matlab function.

Value

A named list with three entries is returned. x - N x 1 vector of power law samples

Author(s)

Sebastian Sippel and Holger Lange

Examples

```
powernoise_series = powernoise(k=2, N=10000)
```

`quadratic_map` *A function to generate a time series from the Quadratic map*

Description

Generates a time series from the Quadradtic map

Usage

```
quadratic_map(N, k, start="rand", disregard_N=0)
```

Arguments

N	length of the time series that is to be generated
k	Quadratic map parameter
start	start value. Default is to random.
disregard_N	Number of values at the beginning of the series to disregard

Value

A vector of length N

Author(s)

Sebastian Sippel

References

Grebogi, C., Ott, E. and Yorke, J.A., 1983. Crises, sudden changes in chaotic attractors, and transient chaos. Physica D: Nonlinear Phenomena, 7(1-3), pp.181-200.

Examples

```
quadratic_map(N = 10^4, k=1.4)
```

---

rank_to_permutation *A function to convert a "ranks-based" permutation notation to an "index-based" permutation scheme.*

Description

Converts permutations denoted by ranks to permutations denoted by indices and back.

Usage

```
rank_to_permutation(pattern, permutation.notation)
```

Arguments

pattern	A numeric vector that denotes a permutation pattern.
permutation.notation	The permutation notation that should be used. Could be "Olivares.2012" or "Keller.2005".

Details

This function converts ranks to indices and back.

Value

A numeric vector, which contains the transformed permutation.

Author(s)

Sebastian Sippel

References

Sebastian Sippel (2014). Master Thesis. University of Bayreuth.

`schuster_map` *A function to generate a time series from the Schuster Map*

Description

Generates a time series from the Schuster map

Usage

```
schuster_map(N, z, start="rand", disregard_N=0)
```

Arguments

`N`	length of the time series that is to be generated
`z`	Schuster map parameter
`start`	start value. Default is to random.
`disregard_N`	Number of values at the beginning of the series to disregard

Value

A vector of length N

Author(s)

Sebastian Sippel

References

Schuster, H.G., 1988. Deterministic chaos. An Introduction.

Examples

```
schuster_map(N = 10^4, z=2)
```

`skew_tent_map` *A function to generate a time series from the logistic map*

Description

Generates a time series from the Skew-Tent map

Usage

```
skew_tent_map(N, a, start="rand", disregard_N=0)
```

Arguments

`N`	length of the time series that is to be generated
`a`	Skew-Tent map parameter, must be in the range [0,1]
`start`	start value. Default is to random.
`disregard_N`	Number of values at the beginning of the series to disregard

Value

A vector of length N

Author(s)

Sebastian Sippel

References

Schuster, H.G., 1988. Deterministic chaos. An Introduction.

Examples

```
skew_tent_map(N = 10^4, a=0.1847)
```

`tent_map` *A function to generate a time series from the logistic map*

Description

Generates a time series from the logistic map

Usage

```
tent_map(N, mu, start="rand", disregard_N=0)
```

Arguments

`N`	length of the time series that is to be generated
`mu`	Tent map parameter, must be in the range [0,2]
`start`	start value. Default is to random.
`disregard_N`	Number of values at the beginning of the series to disregard

Value

A vector of length N

Author(s)

Sebastian Sippel

References

Feldman, D.P., McTague, C.S. and Crutchfield, J.P., 2008. The organization of intrinsic computation: Complexity-entropy diagrams and the diversity of natural information processing. Chaos: An Interdisciplinary Journal of Nonlinear Science, 18(4), p.043106.

Examples

```
tent_map(N = 10^4, mu=1.8)
```

`transformPermCoding` *A function to generate a vector from an index-transformation vector from a permutation coding scheme*

Description

Generates a position vector to change the ordinal pattern distribution in the default permutation coding scheme (i.e. generated by ordinal_pattern_distribution(x, ndemb)) into a user-specified coding scheme. This is a required input for the function changePermCodingOPD.

Usage

```
transformPermCoding(target_pattern, ndemb)
```

Arguments

`target_pattern`	A numeric matrix that specifies the pattern to be transformed into the position vector.
`ndemb`	Embedding dimension of the ordinal patterns (i.e. sliding window size). Should be chosen such as length(x) » ndemb

Details

This function returns a character vector to transform the output of ordinal_pattern_distribution (permutation coding as of Keller and Sinn, 2005) into a user-specified permutation coding scheme. For example, pattern #5 in "lehmerperm" (ndemb = 5) is given by the ranks c(0, 1, 4, 2, 3). This corresponds to pattern #41 in the (original) Keller coding scheme, as given by transformPermCoding(target_pattern = "lehmerperm", ndemb = 5)[5].

Value

A numeric vector of length factorial(ndemb), which contains the positions of the corresponding patterns in the Keller Coding scheme.

Author(s)

Sebastian Sippel

References

Olivares, F., Plastino, A. and Rosso, O.A., 2012. Ambiguities in Bandt-Pompe's methodology for local entropic quantifiers. Physica A: Statistical Mechanics and its Applications, 391(8), pp.2518-2526.

Examples

```
transformPermCoding(target_pattern = "lehmerperm", ndemb = 4)
```

---

weighted_ordinal_pattern_distribution

A function to compute weighted ordinal pattern statistics

Description

Computation of weighted ordinal patterns of a time series. Weights can be generated by a user-specified function (e.g. variance-weighted, see Fadlallah et al 2013).

Usage

```
weighted_ordinal_pattern_distribution(x, ndemb)
```

Arguments

x	A numeric vector (e.g. a time series), from which the weighted ordinal pattern distribution is to be calculated
ndemb	Embedding dimension of the ordinal patterns (i.e. sliding window size). Should be chosen such as length(x) » ndemb

Details

This function returns the distribution of weighted ordinal patterns using the Keller coding scheme, detailed in Physica A 356 (2005) 114-120. NA values are allowed. The function uses old and slow R routines and is only maintained for comparability. For faster routines, see weighted_ordinal_pattern_distribution.

Value

A character vector of length factorial(ndemb) is returned.

Author(s)

Sebastian Sippel

References

Fadlallah, B., Chen, B., Keil, A. and Principe, J., 2013. Weighted-permutation entropy: A complexity measure for time series incorporating amplitude information. Physical Review E, 87(2), p.022911.

See Also

weighted_ordinal_pattern_distribution

Examples

```
x = arima.sim(model=list(ar = 0.3), n = 10^4)
weighted_ordinal_pattern_distribution(x = x, ndemb = 6)
```

Index