Package 'stheoreme'

March 3, 2015

Type Package

Title Klimontovich's S-Theorem Algorithm Implementation and Data Preparation Tools

Version 1.2

Date 2015-02-28

Author Vitaly Efremov

Maintainer Vitaly Efremov <vitaly.efremov@dcu.ie>

Description Functions implementing the procedure of entropy comparison between two data samples after the renormalization of respective probability distributions with the algorithm designed by Klimontovich (Zeitschrift fur Physik B Condensed Matter. 1987, Volume 66, Issue 1, pp 125-127) and extended by Anishchenko (Proc. SPIE 2098, Computer Simulation in Nonlinear Optics. 1994, pp.130-136). The package also includes data preparation tools which can also be used separately for various applications.

License GPL-2

NeedsCompilation no

Repository CRAN

Date/Publication 2015-03-03 20:38:58

R topics documented:

`stheoreme-package` *Klimontovich's S-Theorem Algorithm Implementation and Data Preparation Tools*

Description

Functions in this package implement the procedure based on Klimontovich's S-theorem. This procedure compares relative degree of order between two data arrays. Klimontovich's S-theorem was originally proposed by Klimontovich (1987) as analytical modeling tool for open thermodynamic systems, and it sets the "renormalized entropy" as a criterion of relative degree of order for two probability density functions associated with two states of the thermodynamic system by equating the mean effective energies. Anishchenko (1994) extended the original concept and designed the algorithm for numerical comparison of two data samples. It has been used for many models such as heart rate variability time series by Wessel (1994) & Anishchenko (1995), variability of electroencephalograms of epilepsy patients by Kopizki (2002), relative degree of ordering in images by Herega (2010), self-organization of dissipative optical lattices by Bagci (2009) etc.

Details

Package:	stheoreme
Type:	Package
Version:	1.2
Date:	2015-02-25
License:	GPL-2

Two functions are directly related to Klimontovich's S-theorem algorithm. Function `crit.stheorem` checks the method applicability for certain pair of sample distributions and function `cxds.stheorem` estimates the renormalized entropy difference between two sample distributions after equating the mean energies. The package also contains data preparation tools (for cleaning, formatting, filtering etc.) which can also be used separately, e.g. for descriptive statistics and/or as helpful instruments for time-series analysis, signal and image processing.

Note

It has been created with support of the Science Foundation Ireland under Grant No. 13/TIDA/B2688

Author(s)

Vitaly Efremov <vitaly.efremov@dcu.ie>

References

Y.L.Klimontovich. S-theorem. Zeitschrift fur Physik B Condensed Matter. 1987, Volume 66, Issue 1, pp 125-127.

Yu.L.Klimontovich. Problems of open system statistical theory: criteria of relative state ordering at processes of self-organization. 1989. Usp. Fiz. Nauk. v.158(1) (in Russian)

V.S.Anishchenko, T.G.Anishchenko. On the criterion of the relative degree of order for self-oscillating regimes. Illustration of Klimontovich's S-theorem. Proc.SPIE, v.2098, pp.130-136, 1994.

T.G.Anishchenko, P.I.Saparin, N.B.Igosheva, V.S.Anishchenko. Sex differences in human cardiovascular responses to external excitation. Il Nuovo Cimento D. Luglio-Agosto 1995, Volume17, Issue7-8, pp.699-707.

T.Anishchenko, N.Igosheva, T.Yakusheva, O.Glushkovskaya-Semyachkina, O.Khokhlova. Normalized entropy applied to the analysis of interindividual and gender-related differences in the cardiovascular effects of stress. Eur J Appl Physiol. 2001 Aug; 85(3-4):287-98.

A.N.Herega. On One Criterion of the Relative Degree of Ordering in Images. Technical Physics, 2010, Vol.55, No.5, pp.741-742.

N.Wessel, A.Voss, J.Kurths, P.Saparin, A.Witt, H.J.Kleiner, R.Dietz. Renormalised entropy: a new method of non-linear dynamics for the analysis of heart rate variability. Computers in Cardiology Conf. Proc., 1994, pp.137-140.

G.B.Bagci, U.Tirnakli. Self-organization in dissipative optical lattices. CHAOS. 19, 033113. 2009.

K.Kopizki, P.C.Warnke, P.Saparin, j.Kurths, J.Timmer. Comment on 'Kullback-Leiber and renormalized entropies: Application to electroencephalograms of epilepsy patients'. PHYSICAL REVIEW E 66, 043902 (2002).

`crit.stheorem` *Klimontovich's S-Theorem Convergence Criterion*

Description

Function `crit.stheorem` is applied to a pair of probability vectors or to a pair of vector of counts associated with two states of the same open thermodynamic system in order to check the Klimontovich's S-theorem applicability.

Usage

```
crit.stheorem(distribution0, distribution1)
```

Arguments

`distribution0`	vector of counts representing distribution function of state0 of the (open thermodynamic) system
`distribution1`	vector of counts representing distribution function of state1 of the (open thermodynamic) system

Details

The full text of the theorem and associated algorithm was presented by Klimontovich (1989) and Anischenko (1994).

Let the open thermodynamic system to be characterized by probability density function $f(x)$ and evolving from state0 to state1. Let the state0 be chosen as "chaos". Then the effective Hamiltonian of the system can be written as $H_0(x) = -lnf_0(x)$ and it must not change at evolution to the state1, namely:

$$\int f_0(x)H_0(x)dx = \int f_1(x)H_0(x)dx$$

The latter is "constant Hamiltonian equation" which is not truth for the most of the cases (except the case of identical probability distributions). To make it truth the function $f_0(x)$ has to be represented as a subcase of Gibbs distribution for a state of physical chaos, namely:

$$f^*(x) = exp(\frac{F_0 - H_0(x)}{D}), \int f^*(x)dx = 1$$

where parameter F_0 is to be defined by procedure of normalization and D is an effective temperature. $f_0(x)$ is now to be substituted by the new $f^*(x)$ and temperature is to be varied until "constant Hamiltonian equation" becomes truth. The corresponding D value at which system is thermodynamically balanced is now the indicator of the direction of evolution: if $D > 1$ state0 is actually more chaotic (need to be "heated" for energy balance). Otherwise, if $D < 1$, state1 is more chaotic and a case of $D = 1$ takes place where states are identical.

The next step is to check if the system evolution can be followed in opposite direction: from state1 back to state0. State1 is to be chosen as "chaos" and all the steps to be repeated from very beginning. Another effective temperature D_{bck} is to be calculated representing energy balance of the system at its way back, from state1 to state0 . There are 2 scenarios possible:

A.$D > 1$ and $D_{bck} < 1$ or $D < 1$ and $D_{bck} > 1$

B.$D < 1$ and $D_{bck} < 1$ or $D > 1$ and $D_{bck} > 1$

Scenario A (where "way-forward" and "way-back" temperatures have opposite signs) allows for the system evolution to be consistently explained and more chaotic state to be chosen, whereas scenario B makes a direct evolution between states not possible to be followed.

The actual type of the scenario gives an absolute Klimontovich's S-theorem convergence criterion. Let now a null hypothesis be "two sample distributions correspond to the same open thermodynamic system". Null hypothesis is not rejected at scenario A and the general conclusion is "data samples can be considered as outcomes of the same open system and observed differences can be explained by thermodynamic noise". When scenario B is observed, null hypothesis is to be rejected at some level of significance. In order to determine a criterion of rejection the following steps are to be completed:

1. It is assumed the system has probably passed through the medium state2 on its evolution way from state0 to state1. The corresponding probability density function $f_2(x)$ can be chosen as

$$f_2(x) = \alpha f_0(x) + (1 - \alpha)f_1(x), 0 <= \alpha <= 1, \int f_2(x)dx = 1$$

by varying parameter α until forward-way and back-way effective temperatures have got the opposite signs for all the steps (from state0 to state2 and from state2 to state1 respectively).

2. When α is detected coefficient of consistency is calculated as $r^2 = (2\alpha - 1)^2$. It indicates how far state2 is located from the original states of the system.

3.If $r^2 = 1$ state2 coincides with one of the original states, i.e. scenario A has actually taken place. If $r^2 = 0$ state2 is too far from the original states to follow the system evolution and null hypothesis is surely rejected. Finally, if $0 < r^2 < 1$, the conclusion will depend on significance level chosen.

Value

`r2_val` coefficient of consistency, (`0<=r2_val<=1`)

Author(s)

Vitaly Efremov <vitaly.efremov@dcu.ie>

References

Y.L.Klimontovich. S-theorem. Zeitschrift fur Physik B Condensed Matter. 1987, Volume 66, Issue 1, pp 125-127.

Yu.L.Klimontovich. Problems of open system statistical theory: criteria of relative state ordering at processes of self-organization. 1989. Usp. Fiz. Nauk. v.158(1) (in Russian)

V.S.Anishchenko, T.G.Anishchenko. On the criterion of the relative degree of order for self-oscillating regimes. Illustration of Klimontovich's S-theorem. Proc.SPIE, v.2098, pp.130-136, 1994.

See Also

`cxds.stheorem`, `d1nat`, `pvalign`

Examples

```
#completely different bin counts by their (thermodynamic) nature
h0 <- c(0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1)
h1 <- c(1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0)
crit.stheorem(distribution0=h0, distribution1=h1)

#quazi-gaussian probability vectors with equal means & different variances
f0 <- c(0.0,0.1,0.4,0.4,0.1,0.0)
f1 <- c(0.1,0.15,0.25,0.25,0.15,0.1)
crit.stheorem(f0, f1)

#multimodal bin counts
h0 <- c(1,6,1,6,5,1,2)
h1 <- c(1,6,1,1,9,1,2)
crit.stheorem(h0, h1)

#quazi-gaussian bin counts with a shift between means
h0 <- c(2,2,17,6,1,1,1,0)
h1 <- c(2,3,5,7,7,4,1,0)
crit.stheorem(h0, h1)
```

```
#example of 2-step analysis with Klimontovich's S-theorem for 2
# arrays of outcomes {s0,s1}:
s0<-rep(c(1:11,2),256)
s1<-rep(c(2,3,3,4,5,5,5),55)
# step a. Create probability vectors
b<-d1nat(s0,s1,brks=12); b
# step b. Compare samples with Klimontovich's S-theorem
crit.stheorem(b$f0,b$f1)
```

`cxds.stheorem` *Renormalized Entropy Shift Estimation*

Description

Function `cxds.stheorem` is applied to a pair of probability vectors or to a pair of vector of counts associated with two states of the same open thermodynamic system in order to estimate renormalized entropy shift as the system evolves.

Usage

```
cxds.stheorem(distribution0, distribution1)
```

Arguments

`distribution0`	vector of counts representing distribution function of state0 of the (open thermodynamic) system
`distribution1`	vector of counts representing distribution function of state1 of the (open thermodynamic) system

Details

The function implements the second part of the algorithm based on Klimontovich's S-theorem (the first part is implemented by `crit.stheorem`). More detailes can be found in Klimontovich (1989) and Anischenko (1994).

Let the open thermodynamic system be characterized by probability density function $f(x)$ and evolving from state0 to state1. Entropies corresponding to state0 and state1 are respectively:

$$H_0 = -\int f_0(x) ln f_0(x) dx, H_1 = -\int f_1(x) ln f_1(x) dx$$

This is known Shannon entropy formulation which is most comprehensive indicator of degree of order of the system at the certain state. Shannon entropy shift depends on both external (energy/entropy in-/out-flows) and internal (noise and self-organization processes) changes happened to the system. However, it doesn't make it possible to differentiate between two and estimate,

for instance, the role of self-organization processes in the evolution. The quantifier proposed by Klimontovich (1987) is called renormalized entropy

$$dS_{0\to 1} = -\int f_1(x) ln f_1(x) dx + \int f^*(x) ln f^*(x) dx$$

where $f^*(x)$ is a new probability density function being the result of Gibbs distribution function tranformation of $f_0(x)$ for energy balancing (see the 'details' section for `crit.stheorem`). The formula allows for the final entropy shift to be written in a form

$$dH_{0\to 1} = H_1 - H_0 = dS_{0\to 1} + dI_{0\to 1}$$

where dI is the entropy change caused by external reasons. If direct evolution from the state0 to state1 is thermodynamically forbidden ($r^2 < 1$) the medium state2 of the system evolution is to be found and the final formula for entropy shift will contain additional equation:

$$dS_{0\to 1} = dS_{0\to 2} + dS_{2\to 1}$$

representing the pass of the system through an indirect medium state.

It is recommended to apply the function alongside with Klimontovich's S-theorem convergence criterion function `crit.stheorem`

Value

`dH_val`	Shannon entropy shift
`dS_val`	renormalized entropy shift
`dH_ext`	entropy inflow to the system from outside
`dS_02`	renormalized entropy shift from state0 to medium state2
`dS_21`	renormalized entropy shift from medium state2 to state1

Author(s)

Vitaly Efremov <vitaly.efremov@dcu.ie>

References

Y.L.Klimontovich. S-theorem. Zeitschrift fur Physik B Condensed Matter. 1987, Volume 66, Issue 1, pp 125-127.

Yu.L.Klimontovich. Problems of open system statistical theory: criteria of relative state ordering at processes of self-organization. 1989. Usp. Fiz. Nauk. v.158(1) (in Russian)

V.S.Anishchenko, T.G.Anishchenko. On the criterion of the relative degree of order for self-oscillating regimes. Illustration of Klimontovich's S-theorem. Proc.SPIE, v.2098, pp.130-136, 1994.

K.Kopizki, P.C.Warnke, P.Saparin, j.Kurths, J.Timmer. Comment on 'Kullback-Leiber and renormalized entropies: Application to electroencephalograms of epilepsy patients'. PHYSICAL REVIEW E 66, 043902 (2002).

See Also

crit.stheorem, d1nat, pvalign

Examples

```
#quazi-gaussian probability vectors with equal means & different variances
f0 <- c(0.0,0.1,0.4,0.4,0.1,0.0)
f1 <- c(0.1,0.15,0.25,0.25,0.15,0.1)
cxds.stheorem(distribution0=f0, distribution1=f1)

#quazi-gaussian bin counts with shift between means
h0 <- c(2,2,17,6,1,1,1,0)
h1 <- c(2,3,5,7,7,4,1,0)
crit.stheorem(h0, h1)
cxds.stheorem(h0, h1)

#example of 2-step analysis with Klimontovich's S-theorem for 2
# arrays of outcomes {s0,s1}:
s0<-rep(c(1:11,2),256)
s1<-rep(c(2,3,3,4,5,5,5),55)
# step a. Create probability vectors
b<-d1nat(s0,s1,brks=12); b
# step b. Compare samples with Klimontovich's S-theorem
crit.stheorem(b$f0,b$f1)
cxds.stheorem(b$f0,b$f1)

#example of 3-step analysis with Klimontovich's S-theorem to study two gratings
# random vs regular
s0<-array(c(rep(0,640),rep(1,640)), c(320,320))
s1<-array(runif(5120,0,1), c(64,80))
# step a. Binarize (to make s1 comparable with s0 by its nature as a grating)
a<-utild2bin(s0, s1, method='med')
# step b. Create probability vectors as for angular space (anisotropy study)
# There is no doubt s0 is more regular
b<-d2spec(s0, s1, brks=36, method='ang90'); b
# step c. Compare gratings with Klimontovich's S-theorem. Renormalized entropy shift
# is negligible compared to Shannon's. Evolution from state0 to state1 is possible
# but clearly with external entropy (or energy) inflow
crit.stheorem(b$f0,b$f1)
cxds.stheorem(b$f0,b$f1)

#example of 2-step analysis with Klimontovich's S-theorem for
#two following char arrays:
s0<-c("a","b",rep("c",9),rep("d",2),"e","f","g",rep("h",2),"i","j"); s0
s1<-c(rep("a",16), rep("c",35), rep("i",13)); s1
# step a. Create probability vectors. It seems that s0 has lower level
# of orderliness (Shannon entropy is higher)
b<-d1char.d1nat(s0,s1); b
# step b. Compare samples with Klimontovich's S-theorem. Renormalized entropy indicates
# the opposite: s0 is more ordered and difference in Shannon entropy values was
# due to just "thermodynamic noise"
crit.stheorem(b$f0,b$f1)
```

```
cxds.stheorem(b$f0,b$f1)

#example of 3-step analysis with Klimontovich's S-theorem for 2 random
# arrays of outcomes  {s0,s1}:
s0<-runif(128,0,1)^2
s1<-runif(64,0,1)^2.3
# step a. Convert samples to arrays of sequential 17-point means
a<-utild1group(s0, s1, radius=8, method='splitN')
# step b. Create probability vectors
b<-d1nat(a$group0,a$group1,brks=12,band=c(0,0.8)); b
# step c. Compare samples with Klimontovich's S-theorem
crit.stheorem(b$f0,b$f1)
cxds.stheorem(b$f0,b$f1)
```

`d1char.d1nat` *Probability Mass Function Calculator for Array of Characters*

Description

Function `d1char.d1nat` is applied to a pair of vectors of characters or to a pair of 1d arrays of characters and generates then the pair of corresponding probability vectors by calling `d1nat` function

Usage

```
d1char.d1nat(farr0, farr1, reject = c(""))
```

Arguments

`farr0`	vector of characters
`farr1`	vector of characters
`reject`	vector of rejected characters (the characters excluded from original vectors for analysis)

Details

First, it assigns the identification number for each individual character met in original vectors. Then it works similarly to `hist` function but for the pair of samples with identification numbers as outcomes. It prints the decoding array (listing the identification numbers corresponded for each character) and, as a bonus, it prints basic statistics summary for distributions alongside with technical plot. It is recommended for use as a data preparation step before the following Klimontovich's S-theorem based analysis.

Value

`f0`	probability vector representing state0 of a system
`f1`	probability vector representing state1 of a system

Author(s)

Vitaly Efremov <vitaly.efremov@dcu.ie>

See Also

crit.stheorem, cxds.stheorem, d1nat

Examples

```
s0<-c("?", "!", "1", "a", "b", "b", "s", "x", "y", "z", "z", "z")
s1<-c("1", "1", "2", "b", "b", "s", "s", "x", "y", "z", "z", "z", "z")
b<-d1char.d1nat(farr0=s0,farr1=s1); b

s0<-"three witches watch three swatch watches. which witch watch which swatch watch?"
s1<-"who discovered america five hundred years ago? a brave man! indeed he was! discovered!"
b<-d1char.d1nat(unlist(strsplit(s0,"")),unlist(strsplit(s1,"")),
                reject=c("."," ","!","d","e")); b

#example of 2-step data analysis with Klimontovich's S-theorem
s0<-c("a","b",rep("c",9),rep("d",2),"e","f","g",rep("h",2),"i","j"); s0
s1<-c(rep("a",16), rep("c",35), rep("i",13)); s1

# step a. Create probability vectors. It seems that s0 has lower level
# of orderliness (Shannon entropy is higher)
b<-d1char.d1nat(s0,s1); b
# step b. Compare samples with Klimontovich's S-theorem. Renormalized entropy indicates
# the opposite: s0 is more ordered and difference in Shannon
# entropy values was due to just "thermodynamic noise"
crit.stheorem(b$f0,b$f1)
cxds.stheorem(b$f0,b$f1)
```

---

d1nat *Probability Mass Function Calculator*

Description

Function d1nat is applied to a pair of vectors or 1d arrays (sample outcomes, observation data values, time series data, 1d signal values, etc.) and generates the pair of corresponding probability mass functions (normalized vectors of counts).

Usage

```
d1nat(sample0, sample1, band=c(0,0), brks=0)
```

Arguments

sample0	vector of values (sample outcomes)
sample1	vector of values (sample outcomes)
band	two border values to set a range of considered sample values. The default c(0,0) sets full entire range i.e. range(sample0, sample1)
brks	value giving a number of bins (in a same manner as the number of cells for the histogram). The default brks=0 sets the number of bins chosen automatically as a square root of sample0 size.

Details

It works similarly to hist function but for pair of vectors with sample outcomes. As a bonus it prints basic statistics summary for distributions alongside with technical plot. It is recommended for use as a data preparation step before following Klimontovich's S-theorem based analysis.

Value

f0	probability vector representing state0 of a system
f1	probability vector representing state1 of a system
midpoints	vector of the centres of bins where probability values are calculated

Author(s)

Vitaly Efremov <vitaly.efremov@dcu.ie>

See Also

crit.stheorem, cxds.stheorem, d2nat.d1nat, d1char.d1nat

Examples

```
#two modelling arrays: random with randomness distorted by power
s0<-runif(128,0,1)^2
s1<-runif(64,0,1)^2.3

b<-d1nat(sample0=s0,sample1=s1); b
b<-d1nat(s0,s1,brks=12,band=c(0.2,1)); b

#example of 3-step data analysis with Klimontovich's S-theorem
# step a. Convert samples to arrays of sequential 17-point means
a<-utild1group(s0, s1, radius=8, method='splitN')
# step b. Create probability vectors
b<-d1nat(a$group0,a$group1,brks=12,band=c(0,0.8)); b
# step c. Compare samples with Klimontovich's S-theorem
crit.stheorem(b$f0,b$f1)
cxds.stheorem(b$f0,b$f1)
```

d1spec *Power Spectrum Probability Vector Calculator*

Description

Function d1spec is applied to a pair of vectors (sample outcomes, observation data values, time series data, 1d signal values, etc.) and generates the pair of corresponding spectral power density functions

Usage

```
d1spec(sample0, sample1, band=c(0,0), brks=0, meansub=TRUE)
```

Arguments

sample0	vector of values (sample outcomes)
sample1	vector of values (sample outcomes)
band	two border values to set a range of considered frequencies. The default c(0,0) sets full entire range i.e. from 0 to 0.5 Hz, where 1 Hz = [1/sampling_interval]
brks	value used in a same manner as the number of cells for the histogram. The default brks=0 sets the number of cells equal to `min(length(sample0), length(sample1))/2`
meansub	logical item defining if individual baseline (defined as individual mean value) is subtracted from original vectors before application of Fourier transform

Details

Spectral power density functions are often being used as probability vectors characterizing thermodynamic states of a system. Here fast Fourier transform algorithm is utilized and the frequency values are used as 1 Hz = [1/sampling_interval], i.e. in a range from 0 to 0.5 Hz. In general the function works similarly to d1nat. As a bonus it prints basic statistics summary for power density functions alongside with technical plot. It is recommended for use as a data preparation step before following Klimontovich's S-theorem based analysis.

Value

f0	spectral power density function as a probability vector representing state0 of a system
f1	spectral power density function as a probability vector representing state1 of a system
freqs	vector of coresponding frequency values

Author(s)

Vitaly Efremov <vitaly.efremov@dcu.ie>

References

T.G.Anishchenko, P.I.Saparin, N.B.Igosheva, V.S.Anishchenko. Sex differences in human cardiovascular responses to external excitation. Il Nuovo Cimento D. Luglio-Agosto 1995, Volume17, Issue7-8, pp.699-707.

E.J.Groth. Probability distributions related to power spectra. Astrophysical Journal, 1975. Suppl. Ser., Vol.29, No.286, p. 285-302.

T.Anishchenko, N.Igosheva, T.Yakusheva, O.Glushkovskaya-Semyachkina, O.Khokhlova. Normalized entropy applied to the analysis of interindividual and gender-related differences in the cardiovascular effects of stress. Eur J Appl Physiol. 2001 Aug; 85(3-4):287-98.

See Also

`crit.stheorem`, `cxds.stheorem`, `d2spec`

Examples

```
s0 <- 2+sin(c(1:128))
s1<- array(c(rep(0,8),rep(1,8)), c(256))

b<-d1spec(sample0=s0,sample1=s1); b
b<-d1spec(s0, s1, band=c(0,0.25), brks=16, meansub=FALSE); b

#example of 3-step data analysis with Klimontovich's S-theorem
# step a. Clean samples from outliers (points out of 1.4 sigmas)
a<-utild1clean(s0, s1, method='both', nsigma=1.4)
# step b. Create probability vectors. It seems that s0 has lower level
# of orderliness (Shannon entropy is higher).
b<-d1spec(a$clean0,a$clean1); b
# step c. Compare samples with Klimontovich's S-theorem. Renormalized entropy indicates
# the opposite: s0 is more ordered and difference in Shannon entropy values was
# due to just "thermodynamic noise" (discretization noise in this case)
crit.stheorem(b$f0,b$f1)
cxds.stheorem(b$f0,b$f1)
```

---

d2nat.d1nat *Probability Mass Function Calculator for Matrices*

Description

Function `d2nat.d1nat` is applied to a pair of matrices and generates then the pair of corresponding probability mass functions by calling `d1nat`

Usage

```
d2nat.d1nat(d2arr0, d2arr1, band = c(0, 0), brks = 64, method = "default")
```

Arguments

d2arr0	sample matrix
d2arr1	sample matrix
band	two border values to set a range of considered values in matrices. The default c(0,0) sets full entire range i.e. range(d2arr0, d2arr1)
brks	value giving a number of bins (in a same manner as the number of cells for the histogram). The default value sets the number of bins automatically equal to 64.
method	specifies selection of matrix elements method='default' simply to call d1nat and apply it to ensemble of all matrix elements as to 1d vector of outcomes method='cols' to create 1d array with elements being the mean values of original matrix columns and then simply to call d1nat function method='rows' to create 1d array with elements being the mean values of original matrix rows and then simply to call d1nat function

Details

It works similarly to d1nat function but for pair of matrices. It is recommended for use as a data preparation step before following Klimontovich's S-theorem based analysis. For instance, it can be used for image analysis.

Value

f0	probability vector representing state0 of a system
f1	probability vector representing state1 of a system
midpoints	vector of the centres of bins where probability values are calculated

Author(s)

Vitaly Efremov <vitaly.efremov@dcu.ie>

References

A.N.Herega. On One Criterion of the Relative Degree of Ordering in Images. Technical Physics, 2010, Vol.55, No.5, pp.741-742.

G.B.Bagci, U.Tirnakli. Self-organization in dissipative optical lattices. CHAOS. 19, 033113. 2009.

See Also

crit.stheorem, cxds.stheorem, d1nat, utild2group

Examples

```
#two modelling arrays: random with randomness distorted by power
s0<-array(runif(256,0,1)^2, c(16,16))
s1<-array(runif(512,0,1)^3, c(16,8))

b<-d2nat.d1nat(d2arr0=s0,d2arr1=s1); b
b<-d2nat.d1nat(s0,s1,brks=256); b
b<-d2nat.d1nat(s0,s1,brks=18,band=c(0.1,0.5),method='rows'); b

#example of 3-step data analysis with Klimontovich's S-theorem
# step a. Split matrices to regions with radius 1, create new matrices
# of region means
a<-utild2group(s0, s1, radius=1)
# step b. Create probability vectors
b<-d1nat(a$group0,a$group1,brks=8,band=c(0.1,0.8))
# step c. Compare samples with Klimontovich's S-theorem
crit.stheorem(b$f0,b$f1)
cxds.stheorem(b$f0,b$f1)
```

---

d2spec *Power Spectrum Probability Vector Calculator for Matrices*

Description

Function d2spec is applied to a pair of matrices (2d arrays) and generates the pair of corresponding 1d spectral power density functions (to be used then as probability vectors) specified by the method argument

Usage

```
d2spec(d2arr0, d2arr1, band=c(0,0), brks=0, method='rad', meansub=TRUE)
```

Arguments

d2arr0	sample matrix
d2arr1	sample matrix
band	two border values to set a range of considered frequencies. The default c(0,0) sets full entire range from 0 to 0.5 Hz (where 1 Hz = [1/sampling_interval]) for radial method and from 0 to 180 degrees for both angular methods
brks	value used in a same manner as the number of cells for the histogram. The default brks=0 sets the number of cells equal to `min(length(sample0), length(sample1))/2` for radial method and equal to 180 for both angular methods
method	specifies the method of using the power spectrum as a probability vector. The default value 'rad' defines the calculation of spectral power density for spatial frequency space {u,v} as a function of spectral radius `sqrt(u^2+v^2)`. `method='ang'` to calculate spectral power density as a function of spectral angle `atan(u/v)`

	`method='ang90'` to calculate spectral power density as a function of shifted spectral angle `atan(v/u)`
`meansub`	logical item defining if individual baselines (as mean value) are subtracted from original matrices before application of 2d Fourier transform

Details

Spectral power density is often being used as probability vector/matrix characterizing a state of a system, e.g. in image analysis. Here 2d fast Fourier transform algorithm is utilized to calculate power spectrum matrix $P(u, v)$ and then the 1d spectral power density:

$$f(k_i) = \sum_{k_{min}}^{k_{max}} P(k : k_i < k < k_{i+1})$$

where k is a metrics of frequency space defined with one of the following options: $k = \sqrt{u^2 + v^2}$ (as radius where `method='rad'`); $k = atan(u/v)$ (as angle where `method='ang'`) or $k = atan(v/u)$ (as angle where `method='ang90'`)

and where k_{min}, k_{max} are defined by `band` argument

In general the function works similarly to `d1nat`. As a bonus it prints basic statistics summary for spectral power density functions alongside with technical plot. It is recommended for use as a data preparation step before following Klimontovich's S-theorem based analysis.

Value

`f0`	spectral power density function as a probability vector representing state0 of a system
`f1`	spectral power density function as a probability vector representing state1 of a system
`efs`	vector of corresponding metrics/bin values (radial frequency or spectral angle)

Author(s)

Vitaly Efremov <vitaly.efremov@dcu.ie>

References

N.F.Zhang, A.E.Vladar, M.T.Postek, R.D.Larrabee.Spectral density-based statistical measures for image sharpness. Metrologia, 42(2005), 351-359

E.J.Groth. Probability distributions related to power spectra. Astrophysical Journal, 1975. Suppl. Ser., Vol.29, No.286, p. 285-302.

See Also

`crit.stheorem`, `cxds.stheorem`, `d1spec`, `d1nat`

Examples

```
# compare (harmonic+background) with (harmonic noisy period) matrices
s0<-array(2+sin(c(1:256)/3), c(16,16))
s1<-array(sin(c(1:512)+runif(512,0,2)), c(16,32))

# as radial:
b<-d2spec(d2arr0=s0, d2arr1=s1); b
b<-d2spec(s0, s1, brks=29, band=c(0.15,0.5)); b
# as angular:
b<-d2spec(s0, s1, method='ang', meansub=FALSE); b

#example of 3-step data analysis with Klimontovich's S-theorem
# Study two gratings: random vs regular
s0<-array(c(rep(0,640),rep(1,640)), c(320,320))
s1<-array(runif(5120,0,1), c(64,80))
# step a. Binarize (to make s1 comparable with s0 by its nature as a grating)
a<-utild2bin(s0, s1, method='med')
# step b. Create probability vectors as for angular space (anisotropy study)
# There is no doubt s0 is more regular
b<-d2spec(a$bin0, a$bin1, brks=36, method='ang90'); b
# step c. Compare gratings with Klimontovich's S-theorem. Renormalized entropy shift
# is negligible compared to Shannon's. Evolution from state0 to state1 is possible
# but clearly with external entropy (or energy) inflow
crit.stheorem(b$f0,b$f1)
cxds.stheorem(b$f0,b$f1)
```

`pvalign` *Probability Vector Formatting Tool*

Description

The fuction is applied to a pair of probability vectors or to a pair of vector with counts in order to format them for adequate use at Klimontovich's S-theorem based analysis

Usage

```
pvalign(distribution0, distribution1,
outcomes0=c(0), outcomes1=c(0), tolerance=0)
```

Arguments

`distribution0`	vector of counts / probability vector
`distribution1`	vector of counts / probability vector
`outcomes0`	optionally used vector of bins/cell numerical identifiers corresponding to counts contained in distribution0 vector
`outcomes1`	optionally used vector of bins/cell numerical identifiers corresponding to counts contained in distribution1 vector
`tolerance`	a number to set a level of accuracy at which distributions will be aligned with the default value `tolerance=0.001`

Details

At some applications a pair of probability vectors or vector of counts are not equal in size. At some applications a pair of probability vectors or vector of counts are not corresponded to the same intervals of outcomes or measured at not equidistant intervals. This makes Klimontovich's S-theorem based analysis not adequate or even not possible at all. The function tries to align vectors and convert them to equidistant ones with the level of accuracy specified by parameter of `tolerance`. As a bonus it prints basic statistics summary for distributions alongside with technical plot

Value

`new_scale`	new generated vector of bins/cell numerical identifiers corresponding to counts contained in vectors with counts, common for both distributions
`f0`	aligned and improved probability vector representing state0 of a system
`f1`	aligned and improved probability vector representing state1 of a system

Author(s)

Vitaly Efremov <vitaly.efremov@dcu.ie>

See Also

`crit.stheorem`, `cxds.stheorem`

Examples

```
#consider 2 arrays {h0, h1} representing counts for
#bins where bins are enumerated by corresponding x0 and x1 arrays:
x0 <- -c(10,9,8,7,6,5,4.01,3)
x1 <- c(2,3,4,5,6,7,8)*(1.01)
h0 <- c(0,0,3,4,0,0,0.2,1)
h1 <- c(2,2,2,6,6,0,1)

#align regardless bin identifiers:
b<-pvalign(distribution0=h0, distribution1=h1); b
#align using bin identifiers:
b<-pvalign(h0, h1, outcomes0=x0, outcomes1=x1, tolerance=0.02); b
b$f0; b$f1; b$new_scale
#set small tolerance and observe an error generated:
 # b<-pvalign(h0, h1, x0, x1, tolerance=0.0001)

#example of 2-step data analysis with Klimontovich's S-theorem
# compare two bin counts:
h0<-c(1,1,1,1,1,1,1,1,1,1,1,1)
h1<-c(1,2,2,3,1,0,8,6)
# step a. Create probability vectors. It seems that s0 has lower level
# of orderliness (Shannon entropy is higher)
b<-pvalign(h0, h1); b
# step b. Compare with Klimontovich's S-theorem. Renormalized entropy shift
# is negligible compared to Shannon's. Evolution from state0 to state1 is possible
# but clearly with external entropy (or energy) outflow
```

```
crit.stheorem(b$f0,b$f1)
cxds.stheorem(b$f0,b$f1)
```

---

utild1bin *Vector Binarizer*

Description

Function utild1bin is applied to a pair of vectors and convert them to the pair of binary arrays with a rule specified by user

Usage

```
utild1bin(arr0, arr1, method = "mean", trsh = 0, inverted = FALSE, d2=FALSE)
```

Arguments

arr0	vector of values/outcomes to be binarized
arr1	vector of values/outcomes to be binarized
method	method of setting a threshold: method='mean' threshold is mean(c(arr0, arr1)) for both arrays (used as default) method='imean' setting individual thresholds as mean values method='imed' setting individual thresholds as median values method='i1stQ' setting individual thresholds as 1st quartile values method='i3rdQ' setting individual thresholds as 3rd quartile values method='isubsd' setting individual thresholds as (mean(arrN)-sd(arrN)) method='iaddsd' setting individual thresholds as (mean(arrN)+sd(arrN)) method='med' threshold is median(c(arr0, arr1)) for both arrays method='med0' threshold is median(c(arr0)) for both arrays method='med1' threshold is median(c(arr1)) for both arrays method='mean0' threshold is mean(c(arr0)) for both arrays method='mean1' threshold is mean(c(arr1)) for both arrays method='spec' threshold is specified by the 2 numbers of argument trsh individually for arrays
trsh	specifies the individual threshold values for array binarizations when method='spec' is chosen. If it contains one number it will be interpreted as one threshold for both the arrays
inverted	logical item specifying if the result will be inverted (zeros replaced by ones and ones replaced by zeros) after bianrization
d2	works exactly as utild2bin when TRUE

Value

bin0	result of arr0 binarization
bin1	result of arr1 binarization
counts	summary/report of how the arrays were binarized

Author(s)

Vitaly Efremov <vitaly.efremov@dcu.ie>

See Also

utild2bin

Examples

```
s0<-rep(c(-1,1,3,7,10),2)
s1<-c(1:7)

s0; s1
a<-utild1bin(arr0=s0, arr1=s1); a
a<-utild1bin(s0, s1, method='spec', trsh=c(2,5), inverted=TRUE); a
```

---

utild1clean *Vector Cleaner*

Description

Function utild1clean is applied to a pair of vectors and clean them from outliers with a rule specified by user (outliers will be replaced by the threshold values)

Usage

```
utild1clean(arr0, arr1, method = "bothends", nsigma = 3, d2=FALSE)
```

Arguments

arr0	vector of values/outcomes to be cleaned
arr1	vector of values/outcomes to be cleaned
method	specifies if outliers are to be removed from one or both side of array value distribution method='bothends' remove otliers from both side of a distributon (default) method='left' remove otliers from left side method='right' remove otliers from right side
nsigma	number of sigmas to be considered as threshold for outlier removal
d2	works exactly as utild2clean when TRUE

Value

clean0	result of arr0 cleaning
clean1	result of arr1 cleaning

Author(s)

Vitaly Efremov <vitaly.efremov@dcu.ie>

See Also

utild2clean

Examples

```
s0<-c(rep(c(1,3,7,10),3), -23)
s1<-c(1:7)

s0; s1
a<-utild1clean(arr0=s0, arr1=s1); a
a<-utild1clean(s0, s1, method='right', nsigma=0.8); a
```

---

utild1filt *Gaussian Filter for Vectors*

Description

The function is applied to a pair of vectors as a lowpass gaussian filter to clean them from high frequency components

Usage

```
utild1filt(arr0, arr1, outsize = 2, strong = 1)
```

Arguments

arr0	vector to be filtered
arr1	vector to be filtered
outsize	radius of gaussian filter kernel
strong	multiplication factor defining the sigma of gaussian filter kernel as sigma=outsize*strong

Value

filt0	result of arr0 filtration
filt1	result of arr1 filtration

Author(s)

Vitaly Efremov <vitaly.efremov@dcu.ie>

See Also

utild2filt

Examples

```
s1<-sin(c(1:128)*2)+2-c(1:128)*4/128

a<-utild1filt(arr0=s1, arr1=s1)
plot(s1, type='l')
lines(a$filt1, col='red')

s0<-c(rep(0,15), rep(2,12), rep(-2,12), rep(0,25))
s1<-c(rep(0,45), rep(2,4), rep(-2,4), rep(0,25))

a<-utild1filt(s0, s1, outsize=7, strong=.5)
plot(s0, type='l', ylab='s0, s1, a$filt0, a$filt1')
lines(s1, col='gray')
lines(a$filt0, col='red')
lines(a$filt1, col='purple')
```

---

utild1group *Vector Framer*

Description

The function is applied to a pair of vectors in order to split them into N-point frames and calculate respective arrays of N-point frame means, there N is specified by user

Usage

```
utild1group(arr0, arr1, radius = 1, method = "split1")
```

Arguments

arr0	original vector
arr1	original vector
radius	sets the size of a single frame as N=(2*radius+1) points
method	sets the division of original array by a sequence of frames in different manner method='split1' (the default) sets a sequence of frames as: frame1(1...N), frame2(N+1,..2*N), etc method='splitN' sets a sequence of frames as: frame1(1...N), frame2(2,..N+1), frame3(3,..N+2), etc

Value

group0	array containing mean values of individual frames the original arr0 was divided by
group1	array containing mean values of individual frames the original arr1 was divided by

Author(s)

Vitaly Efremov <vitaly.efremov@dcu.ie>

See Also

utild2group

Examples

```
s0<-rep(c(-1,1,3,7,10),3)
s1<-c(1:14)

s0; s1
a<-utild1group(arr0=s0, arr1=s1); a
a<-utild1group(s0, s1, radius=2); a
a<-utild1group(s0, s1, radius=2, method='splitN'); a
```

---

utild2bin *Matrix Binarizer*

Description

Function utild2bin is applied to a pair of matrices and convert them to matrices of binaries with a rule specified by user

Usage

```
utild2bin(d2arr0, d2arr1, method = "mean", trsh = 0, inverted = FALSE)
```

Arguments

d2arr0	matrix to be binarized
d2arr1	matrix to be binarized
method	method of setting a threshold: method='mean' threshold is mean(c(d2arr0, d2arr1)) for both matrices (used as default) method='imean' setting individual thresholds as mean values method='imed' setting individual thresholds as median values method='i1stQ' setting individual thresholds as 1st quartile values

method='i3rdQ' setting individual thresholds as 3rd quartile values

method='isubsd' setting individual thresholds as (mean(d2arrN)-sd(d2arrN))

method='iaddsd' setting individual thresholds as (mean(d2arrN)+sd(d2arrN))

method='med' threshold is median(c(d2arr0, d2arr1)) for both

method='med0' threshold is median(c(d2arr0)) for both

method='med1' threshold is median(c(d2arr1)) for both

method='mean0' threshold is mean(c(d2arr0)) for both

method='mean1' threshold is mean(c(d2arr1)) for both

method='spec' threshold is specified by the 2 numbers of argument trsh individually for matrices

trsh — specifies the individual threshold values for binarizations when method='spec' is chosen. If it contains one number it will be interpreted as one threshold for both matrices

inverted — logical item specifying if the result will be inverted (zeros replaced by ones and ones replaced by zeros) after bianrization

Value

bin0 — result of d2arr0 binarization

bin1 — result of d2arr1 binarization

counts — summary/report of how the arrays were binarized

Author(s)

Vitaly Efremov <vitaly.efremov@dcu.ie>

See Also

utild1bin

Examples

```
s0<-array(c(-1,1,3,7,10),c(2,5))
s1<-array(c(1:7), c(7,2))

s0; s1
a<-utild2bin(d2arr0=s0, d2arr1=s1); a
a<-utild2bin(s0, s1, method='isubsd', inverted=TRUE); a
a<-utild2bin(s0, s1, method='spec', trsh=c(-1,2)); a
```

utild2clean *Matrix Cleaner*

Description

Function utild2clean is applied to a pair of matrices and clean them from outliers with a rule specified by user (outliers will be replaced by the threshold values)

Usage

```
utild2clean(d2arr0, d2arr1, method = "bothends", nsigma = 3)
```

Arguments

d2arr0	matrix to be cleaned
d2arr1	matrix to be cleaned
method	specifies if outliers are to be removed from one or both side of matrix value distribution method='bothends' remove outliers from both side of a distribution (default) method='left' remove outliers from left side method='right' remove outliers from right side
nsigma	number of sigmas to be considered as threshold for outlier removal

Value

clean0	result of d2arr0 cleaning
clean1	result of d2arr1 cleaning

Author(s)

Vitaly Efremov <vitaly.efremov@dcu.ie>

See Also

utild1clean

Examples

```
s0<-array(c(-1,1,3,7,10),c(2,5))
s1<-array(c(rep(c(1,2),7),20), c(5,3))

s0;s1
a<-utild2clean(d2arr0=s0, d2arr1=s1); a
a<-utild2clean(s0, s1, method='left', nsigma=0.3); a
```

`utild2filt` *Gaussian Filter for Matrices*

Description

The function is applied to a pair of matrices as a lowpass 2d gaussian filter to clean them from high frequency components

Usage

```
utild2filt(d2arr0, d2arr1, outsize = 2, strong = 1)
```

Arguments

`d2arr0`	vector to be filtered
`d2arr1`	vector to be filtered
`outsize`	radius of gaussian filter kernel
`strong`	multiplication factor defining the sigma of gaussian filter kernel as `sigma=outsize*strong`

Value

`filt0`	matrix being the result of d2arr0 filtration
`filt1`	matrix being the result of d2arr1 filtration

Author(s)

Vitaly Efremov <vitaly.efremov@dcu.ie>

See Also

`utild1filt`

Examples

```
s0<-array(c(1,4,5,8,3), c(6,6))
s1<-array(c(1,1,1,1,1,8,1,1,1,1,1,1), c(5,6))

s0; s1
utild2filt(d2arr0=s0, d2arr1=s1)
utild2filt(s0, s1, outsize=1)
utild2filt(s0, s1, strong=.2)
```

utild2group *Matrix Framer*

Description

The function is applied to a pair of matrices in order to split them into NxN pixel frames and calculate respective matrices of frame means, there N is specified by user

Usage

```
utild2group(d2arr0, d2arr1, radius = 1, method = "split1")
```

Arguments

d2arr0	original matrix
d2arr1	original matrix
radius	sets the size of a single frame as N=(2*radius+1) points/pixels
method	sets the division of original array by a sequence of frames in different manner method='split1' (the default) sets a sequence of frames as: frame11(1...N by 1...N), frame12(1...N by N+1,..2*N), frame21(N+1,..2*N by 1,..N), frame22(N+1,..2*N by N+1,..2*N), frame23(N+1,..2*N by 2*N+1,..3*N),etc method='splitN' sets a sequence of frames as: frame11(1...N by 1...N), frame12(1...N by 2,..N+1), frame21(2,..N+1 by 1...N), frame22(2,..N+1 by 2,..N+1), frame23(2,..N+1 by 3...N+2), etc

Value

group0	matrix containing mean values of individual frames the original d2arr0 was divided by
group1	matrix containing mean values of individual frames the original d2arr1 was divided by

Author(s)

Vitaly Efremov <vitaly.efremov@dcu.ie>

See Also

utild1group

Examples

```
s0<-array(c(-1,1,3,7,10),c(14,12))
s1<-array(c(1:156),c(12,13))

s0; s1
a<-utild2group(d2arr0=s0, d2arr1=s1); a
a<-utild2group(s0, s1, radius=2, method='splitN'); a
a<-utild2group(s0, s1, radius=3); a
```

Index