Package ‘tiger’

February 20, 2015

Type Package

Title TIme series of Grouped ERrors

Version 0.2.3.1

Date 2013-08-09

Author Dominik Reusser

Maintainer Dominik Reusser <reusser@pik-potsdam.de>

Description Temporally resolved groups of typical differences (errors) between two time series are determined and visualized

Depends R (>= 2.10.0), e1071, hexbin, qualV, klaR, som

Imports lattice

License GPL-2

NeedsCompilation no

Repository CRAN

Date/Publication 2014-08-10 19:06:53

R topics documented:

tiger-package 2
change.order.clusters 3
color.factor 4
correlated 4
count.diff.direction.error 6
diagnostic 7
eD 9
example.peaks 10
include.others 10
k_hyd 11
k_rel 12
lagtime 13
nashS 14
plots 15
synth.peak.error 18

tiger 20
tiger.res 22
to.uniform 23
validityIndex 24

Index **26**

---

tiger-package *TIme series of Grouped ERrors*

Description

About fifty performance measures are calculated for a gliding window, comparing two time series. The resulting matrix is clustered, such that each time window can be assigned to an error type cluster. The mean performance measures for each cluster can be used to give meaning to each cluster. Additionally, synthetic peaks are used to better characterize the clusters. The package provides functions to calculate and visualize these results.

Details

Use tiger to perform the calculations. See the package vignette for an example, how to perform calculations and how to evaluate information

Author(s)

Dominik Reusser

Maintainer: Dominik Reusser <dreusser@uni-potsdam.de>

References

Reusser, D. E., Blume, T., Schaefli, B., and Zehe, E.: Analysing the temporal dynamics of model performance for hydrological models, Hydrol. Earth Syst. Sci. Discuss., 5, 3169-3211, 2008.

Examples

```
data(tiger.example)
modelled <- tiger.single$modelled
measured <- tiger.single$measured
peaks <- synth.peak.error(rise.factor=2, recession.const=0.02, rise.factor2=1.5)
## Not run: result2 <- tiger(modelled=modelled, measured=measured, window.size=240, synthetic.errors=peaks)
errors.in.time(d.dates, result2, solution=6, show.months=TRUE)
## End(Not run)
```

change.order.clusters *Change numbering of clusters*

Description

Changes the cluster numbering in an fuzzy clustering object.

Usage

```
change.order.clusters(clustering, new.order)
```

Arguments

clustering	Object returned from cmeans
new.order	Vector with new cluster numbering.

Details

Cluster 1 from the old object is assigned the number stored in the frist position in new.order, Cluster 2 the number on the second position and so on.

Value

Identical object as `clustering` except the cluster numbering is changed

Author(s)

Dominik Reusser

References

Reusser, D. E., Blume, T., Schaefli, B., and Zehe, E.: Analysing the temporal dynamics of model performance for hydrological models, Hydrol. Earth Syst. Sci. Discuss., 5, 3169-3211, 2008.

See Also

`cmeans` for the fuzzy clustering

Examples

```
data(tiger.example)

new.order <- c(6,3,2,5,4,1)

cmeans.result <- tiger.single$cluster.assignment[[6]]
str(cmeans.result)
cmeans.result2 <- change.order.clusters(cmeans.result, new.order)
```

`color.factor` *Create colors with intensity according to the magnitude of a value*

Description

Create colors with intensity according to the magnitude of a value

Usage

```
color.factor(color, value, max)
```

Arguments

`color`	The base color(s) to use
`value`	A vector of values
`max`	The maximum value represented by full intensity

Value

A vector of colors, one entry for each value

Author(s)

Dominik Reusser

Examples

```
data <- 1:10
cols=color.factor("red", data, max=10)
plot(data, col=cols)


cols=color.factor(c("red","green","blue"), data, max=10)
plot(data, col=cols)
```

`correlated` *Calculate correlation structure*

Description

Calculate the correlation structure between multiple performance measures

Usage

```
correlated(result, limit = 0.85, plot.scatter = FALSE, keep = NA)
correl(measures, limit = 0.85, plot.scatter = FALSE, keep = NA)
```

Arguments

- `result` — object returned from `tiger`
- `measures` — data.frame for which to determine correlation structure
- `limit` — Limit for absolute correlation, above which data is considered to be correlated
- `plot.scatter` — Boolean, indicating whether to show pairwise plots for correlated measures
- `keep` — Vector with names of measures that must not be excluded because of correlation with other measures

Value

correl returns:

- `pairs` — Matrix with indices of pairwise correlated measures
- `pairs.by.name` — Matrix with measure names of pairwise correlated measures
- `possible.exclusion` — List indicating which measures might be removed to end up with no strongly correlated measures. The list also indicates, which measure is correlated to the removed measures
- `to.drop` — List of indices for measures to drop (according to previous list)
- `to.drop.by.name` — List of measure names (of the previous list)

correlated returns a list of two correl results, one for the original performance measures and one for the transformed measures from a result from `tiger`.

Author(s)

Dominik Reusser

See Also

This method helps to reduce the amount of data to be analyzed from an evaluation using `tiger`

Examples

```
data(tiger.example)
correlated <- correlated(tiger.single, keep=c("CE","RMSE" ))
```

count.diff.direction.error

Compare sign of derivatives

Description

Counts the number of elements for which two vectors show different signs in the derivative.

Usage

```
count.diff.direction.error(x, y)
```

Arguments

x	First vector
y	Second vector

Value

```
sum((diff(x) / diff(y))<0, na.rm=T)
```

Author(s)

Dominik Reusser

See Also

diagnostic_dawson

Examples

```
#All different
count.diff.direction.error(1:10,10:1)

#One different
count.diff.direction.error(1:10,c(1:9, 8))
```

`diagnostic` *Calculate a number of objectives to compare time series*

Description

`diagnostic_dawson` take two vectors (assumed to be time series) and calculates the following objective functions to compare them: correalation, Nash Sutcliffe efficiency, ratio of the integral, lagtime (maximum of the cross correlation), the number of timesteps with opposite sign of the derivative, the highest ratio between recession coefficients and the root mean square error, as well as the ones listed in Dawson 2007.

`diagnostic_window` calcualtes these measures for a part of the time series only. It is used internally by

`diagnostic_series` takes this a step further by calculating the above measures for a gliding window along the time series and calculating additional measures. Similar to `diagnostic`, the function takes two vectors (assumed to be time series) and calculates a number of objectives compare them. In contrast to the more simple `diagnostic`, the same objectives are applied to a gliding window and a few additional objectives are calcualated: the ratio of the derivatives, the ratio of the recession coefficients for each time step and the current quantile of the residuals.

Usage

```
diagnostic_window(position, window.size, measured,
modelled, use_qualV = FALSE,
diff.ecdf=NA)
diagnostic_series(measured, modelled, window.size,
step.size = 1, integral_correction = FALSE,
use_qualV = FALSE)
diagnostic_dawson(modelled, measured, p=NA, m=NA, additional=TRUE,
use_qualV=FALSE, diff.ecdf=NA )
```

Arguments

`modelled`	Modelled time series or array with dimension `c(number_series, dim(measured))`
`measured`	Measured time series
`position`	Index from where to start the calculation
`window.size`	Number of time steps to include
`step.size`	Size of the steps defining the number of scores to be calculating along the time series. For example, with a value of 5 every fifth value is included
`integral_correction`	Boolean. If true, the ratio of the integrals is divided by the total ratio of the entire integral. This way, relative integral errors can be detected.
`p`	The number of free parameters in each model - required to calculate AIC and BIC
`m`	The number of data points that were used in the model calibration - required to calculate AIC and BIC

`additional`	Boolean, indicating whether to calculate additional measures to the ones defined in Dawson 2007
`use_qualV`	Boolean, indicating whether to calculate the additional measures defined in Jachner 2007
`diff.ecdf`	`ecdf`-function of the bias (measured-modelled)

Details

For more details on the objectives, see the see-also-section

Value

A data frame with the described objectives

Author(s)

Dominik Reusser

References

Dawson, C. W.; Abrahart, R. J. & See, L. M. HydroTest: A web-based toolbox of evaluation metrics for the standardised assessment of hydrological forecasts Environmental Modelling & Software, 2007 , 22 , 1034-1052

Jachner, S.; van den Boogaart, K. G. & Petzoldt, T. Statistical Methods for the Qualitative Assessment of Dynamic Models with Time Delay (R Package qualV) Journal of Statistical Software, 2007 , 22 , 1-30

See Also

`qualVcor`, `nashS`,`lagtime`,`count.diff.direction.error`,`k_rel`

Examples

```
  data(example.peaks,package="tiger")

  plot(reference.peak, type="l")
  lines(example.peaks[1,], lty=2)

  diagnostic_dawson(measured = reference.peak, modelled = example.peaks[1,])

  #first half only
  diagnostic_window(measured=reference.peak, modelled=example.peaks[1,],
position = 1, window.size = 45 )

  #gliding window for 20 time steps
  diagnostic_series(measured=reference.peak, modelled=example.peaks[1,],
window.size = 20 )
```

`eD` *Euclidean Distance*

Description

Calculate the euclidean distance for multiple vectors

Usage

```
eD(x, y)
```

Arguments

`x`	matrix with first set of vectors.
`y`	matrix with second set of vectors.

Details

x and y need the following structure to compare multiple vectors at once:

rows contain the k vectors

columns the n coordinates in the n-space

`str(x) ==  matrix [1:k, 1:n]`

Value

vector with the euclidean distance for each pair

Author(s)

Dominik Reusser

Examples

```
eD(1:3, 2:4)
```

example.peaks *Synthetic peak errors*

Description

A number of synthetic peak errors used for testing performance measures and similar

Usage

```
data(example.peaks)
```

Format

The format for example.peaks is: num [1:12, 1:91] 0.1346 0.1346 0.1846 0.0846 0.1346 ...

The format for the reference.peak is: num [1:91] 0.135 0.134 0.134 0.134 0.134 ...

Examples

```
data(example.peaks)
str(example.peaks)
str(reference.peak)
plot(reference.peak,type="line")
lines(example.peaks[,1], lty=2)
diagnostic_dawson(measured = reference.peak, modelled = example.peaks[1,])
## maybe str(peaks) ; plot(peaks) ...
```

include.others *Internal Function: evaluate box plot*

Description

Find clusters with a comparable position on the box plot with respect to the best value. Comparable position means the median of one set of values falls within the interquartile range of the reference set of values

Usage

```
include.others(selected, center, stats, best = FALSE)
```

Arguments

`selected`	index of the best value set.
`center`	where is the best value within the
`stats`	stats element of a boxplot result
`best`	are we comparing against the best set?

Value

Vector of indizes for which elements are comparable

Author(s)

Dominik Reusser

See Also

`box.plots`

`k_hyd` *Hydrological recession constant*

Description

This function calculates the local hydrological recession constant for each point in a time series. The function returns `NA` for periods with increasing discharge.

Usage

```
k_hyd(x)
```

Arguments

`x`	discharge time serie

Details

The function returns `-dx/dt*1/x` if `dx/dt` is larger than 0 and x is not 0. For the other cases, NA is returned.

Value

Vector of recession constants.

Author(s)

Dominik Reusser

References

Blume Recession Paper

See Also

`diagnostic_dawson`

Examples

```
data(example.peaks,package="tiger")

k_hyd(reference.peak)
```

---

k_rel *Mean ratio of hydrological recession constants of two discharge time series*

Description

This function calculates the mean ratio between local hydrological recession constant for each point in two discharge time series.

Usage

```
k_rel(x, y)
```

Arguments

x	discharge time serie
y	discharge time serie

Value

A scalar with the mean ratio

Author(s)

Dominik Reusser

See Also

k_hyd, diagnostic_dawson

Examples

```
data(example.peaks,package="tiger")

k_rel(reference.peak, example.peaks[1,])
```

lagtime — *Lagtime between two time series*

Description

This function calculates the lagtime between x and y, defined as the shift resulting in the maximum cross correlation.

Usage

```
lagtime(x, y)
```

Arguments

x	Time series
y	Time series

Value

The lagtime as scalar. Positive if x is shifted towards later times.

Author(s)

Dominik Reusser

See Also

ccf,diagnostic_dawson

Examples

```
data(example.peaks,package="tiger")
plot(reference.peak, type="l")
lines(example.peaks[7,], lty=2)
lagtime(reference.peak, example.peaks[7,])
```

`nashS` *Calculates the (weighted) Nash Suttcliffe Efficiency coefficient*

Description

Calculates the Nash Suttcliffe Efficiency coefficient.

Usage

```
nashS(modelled, measured, weigth = NA)
nashS_HF(modelled, measured, weigth = NA)
```

Arguments

`modelled`	Vector with modeled data
`measured`	Vector with measured data
`weigth`	If this vector is supplied, each data point is weighted accordingly

Details

The weighting corresponds to the value in the empirical cumulative distribution function.

Value

Returns a scalar between -Inf and 1 corresponding to the agreement between measured and modelled data. 0 means the model agrees equally well as the mean value.

Author(s)

Dominik Reusser

References

http://en.wikipedia.org/wiki/Nash-Sutcliffe_efficiency_coefficient

Examples

```
ref.peak <- synth.peak(rise.factor=2, recession.const=0.02)
peak <- synth.peak(rise.factor=2, recession.const=0.03)
nashS(modelled=peak, measured=ref.peak)
```

plots — *Evaluation plots for temporal dynamics of model performance*

Description

Create various plot to understand the temporal dynamics of model performance

Usage

```
box.plots(result, solution, show.measures = 1:num.measures,
             new.order = 1:solution, show.synthetic.peaks = FALSE,
             synthetic.peaks.col = c(2:8, 2:8), show.timestep = NA,
     show.cell = NA,
                 ref = NULL, ref.new.order = new.order, ref.solutions =
                 solution, col.best.match = "black",
                 clusterPalette = rainbow(solution))
errors.in.time(xval, result, solution, rain.data = NULL, show.months
                 = FALSE, new.order = 1:solution, x.range =
                 1:length(xval), pmax = max(c(result$measured,
                 result$modelled), na.rm = TRUE), data.colors =
                 data.frame(measured = c("grey"), modelled =
                 c("black"), rain = c("black")), clusterPalette =
                 rainbow(solution), color.cut.off = 0, frac.max = 0.7,
                 frac.min = 0.4, grid.nx = 0, legend.pos = "topleft",
                 show.data = TRUE, show.errors = TRUE, show.data.model
                 = show.data, show.data.measured = show.data, ...)
peaks.in.clusters(result, solution,  new.order = 1:solution)
peaks.on.som(result, solution, clusterPalette=rainbow(solution),
 cell.size = 0.9, mfrow=c(2,ceiling(n.errors/2)),
 new.order=1:solution)
peaks.measures(result, show.measures = 1:num.measures,
                 synthetic.peaks.col = c(2:8, 2:8), mfrow = c(2, 3),
                 col.best.match = "black", do.out = rep(TRUE,
                 length(show.measures)), single.errors = FALSE,
                 show.legend = TRUE, show.main = TRUE, y.range = NULL)
scatterplot(measures, show.measures=1:num.measures)
p.validityIndex(result, validity.max)
```

Arguments

`result`	object returned from `tiger`
`measures`	data.frame from which to create a scatter plot. e.g. result\$measures.uniform
`solution`	number of clusters to use for further evaluations (see also `validityIndex`)
`single.errors`	Boolean, indicating weather different synthetic errors should be combined into a single plot or shown in multiple plots
`show.legend`	Boolean, indicating whether to show the legend

`show.main`	Boolean, indicating whether to show performance measure names as plot title
`show.measures`	vector of indices indicating for which performance measures to show the plots
`new.order`	New numbering to assign to clusters. See also `change.order.clusters`
`show.synthetic.peaks`	Show values of the synthetic peaks on top of the box plots.
`synthetic.peaks.col`	Colors to use for synthetic peaks.
`do.out`	vector of booleans indicating whether to exclude outliers when showing the plot
`cell.size`	fraction of the cell square to be filled with color
`show.cell`	the scores for a certain cell on the SOM can be ploted as blue line on the box plot (see examples)
`x.range`	Indizes of x-values to be plotted
`y.range`	Range for y axis
`pmax`	maximum discharge for definition of the plot range
`frac.min`	minimum of the y-range covered by color bars for cluster occurence
`frac.max`	maximum of the y-range covered by color bars for cluster occurence
`clusterPalette`	colors to use for the clusters
`color.cut.off`	Value of cluster occurence below which the color bar is set to transparent (for better readability)
`legend.pos`	Position of the legend
`data.colors`	Color definition for rainfall and runoff
`show.timestep`	timestep for which the values for the performance measures are to be plotted as black lines in the box plot
`xval`	Values to be plotted on the x-axis (e.g. POSIX-date)
`show.months`	Boolean indicating whether to add month ticks to x axis
`mfrow`	see `par`
`ref`	Reference solution to be ploted in grey on the box plot
`ref.new.order`	New numbering to assign to clusters for reference solution on the box plot
`ref.solutions`	Number of clusters for reference solution for which to plot the box plot
`validity.max`	Do not plot solutions with cluster numbers resulting above in a validty index above validity.max
`col.best.match`	Color to use for plotting the line indicating the position of the best match
`rain.data`	vector with rainfall data
`show.data`	boolean, indicating whether to show discharge data
`show.data.measured`	boolean, indicating whether to show measured discharge data
`show.data.model`	boolean, indicating whether to show modeled discharge data
`show.errors`	boolean, indicating whether to show error type bars
`grid.nx`	number of grid lines to be ploted (see `grid`)
`...`	additional parameters passed to plot

Details

box.plots: for each performance measure, a box plot is created showing the values for each cluster

errors.in.time: occurence of the errors cluster along the time dimension

peaks.in.clusters: table of the position of the synthetic peak errors in the clusters.

peaks.measures: responce of the performance measures to the synthetic peak errors.

scatterplot: scatter plot of the performance measures

See package vignette for further details about which plot does what.

Value

used for the side effect of plotting results

Author(s)

Dominik Reusser

References

Reusser, D. E., Blume, T., Schaefli, B., and Zehe, E.: Analysing the temporal dynamics of model performance for hydrological models, Hydrol. Earth Syst. Sci. Discuss., 5, 3169-3211, 2008.

See Also

The package vignette

Examples

```
data(tiger.example)

new.order <- c(6,3,2,5,4,1)
correlated <- correlated(tiger.single, keep=c("CE","RMSE" ))

opar <- par(mfrow=c(3,5))
box.plots(tiger.single, solution=6, new.order=new.order, show.synthetic.peaks=TRUE)
box.plots(tiger.single, solution=6, new.order=new.order, show.cell=data.frame(x=1,y=1))
par(opar)
errors.in.time(xval=d.dates, result= tiger.single, solution=6,
show.months=TRUE, new.order=new.order)
peaks.in.clusters(tiger.single, solution=6, new.order=new.order)
peaks.measures(tiger.single, show.measures=correlated$measures.uniform$to.keep)
scatterplot(tiger.single$measures.uniform, show.measures=correlated$measures.uniform$to.keep)
```

`synth.peak.error` *Generate synthetic peaks and peak errors*

Description

These functions allow generation of synthetic hydrologic peaks generated from a combination of exponetial functions. Also, synthetic errors for the reproduction of a reference peak can be generated in order to subsequentially test the behaviour of performance measures with respect to these errors.

Usage

```
synth.peak(base = 0.07, base.time = 6, rise.time = 5, rise.factor,
recession.const = 0.2, length.out = 240, rez.time = length.out -
ceiling(base.time) - ceiling(rise.time))
synth.peak.error(base = 0.07, base.time = 6, rise.time = 5,
rise.factor, rise.factor2, recession.const = 0.2, length.out = 240,
rez.time = length.out - base.time - rise.time, err1.factor = c(1.2,
1.4, 1.6), err2.factor = c(0.01, 0.02, 0.04), err3.factor = c(2, 4,
8), err4.factor = c(9, 18, 27), err5.factor = c(0.1, 0.2, 0.4),
err6.factor = c(1.5, 2, 3),
err9.factor = c(2, 3, 4.5))
p.synth.peak.error(peaks, y.max = (max(peaks, na.rm = TRUE)),
                 peak.cluster = NULL, peak.palette = grey(c(0, 0.6,
                 0.8)), use.layout = TRUE, show.errors = 1:n.errors,
                 peak.lty = rep(1, n.errors), mfrow = c(2,
                 ceiling(length(show.errors)/2)), plot.legend = TRUE,
                 print.error.nr = TRUE)
```

Arguments

`base`	level of the base flow component
`base.time`	number of time steps before the rise phase starts. May be negative, such that the peak starts outside the window.
`rise.time`	Number of time steps for the rise phase
`rise.factor`	The peak maximum is about `rise.factor` higher than the base flow.
`rise.factor2`	`rise.factor` for the "peaks" which only show a recession phase.
`recession.const`	Recession constant for the peak.
`length.out`	Total length of the time series to be returned.
`rez.time`	Length of the recession phase
`err1.factor`	Factors to use for the first error type: Over- and underestimation of the peak
`err2.factor`	Factors to use for the second error type: shifting of the entire time series
`err3.factor`	Factors to use for the third error type: Recession too fast/too slow

err4.factor	Factors to use for the fourth error type: Lag time
err5.factor	Factors to use for the fifth error type: Correct total volume, but peak over/underestimated
err6.factor	Factors to use for the sixth error type: Peak too wide/too narrow
err9.factor	Factors to use for the ninth error type: Shift during the recession phase
peaks	object returned from synth.peak.error()
y.max	upper limit for the y-axis
peak.cluster	object returned from peaks.in.clusters used for coloring the cluster assignment of synthetic peaks (see examples)
peak.palette	Colors to use if peak.cluster is NULL: first color for reference, second and third for peaks over- and underestimating the reference
use.layout	Boolean, indicating whether to use the predefined layout
plot.legend	Boolean, indicating whether to show the legend
print.error.nr	Boolean, indicating whether to label each subplot with a number
mfrow	mfrow plot parameter (only used, if use.layout=FALSE)
show.errors	Vector with indices indicating which errors to display
peak.lty	Line types for either clusters as defined in peak.cluster or as in peak.palette

Value

synth.peak returns a vector with the synthetic peak according to the provided parameters synth.peak.error returns an array with dimension 3. The first dimension corresponds to the error type. The second dimension to the level of the corresponding error type. The third dimension corresponds to the time.

Author(s)

Dominik Reusser

References

Reusser, D. E., Blume, T., Schaefli, B., and Zehe, E.: Analysing the temporal dynamics of model performance for hydrological models, Hydrol. Earth Syst. Sci. Discuss., 5, 3169-3211, 2008.

See Also

The package vignette

Examples

```
     ref.peak <- synth.peak(rise.factor=2, recession.const=0.02)
peaks <- synth.peak.error(rise.factor=2, recession.const=0.02, rise.factor2=1.5)
peaks2 <- synth.peak.error(rise.factor=2, recession.const=0.02,
     rise.factor2=1.5, err1.factor=c(1.3,1.5,2.0),
     err2.factor = c(0.02,0.03,0.06),
     err3.factor=c(2,4,10),
     err4.factor = c(9,22,40),
     err5.factor = c(0.2,0.3,0.5),
```

```
    err6.factor =c(2,3,5),
    err9.factor=c(1.5,3,6)
  )

   p.synth.peak.error(peaks)
   p.synth.peak.error(peaks2)

   data(tiger.example)
   peak.cluster <- peaks.in.clusters(result=tiger.single,
 solution=5, new.order=c(2,3,5,1,4))
   p.synth.peak.error(peaks=tiger.single$synthetic.errors,
peak.cluster=peak.cluster, peak.palette=rainbow(5))
```

`tiger` *Calculate temporal dynamics of model performance*

Description

About fifty performance measures are calculated for a gliding window, comparing two time series. The resulting matrix is clustered, such that each time window can be assigned to an error type cluster. The mean performance measures for each cluster can be used to give meaning to each cluster. Additionally, synthetic peaks are used to better characterize the clusters.

Usage

```
tiger(modelled, measured, window.size, step.size = 1,
                 use.som = TRUE, som.dim = c(20, 20), som.init =
                 "sample", som.topol = "hexa", maxc = 15,
                 synthetic.errors = NA)
tiger.peaks(result, synthetic.errors)
```

Arguments

`modelled`	Time series of modelled data
`measured`	Time series of measured data
`window.size`	Size of the moving window
`maxc`	Maximum number of clusters to be tested
`synthetic.errors`	Matrix returned from `synth.peak.error`
`result`	object returned from tiger
`use.som`	boolean, indicating whether to use SOM before applying fuzzy clustering
`som.dim`	Dimension of the Self Organizing Map (SOM) c(x,y)
`som.init`	Method to initialize the SOM
`som.topol`	Topology of the SOM
`step.size`	Size of the steps defining the number of scores to be calculating along the time series. For example, with a value of 5 every fifth value is included

Details

See the package vignette.

Value

`maxc`	see input parameter
`window.size`	see input parameter
`modelled`	see input parameter
`measured`	see input parameter
`synthetic.errors`	see input parameter
`measures.synthetic.peaks`	matrix of performance measures for synthetic errors
`measures`	matrix of performance measures for the gliding time window
`na.rows`	vector of boolean, indicating which time windows contain NA values
`names`	names of the perfomance measures
`measures.uniform`	measures, transformed to uniform distribution
`measures.uniform.synthetic.peaks`	measures for synthetic errors, transformed with the corresponding transformation from previous item
`error.names`	names of the synthetic error types
`best.value.location`	list, indicating what the value for "no error" for each performance measure is
`validityMeasure`	vector with validty index for solutions with 2:maxc clusters
`cluster.assignment`	list of 2:maxc objects returned from `cmeans`

Author(s)

Dominik Reusser

References

Reusser, D. E., Blume, T., Schaefli, B., and Zehe, E.: Analysing the temporal dynamics of model performance for hydrological models, Hydrol. Earth Syst. Sci. Discuss., 5, 3169-3211, 2008.

See Also

The package vignette

Examples

```
data(tiger.example)
modelled <- tiger.single$modelled
measured <- tiger.single$measured
peaks <- synth.peak.error(rise.factor=2, recession.const=0.02, rise.factor2=1.5)
## Not run: result2 <- tiger(modelled=modelled, measured=measured, window.size=240, synthetic.errors=peaks)
errors.in.time(d.dates, result2, solution=6, show.months=TRUE)
## End(Not run)

peaks2 <- synth.peak.error(rise.factor=2, recession.const=0.02,
     rise.factor2=1.5, err1.factor=c(1.3,1.5,2.0),
     err2.factor = c(0.02,0.03,0.06),
     err3.factor=c(2,4,10),
     err4.factor = c(9,22,40),
     err5.factor = c(0.2,0.3,0.5),
     err6.factor =c(2,3,5),
     err9.factor=c(1.5,3,6)
   )

## Not run: result3 <- tiger.peaks(result2, peaks2)

   peaks.in.clusters(result2, solution=6)
   x11()
   peaks.in.clusters(result3, solution=6)

## End(Not run)
```

`tiger.res` *Example data for TIGER package*

Description

Example data for tempperform package

Usage

```
data(tiger.example)
```

Format

Object returned from `tiger`.

Source

Reusser, D. E., Blume, T., Schaefli, B., and Zehe, E.: Analysing the temporal dynamics of model performance for hydrological models, Hydrol. Earth Syst. Sci. Discuss., 5, 3169-3211, 2008.

Examples

```
data(tiger.example)
errors.in.time(d.dates, tiger.multi, solution=6, show.months=TRUE)
```

`to.uniform` *Transform data to uniform distribution*

Description

Transform data to uniform distribution. Optionally, a set of values can be transformed against a reference set of data.

Usage

```
to.uniform(ref, val = NA)
```

Arguments

`ref`	Set of values that determince the transformation
`val`	Values to be transformed

Details

If values is NA, the reference set itself will be transformed.

Value

Vector with transformed values.

Author(s)

Dominik Reusser

Examples

```
a <- rnorm(100)
hist(a)
b <- to.uniform(a)
hist(b)
c <- to.uniform(ref=a, val=c(-0.5,0,0.5))
```

`validityIndex` *Validity Index for fuzzy clustering*

Description

Calculate the validity index for fuzzy clusters. A validity index below 1 indicates, that in between clusters is larger than within clusters. Evaluating the validity index for various numbers of desired clusters may help to find the minimum.

Usage

```
validityIndex(cclust, values, verbose = FALSE)
```

Arguments

`cclust`	object returned from `cmeans`
`values`	data provided as `x` to `cmeans`
`verbose`	boolean. If true, values for numerator and denominater are printed.

Value

A single number, the validity index.

Author(s)

Dominik Reusser

References

Reusser, D. E., Blume, T., Schaefli, B., and Zehe, E.: Analysing the temporal dynamics of model performance for hydrological models, Hydrol. Earth Syst. Sci. Discuss., 5, 3169-3211, 2008.

Xie, X. and Beni, G.: A validity measure for fuzzy clustering, IEEE T. Pattern Anal., 13, 841-847, 1991. 3181

See Also

`cmeans` for the fuzzy clustering itself

Examples

```
myOrig <- matrix(c(c(1,0,0,1),
         c(0,0,1,2),
         c(1,1,0,3)), nrow=3, ncol=4, byrow=TRUE)


myData <- rbind(
     matrix(myOrig[1,], nrow=50, ncol=4, byrow=TRUE),
     matrix(myOrig[2,], nrow=50, ncol=4, byrow=TRUE),
```

```
     matrix(myOrig[3,], nrow=50, ncol=4, byrow=TRUE)
     )
str(myData)
myData[,1:3] <- myData[,1:3] + rnorm(3*150)*0.3
myData

maxc <- 10

require(e1071)
    validity <- rep(NA, maxc)
    all.cluster.rer <- list()
    for(centers in 2:maxc){
        cluster.rer<-cmeans(x=myData, centers=centers, method="cmeans", m=2)
        validity[centers] <- validityIndex(cluster.rer , myData)
        all.cluster.rer[[centers]] <- cluster.rer
    }

  plot(validity, type="l")
```

Index

∗Topic **color**
color.factor, 4
∗Topic **datasets**
example.peaks, 10
tiger.res, 22
∗Topic **hplot**
plots, 15
synth.peak.error, 18
∗Topic **package**
tiger-package, 2
∗Topic **univar**
nashS, 14
∗Topic **utilities**
change.order.clusters, 3
correlated, 4
count.diff.direction.error, 6
diagnostic, 7
eD, 9
include.others, 10
k_hyd, 11
k_rel, 12
lagtime, 13
nashS, 14
plots, 15
synth.peak.error, 18
tiger, 20
to.uniform, 23
validityIndex, 24

box.plots, *11*
box.plots (plots), 15

ccf, *13*
change.order.clusters, 3, *16*
cmeans, *3*, *21*, *24*
color.factor, 4
cor, *8*
correl (correlated), 4
correlated, 4
count.diff.direction.error, 6, *8*

d.dates (tiger.res), 22
diagnostic, 7
diagnostic_dawson, *6*, *11–13*
diagnostic_dawson (diagnostic), 7
diagnostic_series (diagnostic), 7
diagnostic_window (diagnostic), 7

ecdf, *8*
eD, 9
errors.in.time (plots), 15
example.peaks, 10

grid, *16*

include.others, 10

k_hyd, 11, *12*
k_rel, *8*, 12

lagtime, *8*, 13

nashS, *8*, 14
nashS_HF (nashS), 14

p.synth.peak.error (synth.peak.error), 18
p.validityIndex (plots), 15
par, *16*
peaks.in.clusters, *19*
peaks.in.clusters (plots), 15
peaks.measures (plots), 15
peaks.on.som (plots), 15
plots, 15

qualV, *8*

reference.peak (example.peaks), 10

scatterplot (plots), 15
synth.peak (synth.peak.error), 18
synth.peak.error, 18, *20*

tiger, *5*, *15*, 20, *22*
tiger-package, 2
tiger.example (tiger.res), 22
tiger.multi (tiger.res), 22
tiger.res, 22
tiger.single (tiger.res), 22
to.uniform, 23

validityIndex, *15*, 24