Package ‘KarsTS’

January 14, 2021

Type Package
Title An Interface for Microclimate Time Series Analysis
Version 2.4.1
Date 2021-01-13
Author Marina Saez [aut, cre],
      David Benavente [ths],
      Soledad Cuezva [ths],
      Concepcion Pla [ctb]
Maintainer Marina Saez <marinasaez_andreu@hotmail.com>
Description An R code with a GUI for microclimate time series, with an emphasis on under-
ground environments. 'KarsTS' provides linear and nonlinear methods, including recur-
rence analysis (Marwan et al. (2007) <doi:10.1016/j.physrep.2006.11.001>) and filling meth-
ods (Moffat et al. (2007) <doi:10.1016/j.agrformet.2007.08.011>), as well as tools to manipu-
late easily time series and gap sets.
License GPL (>= 2)
Note Please, cite this package as: Marina Saez (2021). KarsTS: An Interface For Microclimate Time Series Analysis. R package version 2.4.1.
Imports circular, MVN, tcltk, tcltk2, tseriesChaos, stlplus, tseries, forecast, stinepack, missForest, nonlinearTseries, stats, graphics, utils, zoo, grDevices, rgl, mgcv, infotheo, plot3D
Depends R(>= 3.5.0)
NeedsCompilation no
Repository CRAN
Date/Publication 2021-01-14 20:50:05 UTC

R topics documented:

  KarsTS-package ......................................................... 5
  aboutKTS .............................................................. 6
  aggregateKTS .......................................................... 6
**R topics documented:**

- `createTITLE` ................................................. 32
- `createTitle` ................................................. 33
- `createTsChb` .................................................. 33
- `createTsRb` .................................................. 34
- `cumuKTS` ..................................................... 34
- `destroyMainScreen` ........................................... 35
- `destroyWelcome` .............................................. 35
- `determinismKTS` .............................................. 35
- `diffKTS` ..................................................... 36
- `E1dAndE2d` .................................................. 36
- `embedData` ................................................... 37
- `endingLines` ............................................... 37
- `exportall` .................................................. 38
- `fillWithTwins` ............................................. 38
- `findDateFormat` ............................................ 39
- `findTwins` .................................................. 39
- `fnnKTS` ..................................................... 40
- `funcToExport` .............................................. 41
- `gamKTS` ..................................................... 41
- `gapCheckedTF` ............................................. 42
- `gapDetect` .................................................. 42
- `gapForSelMethod` .......................................... 43
- `genGapExample` ............................................ 44
- `genRmExample` .............................................. 44
- `genTSExample` ............................................. 45
- `getClassEnvir` ............................................ 46
- `getCoordsKTS` ............................................. 46
- `getCRP` ..................................................... 47
- `getDelayCharTimes` ........................................ 48
- `getFANRM2` ................................................ 48
- `getGapsAfterFill` ......................................... 49
- `getMaxNegSlope` .......................................... 50
- `getMaxPosSlope` .......................................... 50
- `getNAsGaps` .............................................. 51
- `getNewGapsInd` ........................................... 51
- `getOtherErrEstim` ........................................ 52
- `getProTaos` ............................................... 53
- `getRecurrencePoints` ..................................... 54
- `getRollStatistics` ........................................ 55
- `getSamPerTable` .......................................... 56
- `getSamPerTable.1Freq` .................................... 56
- `getScreenSize` ............................................ 57
- `getStatistics` ............................................ 57
- `getUniqueSampPer` ....................................... 58
- `goodnessFilling` .......................................... 58
- `groupDates` ............................................... 59
- `groupIndices` ............................................ 59
- `histKTS` ................................................... 60
<table>
<thead>
<tr>
<th>R topics documented:</th>
</tr>
</thead>
<tbody>
<tr>
<td>invariantsKTS</td>
</tr>
<tr>
<td>isTimeAlright</td>
</tr>
<tr>
<td>KarsTS</td>
</tr>
<tr>
<td>laminarityKTS</td>
</tr>
<tr>
<td>linCorrKTS</td>
</tr>
<tr>
<td>linearityKTS</td>
</tr>
<tr>
<td>loadAllTypes</td>
</tr>
<tr>
<td>loadKarsTSFonts</td>
</tr>
<tr>
<td>loessKTS</td>
</tr>
<tr>
<td>mainScreen</td>
</tr>
<tr>
<td>meanValue</td>
</tr>
<tr>
<td>mergeTsOrGap</td>
</tr>
<tr>
<td>missForestKTS</td>
</tr>
<tr>
<td>modeKTS</td>
</tr>
<tr>
<td>mutInf</td>
</tr>
<tr>
<td>mutualKTS</td>
</tr>
<tr>
<td>myApplyVector</td>
</tr>
<tr>
<td>myLinModel</td>
</tr>
<tr>
<td>myScale</td>
</tr>
<tr>
<td>naApproxKTS</td>
</tr>
<tr>
<td>NAs4Resamp</td>
</tr>
<tr>
<td>naSplinesKTS</td>
</tr>
<tr>
<td>normalityKTS</td>
</tr>
<tr>
<td>packagesToImport</td>
</tr>
<tr>
<td>pcaKTS</td>
</tr>
<tr>
<td>plotTimeSeries</td>
</tr>
<tr>
<td>readMultEntryvalues</td>
</tr>
<tr>
<td>refreshDataSetsList</td>
</tr>
<tr>
<td>removeAllTypes</td>
</tr>
<tr>
<td>removeIfExists</td>
</tr>
<tr>
<td>removePoints</td>
</tr>
<tr>
<td>renameAllTypes</td>
</tr>
<tr>
<td>rmCheckedTF</td>
</tr>
<tr>
<td>rmDetect</td>
</tr>
<tr>
<td>rmSlopeOutliers</td>
</tr>
<tr>
<td>rollStatisticsKTS</td>
</tr>
<tr>
<td>roundKTS</td>
</tr>
<tr>
<td>RPKTS</td>
</tr>
<tr>
<td>saveAllTypes</td>
</tr>
<tr>
<td>saveReport</td>
</tr>
<tr>
<td>scaleKTS</td>
</tr>
<tr>
<td>scattTimeSeries</td>
</tr>
<tr>
<td>selectionGaps</td>
</tr>
<tr>
<td>selectionTS</td>
</tr>
<tr>
<td>separateEntry</td>
</tr>
<tr>
<td>setCorrectDate</td>
</tr>
<tr>
<td>setwdKTS</td>
</tr>
<tr>
<td>showHelp</td>
</tr>
</tbody>
</table>
**Description**

An R code with a GUI for microclimate time series, with an emphasis on underground environments. 'KarsTS' provides linear and nonlinear methods, including recurrence analysis (Marwan et al. (2007) <doi:10.1016/j.physrep.2006.11.001>) and filling methods (Moffat et al. (2007) <doi:10.1016/j.agrformet.2007.08.011>), as well as tools to manipulate easily time series and gap sets.

**Details**

*KarsTS* is a package for microclimate time series, with an emphasis on underground environments, such as caves. Microclimate research typically includes CO2 and Rn concentrations, temperature and humidity time series, amongst others. Many of these time series have a strong nonlinear behavior and they often contain significant gaps. *KarsTS* provides linear and non-linear analysis and filling methods, as well as tools to manipulate easily time series and gap sets.

The interface *KarsTS* is opened by running the function `KarsTS` on the R or RStudio console. It has five menus: Time Series, Gap Sets, Analysis, Plots and Filling.

**Time Series menu:** time series basic manipulation (loading, saving, resampling, scaling, rounding, etc.).

**Gap Sets menu:** gap sets basic manipulation (loading, saving, selection, MCAR Little’s test etc.). Gap sets manipulation allows to apply a filling method to a subset of gaps in a time series (for example, gaps smaller than a certain length).

**Analysis menu:** linear and non-linear analytic procedures (statistics, rolling statistics, loess decomposition, invariants, recurrence matrices, stationarity and linearity tests etc.).

**Plots menu:** tools for plotting recurrence matrices, time series, phase portraits, manual removal
of points etc. It contains also analytic procedures with mainly graphical results (linear correlation, mutual information, false nearest neighbors etc.). **Filling menu:** univariate and multivariate methods to fill missing values in time series (interpolation, ARIMA, random forest algorithm etc.).

See the **User’s Guide** for more information.

**Note**

Please, cite this package as:

**Author(s)**

Marina Saez [aut, cre], David Benavente [ths], Soledad Cuezva [ths], Concepcion Pla [ctb]
Maintainer: Marina Saez <marinasaez_andreu@hotmail.com>

<table>
<thead>
<tr>
<th>aboutKTS</th>
<th>aboutKTS: opens information file about KarsTS version</th>
</tr>
</thead>
</table>

**Description**

This function opens a file containing information about KarsTS version

**Author(s)**

Marina Saez Andreu

<table>
<thead>
<tr>
<th>aggregateKTS</th>
<th>aggregateKTS: creation of time series of aggregated values</th>
</tr>
</thead>
</table>

**Description**

This function creates time series of aggregated values. The user choses the inputs via interface: time series to aggregate, aggregation period, statistic, name for the output and NA treatment (see details).

**Details**

The statistics available are: median, mean, minimum, maximum, standard deviation and sum. The NA treatment can be: ignore or propagate. In the first case, the statistic is computed using the observations available in the window, as long as the window is not completely missing. In the second case, incomplete windows are assigned NA.
**anaSamPer**

**Value**

A time series of aggregated values appears in the environment KTSEnv

**Author(s)**

Marina Saez Andreu

---

**Description**

Time series need to have a homogeneous sampling period for most calculations. This function divides a time series in homogeneous-sampling-period pieces that contain either observations or NAs. This information allows to decide how to resample the time series. Note that it also provides a list of gaps.

**Details**

Once chosen the time serie to analyze, KarsTS offers a list of possible sampling periods (time jumps in the time series), but a human decision is necessary to separate safely true sampling periods from gaps. Typically, time jumps corresponding to sampling periods appear many times.

**Value**

A table containing the aforementioned information appears on KarsTS output window.

**Author(s)**

Marina Saez Andreu

---

**applyGap2TSer**

**applyGap2TSer: apply a gap set to a time series**

**Description**

This function applies a set of gaps to a time series.

**Details**

The new time series name is a combination of the original time series and the gap set names.

**Value**

A new time series with NAs at the locations indicated by the gap set
**Author(s)**

Marina Saez Andreu

---

**applyTheiler: apply Theiler's window**

**Description**

It is used to apply a Theiler's window to a matrix.

**Usage**

```r
applyTheiler(RM, thW)
```

**Arguments**

- **RM** The recurrence matrix, in KarsTS format
- **thW** The window, in lags from the diagonal

**Value**

A recurrence matrix, in KarsTS format, where the diagonals from the main diagonal up to the distance given by the window have been removed.

**Author(s)**

Marina Saez Andreu

**Examples**

```r
RM <- genRmExample(InKTSEnv = FALSE, plotRM = FALSE)
RM1 <- applyTheiler(RM, 10)
```
are2TsTimeCompatible

are2TsTimeCompatible: test time compatibility between two time series

Description

It checks for three types of compatibility: the initial dates are the same, the sampling period is the same and the final date is the same. It is used internally to check whether the inputs chosen by the user are appropriated.

Usage

are2TsTimeCompatible (TS1, TS2)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TS1</td>
<td>A time series</td>
</tr>
<tr>
<td>TS2</td>
<td>Another time series</td>
</tr>
</tbody>
</table>

Value

A logical vector, which is c(TRUE, TRUE, TRUE) when all the time compatibility conditions are met.

Author(s)

Marina Saez Andreu

Examples

```r
## Generate two time series
TS1 <- genTSExample(InKTSEnv = FALSE)
TS2 <- genTSExample(InKTSEnv = FALSE)

## Their times are identical
timeComp <- are2TsTimeCompatible (TS1, TS2)
timeComp

## We modify the time of TS2. Now their starts are different, although they still have same sampling period and length
TS2$time <- TS2$time + 1
timeComp <- are2TsTimeCompatible (TS1, TS2)
timeComp
```
areTsGapTimeCompatible

areTsGapTimeCompatible: tests time compatibility between a time series and a gap set

Description

It checks for three types of compatibility: the initial dates are the same, the sampling period is the same and the length is the same. It is used internally to check whether the inputs chosen by the user are appropriated.

Usage

areTsGapTimeCompatible(TS1, GAP1)

Arguments

TS1  A time series
GAP1  A gap set

Value

A logical vector, which is c(TRUE, TRUE, TRUE) when all the time compatibility conditions are met.

Author(s)

Marina Saez Andreu

Examples

## Generate a time series
TS <- genTSExample(InKTSEnv = FALSE)
## Generate a gap set consisting of four gaps of 3 NAs each
GS <- genGapExample(TS, 3, 4, InKTSEnv = FALSE)

## They are obviously compatible since we use TS to create GS
timeComp <- areTsGapTimeCompatible(TS, GS)
timeComp

## We modify the time of TS. The initial dates are not compatible anymore, although the sampling period and the length remain untouched.
TS$time <- TS$time + 1
timeComp <- areTsGapTimeCompatible(TS, GS)
timeComp
**areTsRmTimeCompatible**  
*areTsRmTimeCompatible: test time compatibility between a time series and a recurrence matrix*

**Description**

It checks for three types of compatibility: the initial dates are the same, the sampling period is the same and the length is the same. It is used internally to check whether the inputs chosen by the user are appropriated.

**Usage**

```r
areTsRmTimeCompatible(TS1, RM1)
```

**Arguments**

- **TS1**: A time series
- **RM1**: A recurrence matrix

**Value**

A logical vector, which is `c(TRUE, TRUE, TRUE)` when all the time compatibility conditions are met.

**Author(s)**

Marina Saez Andreu

---

**arimaKalman**  
*arimaKalman: ARIMA + Kalman smoother*

**Description**

This function fits an ARIMA model to a univariate time series and uses the model to feed a Kalman smoother, which is used to fill missing values in the time series. It is used through the ARIMA button in the Filling Menu.

**Details**

This function input panel contains a button called "Estimate ARIMA parameters"; this button calls the function `forecast::auto.arima` to provide an automatic estimation of the ARIMA parameters. These parameters can also be directly introduced by the user. Optionally, the filling can be applied only to a set of gaps in the time series. If the time series does not contain any NAs, KarsTS will fit the ARIMA model anyway and return the parameters.
assignMultiple

Value
The filled time series appears in the environment susEnv

Author(s)
Marina Saez Andreu

arimaXKalman

Description
This function fits an ARIMAX model to a time series and uses the model to feed a Kalman smoother, which is used to fill missing values in the time series. It is used through the ARIMAX button in the Filling Menu.

Details
The only difference between this function and arimaKalman is that arimaXKalman allows the introduction of regressor variables in the model. See arimaKalman for more details.

Value
The filled time series appears in the environment susEnv

Author(s)
Marina Saez Andreu

assignMultiple

Description
This function applies the function assign multiple times

Usage
assignMultiple(namesVector, valuesList, envir = KTSEnv)

Arguments
- namesVector: A vector containing the names to be assigned
- valuesList: The values to which the names will be assigned
- envir: The environment
buttons1

**Author(s)**
Marina Saez Andreu

**Description**
This function creates the buttons corresponding to the Time Series Menu

**Author(s)**
Marina Saez Andreu

buttons2

**Author(s)**
Marina Saez Andreu

**Description**
This function creates the buttons corresponding to the Gap Sets Menu

**Author(s)**
Marina Saez Andreu

buttons3

**Author(s)**
Marina Saez Andreu

**Description**
This function creates the buttons corresponding to the Analysis Menu
buttons4: create the buttons corresponding to the Plots Menu

Description

This function creates the buttons corresponding to the Plots Menu

Author(s)

Marina Saez Andreu

buttons5: create the buttons corresponding to the Filling Menu

Description

This function creates the buttons corresponding to the Filling Menu

Author(s)

Marina Saez Andreu

checkIfAny: check if there are any data sets in the environment

Description

This function checks whether there are any data sets (time series, gap sets or recurrence matrices) in the environment (KTSEnv). If it is the case, it launches a function.

Usage

checkIfAny(action = NULL, envirName = KTSEnv)

Arguments

action The function to be launched
envirName The environment where the data sets are to be found

Author(s)

Marina Saez Andreu
checkIfAnyGapOrTs

checkIfAny: check if there are any time series or gap sets in the environment

Description
This functions checks whether there are any time series or gap sets in the environment (KTSEnv). If it is the case, it launches a function.

Usage
checkIfAnyGapOrTs(action = NULL, envirName = KTSEnv)

Arguments
- action: The function to be launched
- envirName: The environment where the data sets are to be found

Author(s)
Marina Saez Andreu

checkIfAnyGapTs

checkIfAny: check if there are any time series and gap sets in the environment

Description
This functions checks whether there are any time series and gap sets in the environment (KTSEnv). If it is the case, it launches a function.

Usage
checkIfAnyGapTs(action = NULL, envirName = KTSEnv)

Arguments
- action: The function to be launched
- envirName: The environment where the data sets are to be found

Author(s)
Marina Saez Andreu
checkIfAnyRm

**Description**

This function checks whether there are any recurrence matrices in the environment (KTSEnv). If it is the case, it launches a function.

**Usage**

```r
checkIfAnyRm(action = NULL, envirName = KTSEnv)
```

**Arguments**

- `action`  
  The function to be launched
- `envirName`  
  The environment where the data sets are to be found

**Author(s)**

Marina Saez Andreu

---

checkIfAnyRmTs

**Description**

This function checks whether there are any recurrence matrices and time series in the environment (KTSEnv). If it is the case, it launches a function.

**Usage**

```r
checkIfAnyRmTs(action = NULL, envirName = KTSEnv)
```

**Arguments**

- `action`  
  The function to be launched
- `envirName`  
  The environment where the data sets are to be found

**Author(s)**

Marina Saez Andreu
checkIfAnyTs

checkIfAnyTs: check if there is some time series in the environment

Description

This function checks whether there are any time series in the environment (KTSEnv). If it is the case, it launches a function.

Usage

checkIfAnyTs(action = NULL, envirName = KTSEnv)

Arguments

action The function to be launched
envirName The environment where the data sets are to be found

Author(s)

Marina Saez Andreu

cleanEnvir

cleanEnvir: removes a list of variables from KTSEnv

Description

In order to communicate the functions in an interface, it is often unavoidable to create global variables. This function cleans the global variables that are mere intermediate results of the procedures. It is used internally.

Usage

cleanEnvir(envir = KTSEnv)

Arguments

envir The environment where the intermediate objects are to be removed. It defaults to KTSEnv.

Author(s)

Marina Saez Andreu
**compareVecVec**

*compareVecVec: compare the elements of two vectors*

**Description**

Checks whether there are any shared elements between two vectors. It is used internally.

**Usage**

```
compareVecVec(VA, VB)
```

**Arguments**

- **VA**: One vector
- **VB**: Another vector

**Value**

A logical matrix showing the coincidences

**Author(s)**

Marina Saez Andreu

**Examples**

```
compareVecVec(1:10, 7:12)
V1 <- c("Apples", "Strawberries", "Watermelon")
V2 <- c("Ananas", "Apples", "Strawberries", "Coconut")
compareVecVec(V1, V2)
```

---

**composeKTS**

*composeKTS: perform operations on time series*

**Description**

This function allows to perform a number of operations on the values of the time series. The operations currently available are: sum, multiplication, opposite, reciprocal and natural logarithm. It is used through the Operations button in the Analysis Menu.
createChb

Details

Opposite, inverse and logarithm can be applied simultaneously to N time series. The output consists of N output time series the names of which are assigned by default (original names plus Opp, Recip or Ln). On the contrary, the output of sum and multiplication is a single time series and the user must choose its name.

Value

The output time series in the environment KTSEnv

Author(s)

Marina Saez Andreu

createChb: creates a check box on the main screen

Description

This function creates a check box on the main screen

Usage

createChb(labTitle = NULL, variableName = NULL, defaultVal = "0")

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>labTitle</td>
<td>A title for the check box</td>
</tr>
<tr>
<td>variableName</td>
<td>A name for the variable associated to the check box</td>
</tr>
<tr>
<td>defaultVal</td>
<td>Default value (defaults to not selected)</td>
</tr>
</tbody>
</table>

Value

A variable that will be passed to the corresponding OnOk function

Author(s)

Marina Saez Andreu
createChbChb

**createChbChb: two column check box**

**Description**

createChbChb: creates a two column check box on the main screen

**Usage**

```r
createChbChb(ind, elements, prefix1 = NULL, prefix2 = NULL, envir = KTSEnv)
```

**Arguments**

- `ind` Number of rows of the check box
- `elements` Row labels
- `prefix1` A prefix to assign a name to the output variables (column 1)
- `prefix2` A prefix to assign a name to the output variables (column 2)
- `envir` Environment to which the output variables will be assigned

**Value**

An output variable for each box. They are be passed to the corresponding onOk function

**Author(s)**

Marina Saez Andreu

createChbEntry

**createChbEntry: creates a check box with associated text entries**

**Description**

This function creates an element in the input panel consisting of two columns: a check box and the corresponding text entries. It is used internally

**Usage**

```r
createChbEntry(ind, elements, prefix = "scbValue", envir = KTSEnv, dCh = "0", dEn = "")
```

**Arguments**

- `ind` Number of rows
- `elements` Row labels
- `prefix` A prefix to assign names to the output variables
- `envir` Environment (defaults to the environment KTSEnv)
- `dCh` Default value for the checkbox
- `dEn` Default value for the text entry
**Value**

Variables that will be passed to the corresponding OnOk function

**Author(s)**

Marina Saez Andreu

---

**createCrossRM**

**createCrossRM**: creates a cross recurrence matrix

**Description**

This function creates a cross recurrence matrix. It is used through the Cross Recurrence Matrix button in the Analysis Menu

**Details**

The inputs are two time series representing similar variables (for example, two air temperature time series). They must have the same sampling period, although they can have different lengths. The matrix can be visualized by means of the Plot Cross Recurrence Matrix button in the Plots Menu

**Value**

A cross recurrence matrix in the environment susEnv. Cross recurrence matrices are not symmetrical, therefore the entire matrix is stored (not only a triangle).

**Author(s)**

Marina Saez Andreu

**References**

createCrossRMPPlot: creates a cross recurrence plot

Description
This function creates a cross recurrence plot from a previously created cross recurrence matrix. It is used through the Plot Cross Recurrence Plot in the Plots Menu

Author(s)
Marina Saez Andreu

References

createDistMatrix: creates distance matrix

Description
This function calculates and plots a distance matrix via interface.

Details
A distance matrix is an unthresholded recurrence matrix. The norm can be Euclidean or Infinity.

Author(s)
Marina Saez Andreu

createEachRb: creates a radiobutton on the main screen

Description
This function creates a radiobutton. It is used internally

Usage
createEachRb(labTitle = NULL, variable = NULL, panel = KTSEnv$subPanR4C1)
createEntry

Arguments

labTitle  A title for the check box
variable The variable associated to the radiobutton
panel The panel where the radiobutton is to be placed (defaults to the input panel in the main screen)

Value

A variable that will be passed to the corresponding OnOk function

Author(s)

Marina Saez Andreu

createEntry  createChb: creates a text entry on the main screen

Description

This function creates a text entry on the input panel (on the main screen)

Usage

createEntry(labTitle, textVariableName, defaultVal = "", font = KTSEnv$KTSFonts$T1)

Arguments

labTitle The text entry title
textVariableName The name of the variable associated to the text entry
defaultVal Default value (empty entry)
font Title font

Value

A variable that will be passed to the corresponding OnOk function

Author(s)

Marina Saez Andreu
createFAN

**createFAN: creates a fixed amount of neighbors (FAN) matrix.**

**Description**

This function creates a fixed amount of neighbours (FAN) recurrence matrix. It is used through the FAN Recurrence Matrix button in the Analysis Menu.

**Details**

The FAN recurrence matrix can be created from an existing matrix; in this case, only the closest neighbors are kept. The FAN recurrence matrix can be created directly from a time series. In this case, the embedding dimension, delay, tolerance, Theiler’s window and fixed amount of neighbours must be provided via interface. The matrix can be visualized by means of the FAN Recurrence Plot button in the Plots Menu. Note that FAN recurrence matrices are not symmetric.

**Author(s)**

Marina Saez Andreu

**References**


createFANplot

**createFANplot: creates a plot from a fixed amount of neighbors (FAN) recurrence matrix**

**Description**

This function creates a recurrence plot from a previously created fixed amount of neighbors (FAN) recurrence matrix.

**Author(s)**

Marina Saez Andreu

**References**

**createGapChb**

**createGapChb**: creates a check box of gap sets

**Description**
This function creates a check box showing all the gap sets available. It is used internally.

**Usage**
```r
createGapChb(labTitle = "Gap sets", envir = KTSEnv)
```

**Arguments**
- `labTitle`: Title for the check box
- `envir`: Environment

**Value**
Variables to be passed to the corresponding OnOk function

**Author(s)**
Marina Saez Andreu

---

**createGapRb**

**createGapRb**: creates a radio button of gap sets

**Description**
This function creates a radiobutton showing all the gap sets available.

**Usage**
```r
createGapRb(labTitle = "Gap sets", envir = KTSEnv)
```

**Arguments**
- `labTitle`: Title for the check box
- `envir`: Environment

**Value**
A variable to be passed to the corresponding OnOk function

**Author(s)**
Marina Saez Andreu
createJointRM

createJointRM: creates a joint recurrence matrix

**Description**

This function creates a joint recurrence matrix. It is used through the Joint Recurrence Matrix button in the Analysis Menu.

**Details**

The inputs are two time series. They must have the same sampling period. The shorter one will determine the size of the recurrence matrix. The matrix can be visualized by means of the Plot Joint Recurrence Matrix button in the Plots Menu.

**Value**

A joint recurrence matrix in the environment susEnv

**Author(s)**

Marina Saez Andreu

**References**


createJointRMPPlot

createJointRMPPlot: creates a cross recurrence plot

**Description**

This function creates a joint recurrence plot from a previously created joint recurrence matrix. It is used through the Plot Joint Recurrence Plot in the Plots Menu.

**Author(s)**

Marina Saez Andreu

**References**

createNote

createNote is a function that creates a note on the input panel. This function is typically used when you want to provide a note for the user to read or to remember something important.

### Usage

```r
createNote(labTitle = NULL, pady = c(10, 10))
```

### Arguments

- **labTitle**: Text that will be displayed on the note.
- **pady**: Upper and lower margins for the note.

### Description

This function creates a note on the input panel. The note can be used to provide additional information or guidance to the user.

### Author(s)

Marina Saez Andreu

---

createOK

createOK is a function that creates the OK or NEXT button on the input panel. When the button is pressed, the corresponding function is launched.

### Usage

```r
createOK(labTitle = "NEXT", action = NULL, width = 7, panel = KTSEnv$subPanR4C1)
```

### Arguments

- **labTitle**: Usually, its values are "OK" or "NEXT"
- **action**: Function that will be launched when the button is pressed.
- **width**: Button width
- **panel**: Panel where the button will be placed

### Description

This function creates the OK or NEXT button on the input panel. When the button is pressed, the corresponding function is launched.

### Author(s)

Marina Saez Andreu
createRandGaps

**createRandGaps**: create random gaps in a time series

**Description**
This function creates N gaps of length M, randomly distributed through a time series. The inputs are the time series, the number of gaps and their length. It is used through the Random Gaps button in the Gap Sets menu.

**Details**
The new gaps will not overlay previously existing gaps.

**Value**
The function creates a gap set and applies it to a copy of the input time series. Both the gap set and the new time series appear in the environment susEnv.

**Author(s)**
Marina Saez Andreu

createRandName

**createRandName**: creates a random name

**Description**
This function creates a random name consisting of a prefix and a random number from 10000 to 99999. It is used internally.

**Usage**
createRandName(prefix = "panel")

**Arguments**

- **prefix**
The prefix to which the random number will be added

**Author(s)**
Marina Saez Andreu

**Examples**
createRandName("Rodrigo")
createRandName("Rodrigo")
createRandName("Diaz")
createRb

createRb: creates a radio button

Description

This function creates a radio button

Usage

createRb(variable = NULL, dataVector = NULL, panel = KTSEnv$subPanR4C1)

Arguments

- **variable**: The variable to which the radiobutton is linked
- **dataVector**: The names of the buttons
- **panel**: The panel where to place the radiobutton (defaults to the main screen input panel)

Author(s)

Marina Saez Andreu

createRmChb

createRmChb: creates a check box of recurrence matrices

Description

This function creates a check box listing the recurrence matrices that exist in the environment KTSEnv.

Usage

createRmChb(labTitle = "Recurrence matrices", envir = KTSEnv)

Arguments

- **labTitle**: Check box title
- **envir**: Environment (defaults to KTSEnv)

Value

Variables that will be read in the corresponding onOk function

Author(s)

Marina Saez Andreu
createRmRb

createRmRb: creates a radiobutton of recurrence matrices

Description
This function creates a radiobutton listing the recurrence matrices that exist in the environment KTSEnv.

Usage
createRmRb(labTitle = "Recurrence matrices", envir = KTSEnv)

Arguments
- labTitle: Radiobutton title
- envir: Environment (defaults to KTSEnv)

Value
A variable that will be read in the corresponding onOk function

Author(s)
Marina Saez Andreu

createSimpleRM

createSimpleRM: creates a recurrence matrix

Description
This function creates a recurrence matrix. It is used through the Recurrence Matrix button in the Analysis Menu.

Details
The inputs are a time series, the delay, the embedding dimension and a name for the output matrix. The matrix can be visualized by means of the Recurrence Plot button in the Plots Menu Note that there are specific buttons to create and plot cross and joint recurrence matrices.

Value
The recurrence points are represented in a two-column data frame by their positions in the recurrence matrix. Only the upper triangle is stored.

Author(s)
Marina Saez Andreu
createSimpleRMPlot

References

createSimpleRMPlot: creates a recurrence plot

Description
This function creates a recurrence plot from a previously created recurrence matrix. Note that there are specific buttons to create and plot cross and joint recurrence matrices. It is used through the Plot Recurrence Plot in the Plots Menu.

Author(s)
Marina Saez Andreu

createSpecGaps

createSpecGaps: creates a specific gap in a time series

Description
This function creates a gap between two dates chosen by the user. It is used through the Specific Gaps button in the Gap Sets menu.

Value
The function creates a gap set and applies it to a copy of the input time series. Both the gap set and the new time series appear in the environment susEnv.

Author(s)
Marina Saez Andreu
createSubPanR4C1: creates the input panel on the main screen

Description

This function creates a blank input panel on the main screen. It is used whenever is necessary to erase the elements of the input panel (title, check boxes, radiobutton etc.) and write new ones. It is used internally.

Author(s)

Marina Saez Andreu

createTITLE: create a title on the input panel

Description

This function creates (on the input panel) a label that works as a first level title.

Usage

createTITLE(labTitle = "TITLE", panel = KTSEnv$subPanR4C1)

Arguments

labTitle: Text of the title
panel: Panel where the title is to be written (defaults to the input panel on the main screen)

Author(s)

Marina Saez Andreu
createTitle

cREATE TITLE: creates a title in the input panel

description

This function creates (on the input panel) a label that works as a second level title. It is used internally

Usage

createTitle(labTitle = "Title")

Arguments

labTitle The text of the title

Author(s)

Marina Saez Andreu

createTsChb

cREATE TS CHB: creates a check box of time series

description

This function creates a check box listing the time series that exist in the environment KTSEnv.

Usage

createTsChb(labTitle = "Time series", envir = KTSEnv)

Arguments

labTitle Check box title
envir Environment (defaults to KTSEnv)

Value

Variables that will be read in the corresponding onOk function

Author(s)

Marina Saez Andreu
createTsRb  
_createTsRb: creates a radiobutton of time series_

**Description**

This function creates a radiobutton listing the time series that exist in the environment KTSEnv.

**Usage**

```r
createTsRb(labTitle = "Time series", variableName = "selTsP", envir = KTSEnv)
```

**Arguments**

- `labTitle`   Radiobutton title
- `variableName` Name of the variable to be assigned to the radiobutton
- `envir`     Environment (defaults to KTSEnv)

**Value**

A variable that will be read in the corresponding onOk function

**Author(s)**

Marina Saez Andreu

---

cumuKTS  
_cumuKTS: creates a cumulated time series_

**Description**

This function creates a cumulated time series, that is, each point is the sum of all the previous values in the time series. It is particularly useful when the input time series has zero mean. It is used through the button Cumulative sum in the Time Series Menu.

**Value**

The cumulated time series

**Author(s)**

Marina Saez Andreu
destroyMainScreen

A function to destroy KarsTS main screen

Description
A function to destroy KarsTS main screen. It is normally destroyed via interface.

Usage
destroyMainScreen()

Author(s)
Marina Saez Andreu

destroyWelcome

destroyWelcome: destroys KarsTS welcome screen

Description
This function destroys KarsTS welcome screen. It is used through the button Start in the welcome screen.

Author(s)
Marina Saez Andreu

determinismKTS
determinismKTS: estimate determinism

Description
This function is used to study the determinism of a system, based on the number and length of the diagonal lines that contains the recurrence matrix that represents the system. It is used through the button Determinism in the Analysis menu.

Value
The following outputs are written on KarsTS output window: recurrence rate, determinism, ratio and summary of the lengths of the diagonal lines. Besides, a new window containing a histogram of diagonal lines pops up.

Author(s)
Marina Saez Andreu
References


**diffKTS**

*diffKTS: calculate differences of a time series*

**Description**

This function creates a new time series by differencing the input time series. It is used through the button differences in the Time Series Menu.

**Details**

The inputs are: time series, lag and center yes/no. In this context, center the time series means that the differenced time series will be interpolated, so that the output time series has same length and dates as the original time series.

**Value**

The differenced time series

**Author(s)**

Marina Saez Andreu

**E1dAndE2d**

*E1dAndE2d: invariants E1(d) and E2(d)*

**Description**

This function uses the function nonlinearTseries::estimateEmbeddingDim to plot the invariants E1(d) and E2(d) (Cao, 1997). E1(d) helps finding the embedding dimension of a scalar time series and E2(d) can distinguish stochastic from deterministic signals. It is used through the button E1d & E2d in the Plots Menu.

**Author(s)**

Marina Saez Andreu

**References**

**embedData**

**Description**

This function embeds a time series and adds NAs so that the embedded data have the same length as the original time series. It is used internally.

**Usage**

```r
embedData(TSData, embDim, embDelay)
```

**Arguments**

- `TSData`: Values of the time series
- `embDim`: Embedding dimension
- `embDelay`: Delay

**Value**

A matrix of embedded data, with as many NAs as necessary so that it has the same length as the original time series.

**Author(s)**

Marina Saez Andreu

**endingLines**

**Description**

This function adds three lines of asterisks on the output panel. It is used to separate outputs of different functions. It is used internally.

**Usage**

```r
endingLines()
```

**Author(s)**

Marina Saez Andreu
**exportall**  
*exportall: exports all types of data sets*

**Description**

This function is used to export time series, gap sets or recurrence matrices to csv or txt files. It is used through the button Export in the Time Series or Gap Sets menus.

**Details**

All types of data sets can be exported both from the Time Series menu or from the Gap Sets menu. It is possible to export more than one time series to the same file; however, each gap set and recurrence matrix must be stored separately. The files are created in the working directory.

**Author(s)**

Marina Saez Andreu

---

**fillWithTwins**  
*fillWithTwins: fills missing values using twin points*

**Description**

This function finds twin points in a recurrence matrix and uses them to fill adjacent missing values. Many points do not have twins, therefore it is unlikely that all the missing values are filled. It is used through the button twins in the Filling Menu.

**Details**

The inputs to this function are: the time series to fill (TSF), a recurrence matrix that represents the system (RM), the maximum distance (MD) and, optionally, the gap set to fill. A missing value is replaced by the median of its twin points. Case A: TSF was used to build RM. In this case, RM lacks information at the points to fill; therefore, it is necessary to find the twins of the adjacent points, that is, MD must be greater than 0. Case B: the system was reconstructed using other time series, that is, RM was calculated without TSF. In this case, it is possible to set MD to 0, although it can be greater than 0, as well.

**Value**

The filled time series in susEnv and a summary of the procedure on the output window.

**Author(s)**

Marina Saez Andreu
findDateFormat  

**Description**

This function identifies the format of a set of dates from a list of allowed formats.

**Usage**

```r
findDateFormat(X, tz = NULL)
```

**Arguments**

- `X` The date (given as a character string)
- `tz` The time zone

**Details**

The allowed formats are: 

- "%m/%d/%Y %H:%M"
- "%Y/%m/%d %H:%M"
- "%m-%d-%Y %H:%M"

**Value**

It returns the date format, ready to use in the function `strptime`. If the dates have different formats, the function returns "variousFormats". If the format is not one of the allowed ones, it returns "notAllowedformat".

**Author(s)**

Marina Saez Andreu

---

findTwins  

**Description**

This function finds twin points (that is, identical columns) in a recurrence matrix. It is used internally.

**Usage**

```r
findTwins(recMat, pointsToFind = NULL)
```

**Arguments**

- `recMat` The recurrence matrix
- `pointsToFind` The points (columns) whose twins are to be found (defaults to all)
Details

The function classifies the columns in families of twins and assigns an integer number to each family. The numbers are mere codes to identify the families and they do not have any meaning themselves.

Value

A vector of integers. Each number represents a family of twins. Positions corresponding to columns with no twins are assigned NA. When the search of twins is limited to a subset of points, the uninteresting points are assigned NA also.

Author(s)

Marina Saez Andreu

References


Examples

```r
# Generate time series and recurrence matrix
res <- genRmExample(name = "SRM", InKTSEnv = FALSE, plotRM = FALSE)
TS <- res$TS
SRM <- res$newSimpRM

# Find twins
SRMTwins <- findTwins(SRM)

# Interpretation example
aFamily <- SRMTwins[which(is.finite(SRMTwins))][1]
twinsInFamily <- which(SRMTwins == aFamily)
TS[twinsInFamily,]

# The values in TS are very similar
# The columns in the recurrence matrix are identical,
# although this cannot be observed directly
# because of the way KarsTS stores recurrence matrices
```

Description

This function plots embedding dimension vs false nearest neighbors using tseriesChaos::false.nearest and tseriesChaos::plot.false.nearest. It is used to find the minimum embedding dimension. It is used through the button FNN in the Plots Menu.
functToExport

Author(s)

Marina Saez Andreu

functToExport: functions to export

Description

This function exports the function KarsTS::KarsTS. It is used internally.

Usage

functToExport()

Details

Other objects documented here:

makeGlobal is a vector containing the variables to declare global. It is used internally.

KTSEnv is the environment where the loaded data sets are and it can be accessed directly from the R console. We recommend to use KarsTS to manipulate the data sets; however, the user can handle them directly from R in order to apply functionalities not included in KarsTS. If that is the case, be careful to produce data sets with the right format; otherwise, KarsTS will not recognize them. See the User's Guide for more information.

Author(s)

Marina Saez Andreu

gamKTS

A function to fill values by means of a generalized additive model

Description

This function is used to fill gaps using a generalized additive model (gam) with tensor smoothing. It is used through the Multivariate Splines in the Filling Menu. Although the gam model is not necessarily a multivariate spline, the button has this name because we think is more familiar to the users.

Details

The user selects the time series to fill and a set of predictor time series. The function fits a generalized linear model a gap and uses the model to fill the gap. This is done for every gap in the gap set. It is used through the Multivariate Splines in the Filling Menu. The user can choose a fixed d.f. regression spline or a penalized regression spline. It is also possible to choose any of the smoothing bases allowed by the function mgcv::te.
Value

The filled time series in the environment KTSEnv and a summary of the results on the output window.

Author(s)

Marina Saez Andreu

gapCheckedTF

gapCheckedTF: identifies checked gap sets

Description

This function identifies which gap sets have been checked from a check box

Usage

gapCheckedTF(prefix = "gcbValue", envir = KTSEnv)

Arguments

prefix A prefix to reconstruct the names that were assigned to the variables in the check box.

envir The environment

Value

A logical vector

Author(s)

Marina Saez Andreu

gapDetect

gapDetect: identifies the gap sets currently loaded

Description

This function identifies the gap sets currently loaded in the environment KTSEnv

Usage

gapDetect()
Value

A vector containing the names of the gap sets

Author(s)

Marina Saez Andreu

gapForSelMethod

gapForSelMethod: gap set to fill

Description

This function finds which gap the user selected to apply a filling method on it. If none was selected, it creates a gap set containing all the gap sets in the time series.

Usage

gapForSelMethod(selTsName, selTs, envir = KTSEnv)

Arguments

  selTsName       The time series name
  selTs          The time series
  envir          The environment

Value

A list containing the selected gap set name and the gap set. If the user did not select any gap set or there is no gap set in the environment, the function creates a gap set containing all the gaps in the time series. It is name is All plus the time series name.

Author(s)

Marina Saez Andreu
genGapExample  

A function to generate a gap set example

Description
A function to generate a gap set example. It can be assigned to the KTSEnv environment.

Usage
```r
genGapExample(timSer, lGaps, nGaps, name = "GS", InKTSEnv = TRUE)
```

Arguments
- `timSer`: The time series where the gaps are generated
- `lGaps`: The length of the gaps
- `nGaps`: The number of gaps
- `name`: A name for the gap set. It defaults to GS. When InKTSEnv is FALSE, the name is not necessary.
- `InKTSEnv`: Assign to the environment KTSEnv (TRUE) or not (FALSE)

Value
The gap set

Author(s)
Marina Saez Andreu

genRmExample  

A function to generate a recurrence matrix example

Description
A function to generate a recurrence matrix example. It can be assigned to the KTSEnv environment.

Usage
```r
genRmExample(name = "SRM", InKTSEnv = TRUE, plotRM = FALSE)
```

Arguments
- `name`: A name for the recurrence matrix. When the local environment is used instead of KTSEnv, the name is still necessary for the recurrence plot title.
- `InKTSEnv`: Assign the output to the KTSEnv (TRUE) or to the local environment (FALSE)
- `plotRM`: Plot the recurrence matrix (TRUE) or not (FALSE)
Details

The recurrence matrix is calculated on a sinusoidal time series composed with a random walk. The embedding dimension is 2, the delay is 290 and the threshold is 20.

Value

The recurrence matrix and the time series

Author(s)

Marina Saez Andreu

Examples

```r
res <- genRmExample(name = "RMExample", InKTSEnv = FALSE, plotRM = FALSE)
res$newSimpRM
res$TS
```

Description

This function generates an example time series. The dates range from 2015 to 2016 and the sampling period is half an hour. The time series can be white noise or a random walk. It can be assigned to the KTSEnv environment.

Usage

```r
genTSExample(stationary = TRUE, name = "TS", InKTSEnv = TRUE)
```

Arguments

- **stationary** Generate white noise (TRUE) or a random walk (FALSE)
- **name** A name for the time series. It defaults to TS. When InKTSEnv is FALSE, the name is not necessary.
- **InKTSEnv** Assign to the environment KTSEnv (TRUE) or not (FALSE)

Value

The time series

Author(s)

Marina Saez Andreu
**getClassEnvir**

*getClassEnvir: get objects of a class from an environment*

**Description**

This function identifies objects of a class in an environment

**Usage**

```
getClassEnvir(classGet = "list", envir = KTSEnv)
```

**Arguments**

- **classGet**: The class
- **envir**: The environment

**Value**

A vector with the names of the objects

**Author(s)**

Marina Saez Andreu

---

**getCoordsKTS**

*getCoordsKTS: get coordinates from a plot*

**Description**

This function allows to manually select points from a plot and get their coordinates. It is used through the button Get coordinates in the Plots menu.

**Details**

This function has two steps. In the first place, it is necessary to select a time series and plot it. The plot appears on a pop-up window. Then, the user clicks on one or more points and they turn green. When the user presses the Write results button, a list of coordinates (time and value) appears on the output window.

**Author(s)**

Marina Saez Andreu
getCRP

getCRP: gets the correlation probability of recurrence

Description

This function calculates the correlation probability of recurrence of two systems, given by their recurrence matrices.

Usage

getCRP(prob1, prob2, xLims, doPlot = FALSE, main = "plotTitle")

Arguments

prob1 Probability of recurrence of the first matrix
prob2 Probability of recurrence of the second matrix
xLims X range where the correlation probability will be calculated. The diagonals very close or very far from the main diagonal are usually discarded for estimating the CPR.
doPlot Get a plot (TRUE) or not (FALSE)
main Title for the plot

Value

Correlation probability of recurrence (CPR)

Author(s)

Marina Saez Andreu

References

**getDelayCharTimes**

*getDelayCharTimes*: transforms dates from character to numeric.

**Description**

This function transforms a set of dates (given as character) to numeric after testing that their format is allowed by KarsTS.

**Usage**

```r
getDelayCharTimes(initialTimes, tz = NULL)
```

**Arguments**

- `initialTimes`: The dates
- `tz`: The time zone

**Value**

A vector with the times in numeric form

**Author(s)**

Marina Saez Andreu

---

**getFANRM2**

*getFANRM2*: creates a fixed amount of neighbours (FAN) matrix via console.

**Description**

This function creates a fixed amount of neighbours (FAN) matrix via console from a time series. To create a FAN matrix via interface, see the function createFAN.

**Usage**

```r
getFANRM2(selTs, embedDim, lagDelay, threshold, theilerWin, fan)
```

**Arguments**

- `selTs`: A univariate time series in KarsTS format.
- `embedDim`: The embedding dimension.
- `lagDelay`: The delay (in lags).
- `threshold`: The threshold.
- `theilerWin`: The Theiler’s window (in lags).
- `fan`: The fixed amount of neighbours.
getGapsAfterFill

Details
In case you do not want to embed your time series, set embedDim to 1 and lagDelay to 0. If you do not want to apply a Theiler’s window, set theilerWin to 0. The threshold will determine in the first place the number of neighbors of each point; then, the fixed amount of neighbours will determine how many of those are kept. For example, if a point A has 10 neighbors (according to the threshold) and the fan is 5, the point A will have 5 neighbors in the output matrix. However, if a point B has only 4 neighbors (according to the threshold), the number of neighbors of B in the output matrix will be 4. If you want a matrix with a truly fixed amount of neighbors, give the threshold a value high enough to render it useless.

Value
A list containing the X and Y positions of the recurrent points in the FAN recurrence matrix.

Author(s)
Marina Saez Andreu

References

getGapsAfterFill: get the remaining gaps

Description
This function identifies which gaps remain in a time series after applying a filling method. It is used internally.

Usage
getGapsAfterFill(filledTS, selGap, envir = KTSEnv)

Arguments
filledTS The filled time series
selGap The gap set that was meant to be filled
envir The environment

Value
It returns a list containing two data frames: one of them lists the remaining gaps and the other, the filled gaps.

Author(s)
Marina Saez Andreu
getMaxNegSlope: get the maximum negative slope

Description

This function returns the greater negative difference in a time series (greater in absolute value). It is used internally.

Usage

ggetMaxNegSlope(timSerVals)

Arguments

timSerVals  The time series (only the values, not the times)

Author(s)

Marina Saez Andreu

getMaxPosSlope: get the maximum negative slope

Description

This function returns the greater positive difference in a time series. It is used internally.

Usage

ggetMaxPosSlope(timSerVals)

Arguments

timSerVals  The time series (only the values, not the times)

Author(s)

Marina Saez Andreu
getNAsGaps

Description
This function identifies the gaps existing in a time series and returns a table. It is used internally.

Usage
getNAsGaps(y)

Arguments
y
The time series (only values, not times)

Value
A matrix listing the initial and final indices of each gap

Author(s)
Marina Saez Andreu

getNewGapsInd

Description
This function creates nGaps gaps of length lGaps in the time series timSer. It is used internally. The gaps do not overlay amongst them; they do not overlay pre-existing gaps either.

Usage
getNewGapsInd(timSer, lGaps, nGaps)

Arguments
timSer
The time series where the gaps will be created.
lGaps
The length of the gaps
nGaps
The number of gaps.

Value
A vector containing the indices of the gaps
getOtherErrEstim

Author(s)

Marina Saez Andreu

Examples

```r
# Create time series
TS <- genTSExample(InKTSEnv = FALSE)[1:50,]

# Create 3 gaps of 7 NAs each
gInd <- getNewGapsInd(TS, 7, 3)

# Create time series duplicate and apply the gaps
TS1 <- TS
TS1$value[gInd] <- NA

# Compare
cbind(TS, TS1$value)
```

Description

This function gets the relative root mean square error, the mean absolute error and bias error.

Usage

```r
getOtherErrEstim(observed, predicted)
```

Arguments

- `observed`: Observed values
- `predicted`: Predicted values

Author(s)

Marina Saez Andreu
getProTaos: calculate the probability of recurrence

Description

This function calculates the probability of recurrence of a system, given by a recurrence matrix in KarsTS format. The RP is calculated for each diagonal (upper triangle).

Usage

getProTaos(RecMat, xlim = NULL, main = NULL, doPlot = TRUE)

Arguments

RecMat      A recurrence matrix in KarsTS format
xlim        X range of the plot.
main        Plot title
doPlot      Get the plot or not

Value

A list containing

Tao          The diagonals where the RP was calculated
Prob         The RP for each diagonal

Author(s)

Marina Saez Andreu

References

getRecurrencePoints

A function to get recurrent points in a time series

Description

This function is the core of the function createSimpleRm. It finds the recurrence points in a time series, possibly embedded.

Usage

getRecurrencePoints(timSer, embedDim, lagDelay, threshold)

Arguments

- timSer: The time series
- embedDim: The embedding dimension
- lagDelay: The delay for the embedding (in lags)
- threshold: The threshold

Details

Two points in the phase space are recurrent when the distance between them (infinite norm) is less than the threshold.

In a recurrence matrix the X positions read rightwards and the Y positions read upwards. Recurrence matrices are symmetric and their diagonal (line X = Y) cannot contain not-recurrent points. For this reason, only the upper triangle is stored.

Value

A list containing the X and Y positions of the recurrent points in the recurrence matrix (upper triangle)

Author(s)

Marina Saez Andreu

Examples

```
# Generate time series
timSer <- genTSExample(stationary = FALSE, InKTSEnv = FALSE)
graphics::plot(timSer)

# Calculate recurrence matrix
RP <- getRecurrencePoints(timSer, 1, 0, threshold = 0.1)
X <- RP$recPointsX
Y <- RP$recPointsY
```
getRollStatistics

getRollStatistics: get rolling statistics It is used internally

Description

This function calculates a certain statistic in centered sliding windows along a time series. The available statistics are: minimum, first quartile, median, mean, third quartile, maximum and standard deviation. Incomplete windows can be assigned NA (option tailsTS = FALSE); alternatively, the statistic can be calculated with the available values.

Usage

getRollStatistics(selTs, selTsName, slidingWin, tailsTS = FALSE, selStatisTF)

Arguments

selTs The input time series
selTsName The input time series name
slidingWin The sliding window size
tailsTS The action regarding the tails (TRUE or FALSE)
selStatisTF A data frame containing seven columns. Each column corresponds to a statistic. When the user selects a statistic, its value is TRUE; otherwise, it is FALSE.

Value

For each selected statistic, the function creates a time series in the environment susEnv (for example, a time series of mean values).

Author(s)

Marina Saez Andreu
getSamPerTable

getSamPerTable: get sampling periods table

Description

This function divides a time series in pieces with a single sampling period and either values or missing values. Internally, this function separates true missing values from missing values that are the result of the existence of different sampling periods in the time series. Note that KarsTS time series are regular; when a time series with different time steps is loaded, NAs are added to achieve regularity. It is used internally.

Usage

getSamPerTable(timSer, sampPer)

Arguments

- timSer: The input time series
- sampPer: A vector that contains the existing sampling periods.

Value

It returns a data frame listing all the pieces.

Author(s)

Marina Saez Andreu

getSamPerTable.1Freq

getSamPerTable.1Freq: get sampling periods table (time series with one frequency)

Description

This function is a simplified version of getSamPerTable for time series that have a single sampling period. It is used internally.

Author(s)

Marina Saez Andreu
**getScreenSize**

**getScreenSize: gets the computer screen size**

**Description**

This function gets the computer screen size. It is used internally.

**Usage**

getScreenSize()

**Value**

The width and height of the screen are assigned to the environment susEnv.

**Author(s)**

Marina Saez Andreu

---

**getStatistics**

**getStatistics: calculates the statistics of a time series**

**Description**

This function calculates the minimum, first quartile, median, mean, third quartile, maximum and standard deviation of a time series. It is used internally.

**Usage**

getStatistics(selTs)

**Arguments**

| selTs | The time series |

**Value**

A vector containing the statistics values

**Author(s)**

Marina Saez Andreu
**getUniqueSampPer**  
*getUniqueSampPer: get unique sampling periods*

Description

This function get the time steps existing in a time series and sorts them according to the number of times they appear (starting by the most repeated). It returns a table with this information. It is used internally.

Usage

```r
getUniqueSampPer(timeSer)
```

Arguments

- `timeSer`: The time series

Author(s)

Marina Saez Andreu

---

**goodnessFilling**  
*goodnessFilling: estimate the goodness of the filling*

Description

This function performs a linear fit between observed and predicted values. The predicted values come from an artificial gap set that has been filled. When true missing values are imputed, it is impossible to estimate the goodness of the filling. It is used through the Check filling button in the Filling menu.

Value

A summary of the fit in the output window and different plots in pop-up windows.

Author(s)

Marina Saez Andreu
**groupDates**

**groupDates: groups dates**

### Description
This function transforms the output of the function `groupIndices` from indices to dates. It is used internally.

### Usage

```r
groupDates(rawIndices, TimSer)
```

### Arguments

- `rawIndices` The indices
- `TimSer` The time series to which the indices belong.

### Author(s)
Marina Saez Andreu

### Examples

```r
X <- sort(sample(1:50,25))
TS <- genTSExample(InKTSEnv = FALSE)
groupIndices(X)
groupDates(X, TS)
```

---

**groupIndices**

**groupIndices: group indices**

### Description
This function groups a set of sorted indices. For example: 1,2,3,50,100,101,102,103 would be grouped as follows: 1-3, 50, 100-103. It is used internally.

### Usage

```r
groupIndices(rawIndices)
```

### Arguments

- `rawIndices` The indices
**Value**

It returns a data frame. Each row corresponds to a group. The columns are: initial index, final index and group length.

**Author(s)**

Marina Saez Andreu

**Examples**

```r
X <- sort(sample(1:50, 25))
X
groupIndices(X)
```

---

**histKTS**

*histKTS: plots histogram*

**Description**

This function plots the histogram of a time series values. Its inputs are the time series and, optionally, the approximate number of bars. It is used internally through the button Histogram in the Plots menu.

**Author(s)**

Marina Saez Andreu

---

**invariantsKTS**

*invariantsKTS: invariant plots*

**Description**

This function estimates the correlation sum and dimension and the KS Entropy through the functions nonlinearTseries::corrDim and nonlinearTseries::sampleEntropy. It returns three plots: correlation sum, correlation dimension and sample entropy. The first two appear on the same pop-up window. It is used internally through the button Invariants in the Analysis menu.

**Author(s)**

Marina Saez Andreu
**isTimeAlright**

~~isTimeAlright: checks whether a set of dates fits KarsTS format~~

**Description**

This function checks whether a set of dates format is correct according to KarsTS specifications. It is used internally.

**Usage**

```
isTimeAlright(timeCharacter, tz = KTSEnv$timeZone)
```

**Arguments**

- `timeCharacter` A vector containing the dates as character.
- `tz` The time zone.

**Author(s)**

Marina Saez Andreu

---

**KarsTS**

~~KarsTS: launches KarsTS~~

**Description**

This function is used to launch KarsTS interface. This is the only function in the package that is called through the R command window. The rest are used internally or manipulated via interface. If the user closes the interface accidentally, this function can be simply run again.

**Usage**

```
KarsTS(skipWelcome = FALSE)
```

**Arguments**

- `skipWelcome` When TRUE, the interface main screen appears directly

**Author(s)**

Marina Saez Andreu
laminarityKTS  

*laminarityKTS: estimate laminarity*

**Description**

This function is used to study the laminarity of a system, based on the number and length of the vertical lines that contains the recurrence matrix that represents the system. It is used internally through the button Laminarity in the Analysis menu.

**Value**

The following outputs are written on KarsTS output window: recurrence rate, laminarity, ratio and summary of the lengths of the vertical lines. Besides, a new window containing a histogram of vertical lines pops up.

**Author(s)**

Marina Saez Andreu

**References**


---

linCorrKTS  

*linCorrKTS: linear correlation plot*

**Description**

This function plots the autocorrelation and partial autocorrelation functions or the cross correlation function, depending on the number of input time series. It is used internally through the Linear correlation button in the Plots menu.

**Author(s)**

Marina Saez Andreu
linearityKTS

Description

This function runs a number of linearity tests using the functions surrogateTest and nonlinearityTest from package nonlinearTseries. It is used internally through the Linearity button in the Analysis menu.

Author(s)

Marina Saez Andreu

loadAllTypes

Description

This function loads all types of data sets (time series, data sets and recurrence matrices). It generates two buttons: one is used to load R files and the other is used to import csv and txt files. The function automatically recognizes the type of data set, checks whether the format is right and places the data sets in the environment susEnv. It is used through the Load button in the Time Series or the Gap Sets menus.

Author(s)

Marina Saez Andreu

loadKarsTSFonts

Description

This function creates fonts for KarsTS and assign them to the environment susEnv. It is used internally.

Usage

loadKarsTSFonts()

Author(s)

Marina Saez Andreu
loessKTS

**loess: loess smoothing**

**Description**

This function performs a loess smoothing using the function stats::loess. The user has to enter the time series and the alpha parameter. Optionally, the user can enter the control parameter and some predictor time series. It is used through the Loess smooth button in the Analysis menu.

**Author(s)**

Marina Saez Andreu

mainScreen

**mainScreen: launches the main screen**

**Description**

This function launches KarsTS main screen when the welcome screen is destroyed. It is used internally.

**Usage**

mainScreen()

**Author(s)**

Marina Saez Andreu

meanValue

**meanValue: mean value filling**

**Description**

This function replaces each missing value by the mean (or median) of the values located in equivalent positions in the periods around. It is used through the button Mean Value in the Filling menu.

**Details**

The user needs to enter the following inputs: time series, period, number of surrounding periods to consider, maximum number of iterations, minimum number of observations at one side and statistic (median or mean). Note that the surrounding periods can have missing values also; they can even be completely missing. The method can be applied iteratively, although it is not advisable to use many iterations because filling missing values does not increases the real amount of information available. In case the missing values concentrate at one side, the filling might be biased. The minimum number of observations at one side is useful to discard these biased fillings.
mergeTsOrGap

Value
The filled time series in the environment susEnv and a summary of the results on the output window.

Author(s)
Marina Saez Andreu

Description
This function merges two or more time series and two or more gap sets. The time series cannot overlap and must have the same unique sampling period. The gap sets must come from time series with the same initial date and sampling period. It is used through the button Merge of the Time Series or the Gap Sets menus.

Value
The merged time series, the merged gap set or both.

Author(s)
Marina Saez Andreu

missForestKTS

Description
This function is used to fill missing values using the function missForest::missForest. The user must choose the input and output time series, the number of trees and the maximum number of iterations. It is used through the button MissForest in the Filling menu.

Value
The filled time series in the environment KTSEnv and a summary of the results on the output window.

Author(s)
Marina Saez Andreu
modeKTS

modeKTS: computes the mode

Description
This function computes the mode after removing missing values.

Usage
modeKTS(x)

Arguments
x
The data

Author(s)
Marina Saez Andreu

mutInf

mutInf: mutual information

Description
This function calculates the mutual information between two time series via interface. The core functions belong to the package infotheo.

Author(s)
Marina Saez Andreu

mutualKTS

mutualKTS: plots the mutual information

Description
This function computes the average mutual information using the function tseriesChaos::mutual and plots the result. As in the linear correlation button, the input can be one or two time series. It is used through the Mutual button in the Plots menu.

Author(s)
Marina Saez Andreu
Description

This function applies a function to the elements of a vector so that the result for each element is located in a row in a matrix. It is used internally.

Usage

myApplyVector(FUN = NULL, dataVector = NULL, out.ncols = 1, ...)

Arguments

- **FUN**: the function to apply
- **dataVector**: The vector to which elements apply the function
- **out.ncols**: Number of columns for the output matrix
- **...**: Further arguments to passed

Author(s)

Marina Saez Andreu

Examples

```r
exampleF <- function(X){c(X + 5, X*5)}
myApplyVector(FUN = exampleF, dataVector = 1:10, out.ncols = 2)
```

Description

This function is used to perform a linear model between observed and predicted values. It also prepares the outputs to be written on the output window and plots different graphics. It is the core of the goodnessFilling function.

Usage

myLinModel(observed, predicted)
Arguments

<table>
<thead>
<tr>
<th>observed</th>
<th>Observed values</th>
</tr>
</thead>
<tbody>
<tr>
<td>predicted</td>
<td>Predicted values</td>
</tr>
</tbody>
</table>

Author(s)

Marina Saez Andreu

---

myScale

myScale: scaling function

Description

This function is used to scale one or more variables with the possibility of removing the attributes from the base::scale output. It is also possible to perform a robust scaling using the median and the median absolute deviation instead of the mean and the standard deviation. It is used internally.

Usage

myScale(inputMatrix, scaleType = "Robust", outputType = c("outDef", "outList", "outNo"))

Arguments

<table>
<thead>
<tr>
<th>inputMatrix</th>
<th>Matrix containing the variables to scale.</th>
</tr>
</thead>
<tbody>
<tr>
<td>scaleType</td>
<td>Robust scaling or not</td>
</tr>
<tr>
<td>outputType</td>
<td>Three types of outputs: output as in base::scale, attributes removed but stored in a list and no attributes at all.</td>
</tr>
</tbody>
</table>

Author(s)

Marina Saez Andreu

---

naApproxKTS

naApproxKTS: filling by linear interpolation

Description

This function fills missing values in a time series by means of linear interpolation. It is used through the Linear button in the Filling menu.

Value

The filled time series in the environment KTSEnv and a summary of the results on the output window.

Author(s)

Marina Saez Andreu
NAs4Resamp

NAs4Resamp: upsamples a time series.

Description

NAs4Resamp introduces NAs in a time series for upsampling, via interface. The user needs to provide the new sampling period in seconds. This function only introduces NAs, then they need to be filled (menu Filling methods).

Value

The upsampled time series in KarsTS format.

Author(s)

Marina Saez Andreu

naSplinesKTS

naSplinesKTS: filling by spline interpolation

Description

This function fills missing values in a time series by means of splines interpolation. It is used through the Splines button in the Filling menu.

Value

The filled time series in the environment KTSEnv and a summary of the results on the output window.

Author(s)

Marina Saez Andreu

normalityKTS

normalityKTS: normality tests

Description

This function runs a number of normality tests using the functions uniNorm, mardiaTest and hzTest from package MVN. It is used through the Normality button in the Analysis menu.

Author(s)

Marina Saez Andreu
packagesToImport  
*packagesToImport: packages to import to KarsTS*

**Description**

This function imports from other packages the functions that KarsTS needs. It is used internally.

**Usage**

`packagesToImport()`

**Author(s)**

Marina Saez Andreu

pcaKTS  
*pcaKTS: principal component analysis*

**Description**

This function performs a principal component analysis using the function `prcomp`. It is used through the button PCA in the Analysis menu.

**Value**

The punctuations as new time series in the environment KTSEnv. A summary on the output window, including the loadings and the variance explained.

**Author(s)**

Marina Saez Andreu

plotTimeSeries  
*plotTimeSeries: plot time series*

**Description**

This function is used to plot one or more time series. They can be plotted using lines, points or both. The plot can appear on the plot window (main screen) or in a new window. In the second case, it is possible to select a part of the time series and plot it in another window. It is used through the Plots button in the Plots menu.

**Author(s)**

Marina Saez Andreu
**readMultEntryvalues**  
*readMultEntryvalues: verifies multiple entry values*

**Description**

This function verifies that the values of multiple text entries are valid.

**Usage**

```r
readMultEntryvalues(nElements, prefix = "entValue", type = "character")
```

**Arguments**

- `nElements`: Number of entries
- `prefix`: A prefix that has been previously used to name the variables associated to the entries.
- `type`: The required type of element: character, integer or real.

**Value**

The text entry values transformed in integer or real if necessary. If the value is not valid the function returns NA.

**Author(s)**

Marina Saez Andreu

---

**refreshDataSetsList**  
*refreshDataSetsList: refreshes KarsTS data set list*

**Description**

This function identifies the time series, gap sets and recurrence matrices present in the environment susEnv. It returns a list of data sets, which is assigned to susEnv so that it is accessible from all the package functions.

**Usage**

```r
refreshDataSetsList(outp = TRUE)
```

**Arguments**

- `outp`: when `outp = TRUE`, it shows a summary of the data sets on the output window.
Details

The function is used internally every time a new data set is created in order to refresh the data set list. From this list, KarsTS extracts information every time it creates a check box or radiobutton of gap sets. Optionally, information about the data sets appears on the output window (when outp = TRUE ). This happens when the user runs the function by means of the button List in the Time Series or Gap Sets menus. In this case, the user gets to see the information.

Value

It returns a list containing: the time series names, the number of time series, the gap sets names, the number of gap sets, the recurrence matrices names and the number of recurrence matrices.

Author(s)

Marina Saez Andreu

removeAllTypes

removeAllTypes: removes data sets

Description

This function removes all types of data sets (time series, gap sets, recurrence matrices) from the environment susEnv. It is used through the button Remove in the Time Series or Gap Sets menus.

Author(s)

Marina Saez Andreu

removeIfExists

removeIfExists: remove if exists

Description

This function checks whether a variable exists and removes it in case it does. It is used to clean KTSEnv from global variables created as by-products of the procedures. It is used internally.

Usage

removeIfExists(candidates, envir = KTSEnv)

Arguments

candidates The names of the variables to remove.
envir The environment

Author(s)

Marina Saez Andreu
removePoints

removePoints: graphically remove points from a time series

Description

This function is used to remove from a time series a set of graphically selected points. It is useful to remove outliers. It is used through the Remove Points button in the Plots menu.

Details

The points are actually removed from a copy of the time series, which is called by default timeSeriesName_pr. The points to remove are selected by dragging the mouse over them. All points within a radius are selected; the user can control this radius (or threshold) via interface.

Value

A new time series where the selected points values have been replaced by NAs.

Author(s)

Marina Saez Andreu

renameAllTypes

renameAllTypes: rename all types of data sets

Description

This function is used to rename time series, gap sets or recurrence matrices. The data set with the original name is kept. It is used through the Rename button in the Time Series or the Gap Sets menus.

Author(s)

Marina Saez Andreu
### rmCheckedTF

**Description**

This function identifies which time series have been checked from a check box.

**Usage**

```r
rmCheckedTF(prefix = "rcbValue", envir = KTSEnv)
```

**Arguments**

- `prefix`: A prefix to reconstruct the names that were assigned to the variables in the check box.
- `envir`: The environment

**Value**

A logical vector

**Author(s)**

Marina Saez Andreu

### rmDetect

**Description**

This function identifies the recurrence matrices currently loaded in the environment susEnv. It is used internally.

**Usage**

```r
rmDetect()
```

**Value**

A vector containing the names of the recurrence matrices

**Author(s)**

Marina Saez Andreu
Description

This function filters the slope (difference) outliers in a time series. It finds the greater outlier, it removes the value causing it and it tries to re-fill value by linear interpolation. Then the greater outlier in the modified time series is found and so on. It is used internally.

Usage

```r
rmSlopeOutliers(tS = NULL, origMxPosSlope = NULL, origMxNegSlope = NULL, filling = NULL)
```

Arguments

- `tS`: The time series
- `origMxPosSlope`: The maximum allowed positive slope. Greater slopes are considered as outliers.
- `origMxNegSlope`: The maximum allowed negative slope. Greater slopes (in absolute value) are considered as outliers.
- `filling`: The positions in the time series corresponding to filled values (that is, not actual observations). When this argument is not null, the actual observations remain untouched whether if they cause outliers or not.

Value

The smoothed time series. The function stops when there are no more outliers or when it reaches 100000 iterations. This is a safety measure to avoid infinite loops. If the time series still contains outliers, run the function again.

Author(s)

Marina Saez Andreu

---

Description

This function gathers, via interface, the outputs that are necessary to run the function `getRollStatistics`. It is used through the button Rolling statistics in the Analysis menu.

Author(s)

Marina Saez Andreu
roundKTS: rounding

Description

This function rounds the values of one or more time series to a number of decimal places or significant digits. When the user enters both, the significant digits option takes priority. It is used through the button Round in the Time Series menu.

Value

The rounded time series in the environment KTSEnv.

Author(s)

Marina Saez Andreu

Recurrence of probability

Description

RPKTS: This function is used to calculate the recurrence probability via interface.

Details

When two time series are selected, KarsTS calculates their correlation probability of recurrence too.

Author(s)

Marina Saez Andreu

References

saveAllTypes

Description

This function saves all types of data sets (time series, gap sets or recurrence matrices) to one or more R files in the working directory. When various data sets are to be saved to the same file, the user must provide name; otherwise, the file bear the name of the data set. Through the button save in the Time Series or the Gap Sets menu.

Author(s)

Marina Saez Andreu

saveReport

Description

This function saves the contents of the output window to a txt file in the working directory. The file name is report plus the date of creation. Through the button Save report to txt file (lower part of the main screen).

Author(s)

Marina Saez Andreu

scaleKTS

Description

This function is used to scale or more time series. It handles the inputs and outputs of the myScale function. Through the button Scale in the Time Series menu.

Author(s)

Marina Saez Andreu
scattTimeSeries

A function for plotting time series in the phase space

Description

This function plots time series in a two dimensional or three dimensional phase space. The time series can be embedded so that the sum of the dimension of all time series is 2 or 3. Is is used through the Phase Portraits button in the Plots Menu.

Author(s)

Marina Saez Andreu

selectionGaps

selectionGaps: select gaps in a time series

Description

This function is used create a gap set from a time series. The new gap set contains the gaps that meet some criteria. The available criteria are: all gaps, minimum length, maximum length and specific gaps (selected from a list). The user can choose one or more criteria. Through the button Gap selection in the Gap Sets menu.

Details

The criteria can be combined in two ways: a. (minimum length and maximum length) and specific gaps; b. (minimum length or maximum length) and specific gaps. When the user chooses only one length criterion, the combination must be the second (the first one would produce and empty gap).

Value

A gap set gathering the required gaps.

Author(s)

Marina Saez Andreu
**selectionTS**

**Description**

This function is used to cut a piece of time series, resample a time series or both. The user chooses the initial and final dates of the new time series and the resampling factor. The dates default to the original time series extremes and the resampling factor defaults to 1 (no resampling). Through the Cut and resampling button in the Time Series menu.

**Author(s)**

Marina Saez Andreu

---

**separateEntry**

**separateEntry: separates and checks comma-separated entry values**

**Description**

Some text entries in KarsTS consist of a list of values separated by commas. This function separates the values and checks whether they have the appropriate format. It is used internally.

**Usage**

`separateEntry(y, class1 = verifyIntEntry, class2 = verifyCharEntry, noValid = NA)`

**Arguments**

- **y**  
  The value to check
- **class1**  
  The class the first element should be
- **class2**  
  The class the second element should be
- **noValid**  
  The output to return when the input is not valid
**setCorrectDate**

`setCorrectDate`: verifies date entry and returns proper output

**Description**

This function is used inside the verifyDateEntry function, which is used to verify if a date is correct. The function `setCorrectDate` verifies if the year, month, day, hour, minutes or seconds are correct. For example, the day has to be an integer between 1 and 31, the minutes and seconds must be integers between 0 and 59 etc.

**Usage**

`setCorrectDate(x, type)`

**Arguments**

- `x` The element to test
- `type` The type of element: year, month, day, hour, minute or second

**Value**

The element if it is valid; NA otherwise.

**Author(s)**

Marina Saez Andreu

---

**setwdKTS**

`setwdKTS`: set working directory

**Description**

This function is used to change the working directory by means of the Set WD button on the main screen.

**Author(s)**

Marina Saez Andreu
**showHelp**

showHelp: shows KarsTS help

**Description**

This function is used to open a short help file through the Help button on KarsTS main screen. For further help, a User’s Guide in pdf format is available.

**Author(s)**

Marina Saez Andreu

---

**slopeOutliersBut**

slopeOutliersBut: remove slope outliers

**Description**

This function handles the inputs and outputs from function rmSlopeOutliers. It removes, one by one, points that produce abnormally steep slopes and replaces them by linear interpolation. The peaks are progressively filed down and the result is a smoothed time series. Usually, these outliers are caused by wrong measurements; filling missing values can produce them also. It is used through the button Rm slope Outliers in the Filling menu.

**Details**

In the first step, the user must choose the time series to smooth and whether if the outliers must be removed from a filling or from the whole time series. In the second step, the user must provide the maximum positive and negative slopes or, alternatively, a reference time series from which calculate them. When the outliers are removed from a filling, the reference time series is the non-filled time series.

**Value**

The smoothed time series. In order to avoid infinite loops, there is a maximum of 100000 iterations. If, by then, the time series has not been completely smoothed, it will necessary to apply the function again.

**Author(s)**

Marina Saez Andreu
**stationarityKTS**

*stationarityKTS: stationarity tests*

**Description**

This function runs a number of stationarity tests using the functions PP.test and Box.test from package stats and adf.test and kpss.test from package tseries. Through the Stationarity button in the Analysis menu.

**Author(s)**

Marina Saez Andreu

**statisticsKTS**

*statisticsKTS: calculates the statistics of one or more time series*

**Description**

This function handles the inputs and outputs from the function getStatistics, which is used to calculate the statistics of a time series. Through the button Statistics in the Analysis menu.

**Value**

A table on the output window.

**Author(s)**

Marina Saez Andreu

**stinemannKTS**

*stinemannKTS: filling by Stinemann’s interpolation*

**Description**

This function fills missing values in a time series by means of Stinemann’s interpolation. Through the Stinemann’s button in the Filling menu.

**Value**

The filled time series in the environment KTSEnv and a summary of the results on the output window.

**Author(s)**

Marina Saez Andreu
Description

This function performs a loess seasonal decomposition using the function stlplus::stlplus. Through the button Loess.decomp. in the Analysis menu.

Details

The user enters via interface the time series, the period, the seasonal window, the trend window and the type of decomposition (additive or multiplicative). Optionally, the trend can be decomposed using more windows.

Value

The trend, the seasonal and the irregular components. These bear the name of the original time series plus Tr, Sea and Rem, respectively. If the trend is further decomposed, the user must enter names for the output components.

Author(s)

Marina Saez Andreu

Description

It applies a Theiler’s window to a recurrence matrix via interface.

Author(s)

Marina Saez Andreu
**tsCheckedTF**  
*tsCheckedTF: identifies checked time series*

**Description**  
This function identifies which time series have been checked from a check box. It is used internally.

**Usage**  
```
tsCheckedTF(prefix = "scbValue", envir = KTSEnv)
```

**Arguments**
- **prefix**: A prefix to reconstruct the names that were assigned to the variables in the check box.
- **envir**: The environment

**Value**
A logical vector

**Author(s)**
Marina Saez Andreu

---

**tsDetect**  
*tsDetect: identifies the time series currently loaded*

**Description**  
This function identifies the time series currently loaded in the environment susEnv. It is used internally.

**Usage**
```
tsDetect()
```

**Value**
A vector containing the names of the time series

**Author(s)**
Marina Saez Andreu
verifyCharEntry

| verifyCharEntry | verifyCharEntry: verify character entries |

**Description**

This function reads a variable coming from a text entry and checks whether it is character or not. It is used internally.

**Usage**

```r
verifyCharEntry(x, noValid = "isNoValid")
```

**Arguments**

- `x`: The element to check
- `noValid`: The output the function will return when the input is not character.

**Details**

One-character strings are not allowed.

**Author(s)**

Marina Saez Andreu

**Examples**

```r
verifyCharEntry("Strawberry", noValid = "isNoValid")
verifyCharEntry(235, noValid = "isNoValid")
verifyCharEntry(235, noValid = NA)
```

verifyDateEntry

| verifyDateEntry | verifyDateEntry: verify date entries |

**Description**

This function checks whether a date is valid or not. It has six inputs (seconds, minutes, hours, day, month and year); these come from a set of text entries. It is used internally.

**Usage**

```r
verifyDateEntry(valSecs, valMins, valHour, valDay, valMonth, valYear)
```
verifyIntEntry

Arguments
valSecs  Seconds
valMins  Minutes
valHour  Hour
valDay   Day
valMonth Month
valYear  Year

Author(s)
Marina Saez Andreu

Description
This function reads a variable coming from a text entry and checks whether it is integer or not. It is used internally.

Usage
verifyIntEntry(x, noValid = "isNoValid")

Arguments
x  The element to check
noValid  The output the function will return when the input is not character.

Author(s)
Marina Saez Andreu

Examples
verifyIntEntry("Strawberry", noValid = "isNoValid")
verifyIntEntry(235.6, noValid = NA)
verifyIntEntry(235, noValid = NA)
verifyIntEntry(0235, noValid = NA)
verifyIntEntry(235.0, noValid = NA)
verifyRealEntry

verifyRealEntry: verify character entries

Description
This function reads a variable coming from a text entry and checks whether it is real or not. It is used internally.

Usage
verifyRealEntry(x, noValid = "isNoValid")

Arguments
- x: The element to check
- noValid: The output the function will return when the input is not character.

Author(s)
Marina Saez Andreu

Examples
- verifyRealEntry("Strawberry", noValid = "isNoValid")
- verifyRealEntry(235, noValid = NA)
- verifyRealEntry(0235, noValid = NA)
- verifyRealEntry(235.6, noValid = NA)

welcomeScreen

welcomeScreen: launches the welcome screen

Description
This function launches KarsTS welcome screen. It is used internally.

Usage
welcomeScreen()

Author(s)
Marina Saez Andreu
windRoseKTS: wind rose

Description
This function plots a wind rose from directional data. The data must be in sexagesimal degrees. It is used through the button Wind Rose in the Analysis menu.

Author(s)
Marina Saez Andreu

writeMethodSummary: writes a summary of the filling method

Description
This function is internally used when any of the filling methods that KarsTS offers is used. It writes on KarsTS output window information that is shared by all filling methods: which time series and gap set were to fill, which gaps were effectively filled and which remained empty. Filling-method-specific information is written through other functions. It is used internally.

Usage
writeMethodSummary(filledNasTable, remainingNAsInGap, selTsName, selGapName, selGap)

Arguments
filledNasTable  Table of filled gaps
remainingNAsInGap
    Table of gaps that could not be filled. It does not include gaps whose filling was not attempted.
selTsName  Name of the time series that was filled.
selGapName  Name of the gap set that was meant to be filled
selGap  Gap set that was meant to be filled

Author(s)
Marina Saez Andreu
writeMethodTitle: writes a title on KarsTS output window

**Description**

This function writes a title on KarsTS output window. It is used internally.

**Usage**

`writeMethodTitle(titleMethod)`

**Arguments**

- `titleMethod`  The text of the title

**Author(s)**

Marina Saez Andreu
Index

* package
  * KarsTS-package, 5
  * aboutKTS, 6
  * aggregateKTS, 6
  * anaSamPer, 7
  * applyGap2TSer, 7
  * applyTheiler, 8
  * are2TsTimeCompatible, 9
  * areTsGapTimeCompatible, 10
  * areTsRmTimeCompatible, 11
  * arimaKalman, 11
  * arimaXXKalman, 12
  * assignMultiple, 12

  buttons1, 13
  buttons2, 13
  buttons3, 13
  buttons4, 14
  buttons5, 14

  checkIfAny, 14
  checkIfAnyGapOrTs, 15
  checkIfAnyGapTs, 15
  checkIfAnyRm, 16
  checkIfAnyRmTs, 16
  checkIfAnyTs, 17
  cleanEnvir, 17
  compareVecVec, 18
  composeKTS, 18
  createChb, 19
  createChbChb, 20
  createChbEntry, 20
  createCrossRM, 21
  createCrossRMPlot, 22
  createDistMatrix, 22
  createEachRb, 22
  createEntry, 23
  createFAN, 24
  createFANplot, 24

  createGapChb, 25
  createGapRb, 25
  createJointRM, 26
  createJointRMPlot, 26
  createNote, 27
  createOK, 27
  createRandGaps, 28
  createRandName, 28
  createRb, 29
  createRmChb, 29
  createRmRb, 30
  createSimpleRM, 30
  createSimpleRMPlot, 31
  createSpecGaps, 31
  createSubPanR4C1, 32
  createTITLE, 32
  createTitle, 33
  createTsChb, 33
  createTsRb, 34
  cumuKTS, 34

  destroyMainScreen, 35
  destroyWelcome, 35
  determinismKTS, 35
  diffKTS, 36

  E1dAndE2d, 36
  embedData, 37
  endingLines, 37
  exportAll, 38

  fillWithTwins, 38
  findDateFormat, 39
  findTwins, 39
  fnnKTS, 40
  functToExport, 41

  gamKTS, 41
  gapCheckedTF, 42
  gapDetect, 42
INDEX

91

gapForSelMethod, 43
genGapExample, 44
genRmExample, 44
genTSExample, 45
getClassEnvir, 46
getCoordsKTS, 46
getCRP, 47
getDelayCharTimes, 48
getFANRM2, 48
getGapsAfterFill, 49
getMaxNegSlope, 50
getMaxPosSlope, 50
getNAsGaps, 51
getNewGapsInd, 51
getOtherErrEstim, 52
getProTaos, 53
getRecurrencePoints, 54
getRollStatistics, 55
getSamPerTable, 56
getSamPerTable.1Freq, 56
getScreenSize, 57
getStatistics, 57
getUniqueSampPer, 58
goodnessFilling, 58
groupDates, 59
groupIndices, 59

histKTS, 60

invariantsKTS, 60
isTimeAlright, 61

KarsTS, 61
KarsTS-package, 5
KTSEnv (functToExport), 41

laminarityKTS, 62
linCorrKTS, 62
linearityKTS, 63
loadAllTypes, 63
loadKarsTSFonts, 63
loessKTS, 64

mainScreen, 64
makeGlobal (functToExport), 41
meanValue, 64
mergeTsOrGap, 65
missForestKTS, 65
modeKTS, 66

mutInf, 66
mutualKTS, 66
myApplyVector, 67
myLinModel, 67
myScale, 68

naApproxKTS, 68
NAs4Resamp, 69
naSplinesKTS, 69
normalityKTS, 69

packagesToImport, 70
pcaKTS, 70
plotTimeSeries, 70

readMultEntryvalues, 71
refreshDataSetsList, 71
removeAllTypes, 72
removeIfExists, 72
removePoints, 73
renameAllTypes, 73
rmCheckedTF, 74
rmDetect, 74
rmSlopeOutliers, 75
rollStatisticsKTS, 75
roundKTS, 76
RPKTS, 76

saveAllTypes, 77
saveReport, 77
scaleKTS, 77
scattTimeSeries, 78
selectionGaps, 78
selectionTS, 79
separateEntry, 79
setCorrectDate, 80
setwdKTS, 80
showHelp, 81
slopeOutliersBut, 81
stationarityKTS, 81
statisticsKTS, 82
stationarityKTS, 82
statisticsKTS, 82
stlinemannKTS, 82
stlplusKTS, 83

theilerKTS, 83
tscheckedTF, 84
tsDetect, 84

verifyCharEntry, 85
verifyDateEntry, 85
verifyIntEntry, 86
verifyRealEntry, 87
welcomeScreen, 87
windRoseKTS, 88
writeMethodSummary, 88
writeMethodTitle, 89