KarsTS 2.4.1

An interface for microclimate time series analysis

User's guide

The KarsTS package

Author, Creator and Maintainer: Marina Sáez Andreu

Thesis advisors: David Benavente García and Soledad Cuezva Robleño

Contributor: Concepción Pla Bru

Date: January, 2021

License: GPL (>=2)

Citation:

Marina Sáez (2021). KarsTS: An interface for microclimate time series. R package version 2.4.1.

If you have any question or problem, email the maintainer (marinasaez_andreu@hotmail.com). Suggestions for the next version are welcome too. I hope KarsTS helps you with your research. KarsTS took a great effort; please, do not forget to cite it.

I would like to express my gratitude to my thesis advisors for their support and to my dear colleague C. Pla for proofreading this user's guide.

Cover image modified from:
http://freeaussiestock.com/free/New_South_Wales/slides/wombeyan_caves_formations.htm

Contents

1.. Introduction 6
2.. Getting started 6
3.. Note on NAs ad NaNs 6
4.. Data sets 7
5.. Parts of the screen 7
6.. Menus 8
6.1.. Times Series 8
Load 8
Save 8
Remove 8
Export 8
Rename 8
Merge 8
List 9
Sampling periods 9
Cut and resampling 9
Aggregate 9
Operations 10
Round 10
Scale 10
Difference 11
Cumulative sum 11
6.2.. Gap Sets 11
Load, Save, Remove, Export, Rename, Merge and List 11
Gap selection 11
Artificial Random Gaps, Artificial Random Gaps and Apply Gaps to Time Series 11
Upsample 11
6.3.. Analysis 12
Statistics 12
Rolling statistics 12
Loess decomp. (Loess decomposition) 12
Loess smooth. (Loess smoothing) 13
Normality 13

Stationarity 13
Linearity 14
PCA 14
Mutual info 14
Invariants 15
Recurrence matrix 15
Cross recurrence matrix 16
Joint recurrence matrix 16
FAN recurrence matrix 16
Self-repeating rate (Determinism) 17
Laminarity 18
Theiler window 18
6.4.. Plots 18
Plot ts 18
Remove points 19
Get coordinates 19
Linear correlation 19
AMI 20
Wind rose 20
Histogram 20
Phase portraits 20
False Nearest Neighbors 20
Ed(1) and Ed (2) 21
Recurrence plot, FAN recurrence plot and cross recurrence plot. 21
Distance plot 22
RP 22
6.5.. Filling 23
Interpolations 23
ARIMA 23
Position-wise Mean Value 24
Multiv. Splines (Multivariate splines) 25
MissForest 25
Twins 25
ARIMAX 26

Check filling 26
7.. References 26
Appendix I: Functionalities 30
Appendix II: Input files format 34
Time series: 34
.R or .RData. files 34
.csv files 34
txt files 34
Gap sets 35
.R or .RData files 35
.csv files 35
.txt files 35
.R o .RData files 35
csv files 36
txt files 36
Loading frequent problems 36
Appendix III: Filling methods overview 38
Univariate 38
Interpolations 38
ARIMA 38
Position-wise mean value 38
Multivariate 38
Missing Random Algorithm 38
Twins 38
Appendix IV: Graphical parameters 41

1.. Introduction

KarsTS is an R-based interface designed to analyze microclimatic time series with an emphasis on underground environments, such as caves; nonetheless, it can be useful in many other fields of research. Microclimate research typically involves the continuous monitoring of temperature, humidity and gas concentration (CO_2, ^{222}Rn, CH_4 …), amongst other. Many of these variables have a strong non-linear behavior. For this reason, KarsTS gathers a number of linear and non-linear methods in a friendly-user interface: correlation, mutual information, phase portraits, recurrence matrices... Microclimate time series frequently contain gaps that can reach significant lengths; for this reason, KarsTS includes a number of univariate and multivariate gap filling methods as well as the possibility of applying them to subsets of gaps.

KarsTS can import and export txt and csv files. The user can access KarsTS data sets from the R console; however, KarsTS is intended to be autonomous in order that it is not necessary to use another program at the same time.

In this user guide, KarsTS functionality is explained menu by menu. For every method, we describe its utility, applicability, inputs and outputs; however, this user's guide is not meant to provide a wide theoretical background about the methods.

A complete list of KarsTS functionalities can be found in Appendix I. In Appendix II, we provide solutions for frequent problems when loading data from csv or txt files. Appendix III consists of an overview of the filling methods (always from a practical point of view). Appendix IV offers information about the optional graphical parameters available for different types of plots.

2.. Getting started

KarsTS runs both on Windows and Mac.

First usage:

1. If you work on Mac, you may need to update XQuarz.
2. Install R on your computer if it is not already installed.
3. Install R Studio in your computer if it is not already installed
4. Open R Studio and install the package KarsTS
5. Run the following lines on the command window

```
library(KarsTS) #loads the package
KarsTS() #launches the interface
```

Successive usages: go straight to the fifth point.

3.. Note on NAs ad NaNs

In R, and, consequently, in KarsTS, missing values are represented as NA (not available). NA is different from NaN (not a number). For example, $\log_{10}(-1)$ is not a number, whilst a temperature missing observation is a number, even if it is unknown.

4.. Data sets

There are three types of data sets in KarsTS

- Time series
- Sets of gaps
- Recurrence matrices

See Appendix II: Input files format.

5.. Parts of the screen

The main screen has five menu-buttons, which are located in the upper part of the main screen.

1. Time Series: tools to manipulate time series
2. Gap Sets: tools to manipulate gap sets
3. Analysis: tools to analyze time series and create recurrence matrices
4. Plots: data visualization and graphic outputs
5. Filling: filling methods

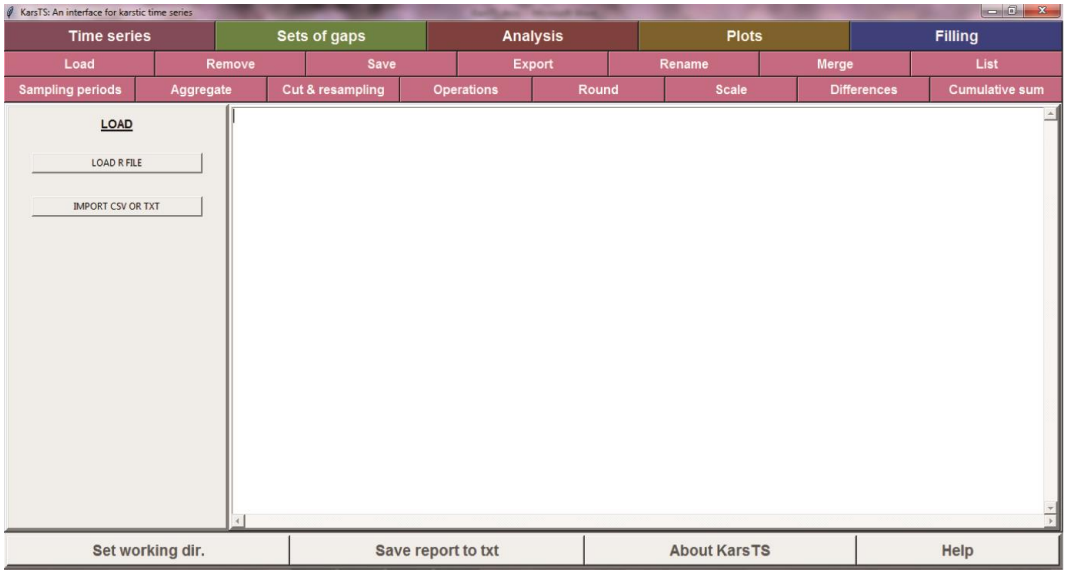

Figure 1: KarsTS main screen

Below the buttons, there is an input panel (at left) and an output window (at right). The output window is editable, that is, the user can write, erase, copy and paste in it. The graphics can also be plotted directly on a pop-up window. In the lower part of the screen four additional buttons allow to: change the working directory, save the outputs to a txt file, get information about the program and open a help file.

Graphical outputs appear on independent windows.

6.. Menus

6.1.. Times Series

Load

The Load button is used to load time series, sets of gaps and recurrence matrices. KarsTS accepts R, RData, csv and txt file types. The R and RData files can contain more than one data set.

The input panel has two buttons. The upper one loads R files whereas the lower one imports txt and csv files.

In order to load txt or csv files, it is necessary to specify the date format. It is convenient to specify the separator, although it defaults to tabulation for txt files and comma for csv files. Note that csv files can use semicolons as separators, depending on the computer language options (the computer where the file was created).

For more details, see Appendix II: Input files format and Loading frequent problems.

Save

Data sets can be saved in separated .R files to the user's working directory. The files are named following the data sets they store. It is possible to save several data sets to a single file too.

Remove

Data sets will be removed from the environment, not from the user's hard disk.

Export

KarsTS can export data sets to .csv or .txt files. Each gap set will be stored in a separated file, named after the data set; even though, multiple time series can be saved to a single file.

Rename

Rename changes the name of a data set. The original data set will not be removed.

Merge

Only time series and gap sets can be merged.

Time series must fulfill the following conditions in order to be merged:

- They cannot overlay (otherwise, the same time could have two different simultaneous values)
- All of them must have the same sampling period

If there are time gaps between the time series to merge, KarsTS will fill them with NAs.

The requirements for gap sets are the following:

- They must come from time series with the same sampling period
- They must come from time series with the same starting date

KarsTS verifies the times series or gap sets compatibility before merging them.

List

This button displays a list and a short description of the loaded data sets on the Output Window.

Sampling periods

This function analyzes the sampling periods and existing gaps in a time series.

When a time series with various sampling periods is imported, KarsTS introduces as many NAs as necessary in order that the whole time series has the minimum sampling period. For the same reason, KarsTS also introduces NAs to fill the time gaps. As a consequence, once the time series is loaded, there are NAs coming from both true missing values and the resampling. The latter result in many gaps with the same length. The user must select the true sampling periods from a checkbox.

The final output of this button is a table in the Output Window. Each row corresponds to a piece of time series that has a unique sampling period and contains either actual values or missing values. The columns of the table correspond to:

- Sampling period (in minutes)
- Initial date
- Final date
- Number of measurements
- Actual values or NAs

Cut and resampling

This function has two main purposes:

- Creating a time series that is a section of another one
- Creating a time series with a different sampling period. For example, if the sampling period of the input time series is equal to 5 minutes and the resampling is 2, the sampling period of the resulting time series will be equal to 10 minutes. Unlike the Aggregate button, it will not average the measurements. This function downsamples the time series; for upsampling use the button Upsample in the Menu Gaps.

Inputs:

- Sampling period: defaults to 1 (the same sampling period)
- Initial date: defaults to the first date in the time series
- Final date: defaults to the last date in the time series

Aggregate

This button builds a time series of grouped values, using one of these functions: mean, median, minimum, maximum, standard deviation or sum.

Inputs:

- Time series
- Period (in lags)
- Statistic: mean, median, minimum, maximum, standard deviation or sum.
- NA treatment: see the examples below
- Initial date: see the examples below

Examples:

Consider a time series with an observation every 30 minutes (sampling period).

1) We want to calculate a time series of minimum daily values. The period is equal to 48 (number of measurements in one day) and we select the minimum. Some days might have missing values (NAs). If the option "propagate NAs" is chosen, the days containing some missing value are assigned NAs. On the contrary, with the option "ignore NAs", the minimum is calculated using the available measurements and a day will be assigned a NA only in case all its measurements are missing. Finally, the starting date of the time series can be also changed. For instance, if the original time series starts at 2009-12-25 00:03:00, we can start at 2009-12-25 00:00:00 for the sake of simplicity.
2) We want to shorten the time series in order to save computing time. Thus, we choose a period of 4 and the mean as statistic. We will obtain a time series with 12 values per day, instead of 48. Note that each value is the mean of four measurements. To resample without averaging, use the Cut and resampling button instead.

Operations

This button is used to perform operations with one or more time series: sum, multiplication, reciprocal, opposite and natural logarithm.

The result of adding or multiplying two or more time series is a single time series, the name of which must be entered by the user. The rest of operations are applied separately to each selected time series and the output time series are assigned default names.

Round

KarsTS does not automatically round the output time series when a calculation is performed. The input panel shows two rounding options: number of significant digits or number of decimal places. When both options are entered, only the number of significant digits is used.

Scale

This button scales one or more time series at the same time. There are two options, depending on the parameters used to center and scale the variable: (i) the median and the median absolute deviation (robust scaling) or (ii) the mean and the standard deviation.

Difference

This button differences one or more time series at the selected lag. The option “Center the time series” interpolates the differenced time series in order to keep the length of the original time series (otherwise, the difference time series has one less value)

Cumulative sum

It is used to calculate the cumulative sum of one or more time series. As in the difference button, it is possible to center the output time series (otherwise, the cumulative time series has one more value)

6.2.. Gap Sets

Load, Save, Remove, Export, Rename, Merge and List

See Times Series

Gap selection

This button extracts a gap set from the gaps of a time series.

There are four selection criteria: all, minimum size, maximum size and specific gaps. The minimum and maximum sizes are measured in lags. When the *Selection* button is selected, a list of gaps appears on the output window to enable the selection of specific gaps.

When the user choses various criteria, they are combined according to these rules:

- The criterion “all” annuls the rest.
- When the maximum and the minimum size are entered, only the gaps that meet simultaneously both conditions will be part of the gap set.
- When both a size criterion and specific gaps are selected, the latter will be added to the gap set regardless of their size.

Artificial Random Gaps, Artificial Random Gaps and Apply Gaps to Time Series

In order to verify the goodness of a filling method on a time series, we recommend creating an artificial set of gaps.

The Artificial Random Gaps button creates a random set of gaps that will not overlay the existing gaps in the time series. The user can choose the number of gaps and their size.

The Artificial Specific Gaps button generates a gap between two dates chosen by the user.

When a set of artificial gaps is created, a new time series containing those gaps is created too. In order to apply the gap set to another time series, use the *Apply Gaps to Time Series* button.

Upsample

The Upsample button introduces NAs in a time series that needs to be upsampled. The user needs to introduce a name for the output time series and the new sampling period in seconds. The new sampling period must be compatible with the original one. For example, for an

original sampling period of 1 hour (3600 s), 15 minutes (900 s) is a compatible sampling period; on the contrary, a sampling period of 14 minutes (840 s) will not be compatible and KarsTS will launch a warning requiring a compatible sampling period.

6.3.. Analysis

Statistics

It displays a table in the Output Window that contains the univariate descriptive statistics of one or more time series.

Rolling statistics

This button calculates the time series statistics in sliding windows. It returns as many time series as statistics were selected by the user. If the median or the mean are selected, the result is a smoothed time series that has the same length as the original one.

Inputs

- Time series
- Sliding window size: in lags
- Statistics: check one or more
- Estimate tails: the sliding window is centered, which implies that the firsts/lasts values in the time series are not surrounded by a complete, symmetric window. For example, let the window length be 48. The first value in the time series has no values at left, although it has 24 values at right. If we do not estimate tails, the first value in the time series will be NA; otherwise, the first value in the time series will be the median of the first 25 values (24 at right plus the value itself). If we estimate the tails, the function will take longer to run.

Loess decomp. (Loess decomposition)

It is used to decompose a time series using the loess seasonal decomposition from package *stlplus* (Hafen, 2016). By default, the time series is decomposed in three components: trend, seasonal component and remainder. The components bear the name of the original time series, plus the suffixes *Tr*, *Sea* and *Rem*, respectively.

At each point, the trend is the result of averaging the values in a window around the point (trend window). A larger window yields a smoother trend. In the same way, the seasonal component is calculated in a window (the seasonal window). A large seasonal window results in a seasonal component whose amplitude changes slowly.

Optionally, more windows of different sizes can be used to decompose the trend into two or more time series.

Inputs:

- Seasonality: number of measurements in one period
- Seasonal window: number of measurements in the seasonal window

- Trend window: number of measurements in the trend window
- Trend decomposition: it allows decomposing the trend into up to seven time series, although usually only two or three will be used. The window size and the name for each output time series are requested. The larger window should be in the uppermost box.

Loess smooth. (Loess smoothing)

Unlike the loess seasonal decomposition, the loess smoothing does not require a periodical phenomenon. The smoothing is the result of locally fitting a linear model in a window around a point. The model predicts the value of the corresponding central point. By default, the model relates the time series values and the time series times. Moreover, the user can include other variables (predictor time series) in it.

Inputs:

- Time series
- Alpha: the parameter alpha controls the size of the sliding window. Basically, alpha is the proportion of points in the time series that will be used in each window; therefore, a higher alpha value yields a smoother time series.
- Predictor time series: optional

Normality

It is used to run a number of stationarity tests from package MVN (Korkmaz *et al.*, 2014), When the user selects more than one time series, KarsTS performs both univariate and multivariate tests. Missing values are removed from the time series because the tests do not admit missing values.

Table 1: Normality tests

Test	Type
Cramer von Mises	Univariate
Lilliefors (KS)	Univariate
Anderson-Darling	Univariate
Henze-Zirkler	Multivariate
Mardia (for skewness and kurtosis)	Multivariate

Stationarity

It performs a number of stationarity tests from packages *stats* (R Core Team, 2017) and *tseries* (Trapletti *et* Hornik, 2017). Missing values are removed from the time series because the tests do not admit missing values. This affects the time correlations; therefore, it is convenient to fill the time series before running the tests.

The next table summarizes the null hypotheses:

Table 2: Stationarity tests

Test	H_0
Philipp-Perron unit root test	It has a unit root (The detrended time series is not stationary)
Box-Ljung test	Independence (There are no self-correlations)
Augmented Dickey-Fuller test (lag=1)	It has a unit root (The detrended time series is not stationary)
KPSS test for Level Stationarity	Level stationarity (The time series is white noise)

Linearity

This button performs a number of linearity tests from the *nonlinearTseries* package (García et al., 2015). It does not admit missing values. In case the time series is not seasonal, the delay for the surrogates test must be 1.

The tests and their null hypotheses are the following:

Table 3: Linearity tests

Test	Null hypothesis
Surrogates test	Data comes from a linear stochastic process
Teraesvirta's neural network test	Linearity in mean
White neural network test	Linearity in mean
Keenan's one-degree test for nonlinearity	The time series follows some AR process
McLeod-Li test	The time series follows some ARIMA process
Tsay's Test for nonlinearity	The time series follows some AR process
Likelihood ratio test for threshold nonlinearity	The time series follows some AR process (the alternative hypothesis is that it follows a TAR process)

PCA

This button performs principal component analysis by singular value decomposition (*prcomp* function from the *stats* package). The user has to choose the time series and a name for the output components. These will appear as new time series. In addition, KarsTS writes on the output window the variance explained and the loadings.

Mutual info

This button returns the mutual information between two time series and their marginal entropies. It is based on the package *infotheo* (Patrick, 2014). In order to estimate these magnitudes, the time series must be discretized previously; KarsTS returns information about the discretization too.

The user has to enter the following inputs:

- Two time series
- Parameters to discretize each time series
    - Type of discretization: equal frequency, equal width or global width. For skewed distributions, the equal frequency method is recommended (Ismail, 2003).
    - Number of bins

Optionally the mutual information can be calculated using a sliding window. In this case, the user needs to specify the length of the window and the result consists of three time series: a time series of mutual information values and two time series of marginal entropies. In order to calculate the mutual information, a sufficient amount of values is required; in other words, the results may not be significant if the sliding window contains few values.

Invariants

This button is useful to estimate two invariants from package *nonlinearTseries* (García *et al.*, 2015), namely, the correlation dimension and KS entropy. In the context of recurrence analysis, the invariants are properties that do not change once the system has been unfolded completely. Because of this, the invariants can be used to estimate the parameters for a correct embedding, specifically the radius and the embedding dimension. The estimation of the correlation dimension and the KS entropy requires long time series.

As the invariants are calculated for different radii and embedding dimensions, the user has to enter the minimum and maximum embedding dimensions as well as the minimum and maximum radii. Moreover, the user must enter the number of radii at which the invariants will be estimated. For example, for 6 radii ranging from 50 to 50.5, the invariants will be calculated at 50, 50.1, 50.2, 50.3, 50.4, 50.5 and 50.6. In general, the curves become horizontal when the radius and embedding dimension are appropriate.

- Correlation sum and dimension:
  The correlation dimension measures the fractal dimension of a geometrical object embedded in a phase space. The correlation sum can be interpreted as the probability of finding a neighbor in a ball of radius r in the phase space.
  The function returns two plots in a single panel. The upper one represents the correlation sum for different radii and embedding dimensions. The user must check that there is a region in this plot where the curves for different embedding dimensions are parallel; otherwise, the estimation of the correlation dimension must be discarded.
- KS Entropy:
  It is a measure of complexity; specifically, smaller values of KS Entropy indicate more self-similarity or less noise.
  This option plots KS entropy curves calculated with different embedding dimensions and radii. The radius values are in the X axis.

Recurrence matrix

A recurrence matrix can be defined as follows:

$$\mathbf{RM}_{i,j}(\varepsilon) = \Theta(\varepsilon - ||\mathbf{x}_i - \mathbf{x}_j||), \quad i, j = 1, \ldots, N,$$

where ε is a threshold distance and Θ is the Heaviside function (Marwan *et al.*, 2007). In other words, $\mathbf{RM}_{i,j}(\varepsilon)$ equals zero when the distance between two points (in the phase space) is greater than the threshold and it equals one otherwise. The time series can be embedded.

For further information, Marwan *et al.* (2007) provides a complete, simple introduction to recurrence analysis. March *et al.* (2005) and Marwan (2011) focus on practical aspects.

The following table shows the buttons that can be used to estimate the inputs needed to create a recurrence matrix:

Table 4: Tools for getting the recurrence matrix parameters

Inputs	Useful buttons
Embedding dimension	False Nearest Neighbors, Invariants
Delay	Mutual information
Tolerance	Invariants
Theiler window	Mutual information

Cross recurrence matrix

Everything said about the simple recurrence matrices in the above paragraph applies to the cross ones. However, there are some differences the user needs to consider.

Whilst a (simple) recurrence matrix compares a time series with itself, a cross recurrence matrix compares two similar variables; therefore, it is necessary to choose two time series. The first one will be assigned to the X-axis and the second one to the Y-axis. KarsTS does not check whether the selected time series are appropriate to create a cross recurrence matrix. If the time series do not reflect similar phenomena, the user should consider a joint recurrence matrix instead.

Unlike a simple recurrence matrix, a cross recurrence matrix does not need to be symmetric. Thus, for the same matrix size, it takes significantly more time to calculate and more memory to store them.

For further information, see Marwan *et al.* (2007)

Joint recurrence matrix

A joint recurrence matrix is the result of multiplying element-by-element two or more simple matrices. The Joint recurrence matrix button requires two simple recurrence matrices and the name for the joint one. In order to create a joint recurrence matrix using more than two simple matrices, use the Joint Recurrence Matrix button repeatedly.

For theoretical aspects, see Marwan *et al.* (2007).

FAN recurrence matrix

A fixed amount of neighbors (FAN) recurrence matrix is a simple recurrence matrix where only the closest neighbors to each point are kept. In other words, in a simple recurrence matrix the neighborhood is determined by the threshold (and possibly the Theiler's window), however, in a FAN recurrence matrix only the closest neighbors are considered, up to a fixed amount of them. Unlike simple recurrence matrices, FAN recurrence matrices are not symmetrical. For theoretical aspects, see Marwan *et al.* (2007). FAN matrices can be visualized using the FAN. Rec. Plot button.

KarsTS can build the FAN recurrence matrix in two ways:

- Directly from a time series
    - Inputs:
        - Time series
        - Maximum number of neighbors
        - Embedding dimension
        - Delay
        - Tolerance or threshold
        - Theiler's window
        - Name for the output matrix
    - Outputs: a FAN recurrence matrix
    - 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.
- By modifying and existing recurrence matrix.
    - Inputs:
        - Recurrence matrix
        - Time series from which the recurrence matrix originated
        - Maximum number of neighbors
        - Name for the output matrix
    - Outputs: a FAN recurrence matrix
    - Details: KarsTS will check that the times of the chosen time series and recurrence matrix are compatible, but this does not guarantee that they were chosen correctly. If you built the matrix a long time ago, consider to build the FAN matrix directly from the time series.

Self-repeating rate (Determinism)

In recurrence analysis, the ratio of recurrent points belonging to diagonal lines to the total amount of recurrent points is usually called *determinism* because it can be considered a measurement of the system *determinism* under certain circumstances (Marwan, 2011). In order to avoid confusion between both senses of the term, we will refer to the former as self-repeating rate instead of determinism. The self-repeating rate regards the predictability and the degree of periodicity of a time series. The output is the same for both simple and joint recurrence matrices, although its meaning is not exactly the same (Marwan *et al.*, 2007).

The input is a recurrence matrix. When the recurrence matrix has many ones, this function may take some time to run (the user can check the number of ones with the List button).

In order to reduce tangential movement, the user can enter a minimum length for the diagonal lines to be considered true diagonal lines (the default minimum is 2).

Recurrence matrices (or plots) are square; as a consequence the length of the diagonal lines located in the borders can be limited by them. To avoid the border effect, the user can limit the analysis to a rhomb inscribed in the recurrence matrix/plot. Note that, in this case, the recurrence rate will we also estimated inside the rhomb.

The result is a list of properties in the Output Window.

- Recurrence rate: number of ones/total number of points in the recurrence matrix.
- Self-repeating: number of points belonging to diagonal lines/number of ones.
- Ratio: points belonging to diagonal lines/points in the recurrence matrix.
- Summary of the lengths of the diagonal lines. The length of the diagonal lines indicates the predictability of the time series. For example, we consider the mean length is 30 points and the sampling period is 30 minutes; it means that, as an average, the series can be predicted 30 measurements (900 minutes) forward.

Laminarity

In the same way the self-repeating rate is related to the diagonals, the laminarity is related to the vertical lines in the recurrence plot (Marwan *et al.*, 2007). The Laminarity button is analogous to the Self-repeating rate button and its outputs are:

1. Recurrence rate: number of ones/total number of points in the recurrence matrix.
2. Laminarity: number of points belonging to vertical lines/number of ones
3. Ratio: points belonging to vertical lines / points in the recurrence matrix.
4. Summary of the lengths of the vertical lines. The length of the vertical lines indicates how long the system remains trapped in the same state. Let say the mean length is 30 points and the sampling period is 30 minutes; it means that, as an average, the system changes its state in 30 points (900 minutes). For example, the state could be the cave being charged or discharged.

Theiler window

This button applies a Theiler's window (Theiler, 1986; Marwan et al, 2007) to a pre-existent recurrence matrix. The user has to choose the matrix and enter the value of the Theiler's window.

6.4.. Plots

Plot ts

This button plots one or more time series. To draw another plot, press the Plot ts button in order to remove the previous choices.

The user can choose to plot points, lines or both. When plotting only lines, consider that the points surrounded by NAs will not be plotted. For example, if the time series is as follows:

3, NA, 6.7, NA, 44, NA, 3.4, NA, 87, NA, 4.5, NA…,

the graphic will be empty.

The user can order the time series; the ones with lower ranks will be plotted above the others.

The plot appears in a new window. The user can select a fragment of time series pressing the button "ZOOM" and clicking on the plot window. The last two clicks will determine the selected fragment. When the user CLOSES the plot window, the selected fragment will appear on another plot window. It will appear also a pop-up window asking whether the user wants to save the selected fragments as new time series.

The plots can be saved to *png* or *tiff* files; the user must specify the width and height of the image in centimeters and the resolution in ppi.

Optionally, several graphical parameters can be chosen (see Appendix IV).

Remove points

This button allows selecting some points on a plot and turning those points into NAs in the corresponding time series. Thus, for example, the user may clean the time series from outliers.

The process takes two steps:

- Plot
- Click the points in the plot.
- Close the plot
- A pop-up window will ask whether you want to create a new time series
- Alternatively, the button *REMOVE POINTS* can be used to create the new time series.

Get coordinates

The user selects a set of points in a time series. Then, when the button *WRITE* is pressed, the function returns their position, time and value in a table on the Output Window.

First, select the time series and press "Plot". Select the points on the plot, they will turn green. Close the plot and write the results. A table with the required information will appear in the Output Window.

Linear correlation

It plots the correlation functions. To calculate the autocorrelation function (acf) and the partial autocorrelation function (pacf), the user must select one time series. The button returns the cross-correlation plot when two time series are selected. NAs are not allowed; therefore, the time series may require a preliminary filling. The correlation will be calculated at delays up to the maximum number of lags, which is one of the inputs.

AMI

It is used to plot the delayed average mutual information (AMI) from the *tseriesChaos* package (Di Narzo *et* Di Narzo, 2013). The AMI is useful to compare non-linear correlations at different lags. In recurrence matrices, the delay is often estimated as the first minimum in the AMI.

As in the linear correlation case, the user can choose one or two time series. NAs are not allowed; therefore the time series requires a preliminary filling. The AMI will be calculated at delays up to the maximum number of lags chosen by the user.

Wind rose

It plots a wind rose from directional data, based on the *circular* package (Agostinelli *et* Lund, 2017). It is necessary to enter the time series that provides the magnitude (for example, wind speed) and the time series that provides the directions (for example, wind direction). The magnitudes must be non-negative and the directions must range from 0 to 360. Optionally, the user can enter the number of bins and the number of ticks.

Histogram

It is used to plot a histogram. The user can choose the approximate number of bars, otherwise the Friedman-Diaconis formula will be used to estimate it.

Phase portraits

It is used to plot time series in the phase space (that is, the time does not appear in any axis). A phase portrait is similar to a scatter plot of time series; however, because of the temporal evolution of the data, the points usually produce recognizable trajectories (for instance, periodic elliptical orbits). The plots can have 2 or 3 dimensions. The time series can be embedded; in order to not embed, set the embedding dimension to 1 and the delay to 0 (default). To use differential embedding, the differenced time series must be calculated previously and treated as independent time series (see the buttons Loess. Smooth and Differences).

Table 5: Phase portraits

Number of time series	Embedding dimension of the time series	Plot dimension
2	1,1	2
3	1,1,1	3
1	2	2
1	3	3
2	1,2	3
2	2,1	3

Optionally, several graphical parameters can be chosen (see Appendix IV).

False Nearest Neighbors

It is based on the tseriesChaos package (Di Narzo *et* Di Narzo, 2013). The false nearest neighbors plot can be used to estimate the embedding dimension of a time series. An insufficient embedding dimension produces false nearest neighbors; therefore, when the phase space is completely unfolded, the number of neighbors remains constant. Consider that

noise can increase the embedding dimension unnecessarily, so it might be convenient to smooth the time series.

The user must provide the following inputs:

- Maximum embedding dimension: the embedding dimension up to which the false nearest neighbors will be calculated. Usually, a maximum embedding dimension of 10 is enough.
- Delay: it depends on the periodicities in the time series.
- Theiler window: it depends on the periodicities in the time series.

Optionally, the user can enter the escape factor and the neighborhood parameter; otherwise the default values will be used.

Ed(1) and Ed (2)

They are properties of a system that remain constant when the parameters of the phase space are well chosen. Thus, they are useful to choose a radius for the neighborhood and an embedding dimension. The function requires a delay, which depends on the periodical properties of the time series.

This function is based on the *tseriesChaos* package (García *et al.*, 2015), where an algorithm proposed by Cao (1997) is used.

Recurrence plot, FAN recurrence plot and cross recurrence plot.

The button *Recurrence plot* is used on symmetrical recurrence matrices (simple or joint); the button *Cross recurrence plot* is used on cross recurrence matrices. The button *FAN recurrence plot* is used on FAN matrices.

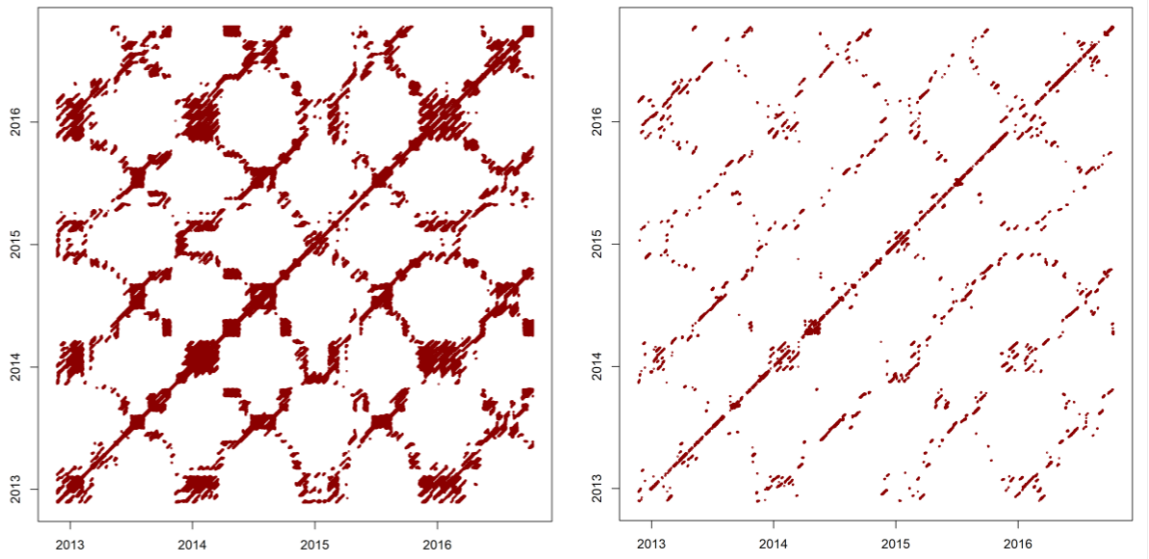

Figure 3: An example of simple recurrence matrix (left) and FAN recurrence matrix (right). The FAN recurrence matrix was created from the simple one.

It is possible to select with the mouse an area of the recurrence plot and zoom it. The user has to mark points: the selected area will be a square encompassing the four points. The user can

correct the position of the points: only the last four will remain. Once marked the points, close the plot window and the zoomed square will appear on another plot.

Recurrence plots can be saved to *png* or *tiff* files; the user must specify the width and height of the image in centimeters and the resolution in ppi.

Plots containing too many points (more than 50000) will fail probably, producing an empty pop-up window. Even though, they can be plotted directly to a *png* or *tiff* file.

The user can select several graphical parameters (see Appendix IV).

Distance plot

A distance plot can be defined as an unthresholded recurrence plot. It represents the distance in the phase space between every pair of points of a possibly embedded time series by means of a color scale. The available distances are the Euclidean and the infinity norm (recurrence matrices are usually built with the infinity norm).

Recurrence plots can be saved to *png* or *tiff* files; the user must specify the width and height of the image in centimeters and the resolution in ppi.

Optionally, the user can choose several graphical parameters (see Appendix IV).

For further information, see March *et al.* (2005).

RP

The probability of recurrence is a measurement of synchronization based on recurrence matrices. In appearance, RP plots are similar to correlation function plots: bar plots where lags are represented in the axis of abscissas and the probability of recurrence, in the axis of ordinates. The RP maxima of two synchronized systems are located at the same lags. This is quantified by the cross probability of recurrence (CRP), which ranges from -1 to 1. Correlation between the amplitudes of the maxima indicates that there is probably a causal relationship between the systems (although, the reverse statement is not necessarily true).

The RP computation is based on the distribution of diagonal lines in the recurrence plot. Lines very close to the main diagonal can produce inflated correlations, especially when no Theiler's window was applied to the matrices. On another hand, diagonals located too far from the main diagonal are too short (because the matrix is not infinite; it has borders that limit the length of these diagonals). As a consequence, in order to estimate correctly the CRP, it is necessary to discard the firsts and the lasts lags.

The inputs for the RP function are two recurrence matrices and the minimum and maximum lags to compute the CPR. KarsTS returns the RP plots and the CRP value. When a single recurrence matrix is entered, KarsTS returns only its RP plot.

For further information, see Marwan *et al.* (2007).

6.5.. Filling

The inputs are time series and, optionally, a gap set to fill. If no gap set is selected, KarsTS will try to fill all the gaps in the time series. On the Output Window will appear a summary that includes the inputs used, the gaps that could be filled and the ones that remain missing.

Interpolations

The output is the filled time series, which will be named after the one with NAs, adding the suffix "_lin" (linear interpolation), "spl" (spline interpolation) or "_sti" (Stineman's interpolation,).

They are based on the packages *zoo* (Wood, 2017) and *stinepack* (Johannesson *et* Bjornsson, 2012).

Optionally, the interpolation can be position-wise (that is, for a missing value, only the values corresponding to the same position in other periods will be used for interpolation). This is useful for gaps greater than the minimum period of the time series. By default, the period is set to 1, that is, convectional interpolation (not position-wise).

ARIMA

This is a univariate filling method that combines a seasonal ARIMA model and a Kalman smoother. The time series needs to be linear, but KarsTS linearity tests do not support missing values. A solution is to perform a temporary filling and to test for linearity. Another option is to try to fill the gaps using the ARIMA button and run the linearity test *a posteriori*.

Necessary inputs

- Time series to fill
- Period (in lags)
- ARIMA parameters: The "Suggest ARIMA parameters" button fills the entries corresponding to the ARIMA parameters automatically; most times, this is the easiest method to perform the ARIMA filling.

Optional inputs

- Gap set: if not provided, KarsTS will build a gap set encompassing all NAs in the selected time series.

Outputs

- Filled time series: It bears the name of the original time series and the suffix "ark", which stands for **AR**IMA and **K**alman.
- Summary on the output window, including:
    - Period
    - ARIMA model: (AR,I,MA)x(seasonal AR, seasonal I, seasonal MA)
    - Time series
    - Gap set
    - Gap set length
    - Number of remaining gaps (within the gap set)

- A table containing the filled gaps

This function is based on the *auto.arima* and the *Arima* functions from package *forecast* (Hyndman *et* Khandakar, 2008).

Position-wise Mean Value

The mean diurnal variation interpolation (MDV) is a filling method used in the field of eddy covariance measurements (Moffat *et al.*, 2007). The position-wise mean value is a generalization of the MDV that can be applied to any time series having a very regular seasonality.

Each missing value is replaced by the mean (or median) of the values located in equivalent positions in the periods around. In case none of these values is available, the central missing value cannot be filled. When some of them are available, the central missing value will be filled or not, depending on the threshold set by the user (minimum number of observations to each side). The process can be applied iteratively, so that the filled time series is the input for the next iteration. Consider using a small number of iterations because filling missing values does not actually increase the amount of information.

Note that a missing value can be replaced by the median or mean of the surrounding points by setting the period to one.

Necessary inputs

- Time series to fill
- Period (in lags)
- Number of periods around: total number of periods to both sides of the period to which the missing value belongs. For example, when the number of periods around is six, the window consists of three periods before and three periods after the central one. Note that the number of periods must be even; otherwise KarsTS will take the next even integer.
- Maximum number of iterations

Optional inputs

- Gap set: if not provided, KarsTS will build a gap set encompassing all NAs in the selected time series.
- Minimum number of observations (one side): when the number of NAs, at least, at one side of the window exceeds this minimum number of observations, KarsTS does not fill the central NA. This avoids the introduction of a biased filling due to an asymmetric distribution of NAs around the NA to fill. The minimum number of iterations defaults to zero.
- Statistic: mean or median.

Outputs

- Filled time series: It bears the name of the original time series and the suffix “mvl”.
- Summary on the output window, including:
    - Period
    - Periods around
    - Minimum number of observations to each side
    - Number of NAs filled in each iteration
    - Time series
    - Gap set
    - Gap set length
    - Number of remaining gaps (within the gap set)
    - A table containing the filled gaps

Multiv. Splines (Multivariate splines)

This multivariate filling method acts on each gap separately. In the first place, it fits a generalized linear model (GAM), where the variable to fill is the response variable. The model is fit in a window around the gap, not in all the time series. The user defines the length of the window in terms of % of the gap length. For example, the gap to fill has 200 missing values. If the user enters “90”, the window will include 180 measurements before the gap and 180 after it. The percentage can be greater than 100. For example, if it is 900 %, the window will include 1800 values before the gap and 1800 after. Note that the same percent will produce windows adapted to the length of the different gaps in the gap set.

The user has to choose between a fixed d.f. regression spline or a penalized regression spline. The allowed smoothing basis are cr (cubic regression spline), tp (thin plate regression spline) or any of the basis allowed in the function *te* from package *mgcv* (Wood, 2017).

MissForest

MissForest is a multivariate filling method from the package *missforest* (Stekhoven, 2013). It performs a non-parametric missing value imputation using a random forest algorithm. It does not require the time series to meet any conditions. The process can be time-consuming, especially when several iterations are performed. The user must enter the maximum number of iterations and the number of trees to grow.

The input time series contain the information needed to fill the gaps. The output time series are the ones to be filled and they are a subset of the input series.

The time series need to be simultaneous and have the same sampling period.

Optionally, the variables can be scaled or embedded. The default setting is not to embed, that is, embedding dimension 1 and delay 0.

Twins

The Twins Filling Method can be univariate or multivariate (see Twins).

The principal inputs are the time series to fill and a recurrence matrix, which can be simple or joint.

When the time series to fill was used to create the recurrence matrix, the twins of the missing points themselves cannot be known. In this case, KarsTS uses the twins of the points close to the missing ones (up to a maximum distance).

It is more advisable to create the recurrence matrix using time series other than the one to fill; in this way, the missing values will be filled regardless of how many missing values surround them.

ARIMAX

It fills gaps using an ARIMA model with external regressor variables (ARIMAX). The user has to choose on or more time series as regressors; apart from this, it works exactly as the ARIMA button.

Check filling

This button tests the appropriateness of the filling. It compares the observed and the predicted values. Therefore, it can only be applied to artificial gaps.

Inputs

- The time series without any artificial gaps
- The time series with artificial gaps
- The time series with filled artificial gaps

It performs a linear fitting between the observed and the predicted values. The summary of the fitting is written on the Output window. The function also produces two plots to check the goodness of the fit.

This button will lead to reject the filling only when it differs greatly from the original values; therefore, it is recommendable to verify the goodness of the filling by plotting the filled time series and the original one (button Plot ts) too.

7.. References

Adler, D. and D. Murdoch (2018). rgl: 3D Visualization Using OpenGL. R package version 0.99.9.

Agostinelli, C. and U. Lund (2017). R package 'circular': Circular Statistics (version 0.4-93).

Alavi, N., J. S. Warland and A. A. Berg (2006). "Filling gaps in evapotranspiration measurements for water budget studies: Evaluation of a Kalman filtering approach." Agricultural and Forest Meteorology **141**: 57-66.

Amritkar, R. and P. P. Kumar (1995). "Interpolation of missing data using nonlinear and chaotic system analysis." Journal of Geophysical Research: Atmospheres **100**(D2): 3149-3154.

Cao, L. (1997). "Practical method for determining the minimum embedding dimension of a scalar time series." Physica D: Nonlinear Phenomena **110**(1-2): 43-50.

Di Narzo, A. and F. Di Narzo (2013). tseriesChaos: Analysis of nonlinear time series. R package version 0.1-13.

García, C. A. and (2015). nonlinearTseries: Nonlinear Time Series Analysis. R package version 0.2.3.

Grosjean, Ph. (2017). SciViews: A GUI API for R. UMONS, Mons, Belgium. URL http://www.sciviews.org/SciViews-R.

Hafen, R. (2016). stlplus: Enhanced Seasonal Decomposition of Time Series by Loess. R package version 0.5.1.

Hyndman, R. J. (2017). forecast: Forecasting functions for time series and linear models. R package version 8.2.

Hyndman, R. J. and Y. Khandakar (2008). "Automatic time series forecasting: the forecast package for R." Journal of Statistical Software **27**(3): 1-22.

Ismail, M. and V. Ciesielski (2003). An Empirical Investigation of the Impact of Discretization on Common Data Distributions. Department of Computer Science. Melbourne RMIT University. Master of Technology (Information Technology).

Johannesson, T., H. Bjornsson and G. Grothendieck (2012). stinepack: Stineman, a consistently well behaved method of interpolation. R package version 1.3.

Korkmaz, S., D. Goksuluk and G. Zararsiz (2014). "MVN: an R package for assessing multivariate normality." The R Journal **6**(2): 151-162.

March, T. K., S. C. Chapman and R. O. Dendy (2005). "Recurrence plot statistics and the effect of embedding." Physica D **200**: 171–184.

Marwan, N. (2011). "How to avoid potential pitfalls in recurrence plot based data analysis." International Journal of Bifurcation and Chaos **21**(4): 1003-1017.

Marwan, N. and J. Kurths (2002). "Nonlinear analysis of bivariate data with cross recurrence plots." Physics Letters A **302**: 299–307

Marwan, N., M. C. Romano, M. Thiel and J. Kurths (2007). "Recurrence plots for the analysis of complex systems." PHYSICS REPORTS **438**: 237 – 329

Moffat , A. M., D. Papale, M. Reichstein, D. Y. Hollinger, A. D. Richardson, A. G. Barr, C. Beckstein, B. H. Braswell, G. Churkina, A. R. Desai, E. Falge, J. H. Gove, M. Heimann, D. Hui, A. J. Jarvis, J. Kattge, A. Noormets and V. J. Stauch (2007). "Comprehensive

comparison of gap-filling techniques for eddy covariance net carbon fluxes." Agricultural and Forest Meteorology **147**: 209-232.

Moritz, S., A. Sardá, T. Bartz-Beielstein, M. Zaefferer and J. Stork (2015). "Comparison of different Methods for Univariate Time Series Imputation in R." arXiv preprint arXiv:1510.03924.

Nau, R (2017). Statistical forecasting: notes on regression and time series analysis: ARIMA models for time series forecasting. https://people.duke.edu/~rnau/411home.htm [17-05-2017]

Patrick, E. M. (2014). infotheo: Information-Theoretic Measures. R package version 1.2.0.

R Core Team (2020). R: A language and environment for statistical computing. Vienna, Austria, R Foundation for Statistical Computing.

RStudio Team (2019). RStudio: Integrated Development for R. Version 1.1.383. Boston, MA, RStudio, Inc.

Sáez, M., Pla, C., Cuezva, S., & Benavente, D. (2019). KarsTS: an R package for microclimate time series analysis. *Earth Science Informatics*, *12*(4), 685-697.

Soetaert, K. (2017). plot3D: Plotting Multi-Dimensional Data. R package version 1.1.1.

Stekhoven, D. J. (2013). missForest: Nonparametric Missing Value Imputation using Random Forest. R package version 1.4..

Stekhoven, D. J. and P. Bühlmann (2011). "MissForest—non-parametric missing value imputation for mixed-type data." Bioinformatics **28**(1): 112-118.

Tierney, L. (2011). tkrplot: TK Rplot. R package version 0.0-23.

Trapletti, A. and K. Hornik (2017). tseries: Time Series Analysis and Computational Finance. R package version 0.10-42.

Wettenhall, J. and P. Grosjean. (2017). "Interactive plots with tkrplot." Sciviews. Reproductible research with R Retrieved 20 September, 2017, from http://www.sciviews.org/recipes/tcltk/TclTk-interactive-plots/.

Wood, S. N. (2017). Generalized additive models: an introduction with R, CRC press.

Zeileis, A. and G. Grothendieck (2005). "zoo: S3 Infrastructure for Regular and Irregular Time Series." Journal of Statistical Software **14**(6).

Zhao, P., L. Xingb and J. Yuc (2009). "Chaotic time series prediction: from one to another." Physics Letters A **373**: 2174–2177.

Zhao, X. and Y. Huang (2015). "A comparison of three gap filling techniques for eddy covariance net carbon fluxes in short vegetation ecosystems." Advances in Meteorology **2015**.

Appendix I: Functionalities

Analyze the distribution of sampling frequencies and NAs in a time series *(Sampling periods)*

Apply an artificial set of gaps to a time series *(Apply gaps to time series)*

Build a time series of daily, monthly... values *(Aggregate)*

Calculate simple, cross and joint recurrence matrices *(Recurrence Matrix, Cross Recurrence Matrix and Joint Recurrence Matrix)*

Calculate the basic statistics of a time series *(Basic Statistics)*

Calculate the basic statistics of a time series in sliding windows *(Rolling Statistics)*

Calculate the reciprocal or the inverse time series *(Operations)*

Check if a time series follows an AR or ARIMA process *(Linearity Tests)*

Check if two time series are correlated *(Linear correlation and Mutual Information)*

Create a set of random artificial gaps *(Artificial Random Gaps)*

Create a specific artificial gap *(Specific Random Gaps)*

Cumulative sum of a time series *(Cumulative sum)*

Cut a piece of time series *(Select by Rg/Freq)*

Differencing a time series *(Differencing)*

Duplicate data sets *(Rename)*

Estimate the correlation dimension (fractal dimension of an object embedded in a phase space) *(Invariants)*

Estimate the correlation sum (probability of finding a neighbor in a ball of radius r in the phase space) *(Invariants)*

Estimate the delay to embed a time series *(AMI)*

Estimate the self-repeating rate (a measure of the determinism of a time series) *(Self-repeating rate)*

Estimate the embedding dimension of a time series *(False Nearest Neighbors, Ed(1)&Ed(2 and Invariants)*

Estimate the KS Entropy (measure of complexity) *(Invariants)*

Estimate the laminarity *(Laminarity)*

Estimate the length of the diagonal lines (predictability) *(Self-repeating rate)*

Estimate the radius necessary to create a recurrence matrix from a time series *(Invariants)*

Estimate the ratio (points belonging to diagonal lines / points in the recurrence matrix) *(Self-repeating rate)*

Estimate the recurrence rate (number of ones/ number of points in the recurrence matrix) *(Self-repeating rate and Laminarity)*

Estimate the trapping time *(Laminarity)*

Export data sets to .txt, .csv, and .mat files *(Export)*

Extract the seasonal component of a time series (the component can change over the time) *(Loess decomp.)*

Extract the trend of a seasonal time series *(Loess decomp.)*

Filling methods:

Univariate

ARIMA model *(ARIMA)*

Mean value *(Mean Value)*

Stinemann's, linear and spline interpolations *(Stinemann's interpolation, linear interpolation and spline interpolations)*

Twins *(Twins)*

Multivariate

ARIMAX model (*ARIMAX*)

Forest random algorithm *(Missforest)*

Generalized additive model (*Multiv. Splines*)

Twins *(Twins)*

Filter outliers causing unusually stiff slopes *(Rm. Slope Outliers)*

Find non-linear correlations or autocorrelations *(Mutual information)*

Get the dates, values and indices of one or more points in a plot *(Get coordinates)*

List the data sets to the Output Window along with a short description of each *(List)*

Load data sets (time series, multiple time series, gaps, recurrence matrices) *(Load)*

Merge consecutive pieces of time series *(Merge)*

Merge gap sets *(Merge)*

Perform a linear regression between the real values and the filled artificial gaps *(Check filling)*

Perform a loess seasonal decomposition on a time series *(Loess decomp.)*

Perform a number of linearity tests (surrogate test, Teraesvirta's, White neural, Keenan's one-degree, McLeod-Li test, Tsay's Test and Likelihood ratio) *(Linearity Tests)*

Perform a number of stationarity tests (Philipp-Perron unit root test, Box-Ljung test, Augmented Dickey-Fuller test and KPSS test for Level Stationarity) *(Stationarity Tests)*

Plot one or more time series *(Plot ts)*

Plot phase 2 or 3 dimensional phase portraits *(Phase portraits)*

Plot simple, cross and joint recurrence matrices *(Recurrence Plot, Cross Recurrence Plot and Joint Recurrence Plot)*

Plot the average mutual information of two time series *(Mutual Information)*

Plot the histogram of the values of a time series *(Histogram)*

Plot the linear autocorrelation and partial-autocorrelation of a time series *(Linear Correlation)*

Plot the linear correlation and partial-correlation of two time series *(Linear Correlation)*

Plot the self-average mutual information of a time series *(Mutual Information)*

Re-compose a time series (additive or multiplicative) after doing some changes to each component *(Operations)*

Remove data sets from the working environment *(Remove)*

Remove outliers manually (on a plot) *(Remove points)*

Rename data sets *(Rename)*

Resample (downsample) a time series (or a piece of it) *(Cut and resampling)*

Resample (upsample) a time series *(Upsample)*

Round a time series to a number of decimal positions *(Round)*

Round a time series to a number of significant digits *(Round)*

Save data sets to the hard disk *(Save)*

Save the output panel content to a txt file (Save to txt)

Scale a time series *(Scale)*

Select a subset of existing NAs in a time series *(Select gaps)*

Smooth a non-seasonal time series using loess (you might take on account other variables to do so) *(Loess smooth.)*

Smooth a seasonal time series *(Loess Decomp and Compose, successively)*

Smooth a time series by averaging the data (ex. daily, monthly...) *(Aggregate)*

Smooth a time series using the mean or the median in sliding windows *(Rollings Statistics)*

Sum up or multiply two or more time series *(Operations)*

Take natural logarithm of a time series *(Operations)*

Two or three-dimensional scatter plots *(Phase portraits)*

Appendix II: Input files format

KarsTS admits the following extensions: .R, .RData, .csv y .txt. Often, time series are stored originally in csv or text files, whilst gap sets and recurrence matrices use to be created with KarsTS. The first two types of file can store multiple data sets. However, cvs or txt files can store only a single matrix, gap set, univariate time series or multivariate time series.

Time series:

Times series can be univariate or multivariate; anyway, times must be located in the first column.

.R or .RData. files

The files can contain several time series, gap sets and recurrence matrices. Each time series is a data frame with two columns: time and values. Times are POSIXct POSIXt dates with R default format (Y-m-d H:M). KarsTS will introduce as many NAs as possible, in order that the difference between two consecutive dates is unique through the whole time series.

.csv files

They can contain univariate or multivariate time series; in any case, times must be placed in the first column. The rest of the columns can contain numbers, NA (missing value) or empty cell (missing value, as well).

The first column of the first row is reserved for a title (for example, thisPlaceData). The second row must contain the variable names (for example, “time”, “temperature”, ”Rn” etc.). When a csv file is loaded, KarsTS decomposes the multivariate time series into univariate ones, assigning them those names (“temperature”, ”Rn” etc.). The loaded time series are R data frames as described in the previous section.

When KarsTS exports a time series to a csv file, the rows containing only missing values are not written in order to save space.

	A	B	C	D
1	ts_examples			
2	time	ts_example	ts_example2	
3	11/22/2012 13:30	289.3243	289.2559	
4	11/22/2012 19:30	289.1798	289.2764	
5	11/23/2012 1:30	289.1922	289.2558	
6	11/23/2012 7:30	289.1612	289.2882	
7	11/23/2012 13:30	289.2083	289.2643	
8	11/23/2012 19:30	289.2007	289.2352	
9	11/24/2012 1:30			
10	11/24/2012 7:30			
11	11/24/2012 13:30			
12	11/24/2012 19:30			
13	11/25/2012 1:30	289.263	289.2095	
14	11/25/2012 7:30	289.2571	289.3084	
15	11/25/2012 13:30	289.2435	289.2585	

	A	B	C	D
1	ts_examples			
2	time	ts_example	ts_example2	
3	11/22/2012 13:30	289.3243	289.2559	
4	11/22/2012 19:30	289.1798	289.2764	
5	11/23/2012 1:30	289.1922	289.2558	
6	11/23/2012 7:30	289.1612	289.2882	
7	11/23/2012 13:30	289.2083	289.2643	
8	11/23/2012 19:30	289.2007	289.2352	
9	11/24/2012 1:30	NA	NA	
10	11/24/2012 7:30	NA	NA	
11	11/24/2012 13:30	NA	NA	
12	11/24/2012 19:30	NA	NA	
13	11/25/2012 1:30	289.263	289.2095	
14	11/25/2012 7:30	289.2571	289.3084	
15	11/25/2012 13:30	289.2435	289.2585	

	A	B	C	D
1	ts_examples			
2	time	ts_example	ts_example2	
3	11/22/2012 13:30	289.3243	289.2559	
4	11/22/2012 19:30	289.1798	289.2764	
5	11/23/2012 1:30	289.1922	289.2558	
6	11/23/2012 7:30	289.1612	289.2882	
7	11/23/2012 13:30	289.2083	289.2643	
8	11/23/2012 19:30	289.2007	289.2352	
9	11/25/2012 1:30	289.263	289.2095	
10	11/25/2012 7:30	289.2571	289.3084	
11	11/25/2012 13:30	289.2435	289.2585	
12	11/25/2012 19:30	289.2681	289.215	
13	11/26/2012 1:30	289.1932	289.2371	
14	11/26/2012 7:30	289.2383	289.2062	
15	11/26/2012 13:30	289.2855	289.2374	

Figure 2: Examples of time series in csv files. All the options are equivalent: missing values represented by empty cells, NA or missing rows.

txt files

Times must occupy the first two columns: year, month and day in the first column; hours and minutes in the second one. Apart from this, they work in the same way csv files do.

Gap sets

.R or .RData files

Each gap set is a list containing the following elements:
$gaps: rows in the original (univariate) time series that correspond to missing values
$tsIni: original time series initial time
$tsEnd: original time series final time
$samPerMin: original time series sampling period (minutes)
$tsLength: number of elements in the original time series.
$tsName: original time series name

.csv files

The elements explained above are displayed in columns. The first column of the first row must contain the gap set name; the second one, the columns names (gaps, tsIni etc.)

.txt files

The initial and final dates are stored in two columns each one. Apart from this, they work in the same way csv files do.

Recurrence matrices

Recurrence matrices contain ones and zeros (and NAs). As recurrence matrices can be huge, KarsTS stores only those positions corresponding to ones. Besides, simple or joint recurrence matrices are symmetric, hence only the upper triangle is stored. Note that in a recurrence matrix the main diagonal is the line X=Y; the lower triangle is the one located to the left and the upper triangle lies to the right.

.R o .RData files

Each recurrence matrix is a list. Besides the positions corresponding to ones, it contains additional information about the time series from which it comes. Note that simple recurrence matrices come from a single time series, cross recurrence matrices from two time series and joint recurrence matrices from two or more time series.
$ones: data frame with two columns. The first one refers to the ones matrix columns or X coordinates in the corresponding recurrence plot. The second one contains the ones rows or Y positions.
$tol: tolerances used to create the recurrence matrix
$tsName: original time series names
$embDim: embedding dimensions used to create the matrix
$delay: delays used to create the matrix
$dist: distances used to create the matrix
$tsLength: original time series lengths
$samPerSec: original time series sampling periods
$tsIni: original time series initial date
$type: type of matrix; it can be “simple” ,“cross” o “joint”

csv files

The elements described above are stored in columns. The matrix name appears in the first column of the first row; the columns names appear in the second row (that is, X, Y, tol, tsName, etc.)

txt files

The initial Date is displayed in two columns.

While KarsTS is performing a costly calculation, the interface might not be visualized correctly. When it is over the interface will recover its normal appearance It might be necessary to drag the main window a little, though. KarsTS will warn you of potentially costly procedures.

Loading frequent problems

The most frequent problems when loading time series files created with other programs are the following.

1. Some date is in a wrong place or it is repeated. Note that the time of the time series must be strictly increasing (nevertheless, there can be different sampling periods and time jumps).
2. The date format is not allowed by KarsTS. The allowed formats are:
   m/d/y H:M, d/m/y H:N, Y/m/d H:M, Y/d/m H:M, m-d-y H:M, d-m-y H:N, Y-m-d H:M, Y-d-m H:M.
3. The date format is not unique. Whether the new format is allowed or not, the loading will fail. Note that the date format can change in a distant part of the file.
4. When a row lacks its date, KarsTS will remove the whole row. Therefore, if the file has empty rows, the user does not need to remove them (unless they are located in the first or the second rows, which must contain the general title and the column titles respectively).
5. Some data columns have values that can be interpreted as text:
   a. Symbols other than NA to indicate missing value (for example, --)
   b. Cells in scientific notation (They might be in a part of the file the user is not watching)
   c. Cells with a mistaken separator (for example, 12,5 in a file where the decimal separator is the point)
6. There is something written in other columns
7. If you .csv file has the date (year, month, day) in a column and the time (hours, minutes) in another one, you can sum them up in a single column. Alternatively, you can export the file to txt.
8. The variables names are in the first row, not in the second. The loading will not fail, but the imported time series will have numbers as names.
9. The time separations are not compatible amongst them. For example: 2004-12-5 00:30, 2004-12-5 01:00, 2004-12-5 01:30,2004-12-5 01:32, 2004-12-5 02:02, 2004-12-5 02:32, 2004-12-5 03:02. In this case, it is necessary to modify it so that the result is:

2004-12-5 00:30, 2004-12-5 01:00, 2004-12-5 01:30, 2004-12-5 01:30, 2004-12-5 02:00. 2004-12-5 02:30, 2004-12-5 03:00.

10. The file was created in a computer with a different date format and the user's Excel interprets wrongly the dates. As a result, the file has more than one date format or the dates are not consecutive.

Appendix III: Filling methods overview

Univariate

Interpolations

Interpolation is not a time-series specific method, that is, it does not account for the autocorrelations. However, it is fast and it can be used on gaps smaller than one period. The Stineman's interpolation is similar to the splines, but it is less sensitive to noise.

KarsTS offers the possibility of position-wise interpolations, to extend the use of these methods to gaps somewhat greater than the minimum period.

ARIMA

Fits an ARIMA model to the time series and uses it to predict the missing values. The time series needs to meet some requirements. Linearity tests (as well as correlations and stationarity tests) are useful to verify whether an ARIMA model is appropriate for the time series. The program can suggest values for the ARIMA parameters, otherwise you can estimate them using the classical procedure (for a clear explanation, see Nau (2017)).

Position-wise mean value

This method requires a time series with a clear, regular period.

Each missing value is replaced by the mean (or median) of the values in equivalent places in the nearby periods. For example, in a time series with daily seasonality, a missing value at 15:00 will be replaced with the mean of surrounding days values at the same hour at 15:00.

The number of periods to use to calculate the median is chosen by the user. If you choose too many periods, the missing value probably will stop being related to the far away values (consider that the time series can have a trend or a change in the period amplitude). If you choose too few periods, maybe some of them have missing values too.

Multivariate

Missing Random Algorithm

A missing random algorithm is used to perform a multiple imputation. The algorithm classifies the multivariate values, thus producing a number of trees. Once the trees are trained, they are used to predict the values. Finally, the results of the different trees are pooled. The function missForest, by itself, does not account for periodicity because it is not time-series specific; however KarsTS allows the user to introduce embedded time series.

Twins

Twins are points that represent the same state of a dynamical system. They are widely used to generate surrogate time series (Marwan, 2007); however, we use them to fill missing values.

The successive states of the system are given by its trajectory through time. Imagine a jet aircraft flying. The state of the jet at every time is given by three coordinates and the jet wake represents all these states simultaneously. Now consider a cave whose microclimate can be described by the CO_2 concentration and the temperature; fig. 4 represents the trajectory of such system. The time-CO_2 projection is the CO_2 time series plot; the time-temperature projection is the temperature time series. Finally, when we remove the time, we say we work in the *phase space*. Thus, the temperature-CO_2 projection is the *phase portrait* of the system, where we see all the states simultaneously, as in the jet wake.

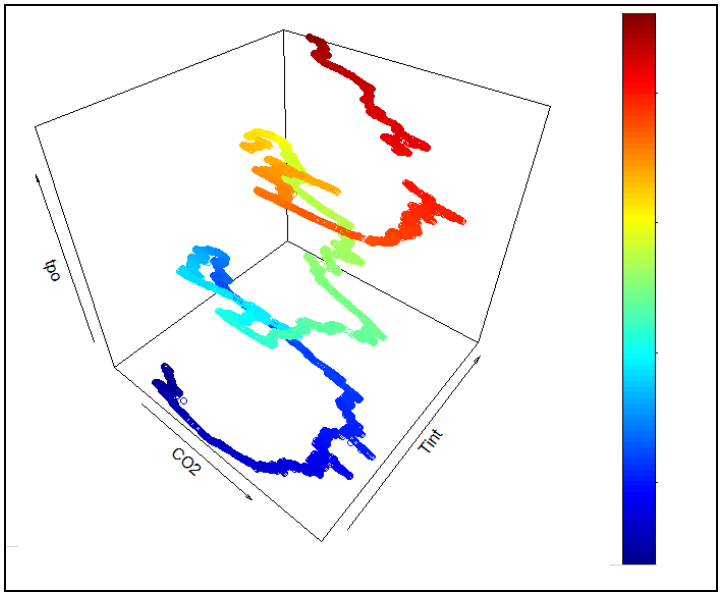

Figure 4: Example of trajectory

In order to represent the same state of a system, two points of a time series must have the same value (in the previous example, the same CO_2 concentration and the same temperature simultaneously). In practice, *the same value* means that their difference is less than a threshold. In a recurrence matrix, it means that they are ones in a column. However, this condition is not sufficient. Twin points are so close that they share also their neighborhood (they have the same points around). Therefore, twin points correspond to identical columns in the recurrence matrix. Note that not all points in a time series have necessarily twins.

Filling with twins

This method is based on finding twin points in a simple or joint recurrence matrix. Each missing value is filled independently from the others; therefore we will explain the process for a single missing value.

Case 1: joint recurrence matrix; the time series to fill was not used to create it.

Consider:

- the time series to fill (Ts.F)
- m predictor time series (Ts.1, Ts.2..., Ts.m)
- a joint recurrence matrix, JRM, created with the m predictor time series
- the position to fill, i. Ts.F (i) is missing and Ts.1 (i), Ts.2 (i)..., Ts.m (i) are known.

In the first place, the function finds the twins of the ith column in the recurrence matrix JRM. Let $j_1,j_2,...,j_n$ be the twin columns. Note that the columns i, $j_1,j_2,...,j_n$ are identical because they represent twin points (points having the same value and the same neighbors in the phase space). Then, the missing value is replaced by the median of Ts.F (j_1), Ts.F (j_2),…, Ts.F (j_h):

$$Ts.F\ (i) = median\ (Ts.F\ (j_1),\ Ts.F\ (j_2),\ldots,\ Ts.F\ (j_h))$$

Case 2: simple recurrence matrix; the time series to fill was not used to create it.

It is a particular case of the previous one, with m=1.

Case 3: joint recurrence matrix; the time series to fill was used to create it.

Consider:

- the time series to fill (Ts.F)
- m predictor time series (Ts.1, Ts.2..., Ts.m)
- a joint recurrence matrix, JRM, created with Ts.F and the m predictor time series
- the position to fill, i. Ts.F (i) is missing and Ts.1 (i), Ts.2 (i)..., Ts.m (i) are known.

The ith column in JRM lacks information because Ts.F (i) is missing and Ts.F was used to build JRM. Therefore, we find the twins of the previous column, that is, the (i-1)th column. Let $j_1,j_2,...,j_n$ be the twin columns. Now, we use Ts.F (j_1+1), Ts.F (j_2+1),…, Ts.F (j_h+1) to replace the missing value:

$$Ts.F\ (i) = median\ (Ts.F\ (j_1+1),\ Ts.F\ (j_2+1),\ldots,\ Ts.F\ (j_h+1))$$

In case, Ts.F (i-1) is also missing, we use the (i-2)th column and replace the missing value by Ts.F (i) = median (Ts.F (j_1+2), Ts.F (j_2+2),…, Ts.F (j_h+2)).

The process continues up to a maximum distance, D, chosen by the user.

Case 4: simple recurrence matrix; the time series to fill was used to create it.

It is a particular case of case 3, with m=1.

Note that when the time series to fill was used to create the recurrence matrix, each gap is filled from the outside in. However, when the time series to fill was not used to create the recurrence matrix, each NA can be filled no matter by how many more NA is surrounded. Note also that in case 1, D can be > 0, but it is not necessary.

Appendix IV: Graphical parameters

X label

Y label

Z label: only available for 3D phase portraits

Labels size: relative to 1

Ticks size: relative to 1

Point/line

Color: the name of the color (for example: red, blue...). In phase portraits, the color "rainBow" results in a color scale that facilitates the visualization of the trajectories. In distance plots, there are two color scales: magenta-yellow-cyan and grey scale.

Line width: relative to 1

Point size: relative to 1

Plot diagonal: in symmetric recurrence plots, the diagonal can be plotted or not.

Left margin: relative to 1

Lower margin: relative to 1

Right margin: relative to 1

Ticks location: controls the distance of the ticks to the axis.

Labels location: controls the distance of the label to the axis.

Minimum distance: for distance plots, minimum distance to be considered

Maximum distance: for distance plots, maximum distance to be considered

Color levels: number of levels for the color scale in distance plots

White outside the range: controls how distances lower than the minimum and higher than the maximum are represented in the distance plot: in white (both) or in color (magenta and cyan, respectively).

Table 6: Graphical parameters available for different types of plots

	Plot ts	Phase portraits+	Rec. plot	Cross rec. plot	Distance plot
X label					
Y label					
Z label					
Labels size					
Ticks size					
X Scale					
Y Scale					
Point/line					
Color					
Line width					
Point size					
Plot diagonal					
Left margin					
Lower margin					
Right margin					
Ticks location					
Labels location					
Min. distance					
Max. distance					
Color levels					
White outside the range					