Package: memoria (via r-universe)

June 10, 2026

Title Quantifying Ecological Memory in Palaeoecological Datasets and Other Long Time-Series

Version 1.1.0

Description Quantifies ecological memory in long time-series using Random Forest models ('Benito', 'Gil-Romera', and 'Birks' 2019 <doi:10.1111/ecog.04772>) fitted with 'ranger' (Wright and Ziegler 2017 <doi:10.18637/jss.v077.i01>). Ecological memory is assessed by modeling a response variable as a function of lagged predictors, distinguishing endogenous memory (lagged response) from exogenous memory (lagged environmental drivers). Designed for palaeoecological datasets and simulated pollen curves from 'virtualPollen', but applicable to any long time-series with environmental drivers and a biotic response.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

Imports ggplot2, ranger, zoo, rlang

Suggests spelling, testthat

URL https://blasbenito.github.io/memoria/

Language en-US

Repository https://blasbenito.r-universe.dev

Date/Publication 2026-02-10 12:56:58 UTC

RemoteUrl https://github.com/blasbenito/memoria

RemoteRef HEAD

RemoteSha 4291380964f464cc44670f83f8dc5695a919ff2b

Contents

`alignTimeSeries` *Align and join multiple time series to a common temporal resolution*

Description

Aligns multiple time series datasets to a common temporal resolution using LOESS interpolation and joins them into a single dataframe. This is useful when combining datasets with different sampling intervals.

Usage

```
alignTimeSeries(
  datasets.list = NULL,
  time.column = NULL,
  interpolation.interval = NULL
)

mergePalaeoData(
  datasets.list = NULL,
  time.column = NULL,
  interpolation.interval = NULL
)
```

Arguments

`datasets.list`	list of dataframes, as in `datasets.list = list(dataset1 = df1, dataset2 = df2)`. The provided dataframes must have a time column with the same column name and the same units of time. Non-numeric columns in these dataframes are ignored. Default: `NULL`.

`time.column` character string, name of the time column of the datasets provided in `datasets.list`. Default: `NULL`.

`interpolation.interval` numeric, temporal resolution of the output data, in the same units as the time columns of the input data. Default: `NULL`.

Details

This function fits a `loess` model of the form `y ~ x`, where `y` is any numeric column in the input datasets and `x` is the column given by the `time.column` argument. The model is used to interpolate column `y` on a regular time series of intervals equal to `interpolation.interval`. All numeric columns in every provided dataset go through this process to generate the final data with samples separated by regular time intervals. Non-numeric columns are ignored and absent from the output dataframe.

Value

A dataframe with every column of the initial dataset interpolated to a regular time grid of resolution defined by `interpolation.interval`. Column names follow the form datasetName.columnName, so the origin of columns can be tracked.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

See Also

Other data_preparation: `lagTimeSeries()`

Examples

```
#loading data
data(pollen)
data(climate)

x <- alignTimeSeries(
 datasets.list = list(
   pollen=pollen,
   climate=climate
 ),
 time.column = "age",
 interpolation.interval = 0.2
 )
```

`climate` *Dataframe with palaeoclimatic data.*

Description

A dataframe containing palaeoclimate data at 1 ky temporal resolution with the following columns:

Usage

```
data(climate)
```

Format

dataframe with 6 columns and 800 rows.

Details

- *age* in kiloyears before present (ky BP).
- *temperatureAverage* average annual temperature in degrees Celsius.
- *rainfallAverage* average annual precipitation in millimetres per day (mm/day).
- *temperatureWarmestMonth* average temperature of the warmest month, in degrees Celsius.
- *temperatureColdestMonth* average temperature of the coldest month, in degrees Celsius.
- *oxigenIsotope* delta O18, global ratio of stable isotopes in the sea floor, see http://lorraine-lisiecki.com/stack.html for further details.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

See Also

Other example_data: palaeodata, palaeodataLagged, palaeodataMemory, pollen

computeMemory *Quantifies ecological memory with Random Forest.*

Description

Takes the output of prepareLaggedData to fit the following model with Random Forest:

$p_t = p_{t-1} + ... + p_{t-n} + d_t + d_{t-1} + ... + d_{t-n} + r$

where:

- d is a driver (several drivers can be added).
- t is the time of any given value of the response p.
- $t - 1$ is the lag number 1 (in time units).
- $p_{t-1} + ... + p_{t-n}$ represents the endogenous component of ecological memory.
- $d_{t-1} + ... + d_{t-n}$ represents the exogenous component of ecological memory.
- d_t represents the concurrent effect of the driver over the response.
- r represents a column of random values, used to test the significance of the variable importance scores returned by Random Forest.

Usage

```
computeMemory(
  lagged.data = NULL,
  response = NULL,
  drivers = NULL,
  random.mode = "autocorrelated",
  repetitions = 10,
  subset.response = "none",
  num.threads = 2
)
```

Arguments

lagged.data	a lagged dataset resulting from prepareLaggedData. See palaeodataLagged as example. Default: NULL.
response	character string, name of the response variable. Not required if 'lagged.data' was generated with [prepareLaggedData]. Default: NULL.
drivers	a character string or character vector with variables to be used as predictors in the model. Not required if 'lagged.data' was generated with [prepareLaggedData]. **Important:** drivers names must not have the character "__" (double underscore). Default: NULL.
random.mode	either "none", "white.noise" or "autocorrelated". See details. Default: "autocorrelated".
repetitions	integer, number of random forest models to fit. Default: 10.

`subset.response`
: character string with values "up", "down" or "none", triggers the subsetting of the input dataset. "up" only models memory on cases where the response's trend is positive, "down" selects cases with negative trends, and "none" selects all cases. Default: `"none"`.

`num.threads`
: integer, number of cores ranger can use for multithreading. Default: `2`.

Details

This function uses the ranger package to fit Random Forest models. Please, check the help of the ranger function to better understand how Random Forest is parameterized in this package. This function fits the model explained above as many times as defined in the argument `repetitions`.

To test the statistical significance of the variable importance scores returned by random forest, on each repetition the model is fitted with a different `r` (random) term, unless `random.mode = "none"`. If `random.mode` equals "autocorrelated", the random term will have a temporal autocorrelation, and if it equals "white.noise", it will be a pseudo-random sequence of numbers generated with `rnorm`, with no temporal autocorrelation. The importance of the random sequence in predicting the response is stored for each model run, and used as a benchmark to assess the importance of the other predictors.

Importance values of other predictors that are above the median of the importance of the random term should be interpreted as non-random, and therefore, significant.

Value

A list with 5 slots:

- `response` character, response variable name.
- `drivers` character vector, driver variable names.
- `memory` dataframe with six columns:
  - `median` numeric, median importance across `repetitions` of the given `variable` according to Random Forest.
  - `sd` numeric, standard deviation of the importance values of the given `variable` across `repetitions`.
  - `min` and `max` numeric, percentiles 0.05 and 0.95 of importance values of the given `variable` across `repetitions`.
  - `variable` character, names of the different variables used to model ecological memory.
  - `lag` numeric, time lag values.
- `R2` vector, values of pseudo R-squared value obtained for the Random Forest model fitted on each repetition. Pseudo R-squared is the Pearson correlation between the observed and predicted data.
- `prediction` dataframe, with the same columns as the dataframe in the slot `memory`, with the median and confidence intervals of the predictions of all random forest models fitted.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

References

- Wright, M. N. & Ziegler, A. (2017). ranger: A fast implementation of random forests for high dimensional data in C++ and R. J Stat Softw 77:1-17. doi:10.18637/jss.v077.i01.
- Breiman, L. (2001). Random forests. Mach Learn, 45:5-32. doi:10.1023/A:1010933404324.
- Hastie, T., Tibshirani, R., Friedman, J. (2009). The Elements of Statistical Learning. Springer, New York. 2nd edition.

See Also

plotMemory, extractMemoryFeatures

Other memoria: extractMemoryFeatures(), plotMemory()

Examples

```
#loading data
data(palaeodataLagged)

# Simplified call - response and drivers auto-detected from attributes
memory.output <- computeMemory(
 lagged.data = palaeodataLagged,
 random.mode = "autocorrelated",
 repetitions = 10
)

str(memory.output)
str(memory.output$memory)

#plotting output
plotMemory(memory.output = memory.output)
```

---

experimentToTable *Turns the outcome of runExperiment into a long table.*

Description

Takes the output of runExperiment, extracts the dataframes containing the ecological memory patterns generated by computeMemory, and binds them together into a single dataframe ready for further analyses or plotting.

Usage

```
experimentToTable(experiment.output = NULL, parameters.file = NULL)
```

Arguments

`experiment.output`
: list, output of `runExperiment`. Default: `NULL`.

`parameters.file`
: dataframe of simulation parameters. Default: `NULL`.

Details

This function is used internally by `plotExperiment`, but it is also available to users in case they want to do other kinds of analyses or plots with the data.

Value

A dataframe.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

See Also

`runExperiment`, `plotExperiment`

Other virtualPollen: `plotExperiment()`, `runExperiment()`

`extractMemoryFeatures` *Extracts ecological memory features from the output of* `computeMemory`.

Description

Computes the following features of the ecological memory patterns returned by `computeMemory`:

- `memory strength` maximum difference in relative importance between each component (endogenous, exogenous, and concurrent) and the median of the random component. This is computed for exogenous, endogenous, and concurrent effect.
- `memory length` proportion of lags over which the importance of a memory component is above the median of the random component. This is only computed for endogenous and exogenous memory.
- `dominance` proportion of the lags above the median of the random term over which a memory component has a higher importance than the other component. This is only computed for endogenous and exogenous memory.

Usage

```
extractMemoryFeatures(
  memory.pattern = NULL,
  exogenous.component = NULL,
  endogenous.component = NULL,
  scale.strength = TRUE
)
```

Arguments

`memory.pattern`	either a list resulting from `computeMemory`, or a dataframe with memory patterns of several taxa generated by `experimentToTable`. When using output from `experimentToTable`, filter to a specific sampling resolution before calling this function (e.g., `data[data$sampling == 25, ]`). Default: `NULL`.
`exogenous.component`	character string or character vector, name of the variable or variables defining the exogenous component. When `memory.pattern` is output from `computeMemory`, this is automatically extracted from the `$drivers` slot if not provided. Required when input is from `experimentToTable`. Default: `NULL`.
`endogenous.component`	character string, name of the variable defining the endogenous component. When `memory.pattern` is output from `computeMemory`, this is automatically extracted from the `$response` slot if not provided. Required when input is from `experimentToTable`. Default: `NULL`.
`scale.strength`	boolean. If `TRUE`, the strength of the ecological memory components, which has the same units as the importance scores yielded by random Forest (percentage of increment in mean squared error when a variable is permuted), is scaled between 0 and 1. Default: `TRUE`.

Details

Warning: this function only works when only one exogenous component (driver) is used to define the model in `computeMemory`. If more than one driver is provided through the argument `exogenous.component`, the maximum importance scores of all exogenous variables is considered. In other words, the importance of exogenous variables is not additive.

Value

A dataframe with 8 columns and 1 row if `memory.pattern` is the output of `computeMemory` and 13 columns and as many rows as taxa are in the input if it is the output of `experimentToTable`. The columns are:

- *label* character string to identify the taxon. It either inherits its values from `experimentToTable`, or sets the default ID as "1".
- *strength.endogenous* numeric, difference between the maximum importance of the endogenous component at any lag and the median of the random component (see details in `computeMemory`). When `scale.strength = TRUE` (default), values are scaled to [0, 1]; otherwise values are in importance units (percentage of increment in MSE).

- *strength.exogenous* numeric, same as above, but for the exogenous component.
- *strength.concurrent* numeric, same as above, but for the concurrent component (driver at lag 0).
- *length.endogenous* numeric in the range [0, 1], proportion of lags over which the importance of the endogenous memory component is above the median of the random component.
- *length.exogenous* numeric in the range [0, 1], same as above but for the exogenous memory component.
- *dominance.endogenous* numeric in the range [0, 1], proportion of the lags above the median of the random term over which a the endogenous memory component has a higher importance than the exogenous component.
- *dominance.exogenous*, opposite as above.
- *maximum.age*, numeric. As every column after this one, only provided if `memory.pattern` is the output of `experimentToTable`. Trait of the given taxon.
- *fecundity* numeric, trait of the given taxon.
- *niche.mean* numeric, trait of the given taxon.
- *niche.sd* numeric, trait of the given taxon.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

See Also

`computeMemory`

Other memoria: `computeMemory()`, `plotMemory()`

Examples

```
# Loading example data (output of computeMemory)
data(palaeodataMemory)

# Simplified call - components auto-detected from computeMemory output
memory.features <- extractMemoryFeatures(
  memory.pattern = palaeodataMemory
)

# Explicit call - still supported for backwards compatibility
memory.features <- extractMemoryFeatures(
  memory.pattern = palaeodataMemory,
  exogenous.component = c(
    "climate.temperatureAverage",
    "climate.rainfallAverage"
  ),
  endogenous.component = "pollen.pinus"
)
```

`lagTimeSeries` *Create lagged versions of time series variables*

Description

Takes a multivariate time series and creates time-lagged columns for modeling. This generates one new column per lag and variable, enabling analysis of how past values influence current observations.

Usage

```
lagTimeSeries(
  input.data = NULL,
  response = NULL,
  drivers = NULL,
  time = NULL,
  oldest.sample = "first",
  lags = NULL,
  time.zoom = NULL,
  scale = FALSE
)

prepareLaggedData(
  input.data = NULL,
  response = NULL,
  drivers = NULL,
  time = NULL,
  oldest.sample = "first",
  lags = NULL,
  time.zoom = NULL,
  scale = FALSE
)
```

Arguments

`input.data`	a dataframe with one time series per column. Default: `NULL`.
`response`	character string, name of the numeric column to be used as response in the model. Default: `NULL`.
`drivers`	character vector, names of the numeric columns to be used as predictors in the model. Default: `NULL`.
`time`	character vector, name of the numeric column with the time. Default: `NULL`.
`oldest.sample`	character string, either "first" or "last". When "first", the first row taken as the oldest case of the time series and the last row is taken as the newest case, so ecological memory flows from the first to the last row of `input.data`. When "last", the last row is taken as the oldest sample, and this is the mode that should be used when `input.data` represents a palaeoecological dataset. Default: `"first"`.

lags	numeric vector, lags to be used in the equation, in the same units as time. The use of seq to define it is highly recommended. If 0 is absent from lags, it is added automatically to allow the consideration of a concurrent effect. Lags should be aligned to the temporal resolution of the data. For example, if the interval between consecutive samples is 100 years, lags should be something like 0, 100, 200, 300. Lags can also be multiples of the time resolution, such as 0, 200, 400, 600 (when time resolution is 100 years). Default: NULL.
time.zoom	numeric vector of two values from the range of the time column, used to subset the data if desired. Default: NULL.
scale	boolean, if TRUE, applies the scale function to normalize the data. Required if the lagged data is going to be used to fit linear models. Default: FALSE.

Details

The function interprets the time column as an index representing the temporal position of each sample. It uses the lag function from the **zoo** package to shift columns by the specified lags, generating one new column per lag and variable.

Value

A dataframe with columns representing time-delayed values of the drivers and the response. Column names have the lag number as a suffix. Has the attributes ‘response‘ and ‘drivers‘, later used by [computeMemory()].

Author(s)

Blas M. Benito <blasbenito@gmail.com>

See Also

computeMemory

Other data_preparation: alignTimeSeries()

Examples

```
#loading data
data(palaeodata)

#adding lags
lagged.data <- lagTimeSeries(
 input.data = palaeodata,
 response = "pollen.pinus",
 drivers = c("climate.temperatureAverage", "climate.rainfallAverage"),
 time = "age",
 oldest.sample = "last",
 lags = seq(0.2, 1, by=0.2)
)

str(lagged.data)
```

```
# Check attributes (used by computeMemory)
attributes(lagged.data)
```

---

palaeodata *Dataframe with pollen and climate data.*

Description

A dataframe with a regular time grid of 0.2 ky resolution resulting from applying `mergePalaeoData` to the datasets `climate` and `pollen`:

Usage

```
data(palaeodata)
```

Format

dataframe with 10 columns and 7986 rows.

Details

- *age* in ky before present (ky BP).
- *pollen.pinus* pollen percentages of Pinus.
- *pollen.quercus* pollen percentages of Quercus.
- *pollen.poaceae* pollen percentages of Poaceae.
- *pollen.artemisia* pollen percentages of Artemisia.
- *climate.temperatureAverage* average annual temperature in degrees Celsius.
- *climate.rainfallAverage* average annual precipitation in millimetres per day (mm/day).
- *climate.temperatureWarmestMonth* average temperature of the warmest month, in degrees Celsius.
- *climate.temperatureColdestMonth* average temperature of the coldest month, in degrees Celsius.
- *climate.oxigenIsotope* delta O18, global ratio of stable isotopes in the sea floor, see `http://lorraine-lisiecki.com/stack.html` for further details.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

See Also

Other example_data: `climate`, `palaeodataLagged`, `palaeodataMemory`, `pollen`

palaeodataLagged *Lagged data generated by* prepareLaggedData.

Description

A dataframe resulting from the application of prepareLaggedData to the dataset palaeodata. The dataframe columns are named using the pattern VariableName__LagValue:

Usage

```
data(palaeodataLagged)
```

Format

dataframe with 19 columns and 3988 rows.

Details

- *pollen.pinus__0* numeric, values of the response variable (pollen counts of Pinus) at lag 0 (current time). This column is used as the response variable by computeMemory.
- *pollen.pinus__0.2-1* numeric, time-delayed values of the response for lags 0.2 to 1 (in ky). These columns represent the endogenous ecological memory.
- *climate.temperatureAverage__0* numeric, temperature values at lag 0 (concurrent effect).
- *climate.rainfallAverage__0* numeric, rainfall values at lag 0 (concurrent effect).
- *climate.temperatureAverage__0.2-1* numeric, time-delayed temperature values for lags 0.2 to 1 (exogenous memory).
- *climate.rainfallAverage__0.2-1* numeric, time-delayed rainfall values for lags 0.2 to 1 (exogenous memory).
- *time* numeric, the time/age column.

The dataframe has attributes response and drivers that are automatically used by computeMemory.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

See Also

Other example_data: climate, palaeodata, palaeodataMemory, pollen

palaeodataMemory *Output of* computeMemory

Description

List containing the output of computeMemory applied to palaeodataLagged. Its slots are:

Usage

```
data(palaeodataMemory)
```

Format

List with five slots.

Details

- response character, response variable name.
- drivers character vector, driver variable names.
- memory dataframe with five columns:
  - variable character, names of the different variables used to model ecological memory.
  - lag numeric, time lag values.
  - median numeric, median importance across repetitions of the given variable according to Random Forest.
  - sd numeric, standard deviation of the importance values of the given variable across repetitions.
  - min and max numeric, percentiles 0.05 and 0.95 of importance values of the given variable across repetitions.
- R2 vector, values of pseudo R-squared value obtained for the Random Forest model fitted on each repetition. Pseudo R-squared is the Pearson correlation between the observed and predicted data.
- prediction dataframe, with the same columns as the dataframe in the slot memory, with the median and confidence intervals of the predictions of all random forest models fitted.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

See Also

Other example_data: climate, palaeodata, palaeodataLagged, pollen

plotExperiment *Plots the output of* runExperiment.

Description

Takes the output of runExperiment, and generates plots of ecological memory patterns for a large number of simulated pollen curves.

Usage

```
plotExperiment(
  experiment.output = NULL,
  parameters.file = NULL,
  ribbon = FALSE
)
```

Arguments

experiment.output	list, output of runExperiment. Default: NULL.
parameters.file	dataframe of simulation parameters. Default: NULL.
ribbon	logical, switches plotting of confidence intervals on (TRUE) and off (FALSE). Default: FALSE.

Value

A ggplot2 object.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

See Also

plotMemory, runExperiment

Other virtualPollen: experimentToTable(), runExperiment()

plotMemory *Plots output of* computeMemory

Description

Plots the ecological memory pattern yielded by computeMemory.

Usage

```
plotMemory(
  memory.output = NULL,
  ribbon = FALSE,
  legend.position = "right",
  ...
)
```

Arguments

memory.output	list, output of computeMemory. Default: NULL.
ribbon	logical, switches plotting of confidence intervals on (TRUE) and off (FALSE). Default: FALSE.
legend.position	character, position of the legend. Default: "right".
...	additional arguments for internal use.

Value

A ggplot object.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

See Also

computeMemory

Other memoria: computeMemory(), extractMemoryFeatures()

Examples

```
#loading data
data(palaeodataMemory)

#plotting memory pattern
plotMemory(memory.output = palaeodataMemory)

#with confidence ribbon
plotMemory(memory.output = palaeodataMemory, ribbon = TRUE)
```

`pollen` *Dataframe with pollen counts.*

Description

A dataframe with the following columns:

Usage

```
data(pollen)
```

Format

dataframe with 5 columns and 639 rows.

Details

- *age* in kiloyears before present (ky BP).
- *pinus* pollen counts of Pinus.
- *quercus* pollen counts of Quercus.
- *poaceae* pollen counts of Poaceae.
- *artemisia* pollen counts of Artemisia.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

See Also

Other example_data: `climate`, `palaeodata`, `palaeodataLagged`, `palaeodataMemory`

`runExperiment` *Computes ecological memory patterns on simulated pollen curves produced by the* `virtualPollen` *package.*

Description

Applies `computeMemory` to assess ecological memory on a large set of virtual pollen curves.

Usage

```
runExperiment(
  simulations.file = NULL,
  selected.rows = NULL,
  selected.columns = NULL,
  parameters.file = NULL,
  parameters.names = NULL,
  driver.column = NULL,
  response.column = "Pollen",
  subset.response = "none",
  time.column = "Time",
  time.zoom = NULL,
  lags = NULL,
  repetitions = 10
)
```

Arguments

`simulations.file`
: List of dataframes produced by `virtualPollen::simulatePopulation`. Each list element is a time series dataframe for one virtual taxon. Can be a 1D list (one sampling scheme) or a 2D matrix-like list (rows = taxa, columns = sampling schemes). See `virtualPollen::simulation` for an example. Default: `NULL`.

`selected.rows`
: Numeric vector indicating which virtual taxa (list elements) from `simulations.file` to analyze. For example, `c(1, 3)` analyzes the 1st and 3rd taxa. Default: `NULL` (analyzes all taxa).

`selected.columns`
: Numeric vector indicating which sampling schemes (columns) from `simulations.file` to analyze. Only relevant when `simulations.file` has a 2D structure with multiple sampling schemes. Default: `NULL` (uses the first sampling scheme only).

`parameters.file`
: Dataframe of simulation parameters produced by `virtualPollen::parametersDataframe`, with one row per virtual taxon. Rows must align with `simulations.file`. See `virtualPollen::parameters` for an example. Default: `NULL`.

`parameters.names`
: Character vector of column names from `parameters.file` to include in output labels. These help identify which simulation settings produced each result. Example: `c("maximum.age", "fecundity")`. Default: `NULL`.

`driver.column`
: Character vector of column names representing environmental drivers in the simulation dataframes. Common choices: `"Driver.A"`, `"Driver.B"`, or `"Suitability"`. Default: `NULL`.

`response.column`
: Character string naming the response variable column in the simulation dataframes. Use `"Pollen"` for pollen abundance from `virtualPollen::simulation`. Default: `"Pollen"`.

`subset.response`
: character string, one of "up", "down" or "none", triggers the subsetting of the input dataset. "up" only models ecological memory on cases where the response's

trend is positive, "down" selects cases with negative trends, and "none" selects all cases. Default: "none".

`time.column` character string, name of the time/age column. Usually, "Time". Default: "Time".

`time.zoom` numeric vector with two numbers defining the time/age extremes of the time interval of interest. Default: NULL.

`lags` numeric vector, lags to be used in the equation, in the same units as time. The use of seq to define it is highly recommended. If 0 is absent from lags, it is added automatically to allow the consideration of a concurrent effect. Lags should be aligned to the temporal resolution of the data. For example, if the interval between consecutive samples is 100 years, lags should be something like 0, 100, 200, 300. Lags can also be multiples of the time resolution, such as 0, 200, 400, 600 (when time resolution is 100 years). Default: NULL.

`repetitions` integer, number of random forest models to fit. Default: 10.

Value

A list with 2 slots:

- names matrix of character strings, with as many rows and columns as simulations.file. Each cell holds a simulation name to be used afterwards, when plotting the results of the ecological memory analysis.
- output a list with as many rows and columns as simulations.file. Each slot holds a an output of computeMemory.
  - memory dataframe with five columns:
    - Variable character, names and lags of the different variables used to model ecological memory.
    - median numeric, median importance across repetitions of the given Variable according to Random Forest.
    - sd numeric, standard deviation of the importance values of the given Variable across repetitions.
    - min and max numeric, percentiles 0.05 and 0.95 of importance values of the given Variable across repetitions.
  - R2 vector, values of pseudo R-squared value obtained for the Random Forest model fitted on each repetition. Pseudo R-squared is the Pearson correlation between the observed and predicted data.
  - prediction dataframe, with the same columns as the dataframe in the slot memory, with the median and confidence intervals of the predictions of all random forest models fitted.
  - multicollinearity multicollinearity analysis on the input data performed with vif_df. A vif value higher than 5 indicates that the given variable is highly correlated with other variables.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

See Also

computeMemory

Other virtualPollen: experimentToTable(), plotExperiment()

Index