Package 'doremi'

March 25, 2019

Type Package

Title Dynamics of Return to Equilibrium During Multiple Inputs

Version 0.1.1

Description Provides models to fit the dynamics of a regulated system experiencing exogenous inputs. The underlying models use differential equations and linear mixed-effects regressions to estimate the characteristic parameters of the equation (the coefficients) and an estimated signal. The package also provides print, summary, plot and predict functions, specific for the models outputs.

License GPL-3

Encoding UTF-8

LazyData true

Imports zoo, data.table, lme4, ggplot2, lmerTest

RoxygenNote 6.1.0

Suggests knitr, rmarkdown, devtools, roxygen2

VignetteBuilder knitr

NeedsCompilation no

Author Mongin Denis [aut],
Uribe Adriana [aut],
Courvoisier Delphine [aut],
Courvoisier Delphine [cre]

Maintainer Courvoisier Delphine <Delphine.Courvoisier@hcuge.ch>

Repository CRAN

Date/Publication 2019-03-25 14:30:03 UTC

R topics documented:

`calculate.gold` *Calculation of derivatives using the GOLD method*

Description

`calculate.gold` estimates the derivatives of a variable using the Generalized Orthogonal Local Derivative (GOLD) method described in Deboeck (2010). This method allows calculating over a number of measurement points (called the embedding number) the first derivative with errors uncorrelated with the signal. It was generalized for non-equidistant time points (variable time steps), in order to account for missing observations.

Usage

```
calculate.gold(signal, time, embedding = 2)
```

Arguments

`signal`	is a vector containing the data from which the derivative is estimated.
`time`	is a vector containing the time values corresponding to the signal. Arguments signal and time must have the same length.
`embedding`	is an integer indicating the number of points to consider for derivative calculation. Embedding must be greater than 1 because at least two points are needed for the calculation of the first derivative and at least 3 for the calculation of the second derivative.

Value

Returns a list containing three columns:

dtime- contains the time values in which the derivative was calculated. That is, the moving average of the input time over embedding points.

dsignal- is a data.frame containing three columns and the same number of rows as the signal. The first column is the moving average of the signal over embedding points, the second is the first derivative, and the third is the second derivative.

embedding- contains the number of points used for the derivative calculation, which is constant.

Examples

```
#In the following example the derivatives for the function y(t) = t^2 are calculated.
#The expected results are:
#y'(t) = 2t and y''(t) = 2
time <- c(1:500)/100
signal <- time^2
result <- calculate.gold(signal = signal, time = time, embedding = 5)
```

---

cardio *Measurements of cardiac frequency in 21 patients during effort tests*

Description

Data containing time, cardiac frequency and load of a resistive bicycle run by patients during effort tests

Usage

```
data(cardio)
```

Format

A data frame with 1686 rows and 4 variables

id positive integer, arbitrary identifier of the patient

time positive real number, time since the beginning of the test, in seconds (s)

load positive real number, load of the resistive bicycle, effort that the patient needs to do, in watts (W)

hr positive real number, patient's cardiac rhythm, in heart beats per minute (1/min)

Source

Mongin et al. 2018, under review (future DOI will be inserted here)

`errorcheck` *Displays error messages for the analysis function according to the nature of the error* `errorcheck` *displays error messages and/or warnings concerning the validity of input arguments provided to the analysis function*

Description

Displays error messages for the analysis function according to the nature of the error `errorcheck` displays error messages and/or warnings concerning the validity of input arguments provided to the analysis function

Usage

```
errorcheck(data, col_var)
```

Arguments

`data`	data.frame or data.table containing the data to be analyzed. Same object that is passed as input argument to the analysis function.
`col_var`	column variable. Contains a string that indicates the name of the column to analyze ("id","input",etc.)

Value

Doesn't return a value. Either displays directly the error message/warning or changes data type in the data.frame/data.table provided

`generate.excitation` *Excitation signal generation*

Description

`generate.excitation` generates a vector of randomly located square pulses with a given amplitude, duration and spacing between the pulses. A pulse is where the excitation passes from value 0 to value amplitude for a given duration and then returns back to 0, thus producing a square shape.

Usage

```
generate.excitation(amplitude = 1, nexc = 1, duration = 2,
  deltatf = 0.1, tmax = 10, minspacing = 1)
```

Arguments

amplitude	is vector of values different than 0 indicating the amplitude of the excitation. It should contain as many values as the number of pulses (nexc). If the elements are less than the number of pulses, the amplitude vector will be "recycled" and the elements from it will be repeated until all the pulses are covered (for instance, if the number of excitations nexc is 6 and the amplitude vector has two elements, pulses 1,3 and 5 will have the same amplitude as the first element of the amplitude vector and pulses 2,4 and 6 that of the second element).
nexc	is an integer greater than 0 indicating the number of pulses to generate.
duration	is a vector of values greater or equal to 0 indicating the duration of each pulse in time units. It should have as many elements as the number of pulses (nexc). If the elements are less than the number of pulses, the amplitude vector will be "recycled" and the elements from it will be repeated until all the pulses are covered.
deltatf	is a value greater than 0 indicating the time step between two consecutive data points.
tmax	is a value greater than 0 indicating the maximum time range of the excitation vector in time units. The time vector generated will go from 0 to tmax.
minspacing	as pulses are generated randomly, minspacing is a value greater than 0 that indicates minimum spacing between pulses, thus avoiding overlapping of the pulses in time. It can be 0 indicating that pulses can follow one another.

Details

Used for simulations in the context of the package. Beware that the following condition should apply:

$$tmax >= (duration + minspacing) * nexc$$

so that the pulses "fit" in the time lapse defined. Compared to pulsew from the seewave package this function can generate pulses of different duration and amplitude.

Value

Returns two vectors:

E- vector containing the values of the excitation generated.

t- vector containing the values of time generated.

Examples

```
generate.excitation (amplitude = 3,
                     nexc = 6,
                     duration = 2,
                     deltatf = 1,
                     tmax = 200,
                     minspacing = 2)
#Vector of length 201 (deltatf x tmax + 1 as it includes 0 as initial time value)
generate.excitation (amplitude = c(1,10,20),
                     nexc = 3,
```

```
duration = c(1,2,4),
deltatf = 0.5,
tmax = 100,
minspacing = 10)
```

`generate.panel.remi` *Simulation of various individual signals with intra and inter noise*

Description

`generate.panel.remi` Generates signals with intra and inter individual noise for several individuals. In order to do this, the function generates a pseudo-continuous signal per individual that is a solution to the first order differential equation:

$$\frac{dy(t)}{dt} - \gamma y(t) = E(t)$$

The analytical solution to this equation is a convolution between the Green function and the excitation term. The function generates internally a pseudo-continuous signal to increase the precision with which the convolution is calculated. From this expanded signal, the function samples points with a constant time step given by deltatf. These operations are repeated as many times as the value set in the input "nind". Once the signal is sampled, intra-individual and inter-individual noise with normal distributions are added.

Usage

```
generate.panel.remi(nind = 1, dampingtime, amplitude = 1, nexc = 1,
  duration = 10, deltatf = 0.5, tmax, minspacing = 10,
  internoise = 0, intranoise = 0)
```

Arguments

`nind`	number of individuals.
`dampingtime`	Signal damping time. It corresponds to the time needed to reach 37% (1/e) of the difference between the equilibrium value and the amplitude of the signal reached when there is no excitation (or 63% of the maximum value for a constant excitation). It should be positive (dampingtime>0).
`amplitude`	is vector of values different than 0 indicating the amplitude of the excitation. It should contain as many values as the number of pulses (nexc). If the elements are less than the number of pulses, the amplitude vector will be "recycled" and the elements from it will be repeated until all the pulses are covered (for instance, if the number of excitations nexc is 6 and the amplitude vector has two elements, pulses 1,3 and 5 will have the same amplitude as the first element of the amplitude vector and pulses 2,4 and 6 that of the second element).
`nexc`	is an integer greater than 0 indicating the number of pulses to generate.

duration	is a vector of values greater or equal to 0 indicating the duration of each pulse in time units. It should have as many elements as the number of pulses (nexc). If the elements are less than the number of pulses, the amplitude vector will be "recycled" and the elements from it will be repeated until all the pulses are covered.
deltatf	is a value greater than 0 indicating the time step between two consecutive data points.
tmax	is a value greater than 0 indicating the maximum time range of the excitation vector in time units. The time vector generated will go from 0 to tmax.
minspacing	as pulses are generated randomly, minspacing is a value greater than 0 that indicates minimum spacing between pulses, thus avoiding overlapping of the pulses in time. It can be 0 indicating that pulses can follow one another.
internoise	Is the inter-individual noise added. The dampingtime across individuals follows a normal distribution centered on the input parameter dampingtime with a standard deviation of internoise*dampingtime, except if any damping time is negative (see Details section).
intranoise	Is the noise added in each individual signal. It also follows a normal distribution with a standard deviation equal to this parameter times the maximum amplitude of the convolution when using an excitation containing a single pulse.

Details

Used for simulations in the context of the package.

The function currently simulates only positive damping times corresponding to a regulated system. When the damping time is low and the inter individual noise is high, some individuals' damping time could be negative. In that case, the damping time distribution is truncated at 0.1*deltatf and values below are set to this limit. High values are symmetrically set at the upper percentile value similar to a Winsorized mean. A warning provides the initial inter individual noise set as input argument and the inter individual noise obtained after truncation.

Value

Returns a data frame with signal and time values starting at 0 and sampled at equal time steps deltatf in the time lapse tmax. It contains the following columns:

- id - individual identifier (from 1 to nind).
- excitation - excitation signal generate through the generate.excitation
- time - time values
- dampedsignalraw - signal with no noise (inter noise added for each individual)
- dampedsignal - signal with intra noise added

See Also

`generate.remi` for calculation of the analytical solution to the differential equation. Call the data frame $fulldata of the result for a full data frame with points generated at a very small deltatf in order to build a pseudo-continuous function that will enhance the quality of the generated signal (see `remi`). and `generate.excitation` for excitation signal generation

Examples

```
generate.panel.remi(nind = 5,
              dampingtime = 10,
              amplitude = c(5,10),
              nexc = 2,
              duration = 20,
              deltatf = 0.5,
              tmax = 200,
              minspacing = 0,
              internoise = 0.2,
              intranoise = 0.1)
```

`generate.remi` *Generation of the first order differential equation solution*

Description

`generate.remi` returns a vector containing the convolution of the Green function of the first order differential equation with a given damping time and an excitation term.

Usage

```
generate.remi(dampingtime, inputvec, inputtim)
```

Arguments

`dampingtime`	Signal damping time. It represents the characteristic response time of the solution of the differential equation. It should be positive.
`inputvec`	Is a vector containing the values of the excitation signal.
`inputtim`	Is a vector containing the time values corresponding to the excitation signal.

Value

Returns a list containing two elements:

- y is a vector containing the values calculated from the convolution of the Green function and the excitation vector.
- t is a vector containing the corresponding time values

Examples

```
generate.remi(10,rep(c(0,1,0),c(20,30,50)),1:100)
```

plot.doremi *S3 method to plot DOREMI objects*

Description

plot.doremi generates a plot with the observed values of the signal, the excitation values and the fitted signal over time for each individual.

Usage

```
## S3 method for class 'doremi'
plot(x, ..., id = NULL)
```

Arguments

x	DOREMI object from remi analysis
...	includes the additional arguments inherited from the generic plot method
id	Identifiers of the individuals to be represented in the plot. By default, it will print the first six individuals.

Value

Returns a plot with axis labels, legend and title. The axis labels and legend include the names of the variables set as input arguments. The title includes the name of the DOREMI object result of the analysis. The function uses ggplot to generate the graphs and so it is possible to override the values of axis labels, legend and title through ggplot commands.

Examples

```
mydata <- generate.panel.remi(nind = 2,
                              dampingtime = 10,
                              amplitude = c(5,10),
                              nexc = 2,
                              duration = 20,
                              deltatf = 2,
                              tmax = 200,
                              minspacing = 0,
                              internoise = 0.2,
                              intranoise = 0.1)
myresult <- remi(data = mydata,
                 id = "id",
                 input = "excitation",
                 time = "time",
                 signal = "dampedsignal",
                 embedding = 5)
plot(myresult)
```

predict.doremi *S3 method to predict signal values in a DOREMI object when entering a new excitation*

Description

predict.doremi predicts signal values with a DOREMI object when providing a new excitation vector(s).

Usage

```
## S3 method for class 'doremi'
predict(object, ..., newdata, verbose = FALSE)
```

Arguments

object	DOREMI object result of an analysis with the function remi
...	Additional arguments inherited from generic predict method.
newdata	includes a data frame containing three columns or more: id (optional), indicating the individual identifier time, containing the time values excitation, being one or several columns containing the different excitations used to estimate a new signal. As in the other methods for the predict function, the columns of newdata must have the same names as those of the original object.
verbose	Is a boolean that displays status messages of the function when set to 1.

Value

Returns a list containing the values of time, the values of the excitation and the predicted values of the signal for the new excitation(s).

Examples

```
myresult <- remi(data = cardio[id == 1],
                 input = "load",
                 time = "time",
                 signal = "hr",
                 embedding = 5)
#Copying cardio into a new data frame and modifying the excitation column
new_exc <- cardio[id == 1, !"id"]

et <- generate.excitation(amplitude = 100,
                                    nexc = 6,
                                    duration = 2,
                                    deltatf = 1,
                                    tmax = 49,
                                    minspacing = 2)
```

```
new_exc$load <- et$exc
new_exc$time <- et$t
predresult <- predict(myresult, newdata = new_exc)
plot(predresult)
```

`print.doremi` *S3 method to print DOREMI objects*

Description

`print.doremi` prints the most important results of a DOREMI object

Usage

```
## S3 method for class 'doremi'
print(x, ...)
```

Arguments

`x`	DOREMI object
`...`	includes the additional arguments inherited from the generic print method

Value

Returns the three coefficients of the differential equation estimated (fixed part, table $resultmean of the DOREMI object)

Examples

```
myresult <- remi(data = cardio,
                 id = "id",
                 input = "load",
                 time = "time",
                 signal = "hr",
                 embedding = 5)
myresult
```

print.doremidata *S3 method to print DOREMI data objects*

Description

print.doremidata prints the most important results of a DOREMIDATA object

Usage

```
## S3 method for class 'doremidata'
print(x, ...)
```

Arguments

x	DOREMIDATA object
...	includes the additional arguments inherited from the generic print method

Value

Returns the table $data of the DOREMIDATA object

Examples

```
mydata <- generate.panel.remi(nind = 5,
                              dampingtime = 10,
                              amplitude = c(5,10),
                              nexc = 2,
                              duration = 20,
                              deltatf = 2,
                              tmax = 200,
                              minspacing = 0,
                              internoise = 0.2,
                              intranoise = 0.1)
mydata
```

remi *DOREMI first order analysis function*

Description

remi estimates the coefficients of a first order differential equation of the form:

$$\frac{1}{\gamma}\dot{y}(t) = -y(t) + \epsilon E(t) + eqvalue$$

using linear mixed-effect models. Where y(t) is the individual's signal, $\dot{y}(t)$ is the derivative and E(t) is the excitation.

Usage

```
remi(data, id = NULL, input = NULL, time = NULL, signal,
  embedding = 2, verbose = FALSE)
```

Arguments

`data`	Is a data frame containing at least one column, that is the signal to be analyzed.
`id`	Is a STRING containing the NAME of the column of data containing the identifier of the individual. If this parameter is not entered when calling the function, a single individual is assumed and a linear regression is done instead of the linear mixed-effects regression.
`input`	Is a STRING or a VECTOR OF STRINGS containing the NAME(s) of data column(s) containing the EXCITATION vector(s). If this parameter is not entered when calling the function, the excitation is assumed to be unknown. In this case, the linear mixed-effect regression is still carried out but no coefficient is calculated for the excitation term. The function then uses the parameters estimated by the regression to carry out an exponential fit of the signal and build the estimated curve. The function will consider an excitation variable each column of data whose name is contained in the input vector. The function will return a coefficient for each one of the excitation variables included.
`time`	Is a STRING containing the NAME of the column of data containing the time vector. If this parameter is not entered when calling the function, it is assumed that time steps are of 1 unit and the time vector is generated internally in the function.
`signal`	Is a STRING containing the NAME of the column of the data frame containing the SIGNAL to be studied.
`embedding`	Is a positive integer containing the number of points to be used for the calculation of the derivatives. Its value by default is 2 as at least two points are needed for the calculation of the first derivative.
`verbose`	Is a boolean that displays status messages of the function when set to 1.

Details

The analysis performs the following linear mixed-effects regression:

$$y'_{ij} \sim b_0 + b_{0j} + b_1 y_{ij} + b_2 E_{ij} + u_{1j} y_{ij} + u_{2j} E_{ij} + e_{ij}$$

with i accounting for the time and j for the different individuals. e_{ij} are the residuals, y'_{ij} is the derivative calculated on embedding points and y and E are the signal and the excitation averaged on embedding points. The coefficients estimated to characterize the signal are calculated as follows:

- Damping time: $\tau_j = \frac{1}{\gamma_j}$ with $\gamma_j = b_1 + u_{1j}$
- Excitation coefficient: $\epsilon_j = \frac{b_2 + u_{2j}}{\gamma_j}$. It is the proportionality between the excitation and the difference between the maximum value reached by the signal and its initial value.
- Equilibrium value: $eqvalue_j = \frac{b_0 + b_{0j}}{\gamma_j}$. It is the stable value reached in the absence of excitation.

The estimation is performed using the function lmer if there are several individuals or lm if there is only one. With the above estimated parameters, the estimated signal can be reconstructed for each individual by first performing the convolution of the excitation with the Green function using the estimated damping rate and then offsetting the resulting signal with the equilibrium value. The function returns five objects:

1. data- A data.frame including the input data and intermediate calculations used to prepare the variables for the fit:

   signal_rollmean - calculation of the moving average of the signal over embedding points.

   signal_derivate1 - calculation of the first derivative of the signal with the GOLD method in embedding points.

   time_derivate - calculation of the moving average of the time vector over embedding points.

   input_rollmean - calculation of the moving average of the excitation vector over embedding points.

2. resultid- A data.frame including for each individual, listed by id number, the damping time, the excitation coefficient and the equilibrium value (see variables presented in the Details section).

3. resultmean- A data.frame including the fixed effects of the three coefficients mentioned above.

4. regression- A list containing the summary of the linear mixed-effects regression.

5. estimated- A data.frame containing the estimated signal calculated as the convolution of the Green function with the estimated damping time, excitation coefficient and equilibrium value and the excitation vector with an added offset (see above). There are two extreme cases in the generation of the signal and these depend on sampling. The excitation vector is expanded to generate a pseudo-continuous signal and increase accuracy when calculating the convolution. Missing data in the exitation signal is completed by using the previous known value (exc_min in the data.frame) or using the next known value (exc_max in the data.frame). With these two imputed excitation vectors, the two extreme cases of the mean estimated signal are calculated by carrying out the convolution between the Green function (decreasing exponential with the damping time calculated by the linear mixed-effects regression). These excitation variables are then used to generate the ymin and ymax signals respectively.

   As seen in the Description section, the print method by default prints only the resultmean element. Each one of the other objects can be accessed by indicating $ and their name after the result, for instance, for a DOREMI object called "result", it is possible to access the regression summary by typing result$regression.

6. embedding - contains the embedding number used to generate the results (same as function input argument)

Value

Returns a summary of the fixed components for the three coefficients: damping time, excitation coefficient and equilibrium value.

See Also

`calculate.gold` to compute the derivatives, for details on embedding.

Examples

```
myresult <- remi(data = cardio,
                 id = "id",
                 input = "load",
                 time = "time",
                 signal = "hr",
                 embedding = 5)
```

`rotation` *Measurements of response time of 17 individuals when carrying out mental rotation tasks*

Description

Data containing reaction time to a mental rotation task over a 60 day period for 17 individuals (Courvoisier et al., 2013).

Usage

```
data(rotation)
```

Format

A data frame with 619 rows and 5 variables

id positive integer, arbitrary identifier of the individual

sex character, sex of the individual, as the study highlighted the difference in response time according to sex

days positive integer,day since the beginning of the experiment

meanRT positive integer, mean response time of the individual to execute the mental rotation task, in milliseconds (ms)

logmeanRT natural logarithm of the mean response time

Source

Delphine S. Courvoisier, Olivier Renaud, Christian Geiser, Kerstin Paschke, Kevin Gaudy, Kirsten Jordan, Sex hormones and mental rotation: An intensive longitudinal investigation,

Hormones and Behavior,

Volume 63, Issue 2,

2013,

Pages 345-351,

https://doi.org/10.1016/j.yhbeh.2012.12.007

summary.doremi *S3 method for DOREMI object summary*

Description

summary.doremi provides a summary of the remi analysis

Usage

```
## S3 method for class 'doremi'
summary(object, ...)
```

Arguments

object	DOREMI object (contains several lists)
...	includes the additional arguments inherited from the generic summary method

Value

Returns a summary containing the five lists of the DOREMI object

Examples

```
myresult <- remi(data = cardio,
                 id = "id",
                 input = "load",
                 time = "time",
                 signal = "hr",
                 embedding = 5)
summary(myresult)
```

Index