airGRteaching: Teaching Hydrological Modelling with the GR Rainfall-Runoff Models ('Shiny' Interface Included). Manual of the R package version 0.2.6.27

Olivier Delaigue, Laurent Coron, Pierre Brigode

Package 'airGRteaching'

April 23, 2019

Type Package

Title Teaching Hydrological Modelling with the GR Rainfall-Runoff Models ('Shiny' Interface Included)

Version 0.2.6.27

Date 2019-04-23

Depends airGR (>= 1.2.13.16)

Imports dygraphs (>= 1.1.1.6), markdown, plotrix, shiny (>= 1.1.0), shinyjs (>= 1.0), xts

Suggests knitr

Description Add-on package to the 'airGR' package that simplifies its use and is aimed at being used for teaching hydrology. The package provides 1) three functions that allow to complete very simply a hydrological modelling exercise 2) plotting functions to help students to explore observed data and to interpret the results of calibration and simulation of the GR ('Génie rural') models 3) a 'Shiny' graphical interface that allows for displaying the impact of model parameters on hydrographs and models internal variables.

License GPL-2

NeedsCompilation no

URL https://hydrogr.github.io/airGRteaching/

Encoding UTF-8

VignetteBuilder knitr

Author Olivier Delaigue [aut, cre] (<https://orcid.org/0000-0002-7668-8468>),
Laurent Coron [aut] (<https://orcid.org/0000-0002-1503-6204>),
Pierre Brigode [aut] (<https://orcid.org/0000-0001-8257-0741>),
Guillaume Thirel [ctb] (<https://orcid.org/0000-0002-1444-1830>)

Maintainer Olivier Delaigue <airGR@irstea.fr>

Repository CRAN

Date/Publication 2019-04-24 06:00:03 UTC

R topics documented:

`airGRteaching` *Tools to Simplify the Use of the airGR Hydrological Package for Education (Including a Shiny Application)*

Description

airGRteaching is an add-on package to the airGR package that simplifies its use and is teaching-oriented. It allows to use with very low programming skills the rainfall-runoff models (GR4H, GR4J, GR5J, GR6J, GR2M, GR1A) and a snow melt and accumulation model (CemaNeige). This package also provides graphical devices to help students to explore data and modelling results.

The airGRteaching package has been designed to fulfil a major requirement: facilitating the use of the airGR functionalities by students. The names of the functions and their arguments were chosen to this end.

The package is mostly based on three families of functions:
- the functions that allow to complete very simply a hydrological modelling exercise;
- plotting functions to help students to explore observed data and to interpret the results of calibration and simulation of the GR models;
- a function which runs a 'Shiny' graphical interface that allows for displaying in real-time model parameters impacts on hydrographs.

— Modelling Functions

Three functions allow to complete very simply a hydrological modelling exercise:
- preparation of data: `PrepGR()`;
- calibration of the models: `CalGR()`;
- simulation with the models: `SimGR()`

— Plotting Functions

airGRteaching provides two types of plotting functions that allow to produce static (`plot()`) or dynamic (`dyplot()`) graphics (incl. mouse events and interactive graphics). The devices allow to explore observed data and to interpret the results of calibration and simulation of the GR models.

— Shiny interface

The package also provides the `ShinyGR()`) function, which allows to run a Shiny interface. Thus its is possible to perform:
- interactive flow simulations with parameters modifications;
- automatic calibration;
- display of internal variables evolution;
- time period selection.

— Models

The six hydrological models and the snow melt and accumulation model already available in airGR are available in airGRteaching.
These models can be called within airGRteaching using the following model names:
- `GR4H`: four-parameter hourly lumped hydrological model (Mathevet, 2005)
- `GR4J`*: four-parameter daily lumped hydrological model (Perrin et al., 2003)
- `GR5J`*: five-parameter daily lumped hydrological model (Le Moine, 2008)
- `GR6J`*: six-parameter daily lumped hydrological model (Pushpalatha et al., 2011)
- `GR2M`: two-parameter monthly lumped hydrological model (Mouelhi, 2003 ; Mouelhi et al., 2006a)
- `GR1A`: one-parameter annual lumped hydrological model (Mouelhi, 2003 ; Mouelhi et al., 2006b)
- `CemaNeige`: two-parameter degree-day snow melt and accumulation daily model (combined with GR4J, GR5J or GR6J) (Valéry et al., 2014)
*: available in the Shiny interface.

— References

- Coron, L., G. Thirel, O. Delaigue, C. Perrin and V. Andréassian (2017), The Suite of Lumped GR Hydrological Models in an R Package. Environmental Modelling and Software, 94, 166–171. doi: 10.1016/j.envsoft.2017.05.002.
- Le Moine, N. (2008), Le bassin versant de surface vu par le souterrain : une voie d'amélioration des performances et du réalisme des modèles pluie-débit ?, PhD thesis (in French), UPMC - Cemagref Antony, Paris, France, 324 pp.
- Mathevet, T. (2005), Quels modèles pluie-débit globaux pour le pas de temps horaire ? Développement empirique et comparaison de modèles sur un large échantillon de bassins versants, PhD thesis (in French), ENGREF - Cemagref Antony, Paris, France, 463 pp.
- Mouelhi S. (2003), Vers une chaîne cohérente de modèles pluie-débit conceptuels globaux aux pas de temps pluriannuel, annuel, mensuel et journalier, PhD thesis (in French), ENGREF - Cemagref Antony, Paris, France, 323 pp.
- Mouelhi, S., C. Michel, C. Perrin and V. Andréassian (2006a), Stepwise development of a two-parameter monthly water balance model, Journal of Hydrology, 318(1-4), 200-214, doi:10.1016/j.jhydrol.2005.06.014.
- Mouelhi, S., C. Michel, C. Perrin. & V. Andreassian (2006b), Linking stream flow to rainfall at the annual time step: the Manabe bucket model revisited, Journal of Hydrology, 328, 283-296, doi:10.1016/j.jhydrol.2005.12.022.

- Perrin, C., C. Michel and V. Andréassian (2003), Improvement of a parsimonious model for streamflow simulation, Journal of Hydrology, 279(1-4), 275-289, doi:10.1016/S0022-1694(03)00225-7.
- Pushpalatha, R., C. Perrin, N. Le Moine, T. Mathevet and V. Andréassian (2011), A downward structural sensitivity analysis of hydrological models to improve low-flow simulation, Journal of Hydrology, 411(1-2), 66-76, doi:10.1016/j.jhydrol.2011.09.034.
- Valéry, A., V. Andréassian and C. Perrin (2014), "As simple as possible but not simpler": What is useful in a temperature-based snow-accounting routine? Part 2 - Sensitivity analysis of the Cemaneige snow accounting routine on 380 catchments, Journal of Hydrology, 517(0): 1176-1187, doi:10.1016/j.jhydrol.2014.04.058.

CalGR *Calibration algorithm that optimises the error criterion selected as objective function*

Description

Calibration algorithm that optimises the error criterion selected as objective function using the Irstea-HBAN procedure described by C. Michel

Usage

```
CalGR(PrepGR, CalCrit = c("NSE", "KGE", "KGE2", "RMSE"),
  WupPer = NULL, CalPer,
  transfo = c("", "sqrt", "log", "inv", "sort"), verbose = TRUE)
```

Arguments

`PrepGR`	[object of class `PrepGR`] see `PrepGR` for details
`CalCrit`	[character] name of the objective function (must be one of `"NSE"`, `"KGE"`, `"KGE2"` or `"RMSE"`)
`WupPer`	(optional) [character] vector of 2 values to define the beginning and end of the warm-up period [`"YYYY-mm-dd"` or `"YYYY-mm-dd HH:MM:SS"`]
`CalPer`	[character] vector of 2 values to define the beginning and end of the calibration period [`"YYYY-mm-dd"` or `"YYYY-mm-dd HH:MM:SS"`]
`transfo`	(optional) [character] name of the transformation transformation applied to discharge for calculating the objective function (must be one of `""`, `"sqrt"`, `"log"`, `"inv"` or `"sort"`)
`verbose`	(optional) [boolean] logical value indicating if the function is run in verbose mode or not

Value

list object of class CalGR containing:

- `OptionsCalib` [list] object of class RunOptions (see: CreateRunOptions)
- `Qobs` [numeric] series of observed discharges [mm/time step]
- `OutputsCalib` [list] object of class OutputsCalib (see: Calibration)
- `OutputsModel` [list] object of class OutputsModel (see: RunModel)
- `TypeModel` [character] name of the function of the hydrological model used
- `CalCrit` [character] name of the function that computes the error criterion during the calibration step
- `PeriodModel` [list] $WarmUp: vector of 2 POSIXct values defining the beginning and end of the warm-up period, $Run: vector of 2 POSIXct values defining the beginning and end of the calibration period

Author(s)

Olivier Delaigue

See Also

CreateRunOptions, CreateInputsCrit, CreateCalibOptions, ErrorCrit_RMSE, ErrorCrit_NSE, ErrorCrit_KGE, ErrorCrit_KGE2, Calibration_Michel

Examples

```
library(airGRteaching)

## data.frame of observed data
data(L0123001, package = "airGR")
BasinObs2 <- BasinObs[, c("DatesR", "P", "E", "Qmm", "T")]

## Preparation of observed data for modelling
PREP <- PrepGR(ObsDF = BasinObs2, HydroModel = "GR4J", CemaNeige = TRUE)

## Calibration step
CAL <- CalGR(PrepGR = PREP, CalCrit = "KGE2",
             WupPer = NULL, CalPer = c("1990-01-01", "1993-12-31"))
str(CAL)
```

---

dyplot *Interactive time series plotting of GR data*

Description

Interactive time series plotting of objects of classes *PrepGR*, *CalGR* or *SimGR*.

Usage

```
## Default S3 method:
dyplot(x, Qsup = NULL, Qsup.name = "Qsup",
  col.Precip = c("royalblue", "lightblue"),
  col.Q = c("black", "orangered", "grey"), col.na = "lightgrey",
  ylab = NULL, main = NULL,
  plot.na = TRUE, RangeSelector = TRUE, Roller = FALSE,
  LegendShow = c("follow", "auto", "always", "onmouseover", "never"), ...)
```

Arguments

`x`	[object of class *PrepGR*, *CalGR* or *SimGR*] see `PrepGR`, `CalGR`, `SimGR` for details
`Qsup`	(optional) [numeric] additional time series of flows (at the same time step than argument `x`) [mm/time step]
`Qsup.name`	(optional) [character] a label for the legend of Qsup
`col.Precip`	(optional) [character] vector of 1 (total precip.) or 2 (liquid and solid precip. with CemaNeige) color codes or names for precipitation (these can be of the form `"#AABBCC"` or `"rgb(255,100,200)"` or `"yellow"`), see `par` and `rgb`
`col.Q`	(optional) [character] vector of up to 3 color codes or names for observed (first value), simulated (second value, if provided) and additional (last value, if provided) flows, respectively (these can be of the form `"#AABBCC"` or `"rgb(255,100,200)"` or `"yellow"`), see `par` and `rgb`
`col.na`	(optional) [character] color code or name for missing values(these can be of the form `"#AABBCC"` or `"rgb(255,100,200)"` or `"yellow"`), see `par` and `rgb`
`ylab`	(optional) [character] a label for the y-axis (flow and precipitation)
`main`	(optional) [character] a main title for the plot
`plot.na`	[boolean] indicating if the missing values are plotted on the x-axis
`RangeSelector`	(optional) [boolean] add a range selector to the bottom of the chart that allows users to pan and zoom to various date ranges (see `dyRangeSelector`)
`Roller`	(optional) [numeric] number of time scale units (e.g. days, months, years) to average values over (see `dyRoller`)
`LegendShow`	(optional) [character] when to display the legend. Specify `"always"` to always show the legend. Specify `"onmouseover"` to only display it when a user mouses over the chart. Specify `"follow"` (default) to have the legend show as overlay to the chart which follows the mouse. See `dyLegend`
`...`	other parameters to be passed through to plotting functions

Author(s)

Olivier Delaigue

See Also

`PrepGR`, `CalGR`, `SimGR`

Examples

```
library(airGRteaching)

## data.frame of observed data
data(L0123001, package = "airGR")
BasinObs2 <- BasinObs[, c("DatesR", "P", "E", "Qmm", "T")]

## Preparation of observed data for modelling
PREP <- PrepGR(ObsDF = BasinObs2, HydroModel = "GR4J", CemaNeige = FALSE)
dyplot(PREP, main = "Observation")

## Calibration step
CAL <- CalGR(PrepGR = PREP, CalCrit = "KGE2",
             WupPer = NULL, CalPer = c("1990-01-01", "1993-12-31"))
dyplot(CAL, main = "Calibration")

## Simulation
SIM <- SimGR(PrepGR = PREP, CalGR = CAL, EffCrit = "KGE2",
             WupPer = NULL, SimPer = c("1994-01-01", "1998-12-31"))
dyplot(SIM, main = "Simulation")
```

`plot.CalGR` *Time series plotting of GR calibration objects*

Description

Time series plotting of GR calibration objects

Usage

```
## S3 method for class 'CalGR'
plot(x, xlab = NULL, ylab = NULL, main = NULL, which = c("perf", "iter", "ts"), ...)
```

Arguments

`x`	[object of class *CalGR*] see `CalGR` for details
`xlab`	(optional) [character] a label for the x-axis when `which = "iter"` (see `title`)
`ylab`	(optional) [character] a label for the y-axis when `which = "iter"` (vector of 1 or 2 values for flow and rainfall respectively; see `title`)
`main`	(optional) [character] a main title for the plot (see `title`)
`which`	[character] choice of the plot type (`"perf"` (default): plot diagnostics; `"iter"`: parameter and calibration criterion values during the iterations of the steepest descent step of the airGR calibration algorithm; `"ts"`: time series of observed precipitation and observed and simulated flows)
`...`	other parameters to be passed through to plotting functions

Author(s)

Olivier Delaigue

See Also

CalGR, plot.OutputsModel

Examples

```
library(airGRteaching)

## data.frame of observed data
data(L0123001, package = "airGR")
BasinObs2 <- BasinObs[, c("DatesR", "P", "E", "Qmm", "T")]

## Preparation of observed data for modelling
PREP <- PrepGR(ObsDF = BasinObs2, HydroModel = "GR5J", CemaNeige = TRUE)

## Calibration step
CAL <- CalGR(PrepGR = PREP, CalCrit = "KGE2",
             WupPer = NULL, CalPer = c("1990-01-01", "1993-12-31"))
plot(CAL, which = "perf")
plot(CAL, which = "ts")
plot(CAL, which = "iter")
```

---

plot.PrepGR *Time series plotting GR observation objects*

Description

Time series plotting GR observation objects

Usage

```
## S3 method for class 'PrepGR'
plot(x, type = "l", col.Precip = "royalblue", col.Q = "black", col.na = "grey",
  xlab = NULL, ylab = NULL, main = NULL,
  plot.na = TRUE, ...)
```

Arguments

x	[object of class *PrepGR*] see PrepGR for details
type	[character] the type of plot that should be drawn (see plot for details)
col.Precip	(optional) [character]color code or name for precipitation, see par
col.Q	(optional) [character] color code or name for observed flow, see par
col.na	(optional) [character] color code or name for missing values, see par
xlab	(optional) [character] a label for the x-axis (see title)

ylab	(optional) [character] a label for the y-axis (vector of 1 or 2 values for rainfall and flow respectively; see title)
main	(optional) [character] a main title for the plot (see title)
plot.na	[boolean] boolean indicating if the missing values are plotted on the x-axis
...	other parameters to be passed through to plotting functions

Author(s)

Olivier Delaigue

See Also

PrepGR

Examples

```
library(airGRteaching)

## data.frame of observed data
data(L0123001, package = "airGR")
BasinObs2 <- BasinObs[, c("DatesR", "P", "E", "Qmm", "T")]

## Preparation of observed data for modelling
PREP <- PrepGR(ObsDF = BasinObs2, HydroModel = "GR4J", CemaNeige = FALSE)

## Observed data plotting
plot(PREP)
```

---

plot.SimGR *Synthetic plotting of of GR model outputs*

Description

Function that creates a screen plot giving an overview of the GR model outputs

Usage

```
## S3 method for class 'SimGR'
plot(x, ...)
```

Arguments

x	[object of class *SimGR*] see SimGR for details
...	other parameters to be passed through to plotting functions

Details

Dashboard of results including various graphs (depending on the model):
(1) time series of total precipitation and simulated flows (and observed flows if provided)
(2) interannual median monthly simulated flow (and observed flows if provided)
(3) correlation plot between simulated and observed flows (if observed flows provided)
(4) cumulative frequency plot for simulated flows (and observed flows if provided)

Value

Screen plot window

Author(s)

Olivier Delaigue, Laurent Coron

See Also

SimGR, plot.OutputsModel

Examples

```
library(airGRteaching)

## data.frame of observed data
data(L0123001, package = "airGR")
BasinObs2 <- BasinObs[, c("DatesR", "P", "E", "Qmm", "T")]

## Preparation of observed data for modelling
PREP <- PrepGR(ObsDF = BasinObs2, HydroModel = "GR4J", CemaNeige = FALSE)

## Calibration step
CAL <- CalGR(PrepGR = PREP, CalCrit = "KGE2",
             WupPer = NULL, CalPer = c("1990-01-01", "1993-12-31"))

## Simulation step using the result of the automatic calibration method to set the model parameters
SIM <- SimGR(PrepGR = PREP, CalGR = CAL, EffCrit = "KGE2",
             WupPer = NULL, SimPer = c("1994-01-01", "1998-12-31"))
plot(SIM)
```

PrepGR *Creation of the inputs required to run the CalGR and SimGR functions*

Description

Creation of the inputs required to run the CalGR and SimGR functions

Usage

```
PrepGR(ObsDF = NULL, DatesR = NULL, Precip = NULL, PotEvap = NULL,
       Qobs = NULL, TempMean = NULL,
       ZInputs = NULL, HypsoData = NULL, NLayers = 5,
       HydroModel, CemaNeige = FALSE)
```

Arguments

`ObsDF`	(optional) [data.frame] data.frame of dates, total precipitation, potential evapotranspiration, observed discharges and mean air temperature (only if CemaNeige is used) (variables must be in this order; see below for the units)
`DatesR`	(optional) [POSIXt] vector of dates required to create the GR and CemaNeige (if used) models inputs. Time zone must be defined as "UTC"
`Precip`	(optional) [numeric] time series of total precipitation (catchment average) [mm/time step], required to create the GR and CemaNeige (if used) models inputs
`PotEvap`	(optional) [numeric] time series of potential evapotranspiration (catchment average) [mm/time step], required to create the GR model inputs
`Qobs`	(optional) [numeric] time series of observed discharges [mm/time step]
`TempMean`	(optional) [numeric] time series of mean air temperature [°C], required to create the CemaNeige model inputs
`ZInputs`	(optional) [numeric] real giving the mean elevation of the Precip and TempMean series (before extrapolation) [m], possibly used to create the CemaNeige (if used) model inputs
`HypsoData`	(optional) [numeric] vector of 101 reals: min, q01 to q99 and max of catchment elevation distribution [m]; if not defined a single elevation is used for CemaNeige (if used)
`NLayers`	(optional) [numeric] integer giving the number of elevation layers requested [-], required to create CemaNeige (if used) model inputs
`HydroModel`	[character] name of the hydrological model (must be one of `"GR1A"`, `"GR2M"`, `"GR4J"`, `"GR5J"`, `"GR6J"` or `"GR4H"`)
`CemaNeige`	[boolean] option indicating whether CemaNeige should be activated (only available when `HydroModel` is equal to any of `"GR4J"`, `"GR5J"` or `"GR6J"`). See details

Details

If the `ObsDF` argument is provided, `DatesR`, `Precip`, `PotEvap`, `Qobs` and `TempMean` are not necessary, and vice-versa. If one variable is provided in `ObsDF` and also separately, then only the data included in `ObsDF` are used.

If the `CemaNeige` argument is set to `TRUE`, the default version of CemaNeige is used (i.e. without the Linear Hysteresis, see the details part in `CreateRunOptions`).

Value

list object of class PrepGR containing the data required to evaluate the model outputs:

InputsModel	[list] object of class *InputsModel* containing the data required to evaluate the model outputs (see: CreateInputsModel outputs)
Qobs	[numeric] time series of observed discharges [mm/time step]
HydroModel	[character] name of the function of the hydrological model used

Author(s)

Olivier Delaigue

See Also

CreateInputsModel

Examples

```
library(airGRteaching)

## data.frame of observed data
data(L0123001, package = "airGR")
BasinObs2 <- BasinObs[, c("DatesR", "P", "E", "Qmm", "T")]

## Preparation of observed data for modelling
PREP <- PrepGR(ObsDF = BasinObs2, HydroModel = "GR4J", CemaNeige = FALSE)
str(PREP)
```

ShinyGR *Interactive Web application to run manually the GR4J, GR5 and GR6J hydrological models with or without CemaNeige*

Description

Shiny application to understand and to display in a interactive way the impact of each parameter of the GR models on the simulated flows

Usage

```
ShinyGR(ObsDF = NULL,
  DatesR = NULL, Precip = NULL, PotEvap = NULL, Qobs = NULL, TempMean = NULL,
  ZInputs = NULL, HypsoData = NULL, NLayers = 5,
  SimPer, NamesObsBV = NULL, theme = "RStudio")
```

Arguments

`ObsDF`	(optional) [data.frame or list of data.frame] `data.frame` of dates, total precipitation, potential evapotranspiration, observed discharge and mean air temperature (only if CemaNeige is used) (variables must be in this order; see below for the units)
`DatesR`	(optional) [POSIXt] vector of dates required to create the GR and CemaNeige models inputs. Time zone must be defined as "UTC"
`Precip`	(optional) [numeric] time series of total precipitation (catchment average) [mm/time step], required to create the GR and CemaNeige models inputs
`PotEvap`	(optional) [numeric] time series of potential evapotranspiration (catchment average) [mm/time step], required to create the GR model inputs
`Qobs`	(optional) [numeric] time series of observed discharge [mm/time step]
`TempMean`	(optional) [numeric] time series of mean air temperature [°C], required to create the CemaNeige model inputs (if used)
`ZInputs`	(optional) [numeric or list of numerics] real giving the mean elevation of the Precip and TempMean series (before extrapolation) [m], used to create the CemaNeige model inputs (if used)
`HypsoData`	(optional) [numeric or list of numerics] vector of 101 reals: min, q01 to q99 and max of catchment elevation distribution [m]; if not defined a single elevation is used for CemaNeige (if used)
`NLayers`	(optional) [numeric or list of numerics] integer giving the number of elevation layers requested [-], required to create CemaNeige model inputs (if used)
`SimPer`	[character or list of characters] vector of 2 values to define the beginning and the end of the simulation period [`"YYYY-mm-dd"` or `"YYYY-mm-dd HH:MM:SS"`], see below for details
`NamesObsBV`	(optional) [character] vector of values to define the data inputs name(s) (if the ObsDF list is not already named)
`theme`	(optional) [character] alternative stylesheet [`"RStudio"` (default), `"Cerulean"`, `"Cyborg"`, `"Flatly"`, `"United"` or `"Yeti"`]

Details

The warm-up period always starts from the first date of the dataset to the time step just before the beginning of the simulation period (`SimPer`).

Author(s)

Olivier Delaigue, Laurent Coron, Pierre Brigode

See Also

`CalGR`, `SimGR`, `plot.CalGR`, `plot.SimGR`

Examples

```
library(airGRteaching)

## data.frame of observed data of a low-land basin
data(L0123001, package = "airGR")
BV_L0123001 <- BasinObs[0001:6000, c("DatesR", "P", "E", "Qmm", "T")]
BI_L0123001 <- BasinInfo

## data.frame of observed data of a mountainous basin
data(L0123002, package = "airGR")
BV_L0123002 <- BasinObs[5000:9999, c("DatesR", "P", "E", "Qmm", "T")]
BI_L0123002 <- BasinInfo

## Interactive simulation step using default parameters
if (interactive()) {
  ShinyGR(ObsDF = list("Low-land basin" = BV_L0123001, "Mountainous basin" = BV_L0123002),
          ZInputs = list(NULL, median(BI_L0123002$HypsoData)),
          HypsoData = list(NULL, BI_L0123002$HypsoData),
          NLayers = list(5, 5),
          SimPer = list(c("1994-01-01", "1998-12-31"), c("2004-01-01", "2006-12-31")),
          theme = "United")
}
```

SimGR *Running one of the GR hydrological models*

Description

Function for running the GR hydrological models

Usage

```
SimGR(PrepGR, CalGR = NULL, Param = NULL, EffCrit = c("NSE", "KGE", "KGE2", "RMSE"),
  WupPer = NULL, SimPer,
  transfo = c("", "sqrt", "log", "inv", "sort"), verbose = TRUE)
```

Arguments

`PrepGR`	[object of class *PrepGR*] see `PrepGR` for details
`CalGR`	(optional) [object of class *CalGR*] see `CalGR` and below for details
`Param`	(optional) [numeric] vector of parameters (the length of the vector depends on the model used), see below for details
`EffCrit`	[character] name of the efficiency criterion (must be one of `"NSE"`, `"KGE"`, `"KGE2"` or `"RMSE"`)
`WupPer`	(optional) [character] vector of 2 values to define the beginning and end of the warm-up period [`"YYYY-mm-dd"` or `"YYYY-mm-dd HH:MM:SS"`]

SimPer	[character] vector of 2 values to define the beginning and end of the simulation period ["YYYY-mm-dd" or "YYYY-mm-dd HH:MM:SS"]
transfo	(optional) [character] name of the transformation applied to discharge for calculating the error criterion (must be one of "", "sqrt", "log", "inv" or "sort")
verbose	(optional) [boolean] logical value indicating if the function is run in verbose mode or not

Details

The user can customize the parameters with the Param argument. The user can also use the parameters resulting from a calibration. In this case, it is necessary to use the CalGR argument.

Value

list object of class SimGR containing:

OptionsSimul	[list] object of class RunOptions (see: CreateRunOptions)
OptionsCrit	[list] object of class InputsCrit (see: CreateInputsCrit)
OutputsModel	[list] object of class OutputsModel (see: RunModel)
Qobs	[numeric] series of observed discharges [mm/time step]
TypeModel	[character] name of the function of the hydrological model used
CalCrit	[character] name of the function that computes the error criterion during the calibration step
EffCrit	[list] name of the function that computes the error criterion during the simulation step
PeriodModel	[list] $WarmUp: vector of 2 POSIXct values defining the beginning and end of the warm-up period; $Run: vector of 2 POSIXct values defining the beginning and end of the calibration period

Author(s)

Olivier Delaigue

See Also

CreateRunOptions, CreateInputsCrit, RunModel, ErrorCrit_RMSE, ErrorCrit_NSE, ErrorCrit_KGE, ErrorCrit_KGE2

Examples

```
library(airGRteaching)

## data.frame of observed data
data(L0123001, package = "airGR")
BasinObs2 <- BasinObs[, c("DatesR", "P", "E", "Qmm", "T")]

## Preparation of observed data for modelling
PREP <- PrepGR(ObsDF = BasinObs2, HydroModel = "GR4J", CemaNeige = FALSE)
```

```
## Calibration step
CAL <- CalGR(PrepGR = PREP, CalCrit = "KGE2",
             WupPer = NULL, CalPer = c("1990-01-01", "1993-12-31"))

## Simulation step using the result of the automatic calibration method to set the model parameters
SIM <- SimGR(PrepGR = PREP, CalGR = CAL, EffCrit = "KGE2",
             WupPer = NULL, SimPer = c("1994-01-01", "1998-12-31"))

## Simulation step using model parameters set by the user
SIM <- SimGR(PrepGR = PREP, Param = c(270.426, 0.984, 108.853, 2.149), EffCrit = "KGE2",
             WupPer = NULL, SimPer = c("1994-01-01", "1998-12-31"))
str(SIM)
```

Index