



airGR: Suite of GR Hydrological Models for Precipitation-Runoff Modelling. Manual of the R package version 1.0.1

Laurent Coron, Charles Perrin, Claude Michel

► To cite this version:

Laurent Coron, Charles Perrin, Claude Michel. airGR: Suite of GR Hydrological Models for Precipitation-Runoff Modelling. Manual of the R package version 1.0.1. 2016, pp.58. 10.15454/EX11NA . hal-02605473

HAL Id: hal-02605473

<https://hal.inrae.fr/hal-02605473>

Submitted on 16 May 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Package ‘airGR’

April 21, 2016

Type Package

Title Suite of GR hydrological models for precipitation-runoff modelling

Version 1.0.1

Date 2016-04-21

Author Laurent Coron, Charles Perrin, with contributions from Vazken Andréassian, Pierre Brigode, Olivier Delaigue, Nicolas Le Moine, Thibaut Mathevet, Safouane Mouelhi, Ludovic Oudin, Raji Pushpalatha, Guillaume Thirel, Audrey Valéry. Based on earlier work by Claude Michel.

Maintainer Olivier Delaigue <airGR@irstea.fr>

Depends R (>= 3.0.1)

Description This package brings into R the hydrological modelling tools developed at Irstea-Antony (HBAN Research Unit, France). The package includes several conceptual rainfall-runoff models (GR4H, GR4J, GR5J, GR6J, GR2M, GR1A), a snowmelt module (Cemaneige) and the associated functions for their calibration and evaluation. Use `help(airGR)` for package description.

License GPL-2

URL <http://webgr.irstea.fr/modeles/?lang=en>

NeedsCompilation yes

Encoding UTF-8

Archs i386, x64

R topics documented:

| | |
|------------------------------|----|
| airGR | 2 |
| BasinInfo | 4 |
| BasinObs | 5 |
| Calibration | 5 |
| Calibration_Michel | 7 |
| CreateCalibOptions | 10 |
| CreateInputsCrit | 12 |
| CreateInputsModel | 15 |
| CreateRunOptions | 17 |

| | |
|---------------------------------------|-----------|
| DataAltExtrapolation_Valery | 19 |
| ErrorCrit | 21 |
| ErrorCrit_KGE | 22 |
| ErrorCrit_KGE2 | 24 |
| ErrorCrit_NSE | 25 |
| ErrorCrit_RMSE | 26 |
| PEdaily_Oudin | 27 |
| plot_OutputsModel | 28 |
| RunModel | 29 |
| RunModel_CemaNeige | 30 |
| RunModel_CemaNeigeGR4J | 31 |
| RunModel_CemaNeigeGR5J | 34 |
| RunModel_CemaNeigeGR6J | 36 |
| RunModel_GR1A | 38 |
| RunModel_GR2M | 40 |
| RunModel_GR4H | 42 |
| RunModel_GR4J | 44 |
| RunModel_GR5J | 46 |
| RunModel_GR6J | 48 |
| SeriesAggreg | 50 |
| TransfoParam | 51 |
| TransfoParam_CemaNeige | 52 |
| TransfoParam_GR1A | 53 |
| TransfoParam_GR2M | 54 |
| TransfoParam_GR4H | 54 |
| TransfoParam_GR4J | 55 |
| TransfoParam_GR5J | 56 |
| TransfoParam_GR6J | 57 |
| Index | 58 |

airGR

Suite of GR hydrological models for precipitation-runoff modelling

Description

This package brings into R the hydrological modelling tools used at Irstea-Antony (HBAN Research Unit, France), including rainfall-runoff models (GR4H, GR4J, GR5J, GR6J, GR2M, GR1A) and a snowmelt module (CemaNeige). Each model core is coded in FORTRAN to ensure low computational time. The other package functions (i.e. mainly the calibration algorithm and the computation of the efficiency criteria) are coded in R.

Functions and objects

The airGR package has been designed to fulfil two major requirements: facilitate the use by non-expert users and allow flexibility regarding the addition of external criteria, models or calibration algorithms. 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 belonging to the [RunModel](#) family require three arguments: *InputsModel*, *RunOptions* and *Param*; please refer to help pages [CreateInputsModel](#) and [CreateRunOptions](#) for further details and examples;

- the functions belonging to the [ErrorCrit](#) family require two arguments: *InputsCrit* and *OutputsModel*; please refer to help pages [CreateInputsCrit](#) and [RunModel](#) for further details and examples;
- the functions belonging to the [Calibration](#) family require four arguments: *InputsModel*, *RunOptions*, *InputsCrit* and *CalibOptions*; please refer to help pages [CreateInputsModel](#), [CreateRunOptions](#), [CreateInputsCrit](#) and [CreateCalibOptions](#) for further details and examples.

In order to limit the risk of mis-use and increase the flexibility of these main functions, we imposed the structure of their arguments and defined their class. Most users will not need to worry about these imposed structures since functions are provided to prepare these arguments for them: [CreateInputsModel](#), [CreateRunOptions](#), [CreateInputsCrit](#), [CreateCalibOptions](#). However, advanced users wishing to supplement the package with their own models will need to comply with these imposed structures and refer to the package source codes to get all the specification requirements.

Models

Six hydrological models and one snow melt and accumulation module are implemented in airGR. The snow module can also be used alone or with the daily hydrological models, and each hydrological model can either be used alone or together with the snow module.

These models can be called within airGR using the following functions:

- [RunModel_GR4H](#): four-parameter hourly lumped hydrological model (Mathevet, 2005)
- [RunModel_GR4J](#): four-parameter daily lumped hydrological model (Perrin et al., 2003)
- [RunModel_GR5J](#): five-parameter daily lumped hydrological model (Le Moine, 2008)
- [RunModel_GR6J](#): six-parameter daily lumped hydrological model (Pushpalatha et al., 2011)
- [RunModel_GR2M](#): two-parameter monthly lumped hydrological model (Mouelhi, 2003 ; Mouelhi et al., 2006a)
- [RunModel_GR1A](#): one-parameter yearly lumped hydrological model (Mouelhi, 2003 ; Mouelhi et al., 2006b)
- [RunModel_CemaNeige](#): two-parameter degree-day snow melt and accumulation daily module (Valéry et al., 2014)
- [RunModel_CemaNeigeGR4J](#): combined use of GR4J and CemaNeige
- [RunModel_CemaNeigeGR5J](#): combined use of GR5J and CemaNeige
- [RunModel_CemaNeigeGR6J](#): combined use of GR6J and CemaNeige

How to get started

To learn how to use the functions from the airGR package, it is recommended to follow the five steps described below:

1. refer to the help for [RunModel_GR4H](#) then run the provided example to assess how to make a simulation ;
2. refer to the help for [CreateInputsModel](#) to understand how the inputs of a model are prepared/organised ;
3. refer to the help for [CreateRunOptions](#) to understand how the run options of a model are parametrised/organised ;
4. refer to the help for [ErrorCrit_NSE](#) and [CreateInputsCrit](#) to understand how the computation of an error criterion is prepared/made ;
5. refer to the help for [Calibration_Michel](#), run the provided example and then refer to the help for [CreateCalibOptions](#) to understand how a model calibration is prepared/made ;

References

- 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 Cemaiege snow accounting routine on 380 catchments, Journal of Hydrology, 517(0): 1176-1187, doi: 1176-1187, doi:10.1016/j.jhydrol.2014.04.058.

BasinInfo

Data sample: characteristics of a fictional catchment (L0123001, L0123002 or L0123003)

Description

R-object containing the code, station's name, area and hypsometric curve of the catchment.

Format

List named 'BasinInfo' containing

- two strings: catchment's code and station's name
- one float: catchment's area in km2
- one numeric vector: catchment's hypsometric curve (min, quantiles 01 to 99 and max) in metres

Examples

```
require(airGR)
data(L0123001)
str(BasinInfo)
```

| | |
|----------|---|
| BasinObs | <i>Data sample: time series of observations of a fictional catchment (L0123001, L0123002 or L0123003)</i> |
|----------|---|

Description

R-object containing the times series of precipitation, temperature, potential evapotranspiration and discharges.

Times series for L0123001 or L0123002 are at the daily time step for use with daily models such as GR4J, GR5J, GR6J, CemaNeigeGR4J, CemaNeigeGR5J and CemaNeigeGR6J. Times series for L0123003 are at the hourly time step for use with hourly models such as GR4H.

Format

Data frame named 'BasinObs' containing

- one POSIXlt vector: time series dates in the POSIXlt format
- five numeric vectors: time series of catchment average precipitation [mm/time step], catchment average air temperature [°C], catchment average potential evapotranspiration [mm/time step], outlet discharge [l/s], outlet discharge [mm/time step]

Examples

```
library(airGR)
data(L0123001)
str(BasinObs)
```

| | |
|-------------|--|
| Calibration | <i>Calibration algorithm which optimises the error criterion selected as objective function using the provided functions</i> |
|-------------|--|

Description

Calibration algorithm which optimises the error criterion selected as objective function using the provided functions.

Usage

```
Calibration(InputsModel, RunOptions, InputsCrit, CalibOptions, FUN_MOD,
  FUN_CRIT, FUN_CALIB = Calibration_Michel, FUN_TRANSFO = NULL,
  verbose = TRUE)
```

Arguments

| | |
|---------------------------|---|
| <code>InputsModel</code> | [object of class <i>InputsModel</i>] see CreateInputsModel for details |
| <code>RunOptions</code> | [object of class <i>RunOptions</i>] see CreateRunOptions for details |
| <code>InputsCrit</code> | [object of class <i>InputsCrit</i>] see CreateInputsCrit for details |
| <code>CalibOptions</code> | [object of class <i>CalibOptions</i>] see CreateCalibOptions for details |
| <code>FUN_MOD</code> | [function] hydrological model function (e.g. <code>RunModel_GR4J</code> , <code>RunModel_CemaNeigeGR4J</code>) |
| <code>FUN_CRIT</code> | [function] error criterion function (e.g. <code>ErrorCrit_RMSE</code> , <code>ErrorCrit_NSE</code>) |
| <code>FUN_CALIB</code> | (optional) [function] calibration algorithm function (e.g. <code>Calibration_Michel</code>), default= <code>Calibration_Michel</code> |
| <code>FUN_TRANSFO</code> | (optional) [function] model parameters transformation function, if the <code>FUN_MOD</code> used is native in the package <code>FUN_TRANSFO</code> is automatically defined |
| <code>verbose</code> | (optional) [boolean] boolean indicating if the function is run in verbose mode or not, default= <code>TRUE</code> |

Value

list see [Calibration_Michel](#)

Author(s)

Laurent Coron (June 2014)

See Also

[Calibration_Michel](#), [ErrorCrit](#), [TransfoParam](#), [CreateInputsModel](#), [CreateRunOptions](#), [CreateInputsCrit](#), [CreateCalibOptions](#).

Examples

```
## loading catchment data
library(airGR)
data(L0123001)

## preparation of InputsModel object
InputsModel <- CreateInputsModel(FUN_MOD = RunModel_GR4J, DatesR = BasinObs$DatesR,
                                Precip = BasinObs$P, PotEvap = BasinObs$E)

## calibration period selection
Ind_Run <- seq(which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="01/01/1990 00:00"),
              which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="31/12/1999 00:00"))

## preparation of RunOptions object
RunOptions <- CreateRunOptions(FUN_MOD = RunModel_GR4J,
                              InputsModel = InputsModel, IndPeriod_Run = Ind_Run)

## calibration criterion: preparation of the InputsCrit object
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_NSE, InputsModel = InputsModel,
                              RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run])

## preparation of CalibOptions object
CalibOptions <- CreateCalibOptions(FUN_MOD = RunModel_GR4J, FUN_CALIB = Calibration_Michel)
```

```

## calibration
OutputsCalib <- Calibration(InputsModel = InputsModel, RunOptions = RunOptions,
                           InputsCrit = InputsCrit, CalibOptions = CalibOptions,
                           FUN_MOD = RunModel_GR4J, FUN_CRIT = ErrorCrit_NSE,
                           FUN_CALIB = Calibration_Michel)

## simulation
Param <- OutputsCalib$ParamFinalR
OutputsModel <- RunModel(InputsModel = InputsModel, RunOptions = RunOptions,
                        Param = Param, FUN = RunModel_GR4J)

## results preview
plot_OutputsModel(OutputsModel = OutputsModel, Qobs = BasinObs$Qmm[Ind_Run])

## efficiency criterion: Nash-Sutcliffe Efficiency
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_NSE, InputsModel = InputsModel,
                              RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run])
OutputsCrit <- ErrorCrit_NSE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ",
          round(OutputsCrit$CritValue, 4), "\n", sep = ""))

## efficiency criterion: Kling-Gupta Efficiency
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_KGE, InputsModel = InputsModel,
                              RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run])
OutputsCrit <- ErrorCrit_KGE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ",
          round(OutputsCrit$CritValue, 4), "\n", sep = ""))

```

Calibration_Michel

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

Description

Calibration algorithm optimises the error criterion selected as objective function.

The algorithm is based on a local search procedure. First, a screening is performed using either a rough predefined grid or a list of parameter sets and then a simple steepest descent local search algorithm is performed.

Usage

```
Calibration_Michel(InputsModel, RunOptions, InputsCrit, CalibOptions,
                  FUN_MOD, FUN_CRIT, FUN_TRANSFO = NULL, verbose = TRUE)
```

Arguments

| | |
|--------------|---|
| InputsModel | [object of class <i>InputsModel</i>] see CreateInputsModel for details |
| RunOptions | [object of class <i>RunOptions</i>] see CreateRunOptions for details |
| InputsCrit | [object of class <i>InputsCrit</i>] see CreateInputsCrit for details |
| CalibOptions | [object of class <i>CalibOptions</i>] see CreateCalibOptions for details |

| | |
|-------------|---|
| FUN_MOD | [function] hydrological model function (e.g. RunModel_GR4J, RunModel_CemaNeigeGR4J) |
| FUN_CRIT | [function] error criterion function (e.g. ErrorCrit_RMSE, ErrorCrit_NSE) |
| FUN_TRANSFO | (optional) [function] model parameters transformation function, if the FUN_MOD used is native in the package FUN_TRANSFO is automatically defined |
| verbose | (optional) [boolean] boolean indicating if the function is run in verbose mode or not, default=TRUE |

Details

A screening is first performed either based on a rough predefined grid (considering various initial values for each parameter) or from a list of initial parameter sets.

The best set identified in this screening is then used as a starting point for the steepest descent local search algorithm.

For this search, since the ranges of parameter values can be quite different, simple mathematical transformations are applied to parameters to make them vary in a similar range and get a similar sensitivity to a predefined search step. This is done using the TransfoParam functions.

During the steepest descent method, at each iteration, we start from a parameter set of NParam values (NParam being the number of free parameters of the chosen hydrological model) and we determine the $2 \times NParam - 1$ new candidates by changing one by one the different parameters (+/- search step).

All these candidates are tested and the best one kept to be the starting point for the next iteration. At the end of each iteration, the search step is either increased or decreased to adapt the progression speed. A composite step can occasionally be done.

The calibration algorithm stops when the search step becomes smaller than a predefined threshold.

Value

list list containing the function outputs organised as follows:

| | |
|------------------------|---|
| <i>\$ParamFinalR</i> | [numeric] parameter set obtained at the end of the calibration |
| <i>\$CritFinal</i> | [numeric] error criterion selected as objective function obtained at the end of the calibration |
| <i>\$NIter</i> | [numeric] number of iterations during the calibration |
| <i>\$NRuns</i> | [numeric] number of model runs done during the calibration |
| <i>\$HistParamR</i> | [numeric] table showing the progression steps in the search for optimal set: parameter values |
| <i>\$HistCrit</i> | [numeric] table showing the progression steps in the search for optimal set: criterion values |
| <i>\$MatBoolCrit</i> | [boolean] table giving the requested and actual time steps over which the model is calibrated |
| <i>\$CritName</i> | [character] name of the calibration criterion used as objective function |
| <i>\$CritBestValue</i> | [numeric] theoretical best criterion value |

Author(s)

Laurent Coron, Claude Michel (August 2013)

References

Michel, C. (1991), Hydrologie appliquée aux petits bassins ruraux, Hydrology handbook (in French), Cemagref, Antony, France.

See Also

[Calibration](#), [RunModel_GR4J](#), [TransfoParam_GR4J](#), [ErrorCrit_RMSE](#), [CreateInputsModel](#), [CreateRunOptions](#),

[CreateInputsCrit](#), [CreateCalibOptions](#).

Examples

```
## loading catchment data
library(airGR)
data(L0123001)

## preparation of InputsModel object
InputsModel <- CreateInputsModel(FUN_MOD = RunModel_GR4J, DatesR = BasinObs$DatesR,
                                Precip = BasinObs$P, PotEvap = BasinObs$E)

## calibration period selection
Ind_Run <- seq(which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="01/01/1990 00:00"),
               which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="31/12/1999 00:00"))

## preparation of RunOptions object
RunOptions <- CreateRunOptions(FUN_MOD = RunModel_GR4J, InputsModel = InputsModel,
                               IndPeriod_Run = Ind_Run)

## calibration criterion: preparation of the InputsCrit object
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_NSE, InputsModel = InputsModel,
                              RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run])

## preparation of CalibOptions object
CalibOptions <- CreateCalibOptions(FUN_MOD = RunModel_GR4J, FUN_CALIB = Calibration_Michel)

## calibration
OutputsCalib <- Calibration_Michel(InputsModel = InputsModel, RunOptions = RunOptions,
                                   InputsCrit = InputsCrit, CalibOptions = CalibOptions,
                                   FUN_MOD = RunModel_GR4J, FUN_CRIT = ErrorCrit_NSE)

## simulation
Param <- OutputsCalib$ParamFinalR
OutputsModel <- RunModel_GR4J(InputsModel = InputsModel,
                              RunOptions = RunOptions, Param = Param)

## results preview
plot_OutputsModel(OutputsModel = OutputsModel, Qobs = BasinObs$Qmm[Ind_Run])

## efficiency criterion: Nash-Sutcliffe Efficiency
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_NSE, InputsModel = InputsModel,
                              RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run])
OutputsCrit <- ErrorCrit_NSE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ",
          round(OutputsCrit$CritValue, 4), "\n", sep = ""))

## efficiency criterion: Kling-Gupta Efficiency
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_KGE, InputsModel = InputsModel,
                              RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run])
OutputsCrit <- ErrorCrit_KGE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ",
          round(OutputsCrit$CritValue, 4), "\n", sep = ""))
```

| | |
|--------------------|---|
| CreateCalibOptions | <i>Creation of the CalibOptions object required but the Calibration functions</i> |
|--------------------|---|

Description

Creation of the CalibOptions object required by the Calibration functions.

Usage

```
CreateCalibOptions(FUN_MOD, FUN_CALIB = Calibration_Michel,
  FUN_TRANSFO = NULL, OptimParam = NULL, FixedParam = NULL,
  SearchRanges = NULL, StartParam = NULL, StartParamList = NULL,
  StartParamDistrib = NULL)
```

Arguments

| | |
|--------------|--|
| FUN_MOD | [function] hydrological model function (e.g. RunModel_GR4J, RunModel_CemaNeigeGR4J) |
| FUN_CALIB | (optional) [function] calibration algorithm function (e.g. Calibration_Michel), default=Calibration_Michel |
| FUN_TRANSFO | (optional) [function] model parameters transformation function, if the FUN_MOD used is native in the package, FUN_TRANSFO is automatically defined |
| OptimParam | (optional) [boolean] vector of booleans indicating which parameters must be optimised (NParam columns, 1 line) |
| FixedParam | (optional) [numeric] vector giving the values set for the non-optimised parameter values (NParam columns, 1 line) |
| SearchRanges | (optional) [numeric] matrix giving the ranges of real parameters (NParam columns, 2 lines) Example: |

| | [X1] | [X2] | [X3] | [...] | [Xi] |
|------|------|------|------|-------|------|
| [1,] | 0 | -1 | 0 | ... | 0.0 |
| [2,] | 3000 | +1 | 100 | ... | 3.0 |

| | |
|------------|---|
| StartParam | (optional) [numeric] vector of parameter values used to start global search calibration procedure Example: |
|------------|---|

| | [X1] | [X2] | [X3] | [...] | [Xi] |
|--|------|------|------|-------|------|
| | 1000 | -0.5 | 22 | ... | 1.1 |

| | |
|----------------|---|
| StartParamList | (optional) [numeric] matrix of parameter sets used for grid-screening calibration procedure (values in columns, sets in line) Example: |
|----------------|---|

| | [X1] | [X2] | [X3] | [...] | [Xi] |
|--------|------|------|------|-------|------|
| [set1] | 800 | -0.7 | 25 | ... | 1.0 |

| | | | | | |
|---------|------|------|-----|-----|-----|
| [set2] | 1000 | -0.5 | 22 | ... | 1.1 |
| [...] | ... | ... | ... | ... | ... |
| [set n] | 200 | -0.3 | 17 | ... | 1.0 |

StartParamDistrib

(optional) [numeric] matrix of parameter values used for grid-screening calibration procedure (values in columns, percentiles in line)

Example:

| | | | | | |
|----------|------|------|------|-------|------|
| | [X1] | [X2] | [X3] | [...] | [Xi] |
| [value1] | 800 | -0.7 | 25 | ... | 1.0 |
| [value2] | 1000 | NA | 50 | ... | 1.2 |
| [value3] | 1200 | NA | NA | ... | 1.6 |

Details

Users wanting to use FUN_MOD, FUN_CALIB or FUN_TRANSFO functions that are not included in the package must create their own CalibOptions object accordingly.

Value

list object of class *CalibOptions* containing the data required to evaluate the model outputs; it can include the following:

| | |
|----------------------------|--|
| <i>\$OptimParam</i> | [boolean] vector of booleans indicating which parameters must be optimised |
| <i>\$FixedParam</i> | [numeric] vector giving the values to allocate to non-optimised parameter values |
| <i>\$SearchRanges</i> | [numeric] matrix giving the ranges of raw parameters |
| <i>\$StartParam</i> | [numeric] vector of parameter values used to start global search calibration procedure |
| <i>\$StartParamList</i> | [numeric] matrix of parameter sets used for grid-screening calibration procedure |
| <i>\$StartParamDistrib</i> | [numeric] matrix of parameter values used for grid-screening calibration procedure |

Author(s)

Laurent Coron (June 2014)

See Also

[Calibration](#), [RunModel](#)

Examples

```
## loading catchment data
library(airGR)
data(L0123001)

## preparation of InputsModel object
InputsModel <- CreateInputsModel(FUN_MOD = RunModel_GR4J, DatesR = BasinObs$DatesR,
                                Precip = BasinObs$P, PotEvap = BasinObs$E)

## calibration period selection
Ind_Run <- seq(which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="01/01/1990 00:00"),
```

```

which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="31/12/1999 00:00"))

## preparation of RunOptions object
RunOptions <- CreateRunOptions(FUN_MOD = RunModel_GR4J,
                              InputsModel = InputsModel, IndPeriod_Run = Ind_Run)

## calibration criterion: preparation of the InputsCrit object
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_NSE, InputsModel = InputsModel,
                              RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run])

## preparation of CalibOptions object
CalibOptions <- CreateCalibOptions(FUN_MOD = RunModel_GR4J, FUN_CALIB = Calibration_Michel)

## calibration
OutputsCalib <- Calibration(InputsModel = InputsModel, RunOptions = RunOptions,
                            InputsCrit = InputsCrit, CalibOptions = CalibOptions,
                            FUN_MOD = RunModel_GR4J, FUN_CRIT = ErrorCrit_NSE,
                            FUN_CALIB = Calibration_Michel)

## simulation
Param <- OutputsCalib$ParamFinalR
OutputsModel <- RunModel(InputsModel = InputsModel, RunOptions = RunOptions,
                        Param = Param, FUN = RunModel_GR4J)

## results preview
plot_OutputsModel(OutputsModel = OutputsModel, Qobs = BasinObs$Qmm[Ind_Run])

## efficiency criterion: Nash-Sutcliffe Efficiency
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_NSE, InputsModel = InputsModel,
                              RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run])
OutputsCrit <- ErrorCrit_NSE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ", round(OutputsCrit$CritValue, 4), "\n", sep = ""))

## efficiency criterion: Kling-Gupta Efficiency
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_KGE, InputsModel = InputsModel,
                              RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run])
OutputsCrit <- ErrorCrit_KGE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ", round(OutputsCrit$CritValue, 4), "\n", sep = ""))

```

CreateInputsCrit

Creation of the InputsCrit object required to the ErrorCrit functions

Description

Creation of the InputsCrit object required to the ErrorCrit functions.

Usage

```

CreateInputsCrit(FUN_CRIT, InputsModel, RunOptions, Qobs, BoolCrit = NULL,
                transfo = "", Ind_zeroes = NULL, epsilon = NULL)

```

Arguments

| | |
|--------------------------|--|
| <code>FUN_CRIT</code> | [function] error criterion function (e.g. <code>ErrorCrit_RMSE</code> , <code>ErrorCrit_NSE</code>) |
| <code>InputsModel</code> | [object of class <i>InputsModel</i>] see CreateInputsModel for details |
| <code>RunOptions</code> | [object of class <i>RunOptions</i>] see CreateRunOptions for details |
| <code>Qobs</code> | [numeric] series of observed discharges [mm/time step] |
| <code>BoolCrit</code> | (optional) [boolean] boolean giving the time steps to consider in the computation (all time steps are consider by default) |
| <code>transfo</code> | (optional) [character] name of the transformation (e.g. "", "sqrt", "log", "inv", "sort") |
| <code>Ind_zeroes</code> | (optional) [numeric] indices of the time steps where zeroes are observed |
| <code>epsilon</code> | (optional) [numeric] epsilon to add to all <code>Qobs</code> and <code>Qsim</code> if <code>\$Ind_zeroes</code> is not empty |

Details

Users wanting to use `FUN_CRIT` functions that are not included in the package must create their own `InputsCrit` object accordingly.

Value

list object of class *InputsCrit* containing the data required to evaluate the model outputs; it can include the following:

| | |
|---------------------------|---|
| <code>\$BoolCrit</code> | [boolean] boolean giving the time steps considered in the computation |
| <code>\$Qobs</code> | [numeric] series of observed discharges [mm/time step] |
| <code>\$transfo</code> | [character] name of the transformation (e.g. "", "sqrt", "log", "inv", "sort") |
| <code>\$Ind_zeroes</code> | [numeric] indices of the time steps where zeroes are observed |
| <code>\$epsilon</code> | [numeric] epsilon to add to all <code>Qobs</code> and <code>Qsim</code> if <code>\$Ind_zeroes</code> is not empty |

Author(s)

Laurent Coron (June 2014)

See Also

[RunModel](#), [CreateInputsModel](#), [CreateRunOptions](#), [CreateCalibOptions](#)

Examples

```
## loading catchment data
library(airGR)
data(L0123001)

## preparation of the InputsModel object
InputsModel <- CreateInputsModel(FUN_MOD = RunModel_GR4J, DatesR = BasinObs$DatesR,
                                Precip = BasinObs$P, PotEvap = BasinObs$E)

## run period selection
Ind_Run <- seq(which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="01/01/1990 00:00"),
               which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="31/12/1999 00:00"))
```

```

## preparation of the RunOptions object
RunOptions <- CreateRunOptions(FUN_MOD = RunModel_GR4J,
                              InputsModel = InputsModel, IndPeriod_Run = Ind_Run)

## simulation
Param <- c(734.568, -0.840, 109.809, 1.971)
OutputsModel <- RunModel(InputsModel = InputsModel, RunOptions = RunOptions,
                        Param = Param, FUN = RunModel_GR4J)

## efficiency criterion: Nash-Sutcliffe Efficiency
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_NSE, InputsModel = InputsModel,
                              RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run])
OutputsCrit <- ErrorCrit_NSE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ", round(OutputsCrit$CritValue, 4), "\n", sep = ""))

## efficiency criterion: Nash-Sutcliffe Efficiency on log-transformed flows
transfo <- "log"
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_NSE, InputsModel = InputsModel,
                              RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run],
                              transfo = transfo)
OutputsCrit <- ErrorCrit_NSE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ",
          round(OutputsCrit$CritValue, 4), "\n", sep = ""))

## efficiency criterion: Nash-Sutcliffe Efficiency above a threshold (q75%)
BoolCrit <- rep(TRUE, length(BasinObs$Qmm[Ind_Run]));
BoolCrit[BasinObs$Qmm[Ind_Run]<quantile(BasinObs$Qmm[Ind_Run], 0.75, na.rm = TRUE)] <- FALSE
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_NSE, InputsModel = InputsModel,
                              RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run],
                              BoolCrit = BoolCrit)
OutputsCrit <- ErrorCrit_NSE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ", round(OutputsCrit$CritValue, 4), "\n", sep = ""))

## efficiency criterion: Kling-Gupta Efficiency
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_KGE, InputsModel = InputsModel,
                              RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run])
OutputsCrit <- ErrorCrit_KGE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ", round(OutputsCrit$CritValue, 4), "\n", sep = ""))
cat(paste("SubCrit ", OutputsCrit$SubCritNames, " ",
          round(OutputsCrit$SubCritValues, 4), "\n", sep = ""))

## efficiency criterion: Kling-Gupta Efficiency below a threshold (q10%) on log-transformed flows
transfo <- "log"
BoolCrit <- rep(TRUE, length(BasinObs$Qmm[Ind_Run]));
BoolCrit[BasinObs$Qmm[Ind_Run]>quantile(BasinObs$Qmm[Ind_Run], 0.10, na.rm = TRUE)] <- FALSE
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_KGE,
                              InputsModel = InputsModel, RunOptions = RunOptions,
                              Qobs = BasinObs$Qmm[Ind_Run], BoolCrit = BoolCrit, transfo = transfo)
OutputsCrit <- ErrorCrit_KGE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ", round(OutputsCrit$CritValue, 4), "\n", sep = ""))
cat(paste("SubCrit ", OutputsCrit$SubCritNames, " ",
          round(OutputsCrit$SubCritValues, 4), "\n", sep = ""))

```

| | |
|-------------------|--|
| CreateInputsModel | <i>Creation of the InputsModel object required to the RunModel functions</i> |
|-------------------|--|

Description

Creation of the InputsModel object required to the RunModel functions.

Usage

```
CreateInputsModel(FUN_MOD, DatesR, Precip, PotEvap = NULL, TempMean = NULL,
  TempMin = NULL, TempMax = NULL, ZInputs = NULL, HypsoData = NULL,
  NLayers = 5, verbose = TRUE)
```

Arguments

| | |
|-----------|--|
| FUN_MOD | [function] hydrological model function (e.g. RunModel_GR4J, RunModel_CemaNeigeGR4J) |
| DatesR | [POSIXlt] vector of dates required to create the GR model and CemaNeige module inputs |
| Precip | [numeric] time series of total precipitation (catchment average) [mm], required to create the GR model and CemaNeige module inputs |
| PotEvap | [numeric] time series of potential evapotranspiration (catchment average) [mm], required to create the GR model inputs |
| TempMean | (optional) [numeric] time series of mean air temperature [°C], required to create the CemaNeige module inputs |
| TempMin | (optional) [numeric] time series of min air temperature [°C], possibly used to create the CemaNeige module inputs |
| TempMax | (optional) [numeric] time series of max air temperature [°C], possibly used to create the CemaNeige module inputs |
| ZInputs | (optional) [numeric] real giving the mean elevation of the Precip and Temp series (before extrapolation) [m] |
| HypsoData | (optional) [numeric] vector of 101 reals: min, q01 to q99 and max of catchment elevation distribution [m], required to create the GR model inputs, if not defined a single elevation is used for CemaNeige |
| NLayers | (optional) [numeric] integer giving the number of elevation layers requested [-], required to create the GR model inputs, default=5 |
| verbose | (optional) [boolean] boolean indicating if the function is run in verbose mode or not, default=TRUE |

Details

Users wanting to use FUN_MOD functions that are not included in the package must create their own InputsModel object accordingly.

Value

list object of class *InputsModel* containing the data required to evaluate the model outputs; it can include the following:

| | |
|-------------------------------|---|
| <i>\$DatesR</i> | [POSIXlt] vector of dates |
| <i>\$Precip</i> | [numeric] time series of total precipitation (catchment average) [mm] |
| <i>\$PotEvap</i> | [numeric] time series of potential evapotranspiration (catchment average) [mm], defined if FUN_MOD includes GR4H, GR4J, GR5J, GR6J, GR2M or GR1A |
| <i>\$LayerPrecip</i> | [list] list of time series of precipitation (layer average) [mm], defined if FUN_MOD includes CemaNeige |
| <i>\$LayerTempMean</i> | [list] list of time series of mean air temperature (layer average) [°C], defined if FUN_MOD includes CemaNeige |
| <i>\$LayerFracSolidPrecip</i> | [list] list of time series of solid precipitation fraction (layer average) [-], defined if FUN_MOD includes CemaNeige |

Author(s)

Laurent Coron (June 2014)

See Also

[RunModel](#), [CreateRunOptions](#), [CreateInputsCrit](#), [CreateCalibOptions](#), [DataAltiExtrapolation_Valery](#)

Examples

```
## loading catchment data
library(airGR)
data(L0123001)

## preparation of the InputsModel object
InputsModel <- CreateInputsModel(FUN_MOD = RunModel_GR4J, DatesR = BasinObs$DatesR,
                                Precip = BasinObs$P, PotEvap = BasinObs$E)

## run period selection
Ind_Run <- seq(which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="01/01/1990 00:00"),
               which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="31/12/1999 00:00"))

## preparation of the RunOptions object
RunOptions <- CreateRunOptions(FUN_MOD = RunModel_GR4J,
                               InputsModel = InputsModel, IndPeriod_Run = Ind_Run)

## simulation
Param <- c(734.568, -0.840, 109.809, 1.971)
OutputsModel <- RunModel(InputsModel = InputsModel, RunOptions = RunOptions, Param = Param,
                         FUN_MOD = RunModel_GR4J)

## results preview
plot_OutputsModel(OutputsModel = OutputsModel, Qobs = BasinObs$Qmm[Ind_Run])

## efficiency criterion: Nash-Sutcliffe Efficiency
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_NSE, InputsModel = InputsModel,
                              RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run])
OutputsCrit <- ErrorCrit_NSE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ", round(OutputsCrit$CritValue, 4), "\n", sep = ""))
```

CreateRunOptions

*Creation of the RunOptions object required to the RunModel functions***Description**

Creation of the RunOptions object required to the RunModel functions.

Usage

```
CreateRunOptions(FUN_MOD, InputsModel, IndPeriod_WarmUp = NULL, IndPeriod_Run,
  IniStates = NULL, IniResLevels = NULL, Outputs_Cal = NULL,
  Outputs_Sim = "all", RunSnowModule = TRUE, MeanAnSolidPrecip = NULL,
  verbose = TRUE)
```

Arguments

| | |
|-------------------|--|
| FUN_MOD | [function] hydrological model function (e.g. RunModel_GR4J, RunModel_CemaNeigeGR4J) |
| InputsModel | [object of class <i>InputsModel</i>] see CreateInputsModel for details |
| IndPeriod_WarmUp | (optional) [numeric] index of period to be used for the model warm-up [-] |
| IndPeriod_Run | [numeric] index of period to be used for the model run [-] |
| IniStates | (optional) [numeric] vector of initial model internal states [mm] |
| IniResLevels | (optional) [numeric] vector of initial filling rates for production and routing stores (2 values between 0 and 1) [-] |
| Outputs_Cal | (optional) [character] vector giving the outputs needed for the calibration (e.g. c("Qsim")), the fewer outputs the faster the calibration |
| Outputs_Sim | (optional) [character] vector giving the requested outputs (e.g. c("DatesR", "Qsim", "SnowPack")), default="all" |
| RunSnowModule | (optional) [boolean] option indicating whether CemaNeige should be activated, default=TRUE |
| MeanAnSolidPrecip | (optional) [numeric] vector giving the annual mean of average solid precipitation for each layer (computed from InputsModel if not defined) [mm/y] |
| verbose | (optional) [boolean] boolean indicating if the function is run in verbose mode or not, default=TRUE |

Details

Users wanting to use FUN_MOD functions that are not included in the package must create their own RunOptions object accordingly.

Initialisation options

The model initialisation options can either be set to a default configuration or be defined by the user.

This is done via three vectors:

IndPeriod_WarmUp, *IniStates*, *IniResLevels*.

A default configuration is used for initialisation if these vectors are not defined.

(1) Default initialisation options:

- *IndPeriod_WarmUp* default setting ensures a one-year warm-up using the time steps preceding the *IndPeriod_Run*. The actual length of this warm-up might be shorter depending on data availability (no missing value of climate inputs being allowed in model input series).
- *IniStates* and *IniResLevels* are automatically set to initialise all the model states at 0, except for the production and routing stores which are initialised at 50 % of their capacity. This initialisation is made at the very beginning of the model call (i.e. at the beginning of *IndPeriod_WarmUp* or at the beginning of *IndPeriod_Run* if the warm-up period is disabled).

(2) Customisation of initialisation options:

- *IndPeriod_WarmUp* can be used to specify the indices of the warm-up period (within the time-series prepared in *InputsModel*).
 - remark 1: for most common cases, indices corresponding to one or several years preceding *IndPeriod_Run* are used (e.g. *IndPeriod_WarmUp* = 1000:1365 and *IndPeriod_Run* = 1366:5000).
 - However, it is also possible to perform a long-term initialisation if other indices than the warm-up ones are set in *IndPeriod_WarmUp* (e.g. *IndPeriod_WarmUp* <- c(1:5000 , 1:5000 , 1:5000 , 1000:1365)).
 - remark 2: it is also possible to completely disable the warm-up period when using *IndPeriod_WarmUp* = 0L.
- *IniStates* and *IniResLevels* can be used to specify the initial model states.
 - remark 1: if *IniStates* is used, all model states must be provided (e.g. 60 floats [mm] are required for GR4J, GR5J and GR6J; 60+2*NLayers floats [mm] are required for CemaNeigeGR4J, CemaNeigeGR5J and CemaNeigeGR6J; see fortran source code for details).
 - remark 2: in addition to *IniStates*, *IniResLevels* allows to set the filling rate of the production and routing stores for the GR models. For instance for GR4J, GR5J and GR6J: *IniResLevels* <- c(0.3,0.5) should be used to obtain initial fillings of 30% and 50% for the production and routing stores, respectively. *IniResLevels* is optional and can only be used if *IniStates* is also defined (the state values corresponding to these two stores in *IniStates* are not used in such case).

Value

list object of class *RunOptions* containing the data required to evaluate the model outputs; it can include the following:

| | |
|--------------------------|--|
| <i>IndPeriod_WarmUp</i> | [numeric] index of period to be used for the model warm-up [-] |
| <i>IndPeriod_Run</i> | [numeric] index of period to be used for the model run [-] |
| <i>IniStates</i> | [numeric] vector of initial model states [mm] |
| <i>IniResLevels</i> | [numeric] vector of initial filling rates for production and routing stores [-] |
| <i>Outputs_Cal</i> | [character] character vector giving only the outputs needed for the calibration |
| <i>Outputs_Sim</i> | [character] character vector giving the requested outputs |
| <i>RunSnowModule</i> | [boolean] option indicating whether CemaNeige should be activated |
| <i>MeanAnSolidPrecip</i> | [numeric] vector giving the annual mean of average solid precipitation for each layer [mm/y] |

Author(s)

Laurent Coron (June 2014)

See Also

[RunModel](#), [CreateInputsModel](#), [CreateInputsCrit](#), [CreateCalibOptions](#)

Examples

```
## loading catchment data
library(airGR)
data(L0123001)

## preparation of the InputsModel object
InputsModel <- CreateInputsModel(FUN_MOD = RunModel_GR4J, DatesR = BasinObs$DatesR,
                                Precip = BasinObs$P, PotEvap = BasinObs$E)

## run period selection
Ind_Run <- seq(which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="01/01/1990 00:00"),
               which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="31/12/1999 00:00"))

## preparation of the RunOptions object
RunOptions <- CreateRunOptions(FUN_MOD = RunModel_GR4J,
                               InputsModel = InputsModel, IndPeriod_Run = Ind_Run)

## simulation
Param <- c(734.568, -0.840, 109.809, 1.971)
OutputsModel <- RunModel(InputsModel = InputsModel, RunOptions = RunOptions, Param = Param,
                        FUN_MOD = RunModel_GR4J)

## results preview
plot_OutputsModel(OutputsModel = OutputsModel, Qobs = BasinObs$Qmm[Ind_Run])

## efficiency criterion: Nash-Sutcliffe Efficiency
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_NSE, InputsModel = InputsModel,
                              RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run])
OutputsCrit <- ErrorCrit_NSE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ", round(OutputsCrit$CritValue, 4), "\n", sep = ""))
```

DataAltiExtrapolation_Valery

NA

Description

Function which extrapolates the precipitation and air temperature series for different elevation layers (method from Valery, 2010).

Usage

```
DataAltiExtrapolation_Valery(DatesR, Precip, TempMean, TempMin = NULL,
                             TempMax = NULL, ZInputs, HypsoData, NLayers, verbose = TRUE)
```

Arguments

| | |
|-----------|---|
| DatesR | [POSIXlt] vector of dates |
| Precip | [numeric] time series of daily total precipitation (catchment average) [mm] |
| TempMean | [numeric] time series of daily mean air temperature [°C] |
| TempMin | (optional) [numeric] time series of daily min air temperature [°C] |
| TempMax | (optional) [numeric] time series of daily max air temperature [°C] |
| ZInputs | [numeric] real giving the mean elevation of the Precip and Temp series (before extrapolation) [m] |
| HypsoData | [numeric] vector of 101 reals: min, q01 to q99 and max of catchment elevation distribution [m] |
| NLayers | [numeric] integer giving the number of elevation layers requested [-] |
| verbose | (optional) [boolean] boolean indicating if the function is run in verbose mode or not, default=TRUE |

Details

Elevation layers of equal surface are created the 101 elevation quantiles (*HypsoData*) and the number requested elevation layers (*NLayers*).

Forcing data (precipitation and air temperature) are extrapolated using gradients from Valéry (2010). (e.g. gradP=0.0004 [m-1] for France and gradT=0.434 [degreC/100m] for January, 1st).

This function is used by the *CreateInputsModel* function.

Value

list containing the extrapolated series of precip. and air temp. on each elevation layer

| | |
|-------------------------------|---|
| <i>\$LayerPrecip</i> | [list] list of time series of daily precipitation (layer average) [mm] |
| <i>\$LayerTempMean</i> | [list] list of time series of daily mean air temperature (layer average) [°C] |
| <i>\$LayerTempMin</i> | [list] list of time series of daily min air temperature (layer average) [°C] |
| <i>\$LayerTempMax</i> | [list] list of time series of daily max air temperature (layer average) [°C] |
| <i>\$LayerFracSolidPrecip</i> | [list] list of time series of daily solid precip. fract. (layer average) [-] |
| <i>\$ZLayers</i> | [numeric] vector of median elevation for each layer |

Author(s)

Laurent Coron, Audrey Valéry, Pierre Brigode (June 2014)

References

- Turcotte, R., L.-G. Fortin, V. Fortin, J.-P. Fortin and J.-P. Villeneuve (2007), Operational analysis of the spatial distribution and the temporal evolution of the snowpack water equivalent in southern Quebec, Canada, *Nordic Hydrology*, 38(3), 211, doi:10.2166/nh.2007.009.
- Valéry, A. (2010), Modélisation précipitations-débit sous influence nivale ? : Elaboration d'un module neige et évaluation sur 380 bassins versants, PhD thesis (in french), AgroParisTech, Paris, France.
- USACE (1956), *Snow Hydrology*, pp. 437, U.S. Army Corps of Engineers (USACE) North Pacific Division, Portland, Oregon, USA.

See Also

[CreateInputsModel](#), [RunModel_CemaNeigeGR4J](#)

| | |
|-----------|--|
| ErrorCrit | <i>Error criterion using the provided function</i> |
|-----------|--|

Description

Function which computes an error criterion with the provided function.

Usage

```
ErrorCrit(InputsCrit, OutputsModel, FUN_CRIT, verbose = TRUE)
```

Arguments

| | |
|--------------|--|
| InputsCrit | [object of class <i>InputsCrit</i>] see CreateInputsCrit for details |
| OutputsModel | [object of class <i>OutputsModel</i>] see RunModel_GR4J or RunModel_CemaNeigeGR4J for details |
| FUN_CRIT | [function] error criterion function (e.g. ErrorCrit_RMSE , ErrorCrit_NSE) |
| verbose | (optional) [boolean] boolean indicating if the function is run in verbose mode or not, default=TRUE |

Value

list list containing the function outputs, see [ErrorCrit_RMSE](#) or [ErrorCrit_NSE](#) for details

Author(s)

Laurent Coron (June 2014)

See Also

[ErrorCrit_RMSE](#), [ErrorCrit_NSE](#), [ErrorCrit_KGE](#)

Examples

```
## loading catchment data
library(airGR)
data(L0123001)

## preparation of the InputsModel object
InputsModel <- CreateInputsModel(FUN_MOD = RunModel_GR4J, DatesR = BasinObs$DatesR,
                                Precip = BasinObs$P, PotEvap = BasinObs$E)

## run period selection
Ind_Run <- seq(which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="01/01/1990 00:00"),
              which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="31/12/1999 00:00"))

## preparation of the RunOptions object
RunOptions <- CreateRunOptions(FUN_MOD = RunModel_GR4J,
                              InputsModel = InputsModel, IndPeriod_Run = Ind_Run)
```

```

## simulation
Param <- c(734.568, -0.840, 109.809, 1.971)
OutputsModel <- RunModel(InputsModel = InputsModel, RunOptions = RunOptions,
                        Param = Param, FUN = RunModel_GR4J)

## efficiency criterion: Nash-Sutcliffe Efficiency
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_NSE, InputsModel = InputsModel,
                             RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run])
OutputsCrit <- ErrorCrit_NSE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ", round(OutputsCrit$CritValue, 4), "\n", sep = ""))

## efficiency criterion: Nash-Sutcliffe Efficiency on log-transformed flows
transfo <- "log"
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_NSE, InputsModel = InputsModel,
                             RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run],
                             transfo = transfo)
OutputsCrit <- ErrorCrit_NSE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ", round(OutputsCrit$CritValue, 4), "\n", sep = ""))

## efficiency criterion: Nash-Sutcliffe Efficiency above a threshold (q75%)
BoolCrit <- rep(TRUE, length(BasinObs$Qmm[Ind_Run]))
BoolCrit[BasinObs$Qmm[Ind_Run]<quantile(BasinObs$Qmm[Ind_Run], 0.75, na.rm = TRUE)] <- FALSE
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_NSE, InputsModel = InputsModel,
                             RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run],
                             BoolCrit = BoolCrit)
OutputsCrit <- ErrorCrit_NSE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ", round(OutputsCrit$CritValue, 4), "\n", sep = ""))

## efficiency criterion: Kling-Gupta Efficiency
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_KGE, InputsModel = InputsModel,
                             RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run])
OutputsCrit <- ErrorCrit_KGE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ", round(OutputsCrit$CritValue, 4), "\n", sep = ""))
cat(paste("SubCrit ", OutputsCrit$SubCritNames, " ",
        round(OutputsCrit$SubCritValues, 4), "\n", sep = ""))

## efficiency criterion: Kling-Gupta Efficiency below a threshold (q10%) on log-transformed flows
transfo <- "log"
BoolCrit <- rep(TRUE, length(BasinObs$Qmm[Ind_Run]))
BoolCrit[BasinObs$Qmm[Ind_Run]>quantile(BasinObs$Qmm[Ind_Run], 0.10, na.rm = TRUE)] <- FALSE
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_KGE, InputsModel = InputsModel,
                             RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run],
                             BoolCrit = BoolCrit, transfo = transfo)
OutputsCrit <- ErrorCrit_KGE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ", round(OutputsCrit$CritValue, 4), "\n", sep = ""))
cat(paste("SubCrit ", OutputsCrit$SubCritNames, " ",
        round(OutputsCrit$SubCritValues, 4), "\n", sep = ""))

```

Description

Function which computes an error criterion based on the KGE formula proposed by Gupta et al. (2009).

Usage

```
ErrorCrit_KGE(InputsCrit, OutputsModel, verbose = TRUE)
```

Arguments

| | |
|--------------|--|
| InputsCrit | [object of class <i>InputsCrit</i>] see CreateInputsCrit for details |
| OutputsModel | [object of class <i>OutputsModel</i>] see RunModel_GR4J or RunModel_CemaNeigeGR4J for details |
| verbose | (optional) [boolean] boolean indicating if the function is run in verbose mode or not, default=TRUE |

Details

In addition to the criterion value, the function outputs include a multiplier (-1 or +1) which allows the use of the function for model calibration: the product CritValue*Multiplier is the criterion to be minimised (Multiplier=-1 for KGE).

Value

list list containing the function outputs organised as follows:

| | |
|--------------------------|--|
| <i>\$CritValue</i> | [numeric] value of the criterion |
| <i>\$CritName</i> | [character] name of the criterion |
| <i>\$SubCritValues</i> | [numeric] values of the sub-criteria |
| <i>\$SubCritNames</i> | [character] names of the components of the criterion |
| <i>\$CritBestValue</i> | [numeric] theoretical best criterion value |
| <i>\$Multiplier</i> | [numeric] integer indicating whether the criterion is indeed an error (+1) or an efficiency (-1) |
| <i>\$Ind_notcomputed</i> | [numeric] indices of the time steps where InputsCrit\$BoolCrit=FALSE or no data is available |

Author(s)

Laurent Coron (June 2014)

References

Gupta, H. V., Kling, H., Yilmaz, K. K. and Martinez, G. F. (2009), Decomposition of the mean squared error and NSE performance criteria: Implications for improving hydrological modelling, *Journal of Hydrology*, 377(1-2), 80-91, doi:10.1016/j.jhydrol.2009.08.003.

See Also

[ErrorCrit_RMSE](#), [ErrorCrit_NSE](#), [ErrorCrit_KGE2](#)

Examples

```
## see example of the ErrorCrit function
```


ErrorCrit_KGE2

*Error criterion based on the KGE' formula***Description**

Function which computes an error criterion based on the KGE' formula proposed by Kling et al. (2012).

Usage

```
ErrorCrit_KGE2(InputsCrit, OutputsModel, verbose = TRUE)
```

Arguments

| | |
|--------------|--|
| InputsCrit | [object of class <i>InputsCrit</i>] see CreateInputsCrit for details |
| OutputsModel | [object of class <i>OutputsModel</i>] see RunModel_GR4J or RunModel_CemaNeigeGR4J for details |
| verbose | (optional) [boolean] boolean indicating if the function is run in verbose mode or not, default=TRUE |

Details

In addition to the criterion value, the function outputs include a multiplier (-1 or +1) which allows the use of the function for model calibration: the product CritValue*Multiplier is the criterion to be minimised (Multiplier=-1 for KGE2).

Value

list list containing the function outputs organised as follows:

| | |
|--------------------------|--|
| <i>\$CritValue</i> | [numeric] value of the criterion |
| <i>\$CritName</i> | [character] name of the criterion |
| <i>\$SubCritValues</i> | [numeric] values of the sub-criteria |
| <i>\$SubCritNames</i> | [character] names of the components of the criterion |
| <i>\$CritBestValue</i> | [numeric] theoretical best criterion value |
| <i>\$Multiplier</i> | [numeric] integer indicating whether the criterion is indeed an error (+1) or an efficiency (-1) |
| <i>\$Ind_notcomputed</i> | [numeric] indices of the time steps where InputsCrit\$BoolCrit=FALSE or no data is available |

Author(s)

Laurent Coron (June 2014)

References

Gupta, H. V., Kling, H., Yilmaz, K. K. and Martinez, G. F. (2009), Decomposition of the mean squared error and NSE performance criteria: Implications for improving hydrological modelling, *Journal of Hydrology*, 377(1-2), 80-91, doi:10.1016/j.jhydrol.2009.08.003.

Kling, H., Fuchs, M. and Paulin, M. (2012), Runoff conditions in the upper Danube basin under an ensemble of climate change scenarios, *Journal of Hydrology*, 424-425, 264-277, doi:10.1016/j.jhydrol.2012.01.011.

See Also

[ErrorCrit_RMSE](#), [ErrorCrit_NSE](#), [ErrorCrit_KGE](#)

Examples

```
## see example of the ErrorCrit function
```

| | |
|---------------|---|
| ErrorCrit_NSE | <i>Error criterion based on the NSE formula</i> |
|---------------|---|

Description

Function which computes an error criterion based on the NSE formula proposed by Nash & Sutcliffe (1970).

Usage

```
ErrorCrit_NSE(InputsCrit, OutputsModel, verbose = TRUE)
```

Arguments

| | |
|--------------|--|
| InputsCrit | [object of class <i>InputsCrit</i>] see CreateInputsCrit for details |
| OutputsModel | [object of class <i>OutputsModel</i>] see RunModel_GR4J or RunModel_CemaNeigeGR4J for details |
| verbose | (optional) [boolean] boolean indicating if the function is run in verbose mode or not, default=TRUE |

Details

In addition to the criterion value, the function outputs include a multiplier (-1 or +1) which allows the use of the function for model calibration: the product CritValue*Multiplier is the criterion to be minimised (Multiplier=-1 for NSE).

Value

list list containing the function outputs organised as follows:

| | |
|--------------------------|--|
| <i>\$CritValue</i> | [numeric] value of the criterion |
| <i>\$CritName</i> | [character] name of the criterion |
| <i>\$CritBestValue</i> | [numeric] theoretical best criterion value |
| <i>\$Multiplier</i> | [numeric] integer indicating whether the criterion is indeed an error (+1) or an efficiency (-1) |
| <i>\$Ind_notcomputed</i> | [numeric] indices of the time steps where InputsCrit\$BoolCrit=FALSE or no data is available |

Author(s)

Laurent Coron (June 2014)

References

Nash, J.E. and Sutcliffe, J.V. (1970), River flow forecasting through conceptual models part 1. A discussion of principles, *Journal of Hydrology*, 10(3), 282-290, doi:10.1016/0022-1694(70)90255-6.

See Also

[ErrorCrit_RMSE](#), [ErrorCrit_KGE](#), [ErrorCrit_KGE2](#)

Examples

```
## see example of the ErrorCrit function
```

| | |
|----------------|--|
| ErrorCrit_RMSE | <i>Error criterion based on the RMSE</i> |
|----------------|--|

Description

Function which computes an error criterion based on the root mean square error (RMSE).

Usage

```
ErrorCrit_RMSE(InputsCrit, OutputsModel, verbose = TRUE)
```

Arguments

| | |
|--------------|--|
| InputsCrit | [object of class <i>InputsCrit</i>] see CreateInputsCrit for details |
| OutputsModel | [object of class <i>OutputsModel</i>] see RunModel_GR4J or RunModel_CemaNeigeGR4J for details |
| verbose | (optional) [boolean] boolean indicating if the function is run in verbose mode or not, default=TRUE |

Details

In addition to the criterion value, the function outputs include a multiplier (-1 or +1) which allows the use of the function for model calibration: the product CritValue*Multiplier is the criterion to be minimised (Multiplier=+1 for RMSE).

Value

list list containing the function outputs organised as follows:

| | |
|--------------------------|--|
| <i>\$CritValue</i> | [numeric] value of the criterion |
| <i>\$CritName</i> | [character] name of the criterion |
| <i>\$CritBestValue</i> | [numeric] theoretical best criterion value |
| <i>\$Multiplier</i> | [numeric] integer indicating whether the criterion is indeed an error (+1) or an efficiency (-1) |
| <i>\$Ind_notcomputed</i> | [numeric] indices of the time steps where InputsCrit\$BoolCrit=FALSE or no data is available |

Author(s)

Laurent Coron, Ludovic Oudin (June 2014)

See Also

[ErrorCrit_NSE](#), [ErrorCrit_KGE](#), [ErrorCrit_KGE2](#)

Examples

```
## see example of the ErrorCrit function
```

| | |
|---------------|---|
| PEdaily_Oudin | <i>Computation of daily series of potential evapotranspiration with Oudin's formula</i> |
|---------------|---|

Description

Function which computes daily PE using the formula from Oudin et al. (2005).

Usage

```
PEdaily_Oudin(JD, Temp, LatRad)
```

Arguments

| | |
|--------|--|
| JD | [numeric] time series of julian day [-] |
| Temp | [numeric] time series of daily mean air temperature [°C] |
| LatRad | [numeric] latitude of measurement for the temperature series [rad] |

Value

numeric time series of daily potential evapotranspiration [mm/d]

Author(s)

Laurent Coron, Ludovic Oudin (December 2013)

References

Oudin, L., F. Hervieu, C. Michel, C. Perrin, V. Andréassian, F. Anctil and C. Loumagne (2005), Which potential evapotranspiration input for a lumped rainfall-runoff model?: Part 2-Towards a simple and efficient potential evapotranspiration model for rainfall-runoff modelling, *Journal of Hydrology*, 303(1-4), 290-306, doi:10.1016/j.jhydrol.2004.08.026.

Examples

```
require(airGR)
data(L0123001)
PotEvap <- PEdaily_Oudin(JD=as.POSIXlt(BasinObs$DatesR)$yday, Temp=BasinObs$T, LatRad=0.8)
```

| | |
|-------------------|---|
| plot_OutputsModel | <i>Default preview of model outputs</i> |
|-------------------|---|

Description

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

Usage

```
plot_OutputsModel(OutputsModel, Qobs = NULL, IndPeriod_Plot = NULL,
  BasinArea = NULL, PlotChoice = "all", verbose = TRUE)
```

Arguments

| | |
|----------------|--|
| OutputsModel | [object of class <i>OutputsModel</i>] list of model outputs (which must at least include DatesR, Precip and Qsim) [POSIXlt, mm, mm] |
| Qobs | (optional) [numeric] time series of observed flow (for the same time steps than simulated) [mm/time step] |
| IndPeriod_Plot | (optional) [numeric] indices of the time steps to be plotted (among the OutputsModel series) |
| BasinArea | (optional) [numeric] basin area [km2], used to plot flow axes in m3/s |
| PlotChoice | (optional) [character] choice of plots (e.g. c("Precip", "SnowPack", "Flows", "Regime", "CumFreq", "CorQQ")), default="all" |
| verbose | (optional) [boolean] boolean indicating if the function is run in verbose mode or not, default=TRUE |

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)

Laurent Coron (June 2014)

RunModel

*Run with the provided hydrological model function***Description**

Function which performs a single model run with the provided function over the selected period.

Usage

```
RunModel(InputsModel, RunOptions, Param, FUN_MOD)
```

Arguments

| | |
|-------------|---|
| InputsModel | [object of class <i>InputsModel</i>] see CreateInputsModel for details |
| RunOptions | [object of class <i>RunOptions</i>] see CreateRunOptions for details |
| Param | [numeric] vector of model parameters |
| FUN_MOD | [function] hydrological model function (e.g. <code>RunModel_GR4J</code> , <code>RunModel_CemaNeigeGR4J</code>) |

Value

list see [RunModel_GR4J](#) or [RunModel_CemaNeigeGR4J](#) for details

Author(s)

Laurent Coron (June 2014)

See Also

[RunModel_GR4J](#), [RunModel_CemaNeigeGR4J](#), [CreateInputsModel](#), [CreateRunOptions](#).

Examples

```
## loading catchment data
library(airGR)
data(L0123001)

## preparation of the InputsModel object
InputsModel <- CreateInputsModel(FUN_MOD = RunModel_GR4J, DatesR = BasinObs$DatesR,
                                Precip = BasinObs$P, PotEvap = BasinObs$E)

## run period selection
Ind_Run <- seq(which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="01/01/1990 00:00"),
               which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="31/12/1999 00:00"))

## preparation of the RunOptions object
RunOptions <- CreateRunOptions(FUN_MOD = RunModel_GR4J,
                               InputsModel = InputsModel, IndPeriod_Run = Ind_Run)

## simulation
Param <- c(734.568, -0.840, 109.809, 1.971)
OutputsModel <- RunModel(InputsModel = InputsModel, RunOptions = RunOptions, Param = Param,
                         FUN_MOD = RunModel_GR4J)
```

```
## results preview
plot_OutputsModel(OutputsModel = OutputsModel, Qobs = BasinObs$Qmm[Ind_Run])

## efficiency criterion: Nash-Sutcliffe Efficiency
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_NSE, InputsModel = InputsModel,
                              RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run])
OutputsCrit <- ErrorCrit_NSE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ", round(OutputsCrit$CritValue, 4), "\n", sep = ""))
```

| | |
|--------------------|---|
| RunModel_CemaNeige | <i>Run with the CemaNeige snow module</i> |
|--------------------|---|

Description

Function which performs a single run for the CemaNeige daily snow module.

Usage

```
RunModel_CemaNeige(InputsModel, RunOptions, Param)
```

Arguments

| | |
|--------------|---|
| InputsModel | [object of class <i>InputsModel</i>] see CreateInputsModel for details |
| RunOptions | [object of class <i>RunOptions</i>] see CreateRunOptions for details |
| Param | [numeric] vector of 2 parameters |
| CemaNeige X1 | weighting coefficient for snow pack thermal state [-] |
| CemaNeige X2 | degree-day melt coefficient [mm/°C/d] |

Details

For further details on the model, see the references section. For further details on the argument structures and initialisation options, see [CreateRunOptions](#).

Value

list list containing the function outputs organised as follows:

| | |
|--|--|
| <i>\$DatesR</i> | [POSIXlt] series of dates |
| <i>\$CemaNeigeLayers</i> | [list] list of CemaNeige outputs (1 list per layer) |
| <i>\$CemaNeigeLayers[[iLayer]]\$Pliq</i> | [numeric] series of liquid precip. [mm/d] |
| <i>\$CemaNeigeLayers[[iLayer]]\$Psol</i> | [numeric] series of solid precip. [mm/d] |
| <i>\$CemaNeigeLayers[[iLayer]]\$SnowPack</i> | [numeric] series of snow pack [mm] |
| <i>\$CemaNeigeLayers[[iLayer]]\$ThermalState</i> | [numeric] series of snow pack thermal state [°C] |
| <i>\$CemaNeigeLayers[[iLayer]]\$Gratio</i> | [numeric] series of Gratio [0-1] |
| <i>\$CemaNeigeLayers[[iLayer]]\$PotMelt</i> | [numeric] series of potential snow melt [mm] |
| <i>\$CemaNeigeLayers[[iLayer]]\$Melt</i> | [numeric] series of actual snow melt [mm] |
| <i>\$CemaNeigeLayers[[iLayer]]\$PliqAndMelt</i> | [numeric] series of liquid precip. + actual snow melt [mm] |
| <i>\$StateEnd</i> | [numeric] states at the end of the run: CemaNeige states [mm & °C] |

(refer to the provided references or to the package source code for further details on these model outputs)

Author(s)

Laurent Coron (January 2014)

References

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 1 - Comparison of six snow accounting routines on 380 catchments, Journal of Hydrology, doi:10.1016/j.jhydrol.2014.04.059.
 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, doi:10.1016/j.jhydrol.2014.04.058.

See Also

[RunModel_CemaNeigeGR4J](#), [CreateInputsModel](#), [CreateRunOptions](#).

Examples

```
## load of catchment data
require(airGR)
data(L0123002)

## preparation of the InputsModel object
InputsModel <- CreateInputsModel(FUN_MOD=RunModel_CemaNeige,DatesR=BasinObs$DatesR,
                                Precip=BasinObs$P,TempMean=BasinObs$T,
                                ZInputs=BasinInfo$HypsoData[51],HypsoData=BasinInfo$HypsoData,
                                NLayers=5)

## run period selection
Ind_Run <- seq(which(format(BasinObs$DatesR,format="%d/%m/%Y %H:%M")=="01/01/1990 00:00"),
               which(format(BasinObs$DatesR,format="%d/%m/%Y %H:%M")=="31/12/1999 00:00"))

## preparation of the RunOptions object
RunOptions <- CreateRunOptions(FUN_MOD=RunModel_CemaNeige,InputsModel=InputsModel,
                               IndPeriod_Run=Ind_Run)

## simulation
Param <- c(0.962,2.249)
OutputsModel <- RunModel_CemaNeige(InputsModel=InputsModel,RunOptions=RunOptions,Param=Param)

## results preview
plot_OutputsModel(OutputsModel=OutputsModel)
```


Description

Function which performs a single run for the CemaNeige-GR4J daily lumped model over the test period.

Usage

```
RunModel_CemaNeigeGR4J(InputsModel, RunOptions, Param)
```

Arguments

| | |
|--------------|---|
| InputsModel | [object of class <i>InputsModel</i>] see CreateInputsModel for details |
| RunOptions | [object of class <i>RunOptions</i>] see CreateRunOptions for details |
| Param | [numeric] vector of 6 parameters |
| GR4J X1 | production store capacity [mm] |
| GR4J X2 | intercatchment exchange coefficient [mm/d] |
| GR4J X3 | routing store capacity [mm] |
| GR4J X4 | unit hydrograph time constant [d] |
| CemaNeige X1 | weighting coefficient for snow pack thermal state [-] |
| CemaNeige X2 | degree-day melt coefficient [mm/°C/d] |

Details

For further details on the model, see the references section. For further details on the argument structures and initialisation options, see [CreateRunOptions](#).

Value

list list containing the function outputs organised as follows:

| | |
|--|---|
| <i>\$DatesR</i> | [POSIXlt] series of dates |
| <i>\$PotEvap</i> | [numeric] series of input potential evapotranspiration [mm/d] |
| <i>\$Precip</i> | [numeric] series of input total precipitation [mm/d] |
| <i>\$Prod</i> | [numeric] series of production store level [mm] |
| <i>\$AE</i> | [numeric] series of actual evapotranspiration [mm/d] |
| <i>\$Perc</i> | [numeric] series of percolation (PERC) [mm/d] |
| <i>\$PR</i> | [numeric] series of PR=PN-PS+PERC [mm/d] |
| <i>\$Q9</i> | [numeric] series of UH1 outflow (Q9) [mm/d] |
| <i>\$Q1</i> | [numeric] series of UH2 outflow (Q1) [mm/d] |
| <i>\$Rout</i> | [numeric] series of routing store level [mm] |
| <i>\$Exch</i> | [numeric] series of potential semi-exchange between catchments [mm/d] |
| <i>\$AExch</i> | [numeric] series of actual exchange between catchments (1+2) [mm/d] |
| <i>\$QR</i> | [numeric] series of routing store outflow (QR) [mm/d] |
| <i>\$QD</i> | [numeric] series of direct flow from UH2 after exchange (QD) [mm/d] |
| <i>\$Qsim</i> | [numeric] series of Qsim [mm/d] |
| <i>\$CemaNeigeLayers</i> | [list] list of CemaNeige outputs (1 list per layer) |
| <i>\$CemaNeigeLayers[[iLayer]]\$Pliq</i> | [numeric] series of liquid precip. [mm/d] |
| <i>\$CemaNeigeLayers[[iLayer]]\$Psol</i> | [numeric] series of solid precip. [mm/d] |
| <i>\$CemaNeigeLayers[[iLayer]]\$SnowPack</i> | [numeric] series of snow pack [mm] |
| <i>\$CemaNeigeLayers[[iLayer]]\$ThermalState</i> | [numeric] series of snow pack thermal state [°C] |
| <i>\$CemaNeigeLayers[[iLayer]]\$Gratio</i> | [numeric] series of Gratio [0-1] |

| | |
|---|---|
| <i>\$CemaNeigeLayers[[iLayer]]\$PotMelt</i> | [numeric] series of potential snow melt [mm/d] |
| <i>\$CemaNeigeLayers[[iLayer]]\$Melt</i> | [numeric] series of actual snow melt [mm/d] |
| <i>\$CemaNeigeLayers[[iLayer]]\$PliqAndMelt</i> | [numeric] series of liquid precip. + actual snow melt [mm/d] |
| <i>\$StateEnd</i> | [numeric] states at the end of the run: store & unit hydrographs levels [mm], CemaNeige states [mm & °C] |

(refer to the provided references or to the package source code for further details on these model outputs)

Author(s)

Laurent Coron, Audrey Valéry, Claude Michel, Charles Perrin, Vazken Andréassian (December 2013)

References

Perrin, C., C. Michel and V. Andréassian (2003), Improvement of a parsimonious model for stream-flow simulation, *Journal of Hydrology*, 279(1-4), 275-289, doi:10.1016/S0022-1694(03)00225-7.
 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 1 - Comparison of six snow accounting routines on 380 catchments, *Journal of Hydrology*, doi:10.1016/j.jhydrol.2014.04.059.
 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*, doi:10.1016/j.jhydrol.2014.04.058.

See Also

[RunModel_CemaNeigeGR5J](#), [RunModel_CemaNeigeGR6J](#), [RunModel_GR4J](#), [CreateInputsModel](#), [CreateRunOptions](#).

Examples

```
## loading catchment data
library(airGR)
data(L0123002)

## preparation of the InputsModel object
InputsModel <- CreateInputsModel(FUN_MOD = RunModel_CemaNeigeGR4J, DatesR = BasinObs$DatesR,
                                Precip = BasinObs$P, PotEvap = BasinObs$E, TempMean = BasinObs$T,
                                ZInputs = BasinInfo$HypsoData[51],
                                HypsoData = BasinInfo$HypsoData, NLayers = 5)

## run period selection
Ind_Run <- seq(which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="01/01/1990 00:00"),
               which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="31/12/1999 00:00"))

## preparation of the RunOptions object
RunOptions <- CreateRunOptions(FUN_MOD = RunModel_CemaNeigeGR4J, InputsModel = InputsModel,
                               IndPeriod_Run = Ind_Run)

## simulation
Param <- c(408.774, 2.646, 131.264, 1.174, 0.962, 2.249)
OutputsModel <- RunModel_CemaNeigeGR4J(InputsModel = InputsModel,
                                       RunOptions = RunOptions, Param = Param)
```

```
## results preview
plot_OutputsModel(OutputsModel = OutputsModel, Qobs = BasinObs$Qmm[Ind_Run])

## efficiency criterion: Nash-Sutcliffe Efficiency
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_NSE, InputsModel = InputsModel,
                             RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run])
OutputsCrit <- ErrorCrit_NSE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ", round(OutputsCrit$CritValue, 4), "\n", sep = ""))
```

RunModel_CemaNeigeGR5J

Run with the CemaNeigeGR5J hydrological model

Description

Function which performs a single run for the CemaNeige-GR5J daily lumped model.

Usage

```
RunModel_CemaNeigeGR5J(InputsModel, RunOptions, Param)
```

Arguments

| | |
|--------------|---|
| InputsModel | [object of class <i>InputsModel</i>] see CreateInputsModel for details |
| RunOptions | [object of class <i>RunOptions</i>] see CreateRunOptions for details |
| Param | [numeric] vector of 7 parameters |
| GR5J X1 | production store capacity [mm] |
| GR5J X2 | intercatchment exchange coefficient [mm/d] |
| GR5J X3 | routing store capacity [mm] |
| GR5J X4 | unit hydrograph time constant [d] |
| GR5J X5 | intercatchment exchange threshold [-] |
| CemaNeige X1 | weighting coefficient for snow pack thermal state [-] |
| CemaNeige X2 | degree-day melt coefficient [mm/°C/d] |

Details

For further details on the model, see the references section. For further details on the argument structures and initialisation options, see [CreateRunOptions](#).

Value

list list containing the function outputs organised as follows:

| | |
|------------------|---|
| <i>\$DatesR</i> | [POSIXlt] series of dates |
| <i>\$PotEvap</i> | [numeric] series of input potential evapotranspiration [mm/d] |
| <i>\$Precip</i> | [numeric] series of input total precipitation [mm/d] |
| <i>\$Prod</i> | [numeric] series of production store level [mm] |
| <i>\$AE</i> | [numeric] series of actual evapotranspiration [mm/d] |
| <i>\$Perc</i> | [numeric] series of percolation (PERC) [mm/d] |

| | |
|--|---|
| <i>\$PR</i> | [numeric] series of PR=PN-PS+PERC [mm/d] |
| <i>\$Q9</i> | [numeric] series of UH1 outflow (Q9) [mm/d] |
| <i>\$Q1</i> | [numeric] series of UH2 outflow (Q1) [mm/d] |
| <i>\$Rout</i> | [numeric] series of routing store level [mm] |
| <i>\$Exch</i> | [numeric] series of potential semi-exchange between catchments [mm/d] |
| <i>\$AExch</i> | [numeric] series of actual exchange between catchments (1+2) [mm/d] |
| <i>\$QR</i> | [numeric] series of routing store outflow (QR) [mm/d] |
| <i>\$QD</i> | [numeric] series of direct flow from UH2 after exchange (QD) [mm/d] |
| <i>\$Qsim</i> | [numeric] series of Qsim [mm/d] |
| <i>\$CemaNeigeLayers</i> | [list] list of CemaNeige outputs (1 list per layer) |
| <i>\$CemaNeigeLayers[[iLayer]]\$Pliq</i> | [numeric] series of liquid precip. [mm/d] |
| <i>\$CemaNeigeLayers[[iLayer]]\$Psol</i> | [numeric] series of solid precip. [mm/d] |
| <i>\$CemaNeigeLayers[[iLayer]]\$SnowPack</i> | [numeric] series of snow pack [mm] |
| <i>\$CemaNeigeLayers[[iLayer]]\$ThermalState</i> | [numeric] series of snow pack thermal state [°C] |
| <i>\$CemaNeigeLayers[[iLayer]]\$Gratio</i> | [numeric] series of Gratio [0-1] |
| <i>\$CemaNeigeLayers[[iLayer]]\$PotMelt</i> | [numeric] series of potential snow melt [mm/d] |
| <i>\$CemaNeigeLayers[[iLayer]]\$Melt</i> | [numeric] series of actual snow melt [mm/d] |
| <i>\$CemaNeigeLayers[[iLayer]]\$PliqAndMelt</i> | [numeric] series of liquid precip. + actual snow melt [mm/d] |
| <i>\$StateEnd</i> | [numeric] states at the end of the run: store & unit hydrographs levels [mm], CemaNeige states [mm & °C] |

(refer to the provided references or to the package source code for further details on these model outputs)

Author(s)

Laurent Coron, Audrey Valéry, Claude Michel, Nicolas Le Moine, Charles Perrin, Vazken Andréassian (December 2013)

References

- 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 (french), UPMC, Paris, France.
- 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 1 - Comparison of six snow accounting routines on 380 catchments, *Journal of Hydrology*, doi:10.1016/j.jhydrol.2014.04.059.
- 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*, doi:10.1016/j.jhydrol.2014.04.058.

See Also

[RunModel_CemaNeigeGR4J](#), [RunModel_CemaNeigeGR6J](#), [RunModel_GR5J](#), [CreateInputsModel](#), [CreateRunOptions](#).

Examples

```
## loading catchment data
library(airGR)
data(L0123002)
```

```

## preparation of the InputsModel object
InputsModel <- CreateInputsModel(FUN_MOD = RunModel_CemaNeigeGR5J, DatesR = BasinObs$DatesR,
                                Precip = BasinObs$P, PotEvap = BasinObs$E, TempMean = BasinObs$T,
                                ZInputs = BasinInfo$HypsoData[51],
                                HypsoData = BasinInfo$HypsoData, NLayers = 5)

## run period selection
Ind_Run <- seq(which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="01/01/1990 00:00"),
               which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="31/12/1999 00:00"))

## preparation of the RunOptions object
RunOptions <- CreateRunOptions(FUN_MOD = RunModel_CemaNeigeGR5J, InputsModel = InputsModel,
                               IndPeriod_Run = Ind_Run)

## simulation
Param <- c(179.139, -0.100, 203.815, 1.174, 2.478, 0.977, 2.774)
OutputsModel <- RunModel_CemaNeigeGR5J(InputsModel = InputsModel,
                                         RunOptions = RunOptions, Param = Param)

## results preview
plot_OutputsModel(OutputsModel = OutputsModel, Qobs = BasinObs$Qmm[Ind_Run])

## efficiency criterion: Nash-Sutcliffe Efficiency
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_NSE, InputsModel = InputsModel,
                              RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run])
OutputsCrit <- ErrorCrit_NSE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ", round(OutputsCrit$CritValue, 4), "\n", sep = ""))

```

RunModel_CemaNeigeGR6J

Run with the CemaNeigeGR6J hydrological model

Description

Function which performs a single run for the CemaNeige-GR6J daily lumped model.

Usage

```
RunModel_CemaNeigeGR6J(InputsModel, RunOptions, Param)
```

Arguments

| | |
|-------------|---|
| InputsModel | [object of class <i>InputsModel</i>] see CreateInputsModel for details |
| RunOptions | [object of class <i>RunOptions</i>] see CreateRunOptions for details |
| Param | [numeric] vector of 8 parameters |
| GR6J X1 | production store capacity [mm] |
| GR6J X2 | intercatchment exchange coefficient [mm/d] |
| GR6J X3 | routing store capacity [mm] |
| GR6J X4 | unit hydrograph time constant [d] |
| GR6J X5 | intercatchment exchange threshold [-] |

| | |
|--------------|---|
| GR6J X6 | coefficient for emptying exponential store [mm] |
| CemaNeige X1 | weighting coefficient for snow pack thermal state [-] |
| CemaNeige X2 | degree-day melt coefficient [mm/°C/d] |

Details

For further details on the model, see the references section. For further details on the argument structures and initialisation options, see [CreateRunOptions](#).

Value

list list containing the function outputs organised as follows:

| | |
|--|---|
| <i>\$DatesR</i> | [POSIXlt] series of dates |
| <i>\$PotEvap</i> | [numeric] series of input potential evapotranspiration [mm/d] |
| <i>\$Precip</i> | [numeric] series of input total precipitation [mm/d] |
| <i>\$Prod</i> | [numeric] series of production store level [mm] |
| <i>\$AE</i> | [numeric] series of actual evapotranspiration [mm/d] |
| <i>\$Perc</i> | [numeric] series of percolation (PERC) [mm/d] |
| <i>\$PR</i> | [numeric] series of PR=PN-PS+PERC [mm/d] |
| <i>\$Q9</i> | [numeric] series of UH1 outflow (Q9) [mm/d] |
| <i>\$Q1</i> | [numeric] series of UH2 outflow (Q1) [mm/d] |
| <i>\$Rout</i> | [numeric] series of routing store level [mm] |
| <i>\$Exch</i> | [numeric] series of potential semi-exchange between catchments [mm/d] |
| <i>\$AExch</i> | [numeric] series of actual exchange between catchments (1+2) [mm/d] |
| <i>\$QR</i> | [numeric] series of routing store outflow (QR) [mm/d] |
| <i>\$QR1</i> | [numeric] series of exponential store outflow (QR1) [mm/d] |
| <i>\$Exp</i> | [numeric] series of exponential store level (negative) [mm] |
| <i>\$QD</i> | [numeric] series of direct flow from UH2 after exchange (QD) [mm/d] |
| <i>\$Qsim</i> | [numeric] series of Qsim [mm/d] |
| <i>\$CemaNeigeLayers</i> | [list] list of CemaNeige outputs (1 list per layer) |
| <i>\$CemaNeigeLayers[[iLayer]]\$Pliq</i> | [numeric] series of liquid precip. [mm/d] |
| <i>\$CemaNeigeLayers[[iLayer]]\$Psol</i> | [numeric] series of solid precip. [mm/d] |
| <i>\$CemaNeigeLayers[[iLayer]]\$SnowPack</i> | [numeric] series of snow pack [mm] |
| <i>\$CemaNeigeLayers[[iLayer]]\$ThermalState</i> | [numeric] series of snow pack thermal state [°C] |
| <i>\$CemaNeigeLayers[[iLayer]]\$Gratio</i> | [numeric] series of Gratio [0-1] |
| <i>\$CemaNeigeLayers[[iLayer]]\$PotMelt</i> | [numeric] series of potential snow melt [mm/d] |
| <i>\$CemaNeigeLayers[[iLayer]]\$Melt</i> | [numeric] series of actual snow melt [mm/d] |
| <i>\$CemaNeigeLayers[[iLayer]]\$PliqAndMelt</i> | [numeric] series of liquid precip. + actual snow melt [mm/d] |
| <i>\$StateEnd</i> | [numeric] states at the end of the run: store & unit hydrographs levels [mm], CemaNeige states [mm & °C] |

(refer to the provided references or to the package source code for further details on these model outputs)

Author(s)

Laurent Coron, Audrey Valéry, Claude Michel, Charles Perrin, Raji Pushpalatha, Nicolas Le Moine, Vazken Andréassian (December 2013)

References

- 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 1 - Comparison of six snow accounting routines on 380 catchments, *Journal of Hydrology*, doi:10.1016/j.jhydrol.2014.04.059.
- 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*, doi:10.1016/j.jhydrol.2014.04.058.

See Also

[RunModel_CemaNeigeGR4J](#), [RunModel_CemaNeigeGR5J](#), [RunModel_GR6J](#), [CreateInputsModel](#), [CreateRunOptions](#).

Examples

```
## loading catchment data
library(airGR)
data(L0123002)

## preparation of the InputsModel object
InputsModel <- CreateInputsModel(FUN_MOD = RunModel_CemaNeigeGR6J, DatesR = BasinObs$DatesR,
                                Precip = BasinObs$P, PotEvap = BasinObs$E, TempMean = BasinObs$T,
                                ZInputs = BasinInfo$HypsoData[51],
                                HypsoData = BasinInfo$HypsoData, NLayers = 5)

## run period selection
Ind_Run <- seq(which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="01/01/1990 00:00"),
               which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="31/12/1999 00:00"))

## preparation of the RunOptions object
RunOptions <- CreateRunOptions(FUN_MOD = RunModel_CemaNeigeGR6J, InputsModel = InputsModel,
                               IndPeriod_Run = Ind_Run)

## simulation
Param <- c(116.482, 0.500, 72.733, 1.224, 0.278, 30.333, 0.977, 2.776)
OutputsModel <- RunModel_CemaNeigeGR6J(InputsModel = InputsModel,
                                       RunOptions = RunOptions, Param = Param)

## results preview
plot_OutputsModel(OutputsModel = OutputsModel, Qobs = BasinObs$Qmm[Ind_Run])

## efficiency criterion: Nash-Sutcliffe Efficiency
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_NSE, InputsModel = InputsModel,
                              RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run])
OutputsCrit <- ErrorCrit_NSE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ", round(OutputsCrit$CritValue, 4), "\n", sep = ""))
```

Description

Function which performs a single run for the GR1A annual lumped model over the test period.

Usage

```
RunModel_GR1A(InputsModel, RunOptions, Param)
```

Arguments

| | |
|-------------|---|
| InputsModel | [object of class <i>InputsModel</i>] see CreateInputsModel for details |
| RunOptions | [object of class <i>RunOptions</i>] see CreateRunOptions for details |
| Param | [numeric] vector of 1 parameter |

GR1A X1 model parameter [-]

Details

For further details on the model, see the references section. For further details on the argument structures and initialisation options, see [CreateRunOptions](#).

Value

list list containing the function outputs organised as follows:

| | |
|-------------------|---|
| <i>\$DatesR</i> | [POSIXlt] series of dates |
| <i>\$PotEvap</i> | [numeric] series of input potential evapotranspiration [mm/y] |
| <i>\$Precip</i> | [numeric] series of input total precipitation [mm/y] |
| <i>\$Qsim</i> | [numeric] series of Qsim [mm/y] |
| <i>\$StateEnd</i> | [numeric] states at the end of the run (NULL) [-] |

(refer to the provided references or to the package source code for further details on these model outputs)

Author(s)

Laurent Coron, Claude Michel (March 2015)

References

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, France.

Examples

```
## loading catchment data
library(airGR)
data(L0123001)

## conversion of example data from daily to yearly time step
```



```

TabSeries      <- data.frame(BasinObs$DatesR, BasinObs$P, BasinObs$E, BasinObs$T, BasinObs$Qmm)
TimeFormat     <- "daily"
NewTimeFormat  <- "yearly"
ConvertFun     <- c("sum", "sum", "mean", "sum")
YearFirstMonth <- 09;
NewTabSeries   <- SeriesAggreg(TabSeries = TabSeries, TimeFormat = TimeFormat,
                               NewTimeFormat = NewTimeFormat, ConvertFun = ConvertFun,
                               YearFirstMonth = YearFirstMonth)

BasinObs       <- NewTabSeries
names(BasinObs) <- c("DatesR", "P", "E", "T", "Qmm")

## preparation of the InputsModel object
InputsModel <- CreateInputsModel(FUN_MOD = RunModel_GR1A, DatesR = BasinObs$DatesR,
                                Precip = BasinObs$P, PotEvap = BasinObs$E)

## run period selection
Ind_Run <- seq(which(format(BasinObs$DatesR, format = "%Y")=="1990"),
               which(format(BasinObs$DatesR, format = "%Y")=="1999"))

## preparation of the RunOptions object
RunOptions <- CreateRunOptions(FUN_MOD = RunModel_GR1A,
                               InputsModel = InputsModel, IndPeriod_Run = Ind_Run)

## simulation
Param <- c(0.840)
OutputsModel <- RunModel_GR1A(InputsModel = InputsModel, RunOptions = RunOptions, Param = Param)

## results preview
plot_OutputsModel(OutputsModel = OutputsModel, Qobs = BasinObs$Qmm[Ind_Run])

## efficiency criterion: Nash-Sutcliffe Efficiency
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_NSE, InputsModel = InputsModel,
                              RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run])
OutputsCrit <- ErrorCrit_NSE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ", round(OutputsCrit$CritValue, 4), "\n", sep = ""))

```

RunModel_GR2M

Run with the GR2M hydrological model

Description

Function which performs a single run for the GR2M monthly lumped model over the test period.

Usage

```
RunModel_GR2M(InputsModel, RunOptions, Param)
```

Arguments

| | |
|-------------|---|
| InputsModel | [object of class <i>InputsModel</i>] see CreateInputsModel for details |
| RunOptions | [object of class <i>RunOptions</i>] see CreateRunOptions for details |
| Param | [numeric] vector of 2 parameters |

| | |
|---------|--------------------------------------|
| GR2M X1 | production store capacity [mm] |
| GR2M X2 | groundwater exchange coefficient [-] |

Details

For further details on the model, see the references section. For further details on the argument structures and initialisation options, see [CreateRunOptions](#).

Value

list list containing the function outputs organised as follows:

| | |
|-------------------|--|
| <i>\$DatesR</i> | [POSIXlt] series of dates |
| <i>\$PotEvap</i> | [numeric] series of input potential evapotranspiration [mm/month] |
| <i>\$Precip</i> | [numeric] series of input total precipitation [mm/month] |
| <i>\$Qsim</i> | [numeric] series of Qsim [mm/month] |
| <i>\$StateEnd</i> | [numeric] states at the end of the run (production store level and routing store level) [mm] |

(refer to the provided references or to the package source code for further details on these model outputs)

Author(s)

Laurent Coron, Claude Michel, Safouane Mouelhi (March 2015)

References

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, France.

Mouelhi, S., C. Michel, C. Perrin and V. Andréassian (2006), 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.

Examples

```
## loading catchment data
library(airGR)
data(L0123001)

## conversion of example data from daily to monthly time step
TabSeries <- data.frame(BasinObs$DatesR, BasinObs$P, BasinObs$E, BasinObs$T, BasinObs$Qmm)
TimeFormat <- "daily"
NewTimeFormat <- "monthly"
ConvertFun <- c("sum", "sum", "mean", "sum")
NewTabSeries <- SeriesAggreg(TabSeries = TabSeries, TimeFormat = TimeFormat,
                             NewTimeFormat = NewTimeFormat, ConvertFun = ConvertFun)
BasinObs <- NewTabSeries
names(BasinObs) <- c("DatesR", "P", "E", "T", "Qmm")

## preparation of the InputsModel object
InputsModel <- CreateInputsModel(FUN_MOD = RunModel_GR2M, DatesR = BasinObs$DatesR,
                                 Precip = BasinObs$P, PotEvap = BasinObs$E)
```

```

## run period selection
Ind_Run <- seq(which(format(BasinObs$DatesR, format = "%m/%Y")==="01/1990"),
               which(format(BasinObs$DatesR, format = "%m/%Y")==="12/1999"))

## preparation of the RunOptions object
RunOptions <- CreateRunOptions(FUN_MOD = RunModel_GR2M,
                               InputsModel = InputsModel, IndPeriod_Run = Ind_Run)

## simulation
Param <- c(265.072, 1.040)
OutputsModel <- RunModel_GR2M(InputsModel = InputsModel, RunOptions = RunOptions, Param = Param)

## results preview
plot_OutputsModel(OutputsModel = OutputsModel, Qobs = BasinObs$Qmm[Ind_Run])

## efficiency criterion: Nash-Sutcliffe Efficiency
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_NSE, InputsModel = InputsModel,
                               RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run])
OutputsCrit <- ErrorCrit_NSE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ", round(OutputsCrit$CritValue, 4), "\n", sep = ""))

```

RunModel_GR4H

Run with the GR4H hydrological model

Description

Function which performs a single run for the GR4H hourly lumped model.

Usage

```
RunModel_GR4H(InputsModel, RunOptions, Param)
```

Arguments

| | |
|-------------|---|
| InputsModel | [object of class <i>InputsModel</i>] see CreateInputsModel for details |
| RunOptions | [object of class <i>RunOptions</i>] see CreateRunOptions for details |
| Param | [numeric] vector of 4 parameters |

| | |
|---------|---|
| GR4H X1 | production store capacity [mm] |
| GR4H X2 | groundwater exchange coefficient [mm/h] |
| GR4H X3 | routing store capacity [mm] |
| GR4H X4 | unit hydrograph time constant [h] |

Details

For further details on the model, see the references section. For further details on the argument structures and initialisation options, see [CreateRunOptions](#).

Value

list list containing the function outputs organised as follows:

| | |
|-------------------|---|
| <i>\$DatesR</i> | [POSIXlt] series of dates |
| <i>\$PotEvap</i> | [numeric] series of input potential evapotranspiration [mm/h] |
| <i>\$Precip</i> | [numeric] series of input total precipitation [mm/h] |
| <i>\$Prod</i> | [numeric] series of production store level [mm] |
| <i>\$AE</i> | [numeric] series of actual evapotranspiration [mm/h] |
| <i>\$Perc</i> | [numeric] series of percolation (PERC) [mm/h] |
| <i>\$PR</i> | [numeric] series of $PR=PN-PS+PERC$ [mm/h] |
| <i>\$Q9</i> | [numeric] series of UH1 outflow (Q9) [mm/h] |
| <i>\$Q1</i> | [numeric] series of UH2 outflow (Q1) [mm/h] |
| <i>\$Rout</i> | [numeric] series of routing store level [mm] |
| <i>\$Exch</i> | [numeric] series of potential semi-exchange between catchments [mm/h] |
| <i>\$AExch</i> | [numeric] series of actual exchange between catchments (1+2) [mm/h] |
| <i>\$QR</i> | [numeric] series of routing store outflow (QR) [mm/h] |
| <i>\$QD</i> | [numeric] series of direct flow from UH2 after exchange (QD) [mm/h] |
| <i>\$Qsim</i> | [numeric] series of Qsim [mm/h] |
| <i>\$StateEnd</i> | [numeric] states at the end of the run (res. levels, UH1 levels, UH2 levels) [mm] |

(refer to the provided references or to the package source code for further details on these model outputs)

Author(s)

Laurent Coron (July 2014)

References

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.

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 (french), UPMC, Paris, France.

See Also

[RunModel_GR4J](#), [CreateInputsModel](#), [CreateRunOptions](#).

Examples

```
## load of catchment data
require(airGR)
data(L0123003)

## preparation of the InputsModel object
InputsModel <- CreateInputsModel(FUN_MOD=RunModel_GR4H,DatesR=BasinObs$DatesR,
                                Precip=BasinObs$P,PotEvap=BasinObs$E)

## run period selection
Ind_Run <- seq(which(format(BasinObs$DatesR,format="%d/%m/%Y %H:%M")=="01/03/2004 00:00"),
              which(format(BasinObs$DatesR,format="%d/%m/%Y %H:%M")=="31/12/2008 00:00"))

## preparation of the RunOptions object
RunOptions <- CreateRunOptions(FUN_MOD=RunModel_GR4H,InputsModel=InputsModel,IndPeriod_Run=Ind_Run)
```

```
## simulation
Param <- c(521.113,-2.918,218.009,4.124)
OutputsModel <- RunModel_GR4H(InputsModel=InputsModel,RunOptions=RunOptions,Param=Param)

## results preview
plot_OutputsModel(OutputsModel=OutputsModel,Qobs=BasinObs$Qmm[Ind_Run])

## efficiency criterion: Nash-Sutcliffe Efficiency
InputsCrit <- CreateInputsCrit(FUN_CRIT=ErrorCrit_NSE,InputsModel=InputsModel,
                              RunOptions=RunOptions,Qobs=BasinObs$Qmm[Ind_Run])
OutputsCrit <- ErrorCrit_NSE(InputsCrit=InputsCrit,OutputsModel=OutputsModel)
cat(paste(" Crit ",OutputsCrit$CritName," ",round(OutputsCrit$CritValue,4),"\\n",sep=""))
```

RunModel_GR4J

*Run with the GR4J hydrological model***Description**

Function which performs a single run for the GR4J daily lumped model over the test period.

Usage

```
RunModel_GR4J(InputsModel, RunOptions, Param)
```

Arguments

| | |
|-------------|---|
| InputsModel | [object of class <i>InputsModel</i>] see CreateInputsModel for details |
| RunOptions | [object of class <i>RunOptions</i>] see CreateRunOptions for details |
| Param | [numeric] vector of 4 parameters |
| GR4J X1 | production store capacity [mm] |
| GR4J X2 | intercatchment exchange coefficient [mm/d] |
| GR4J X3 | routing store capacity [mm] |
| GR4J X4 | unit hydrograph time constant [d] |

Details

For further details on the model, see the references section. For further details on the argument structures and initialisation options, see [CreateRunOptions](#).

Value

list list containing the function outputs organised as follows:

| | |
|------------------|---|
| <i>\$DatesR</i> | [POSIXlt] series of dates |
| <i>\$PotEvap</i> | [numeric] series of input potential evapotranspiration [mm/d] |
| <i>\$Precip</i> | [numeric] series of input total precipitation [mm/d] |
| <i>\$Prod</i> | [numeric] series of production store level [mm] |
| <i>\$AE</i> | [numeric] series of actual evapotranspiration [mm/d] |

| | |
|-------------------|---|
| <i>\$Perc</i> | [numeric] series of percolation (PERC) [mm/d] |
| <i>\$PR</i> | [numeric] series of $PR=PN-PS+PERC$ [mm/d] |
| <i>\$Q9</i> | [numeric] series of UH1 outflow (Q9) [mm/d] |
| <i>\$Q1</i> | [numeric] series of UH2 outflow (Q1) [mm/d] |
| <i>\$Rout</i> | [numeric] series of routing store level [mm] |
| <i>\$Exch</i> | [numeric] series of potential semi-exchange between catchments [mm/d] |
| <i>\$AExch</i> | [numeric] series of actual exchange between catchments (1+2) [mm/d] |
| <i>\$QR</i> | [numeric] series of routing store outflow (QR) [mm/d] |
| <i>\$QD</i> | [numeric] series of direct flow from UH2 after exchange (QD) [mm/d] |
| <i>\$Qsim</i> | [numeric] series of Qsim [mm/d] |
| <i>\$StateEnd</i> | [numeric] states at the end of the run (res. levels, UH1 levels, UH2 levels) [mm] |

(refer to the provided references or to the package source code for further details on these model outputs)

Author(s)

Laurent Coron, Claude Michel, Charles Perrin (December 2013)

References

Perrin, C., C. Michel and V. Andréassian (2003), Improvement of a parsimonious model for stream-flow simulation, Journal of Hydrology, 279(1-4), 275-289, doi:10.1016/S0022-1694(03)00225-7.

See Also

[RunModel_GR5J](#), [RunModel_GR6J](#), [RunModel_CemaNeigeGR4J](#), [CreateInputsModel](#), [CreateRunOptions](#).

Examples

```
## loading catchment data
library(airGR)
data(L0123001)

## preparation of the InputsModel object
InputsModel <- CreateInputsModel(FUN_MOD = RunModel_GR4J, DatesR = BasinObs$DatesR,
                                Precip = BasinObs$P, PotEvap = BasinObs$E)

## run period selection
Ind_Run <- seq(which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="01/01/1990 00:00"),
               which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="31/12/1999 00:00"))

## preparation of the RunOptions object
RunOptions <- CreateRunOptions(FUN_MOD = RunModel_GR4J,
                               InputsModel = InputsModel, IndPeriod_Run = Ind_Run)

## simulation
Param <- c(257.238, 1.012, 88.235, 2.208)
OutputsModel <- RunModel_GR4J(InputsModel = InputsModel, RunOptions = RunOptions, Param = Param)

## results preview
plot_OutputsModel(OutputsModel = OutputsModel, Qobs = BasinObs$Qmm[Ind_Run])

## efficiency criterion: Nash-Sutcliffe Efficiency
```

```
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_NSE, InputsModel = InputsModel,
                             RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run])
OutputsCrit <- ErrorCrit_NSE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ", round(OutputsCrit$CritValue, 4), "\n", sep = ""))
```

RunModel_GR5J

Run with the GR5J hydrological model

Description

Function which performs a single run for the GR5J daily lumped model over the test period.

Usage

```
RunModel_GR5J(InputsModel, RunOptions, Param)
```

Arguments

| | |
|-------------|---|
| InputsModel | [object of class <i>InputsModel</i>] see CreateInputsModel for details |
| RunOptions | [object of class <i>RunOptions</i>] see CreateRunOptions for details |
| Param | [numeric] vector of 5 parameters |
| | GR5J X1 production store capacity [mm] |
| | GR5J X2 intercatchment exchange coefficient [mm/d] |
| | GR5J X3 routing store capacity [mm] |
| | GR5J X4 unit hydrograph time constant [d] |
| | GR5J X5 intercatchment exchange threshold [-] |

Details

For further details on the model, see the references section. For further details on the argument structures and initialisation options, see [CreateRunOptions](#).

Value

list list containing the function outputs organised as follows:

| | |
|------------------|---|
| <i>\$DatesR</i> | [POSIXlt] series of dates |
| <i>\$PotEvap</i> | [numeric] series of input potential evapotranspiration [mm/d] |
| <i>\$Precip</i> | [numeric] series of input total precipitation [mm/d] |
| <i>\$Prod</i> | [numeric] series of production store level [mm] |
| <i>\$AE</i> | [numeric] series of actual evapotranspiration [mm/d] |
| <i>\$Perc</i> | [numeric] series of percolation (PERC) [mm/d] |
| <i>\$PR</i> | [numeric] series of PR=PN-PS+PERC [mm/d] |
| <i>\$Q9</i> | [numeric] series of UH1 outflow (Q9) [mm/d] |
| <i>\$Q1</i> | [numeric] series of UH2 outflow (Q1) [mm/d] |
| <i>\$Rout</i> | [numeric] series of routing store level [mm] |
| <i>\$Exch</i> | [numeric] series of potential semi-exchange between catchments [mm/d] |
| <i>\$AExch</i> | [numeric] series of actual exchange between catchments (1+2) [mm/d] |

\$QR [numeric] series of routing store outflow (QR) [mm/d]
\$QD [numeric] series of direct flow from UH2 after exchange (QD) [mm/d]
\$Qsim [numeric] series of Qsim [mm/d]
\$StateEnd [numeric] states at the end of the run (res. levels, UH1 levels, UH2 levels) [mm]

(refer to the provided references or to the package source code for further details on these model outputs)

Author(s)

Laurent Coron, Claude Michel, Nicolas Le Moine (December 2013)

References

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 (french), UPMC, Paris, France.
 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.

See Also

[RunModel_GR4J](#), [RunModel_GR6J](#), [RunModel_CemaNeigeGR5J](#), [CreateInputsModel](#), [CreateRunOptions](#).

Examples

```
## loading catchment data
library(airGR)
data(L0123001)

## preparation of the InputsModel object
InputsModel <- CreateInputsModel(FUN_MOD = RunModel_GR5J, DatesR = BasinObs$DatesR,
                                Precip = BasinObs$P, PotEvap = BasinObs$E)

## run period selection
Ind_Run <- seq(which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="01/01/1990 00:00"),
              which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="31/12/1999 00:00"))

## preparation of the RunOptions object
RunOptions <- CreateRunOptions(FUN_MOD = RunModel_GR5J,
                              InputsModel = InputsModel, IndPeriod_Run = Ind_Run)

## simulation
Param <- c(245.918, 1.027, 90.017, 2.198, 0.434)
OutputsModel <- RunModel_GR5J(InputsModel = InputsModel,
                              RunOptions = RunOptions, Param = Param)

## results preview
plot_OutputsModel(OutputsModel = OutputsModel, Qobs = BasinObs$Qmm[Ind_Run])

## efficiency criterion: Nash-Sutcliffe Efficiency
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_NSE, InputsModel = InputsModel,
                              RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run])
```



```
OutputsCrit <- ErrorCrit_NSE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
cat(paste(" Crit ", OutputsCrit$CritName, " ", round(OutputsCrit$CritValue, 4), "\n", sep = ""))
```

RunModel_GR6J

Run with the GR6J hydrological model

Description

Function which performs a single run for the GR6J daily lumped model over the test period.

Usage

```
RunModel_GR6J(InputsModel, RunOptions, Param)
```

Arguments

| | |
|-------------|---|
| InputsModel | [object of class <i>InputsModel</i>] see CreateInputsModel for details |
| RunOptions | [object of class <i>RunOptions</i>] see CreateRunOptions for details |
| Param | [numeric] vector of 6 parameters |
| | GR6J X1 production store capacity [mm] |
| | GR6J X2 intercatchment exchange coefficient [mm/d] |
| | GR6J X3 routing store capacity [mm] |
| | GR6J X4 unit hydrograph time constant [d] |
| | GR6J X5 intercatchment exchange threshold [-] |
| | GR6J X6 coefficient for emptying exponential store [mm] |

Details

For further details on the model, see the references section. For further details on the argument structures and initialisation options, see [CreateRunOptions](#).

Value

list list containing the function outputs organised as follows:

| | |
|------------------|---|
| <i>\$DatesR</i> | [POSIXlt] series of dates |
| <i>\$PotEvap</i> | [numeric] series of input potential evapotranspiration [mm/d] |
| <i>\$Precip</i> | [numeric] series of input total precipitation [mm/d] |
| <i>\$Prod</i> | [numeric] series of production store level [mm] |
| <i>\$AE</i> | [numeric] series of actual evapotranspiration [mm/d] |
| <i>\$Perc</i> | [numeric] series of percolation (PERC) [mm/d] |
| <i>\$PR</i> | [numeric] series of PR=PN-PS+PERC [mm/d] |
| <i>\$Q9</i> | [numeric] series of UH1 outflow (Q9) [mm/d] |
| <i>\$Q1</i> | [numeric] series of UH2 outflow (Q1) [mm/d] |
| <i>\$Rout</i> | [numeric] series of routing store level [mm] |
| <i>\$Exch</i> | [numeric] series of potential semi-exchange between catchments [mm/d] |
| <i>\$AExch</i> | [numeric] series of actual exchange between catchments (1+2) [mm/d] |
| <i>\$QR</i> | [numeric] series of routing store outflow (QR) [mm/d] |

| | |
|-------------------|---|
| <i>\$QR1</i> | [numeric] series of exponential store outflow (QR1) [mm/d] |
| <i>\$Exp</i> | [numeric] series of exponential store level (negative) [mm] |
| <i>\$QD</i> | [numeric] series of direct flow from UH2 after exchange (QD) [mm/d] |
| <i>\$Qsim</i> | [numeric] series of Qsim [mm/d] |
| <i>\$StateEnd</i> | [numeric] states at the end of the run (res. levels, UH1 levels, UH2 levels) [mm] |

(refer to the provided references or to the package source code for further details on these model outputs)

Author(s)

Laurent Coron, Claude Michel, Charles Perrin, Raji Pushpalatha, Nicolas Le Moine (December 2013)

References

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.

See Also

[RunModel_GR4J](#), [RunModel_GR5J](#), [RunModel_CemaNeigeGR6J](#), [CreateInputsModel](#), [CreateRunOptions](#).

Examples

```
## loading catchment data
library(airGR)
data(L0123001)

## preparation of the InputsModel object
InputsModel <- CreateInputsModel(FUN_MOD = RunModel_GR6J, DatesR = BasinObs$DatesR,
                                Precip = BasinObs$P, PotEvap = BasinObs$E)

## run period selection
Ind_Run <- seq(which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="01/01/1990 00:00"),
              which(format(BasinObs$DatesR, format = "%d/%m/%Y %H:%M")=="31/12/1999 00:00"))

## preparation of the RunOptions object
RunOptions <- CreateRunOptions(FUN_MOD = RunModel_GR6J,
                              InputsModel = InputsModel, IndPeriod_Run = Ind_Run)

## simulation
Param <- c(242.257, 0.637, 53.517, 2.218, 0.424, 4.759)
OutputsModel <- RunModel_GR6J(InputsModel = InputsModel, RunOptions = RunOptions, Param = Param)

## results preview
plot_OutputsModel(OutputsModel = OutputsModel, Qobs = BasinObs$Qmm[Ind_Run])

## efficiency criterion: Nash-Sutcliffe Efficiency
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_NSE, InputsModel = InputsModel,
                              RunOptions = RunOptions, Qobs = BasinObs$Qmm[Ind_Run])
OutputsCrit <- ErrorCrit_NSE(InputsCrit = InputsCrit, OutputsModel = OutputsModel)
```

```
cat(paste(" Crit ", OutputsCrit$CritName, " ", round(OutputsCrit$CritValue, 4), "\n", sep = ""))
```

| | |
|--------------|--|
| SeriesAggreg | <i>Conversion of time series to another time step (aggregation only)</i> |
|--------------|--|

Description

Conversion of time series to another time step (aggregation only).

Warning : on the aggregated outputs, the dates correpond to the beginning of the time step

(e.g. for daily time-series 01/03/2005 00:00 = value for period 01/03/2005 00:00 - 01/03/2005 23:59)

(e.g. for monthly time-series 01/03/2005 00:00 = value for period 01/03/2005 00:00 - 31/03/2005 23:59)

(e.g. for yearly time-series 01/03/2005 00:00 = value for period 01/03/2005 00:00 - 28/02/2006 23:59)

Usage

```
SeriesAggreg(TabSeries, TimeFormat, NewTimeFormat, ConvertFun,
  YearFirstMonth = 1, verbose = TRUE)
```

Arguments

| | |
|----------------|--|
| TabSeries | [POSIXlt+numeric] dataframe containing the vector of dates and the time series values |
| TimeFormat | [character] desired format (i.e. "hourly", "daily", "monthly" or "yearly") |
| NewTimeFormat | [character] desired format (i.e. "hourly", "daily", "monthly" or "yearly") |
| ConvertFun | [character] names of aggregation functions (e.g. for P[mm], T[deg], Q[mm] : ConvertFun <- c("sum", "mean", "sum")) |
| YearFirstMonth | (optional) [numeric] integer used when NewTimeFormat = "yearly" to set when the starting month of the year (e.g. 01 for calendar year or 09 for hydrological year starting in september) |
| verbose | (optional) [boolean] boolean indicating if the function is run in verbose mode or not, default = FALSE |

Value

POSIXlt+numeric dataframe containing a vector of aggregated dates and time series values

Author(s)

Laurent Coron (March 2015)

Examples

```
## loading catchment data
library(airGR)

## preparation of the initial time series data frame at the daily time step
data(L0123002)
TabSeries <- data.frame(BasinObs$DatesR, BasinObs$P, BasinObs$E, BasinObs$T, BasinObs$Qmm)
TimeFormat <- "daily"

## conversion at the monthly time step
NewTimeFormat <- "monthly"
ConvertFun <- c("sum", "sum", "mean", "sum")
NewTabSeries <- SeriesAggreg(TabSeries = TabSeries, TimeFormat, NewTimeFormat, ConvertFun)

## conversion at the yearly time step
NewTimeFormat <- "yearly"
ConvertFun <- c("sum", "sum", "mean", "sum")
NewTabSeries <- SeriesAggreg(TabSeries = TabSeries, TimeFormat, NewTimeFormat, ConvertFun)
```

TransfoParam

Transformation of the parameters using the provided function

Description

Function which transforms model parameters using the provided function (from raw to transformed parameters and vice versa).

Usage

```
TransfoParam(ParamIn, Direction, FUN_TRANSFO)
```

Arguments

| | |
|-------------|--|
| ParamIn | [numeric] matrix of parameter sets (sets in line, parameter values in column) |
| Direction | [character] direction of the transformation: use "RT" for Raw->Transformed and "TR" for Transformed->Raw |
| FUN_TRANSFO | [function] model parameters transformation function (e.g. TransfoParam_GR4J, TransfoParam_CemaNeigeGR4J) |

Value

ParamOut [numeric] matrix of parameter sets (sets in line, parameter values in column)

Author(s)

Laurent Coron (June 2014)

See Also

[TransfoParam_GR4J](#), [TransfoParam_GR5J](#), [TransfoParam_GR6J](#), [TransfoParam_CemaNeige](#)

Examples

```
library(airGR)

## transformation Raw->Transformed for the GR4J model
Xraw <- matrix(c(+221.41, -3.63, +30.00, +1.37,
                +347.23, -1.03, +60.34, +1.76,
                +854.06, -0.10, +148.41, +2.34),
              ncol = 4, byrow = TRUE)
Xtran <- TransfoParam(ParamIn = Xraw, Direction = "RT", FUN_TRANSFO = TransfoParam_GR4J)

## transformation Transformed->Raw for the GR4J model
Xtran <- matrix(c(+3.60, -2.00, +3.40, -9.10,
                +3.90, -0.90, +4.10, -8.70,
                +4.50, -0.10, +5.00, -8.10),
              ncol = 4, byrow = TRUE)
Xraw <- TransfoParam(ParamIn = Xtran, Direction = "TR", FUN_TRANSFO = TransfoParam_GR4J)
```

TransfoParam_CemaNeige

Transformation of the parameters of the CemaNeige module

Description

Function which transforms model parameters of the CemaNeige module (from raw to transformed parameters and vice versa).

Usage

```
TransfoParam_CemaNeige(ParamIn, Direction)
```

Arguments

| | |
|-----------|--|
| ParamIn | [numeric] matrix of parameter sets (sets in line, parameter values in column) |
| Direction | [character] direction of the transformation: use "RT" for Raw->Transformed and "TR" for Transformed->Raw |

Value

ParamOut [numeric] matrix of parameter sets (sets in line, parameter values in column)

Author(s)

Laurent Coron (December 2013)

See Also

[TransfoParam](#), [TransfoParam_GR4J](#), [TransfoParam_GR5J](#), [TransfoParam_GR6J](#)

Examples

```
library(airGR)

## transformation Raw->Transformed for the CemaNeige module
Xraw <- matrix(c(+0.19, +1.73,
                +0.39, +2.51,
                +0.74, +4.06),
              ncol = 2, byrow = TRUE)
Xtran <- TransfoParam_CemaNeige(ParamIn = Xraw , Direction = "RT")

## transformation Transformed->Raw for the CemaNeige module
Xtran <- matrix(c(-6.26, +0.55,
                 -2.13, +0.92,
                 +4.86, +1.40),
               ncol = 2, byrow = TRUE)
Xraw <- TransfoParam_CemaNeige(ParamIn = Xtran, Direction = "TR")
```

| | |
|-------------------|---|
| TransfoParam_GR1A | <i>Transformation of the parameters of the GR1A model</i> |
|-------------------|---|

Description

Function which transforms model parameters of the GR1A model (from real to transformed parameters and vice versa).

Usage

```
TransfoParam_GR1A(ParamIn, Direction)
```

Arguments

| | |
|-----------|--|
| ParamIn | [numeric] matrix of parameter sets (sets in line, parameter values in column) |
| Direction | [character] direction of the transformation: use "RT" for Raw->Transformed and "TR" for Transformed->Raw |

Value

ParamOut [numeric] matrix of parameter sets (sets in line, parameter values in column)

Author(s)

Laurent Coron (March 2015)

| | |
|-------------------|---|
| TransfoParam_GR2M | <i>Transformation of the parameters of the GR2M model</i> |
|-------------------|---|

Description

Function which transforms model parameters of the GR2M model (from real to transformed parameters and vice versa).

Usage

```
TransfoParam_GR2M(ParamIn, Direction)
```

Arguments

| | |
|-----------|--|
| ParamIn | [numeric] matrix of parameter sets (sets in line, parameter values in column) |
| Direction | [character] direction of the transformation: use "RT" for Raw->Transformed and "TR" for Transformed->Raw |

Value

ParamOut [numeric] matrix of parameter sets (sets in line, parameter values in column)

Author(s)

Laurent Coron (March 2015)

| | |
|-------------------|---|
| TransfoParam_GR4H | <i>Transformation of the parameters of the GR4H model</i> |
|-------------------|---|

Description

Function which transforms model parameters of the GR4H model (from real to transformed parameters and vice versa).

Usage

```
TransfoParam_GR4H(ParamIn, Direction)
```

Arguments

| | |
|-----------|--|
| ParamIn | [numeric] matrix of parameter sets (sets in line, parameter values in column) |
| Direction | [character] direction of the transformation: use "RT" for Raw->Transformed and "TR" for Transformed->Raw |

Value

ParamOut [numeric] matrix of parameter sets (sets in line, parameter values in column)

Author(s)

Laurent Coron, Claude Michel, Thibault Mathevet (July 2014)

| | |
|-------------------|---|
| TransfoParam_GR4J | <i>Transformation of the parameters of the GR4J model</i> |
|-------------------|---|

Description

Function which transforms model parameters of the GR4J model (from real to transformed parameters and vice versa).

Usage

```
TransfoParam_GR4J(ParamIn, Direction)
```

Arguments

| | |
|-----------|--|
| ParamIn | [numeric] matrix of parameter sets (sets in line, parameter values in column) |
| Direction | [character] direction of the transformation: use "RT" for Raw->Transformed and "TR" for Transformed->Raw |

Value

ParamOut [numeric] matrix of parameter sets (sets in line, parameter values in column)

Author(s)

Laurent Coron (December 2013)

See Also

[TransfoParam](#), [TransfoParam_GR5J](#), [TransfoParam_GR6J](#), [TransfoParam_CemaNeige](#)

Examples

```
library(airGR)

## transformation Raw->Transformed for the GR4J model
Xraw <- matrix(c(+221.41, -3.63, +30.00, +1.37,
                +347.23, -1.03, +60.34, +1.76,
                +854.06, -0.10, +148.41, +2.34),
              ncol = 4, byrow = TRUE)
Xtran <- TransfoParam_GR4J(ParamIn = Xraw , Direction = "RT")

## transformation Transformed->Raw for the GR4J model
Xtran <- matrix(c(+3.60, -2.00, +3.40, -9.10,
                +3.90, -0.90, +4.10, -8.70,
                +4.50, -0.10, +5.00, -8.10),
              ncol = 4, byrow = TRUE)
Xraw <- TransfoParam_GR4J(ParamIn = Xtran, Direction = "TR")
```

| | |
|-------------------|---|
| TransfoParam_GR5J | <i>Transformation of the parameters of the GR5J model</i> |
|-------------------|---|

Description

Function which transforms model parameters of the GR5J model (from real to transformed parameters and vice versa).

Usage

```
TransfoParam_GR5J(ParamIn, Direction)
```

Arguments

| | |
|-----------|--|
| ParamIn | [numeric] matrix of parameter sets (sets in line, parameter values in column) |
| Direction | [character] direction of the transformation: use "RT" for Raw->Transformed and "TR" for Transformed->Raw |

Value

ParamOut [numeric] matrix of parameter sets (sets in line, parameter values in column)

Author(s)

Laurent Coron (December 2013)

See Also

[TransfoParam](#), [TransfoParam_GR4J](#), [TransfoParam_GR6J](#), [TransfoParam_CemaNeige](#)

Examples

```
library(airGR)

## transformation Raw->Transformed for the GR5J model
Xraw <- matrix(c(+221.41, -2.65, +27.11, +1.37, -0.76,
               +347.23, -0.64, +60.34, +1.76, +0.30,
               +854.01, -0.10, +148.41, +2.34, +0.52),
              ncol = 5, byrow = TRUE)
Xtran <- TransfoParam_GR5J(ParamIn = Xraw , Direction = "RT")

## transformation Transformed->Raw for the GR5J model
Xtran <- matrix(c(+3.60, -1.70, +3.30, -9.10, -0.70,
               +3.90, -0.60, +4.10, -8.70, +0.30,
               +4.50, -0.10, +5.00, -8.10, +0.50),
              ncol = 5, byrow = TRUE)
Xraw <- TransfoParam_GR5J(ParamIn = Xtran, Direction = "TR")
```

| | |
|-------------------|---|
| TransfoParam_GR6J | <i>Transformation of the parameters of the GR6J model</i> |
|-------------------|---|

Description

Function which transforms model parameters of the GR6J model (from real to transformed parameters and vice versa).

Usage

```
TransfoParam_GR6J(ParamIn, Direction)
```

Arguments

| | |
|-----------|--|
| ParamIn | [numeric] matrix of parameter sets (sets in line, parameter values in column) |
| Direction | [character] direction of the transformation: use "RT" for Raw->Transformed and "TR" for Transformed->Raw |

Value

ParamOut [numeric] matrix of parameter sets (sets in line, parameter values in column)

Author(s)

Laurent Coron (December 2013)

See Also

[TransfoParam](#), [TransfoParam_GR4J](#), [TransfoParam_GR5J](#), [TransfoParam_CemaNeige](#)

Examples

```
library(airGR)

## transformation Raw->Transformed for the GR6J model
Xraw <- matrix(c(+221.41, -1.18, +27.11, 1.37, -0.18, +20.09,
                +347.23, -0.52, +60.34, 1.76, +0.02, +54.60,
                +854.06, +0.52, +148.41, 2.34, +0.22, +148.41),
              ncol = 6, byrow = TRUE)
Xtran <- TransfoParam_GR6J(ParamIn = Xraw , Direction = "RT")

## transformation Transformed->Raw for the GR6J model
Xtran <- matrix(c(+3.60, -1.00, +3.30, -9.10, -0.90, +3.00,
                +3.90, -0.50, +4.10, -8.70, +0.10, +4.00,
                +4.50, +0.50, +5.00, -8.10, +1.10, +5.00),
              ncol = 6, byrow = TRUE)
Xraw <- TransfoParam_GR6J(ParamIn = Xtran, Direction = "TR")
```

Index

*Topic **package, hydrology, modelling**
airGR, 2

airGR, 2

BasinInfo, 4

BasinObs, 5

Calibration, 3, 5, 8, 11

Calibration_Michel, 3, 6, 7

CreateCalibOptions, 3, 6, 7, 9, 10, 13, 16, 19

CreateInputsCrit, 3, 6, 7, 9, 12, 16, 19, 21,
23–26

CreateInputsModel, 2, 3, 6–8, 13, 15, 17, 19,
21, 29–36, 38–40, 42–49

CreateRunOptions, 2, 3, 6–8, 13, 16, 17,
29–49

DataAltiExtrapolation_Valery, 16, 19

ErrorCrit, 3, 6, 21

ErrorCrit_KGE, 21, 22, 25–27

ErrorCrit_KGE2, 23, 24, 26, 27

ErrorCrit_NSE, 3, 21, 23, 25, 25, 27

ErrorCrit_RMSE, 8, 21, 23, 25, 26, 26

PEdaily_Oudin, 27

plot_OutputsModel, 28

RunModel, 2, 3, 11, 13, 16, 19, 29

RunModel_CemaNeige, 3, 30

RunModel_CemaNeigeGR4J, 3, 21, 23–26, 29,
31, 31, 35, 38, 45

RunModel_CemaNeigeGR5J, 3, 33, 34, 38, 47

RunModel_CemaNeigeGR6J, 3, 33, 35, 36, 49

RunModel_GR1A, 3, 38

RunModel_GR2M, 3, 40

RunModel_GR4H, 3, 42

RunModel_GR4J, 3, 8, 21, 23–26, 29, 33, 43,
44, 47, 49

RunModel_GR5J, 3, 35, 45, 46, 49

RunModel_GR6J, 3, 38, 45, 47, 48

SeriesAggreg, 50

TransfoParam, 6, 51, 52, 55–57

TransfoParam_CemaNeige, 51, 52, 55–57

TransfoParam_GR1A, 53

TransfoParam_GR2M, 54

TransfoParam_GR4H, 54

TransfoParam_GR4J, 8, 51, 52, 55, 56, 57

TransfoParam_GR5J, 51, 52, 55, 56, 57

TransfoParam_GR6J, 51, 52, 55, 56, 57