Package 'PRSim'

January 20, 2025

Type Package

Title Stochastic Simulation of Streamflow Time Series using Phase Randomization

Version 1.5

Date 2024-03-14

Author Manuela Brunner [aut, cre] (<https://orcid.org/0000-0001-8824-877X>), Reinhard Furrer [aut] (<https://orcid.org/0000-0002-6319-2332>), R Core Teamn [ctb, cph] (ks_test.c)

Maintainer Manuela Brunner <manuela.brunner@env.ethz.ch>

Description Provides a simulation framework to simulate streamflow time series with similar main characteristics as observed data. These characteristics include the distribution of daily streamflow values and their temporal correlation as expressed by short- and long-range dependence. The approach is based on the randomization of the phases of the Fourier transform or the phases of the wavelet transform. The function prsim() is applicable to single site simulation and uses the Fourier transform. The function prsim.wave() extends the approach to multiple sites and is based on the complex wavelet transform. The function prsim.weather() extends the approach to multiple variables for weather generation. We further use the flexible four-parameter Kappa distribution, which allows for the extrapolation to yet unobserved low and high flows. Alternatively, the empirical or any other distribution can be used. A detailed description of the simulation approach for single sites and an application example can be found in Brunner et al. (2019) <doi:10.5194/hess-23-3175-2019>. A detailed description and evaluation of the wavelet-based multi-site approach can be found in Brunner and Gilleland (2020) <doi:10.5194/hess-24-3967-2020>. A detailed description and evaluation of the multi-variable and multi-site weather generator can be found in Brunner et al. (2021) <doi:10.5194/esd-12-621-2021>. A detailed description and evaluation of the non-stationary streamflow generator can be found in Brunner and Gilleland (2024) <doi:10.1029/2023EF004238>.

URL https://git.math.uzh.ch/reinhard.furrer/PRSim-devel

BugReports https://git.math.uzh.ch/reinhard.furrer/PRSim-devel/-/issues

License GPL-3

Encoding UTF-8

LazyData true

Depends R (>= 3.5.0)

Suggests lattice, ismev, evd, GB2, boot, MASS

Imports stats, methods, lmomco, mev, goftest, wavScalogram, splus2R

RoxygenNote 7.2.3

NeedsCompilation yes

Repository CRAN

Date/Publication 2024-04-08 08:33:00 UTC

Contents

`PRSim-package` *Stochastic Simulation of Streamflow Time Series using Phase Randomization*

Description

Provides a simulation framework to simulate streamflow time series with similar main characteristics as observed data. These characteristics include the distribution of daily streamflow values and their temporal correlation as expressed by short- and long-range dependence. The approach is based on the randomization of the phases of the Fourier transform or the phases of the wavelet transform. The function prsim() is applicable to single site simulation and uses the Fourier transform. The function prsim.wave() extends the approach to multiple sites and is based on the complex wavelet transform. The function prsim.weather() extends the approach to multiple variables for weather generation. We further use the flexible four-parameter Kappa distribution, which allows for the extrapolation to yet unobserved low and high flows. Alternatively, the empirical or any other distribution can be used. A detailed description of the simulation approach for single sites and an application example can be found in Brunner et al. (2019) <doi:10.5194/hess-23-3175-2019>. A detailed description and evaluation of the wavelet-based multi-site approach can be found in Brunner and Gilleland (2020) <doi:10.5194/hess-24-3967-2020>. A detailed description and evaluation of the multi-variable and multi-site weather generator can be found in Brunner et al.

(2021) <doi:10.5194/esd-12-621-2021>. A detailed description and evaluation of the non-stationary streamflow generator can be found in Brunner and Gilleland (2024) <doi:10.1029/2023EF004238>.

Details

The DESCRIPTION file:

Package:	PRSim
Type:	Package
Title:	Stochastic Simulation of Streamflow Time Series using Phase Randomization
Version:	1.5
Date:	2024-03-14
Authors@R:	c(person("Manuela", "Brunner", role = c("aut", "cre"), email = "manuela.brunner@env.ethz.ch", comment =
Author:	Manuela Brunner [aut, cre] (<https://orcid.org/0000-0001-8824-877X>), Reinhard Furrer [aut] (<https://orci
Maintainer:	Manuela Brunner <manuela.brunner@env.ethz.ch>
Description:	Provides a simulation framework to simulate streamflow time series with similar main characteristics as obse
URL:	https://git.math.uzh.ch/reinhard.furrer/PRSim-devel
BugReports:	https://git.math.uzh.ch/reinhard.furrer/PRSim-devel/-/issues
License:	GPL-3
Encoding:	UTF-8
LazyData:	true
Depends:	R (>= 3.5.0)
Suggests:	lattice, ismev, evd, GB2, boot, MASS
Imports:	stats, methods, lmomco, mev, goftest, wavScalogram, splus2R
RoxygenNote:	7.2.3
Archs:	x64

Index of help topics:

`weather_sim_multi_sites`
: Simulated temperature and precipitation for two grid cells

Contains two functions for the stochastic simulation of continuous discharge time series: prsim and prsim.wave both using phase randomization. prsim is based on the Fourier transform while prsim.wave uses the wavelet transform.

prsim: Simulation in the frequency domain is based on the randomization of the phases of the Fourier transform. We here combine phase randomization simulation with the flexible, four-parameter kappa distribution, which allows for the extrapolation to yet unobserved low and high flows. Alternative distributions or the empirical distribution can be used instead. The simulation approach consists of eight steps: (1) fitting of theoretical Kappa distribution, (2) normalization and deseasonalization, (3) Fourier transformation, (4) Fourier phases computation, (5) random phase generation, (6) inverse Fourier transformation, (7) back transformation, and (8) simulation.

prsim.wave: Simulation for multiple sites in the frequency domain based on the randomization of the phases of the continuous wavelet transform. We combine phase randomization with the flexible, four-parameter kappa distribution. Alternative theoretical distributions or the empirical distribution can be used instead. The simulation procedure consists of five steps: (1) Derivation of random phases from a white noise time series, (2) Fitting of kappa distribution, (3) Wavelet transform, (4) Inverse wavelet transform, and (5) Transformation to the kappa distribution (or the distribution of choice).

prsim.weather: Simulation of two variables (temperature and precipitation) for multiple sites in the frequency domain based on the randomization of the phases of the continuous wavelet transform. We combine phase randomization with the flexible, skewed exponential power (sep) and extended generalized pareto distributions (egpd). Alternative theoretical distributions can be used instead. The simulation procedure consists of five steps: (1) Derivation of random phases from a randomly sampled time series, (2) Fitting of temperature and precipitation disstributions, (3) Wavelet transform, (4) Inverse wavelet transform, and (5) Transformation to the desired distributions.

Author(s)

Manuela Brunner [aut, cre] (<https://orcid.org/0000-0001-8824-877X>), Reinhard Furrer [aut] (<https://orcid.org/0000-0002-6319-2332>), R Core Teamn [ctb, cph] (ks_test.c)

Maintainer: Manuela Brunner <manuela.brunner@env.ethz.ch>

References

Brunner, M. I., A. Bárdossy, and R. Furrer (2019). Technical note: Stochastic simulation of streamflow time series using phase randomization. Hydrology and Earth System Sciences, 23, 3175-3187, https://doi.org/10.5194/hess-23-3175-2019.

Brunner, M. I., and E. Gilleland (2020). Stochastic simulation of streamflow and spatial extremes: a continuous, wavelet-based approach, Hydrology and Earth System Sciences, https://doi.org/10.5194/hess-24-3967-2020.

Brunner, M. I., and E. Gilleland (2021). Spatial compound hot-dry events in the United States: assessment using a multi-site multi-variable weather generator, in preparation.

Examples

```
demo("PRSim")
demo("PRSim-validate")
demo("PRSim_wave")
demo("PRSim_wave-validate")
demo("PRSim_weather")
demo("PRSim_weather-validate")
```

---

pRsim *Simulate for one station*

Description

Applies the algorithm to a single station

Usage

```
prsim(data, station_id="Qobs", number_sim=1, win_h_length=15,
      marginal=c("kappa","empirical"), n_par=4, marginalpar=TRUE,
      GoFtest=NULL, verbose=TRUE, suppWarn=FALSE, ...)
```

Arguments

data	data frame containing the time indications and runoff of at least one station. See 'Details'.
station_id	identifies the station in case several runoffs are present in data. See 'Details'.
number_sim	number of simulations to be carried out.
win_h_length	(half-)length of moving window size.
marginal	marginal distribution to be used for the backtransformation. Can be either "kappa", "empirical", or any type of CDF (see 'Details'). "kappa" uses the four-parameter kappa distribution for backtransformation, "empirical" uses the empirical distribution. CDF allows for specifying any distribution 'Examples'.
n_par	number of parameters of the marginal distribution used
GoFtest	If (non-null) a GoF test for daily data should be performed: "KS" performs a Kolmogorof-Smirnov test, and "AD" performs an Anderson-Darling test. see 'Details')
verbose	logical. Should progress be reported?
marginalpar	logical. Should the estimated parameters of the distribution used be returned?
suppWarn	logical. See 'Details'.
...	any other argument passed to the sub-function specifying the cdf for fitting. See 'Details' and 'Examples'.

Details

Time can be given with three columns named "YYYY", "MM", "DD", or as in POSIXct format YYYY-MM-DD. All leap days (Feb 29th) will be omitted from the analysis, but no missing observations are allowed.

Stations are identified by column name (default "Qobs"), or by column index.

The function homtest::par.kappa might issue quite a few warnings of type In fn(par, ...) : value out of range in 'gammafn'. The argument suppWarn allows to silence warnings for the specific function call via suppressWarnings(). Of course, a subsequent check via warnings() is recommended.

Alternative distributions can be specified by providing three functions: (1) a function fitting the parameters of a distributions and providing a vector of these parameters as output (CDF_fit), (2) a function simulating random numbers from this distribution (rCDF), and (3) a function specifying the distribution (pCDF). See 'Examples' for the generalized beta for the second kind and for the Generalized Extreme Values (GEV) distribution.

When using the kappa distribution, the AD test can for certain values of the parameter h not be performed.

Value

A list with elements

simulation	A data frame with time information, observations, deseaonalized observations and number_sim columns containing the simulated runoff.
pars	A matrix containing the estimated parameters of the marginal distribution (if marginalpar).
p_val	A vector containing the p-values of ks.test or ad.test applied to the daily detrended data (if GoFtest is not NULL)

Author(s)

Manuela Brunner

References

Brunner, M. I., A. Bárdossy, and R. Furrer (2019). Technical note: Stochastic simulation of streamflow time series using phase randomization. Hydrology and Earth System Sciences, 23, 3175-3187, https://doi.org/10.5194/hess-23-3175-2019.

See Also

ks.test

Examples

```
data(runoff)
out <- prsim( runoff[ runoff$YYYY<1980, ], "Qobs", 1, suppWarn=TRUE)
# warnings() # as a follow-up to `suppWarn=TRUE`
```

```
## Specifying particular CDFs:
## (1) example with the Generalized Extreme Value (GEV) distribution
require("evd")
require("ismev")
rGEV <- function(n, theta)   rgev(n, theta[1], theta[2], theta[3])
pGEV <- function(x, theta)   pgev(x, theta[1], theta[2], theta[3])
GEV_fit <- function( xdat, ...)   gev.fit( xdat, ...)$mle


## (2) example with generalized Beta distribution of the second kind

require( "GB2")
rGB2 <- function(n, theta)  rgb2(n, theta[1], theta[2], theta[3], theta[4])
pGB2 <- function(x, theta)  pgb2(x, theta[1], theta[2], theta[3], theta[4])
GB2_fit <- function( xdat, ...)   ml.gb2( xdat, ...)$opt1$par
```

`pRsim.wave` *Simulate for multiple stations*

Description

Applies the wavelet-based simulation algorithm to multiple sites (single site possible as well)

Usage

```
prsim.wave(data, station_id="Qobs", number_sim=1, win_h_length=15,
            marginal=c("kappa","empirical"), n_par=4, n_wave=100, marginalpar=TRUE,
                  GoFtest=NULL, verbose=TRUE, suppWarn=FALSE, ...)
```

Arguments

`data`	list of data frames. One list entry, i.e. data frame, corresponds to one station. Each data frame contains the time indications and runoff of one station. See 'Details'.
`station_id`	identifies the station in case several time series are present in `data`. See 'Details'.
`number_sim`	number of simulations to be carried out.
`win_h_length`	(half-)length of moving window size.
`marginal`	marginal distribution to be used for the backtransformation. Can be either `"kappa"`, `"empirical"`, or any type of CDF (see 'Details'). `"kappa"` uses the four-parameter kappa distribution for backtransformation, `"empirical"` uses the empirical distribution. CDF allows for specifying any distribution 'Examples'.
`n_par`	number of parameters of the marginal distribution used

GoFtest	If (non-null) a GoF test for daily data should be performed: "KS" performs a Kolmogorof-Smirnov test, and "AD" performs an Anderson-Darling test. see 'Details')
verbose	logical. Should progress be reported?
marginalpar	logical. Should the estimated parameters of the distribution used be returned?
n_wave	number of scales to be considered in the continuous wavelet transform.
suppWarn	logical. See 'Details'.
...	any other argument passed to the sub-function specifying the cdf for fitting. See 'Details' and 'Examples'.

Details

Time can be given with three columns named "YYYY", "MM", "DD", or as in POSIXct format YYYY-MM-DD. All leap days (Feb 29th) will be omitted from the analysis, but no missing observations are allowed.

Stations are identified by list index.

The function homtest::par.kappa might issue quite a few warnings of type In fn(par, ...) : value out of range in 'gammafn'. The argument suppWarn allows to silence warnings for the specific function call via suppressWarnings(). Of course, a subsequent check via warnings() is recommended.

Alternative distributions can be specified by providing three functions: (1) a function fitting the parameters of a distributions and providing a vector of these parameters as output (CDF_fit), (2) a function simulating random numbers from this distribution (rCDF), and (3) a function specifying the distribution (pCDF). See 'Examples' for the generalized beta for the second kind and for the Generalized Extreme Values (GEV) distribution.

When using the kappa distribution, the AD test can for certain values of the parameter h not be performed.

Value

A list with elements

simulation	A data frame with time information, observations, and number_sim columns containing the simulated runoff.
pars	A matrix containing the estimated parameters of the marginal distribution (if marginalpar).
p_val	A vector containing the p-values of ks.test or ad.test applied to the daily detrended data (if GoFtest is not NULL)

Author(s)

Manuela Brunner

References

Brunner, M. I., and E. Gilleland (2020). Stochastic simulation of streamflow and spatial extremes: a continuous, wavelet-based approach, Hydrology and Earth System Sciences, https://doi.org/10.5194/hess-24-3967-2020.

See Also

ks.test

Examples

```
data(runoff_multi_sites)

## Specifying particular CDFs:
## (1) example with the Generalized Extreme Value (GEV) distribution

require("evd")
require("ismev")
rGEV <- function(n, theta)   rgev(n, theta[1], theta[2], theta[3])
pGEV <- function(x, theta)   pgev(x, theta[1], theta[2], theta[3])
GEV_fit <- function( xdat, ...)   gev.fit( xdat, ...)$mle


## (2) example with generalized Beta distribution of the second kind

require( "GB2")
rGB2 <- function(n, theta)  rgb2(n, theta[1], theta[2], theta[3], theta[4])
pGB2 <- function(x, theta)  pgb2(x, theta[1], theta[2], theta[3], theta[4])
GB2_fit <- function( xdat, ...)   ml.gb2( xdat, ...)$opt1$par
```

---

pRsim.wave.nonstat *Simulate for multiple stations under non-stationary conditions*

Description

Applies the wavelet-based and non-stationary simulation algorithm to multiple sites (single site possible as well)

Usage

```
prsim.wave.nonstat(data, station_id="Qobs", number_sim=1, win_h_length=15,
             marginal=c("kappa","empirical"), n_par=4, n_wave=100, cov_name='T',
                  marginalpar=TRUE, GoFtest=NULL, verbose=TRUE,
                  suppWarn=FALSE, warming_level, ...)
```

Arguments

data	list of data frames. One list entry, i.e. data frame, corresponds to one station. Each data frame contains the time indications and runoff of one station. See 'Details'.
station_id	identifies the station in case several time series are present in data. See 'Details'.

number_sim	number of simulations to be carried out.
win_h_length	(half-)length of moving window size.
marginal	marginal distribution to be used for the backtransformation. Can be either "kappa", "empirical", or any type of CDF (see 'Details'). "kappa" uses the four-parameter kappa distribution for backtransformation, "empirical" uses the empirical distribution. CDF allows for specifying any distribution 'Examples'.
n_par	number of parameters of the marginal distribution used
GoFtest	If (non-null) a GoF test for daily data should be performed: "KS" performs a Kolmogorof-Smirnov test, and "AD" performs an Anderson-Darling test. see 'Details')
verbose	logical. Should progress be reported?
cov_name	character. 'T' for temperature. Has to correspond to covariate name used in data list.
marginalpar	logical. Should the estimated parameters of the distribution used be returned?
n_wave	number of scales to be considered in the continuous wavelet transform.
suppWarn	logical. See 'Details'.
warming_level	a vector of station-specific warming levels. Each vector entry contains the warming level for the corresponding station part of the data list. For example, vector entry 1 represents the warming level for station 1 in the data list.
...	any other argument passed to the sub-function specifying the cdf for fitting. See 'Details' and 'Examples'.

Details

Time can be given with three columns named "YYYY", "MM", "DD", or as in POSIXct format YYYY-MM-DD. All leap days (Feb 29th) will be omitted from the analysis, but no missing observations are allowed.

Stations are identified by list index.

The function homtest::par.kappa might issue quite a few warnings of type In fn(par, ...) : value out of range in 'gammafn'. The argument suppWarn allows to silence warnings for the specific function call via suppressWarnings(). Of course, a subsequent check via warnings() is recommended.

Alternative distributions can be specified by providing three functions: (1) a function fitting the parameters of a distributions and providing a vector of these parameters as output (CDF_fit), (2) a function simulating random numbers from this distribution (rCDF), and (3) a function specifying the distribution (pCDF). See 'Examples' for the generalized beta for the second kind and for the Generalized Extreme Values (GEV) distribution.

When using the kappa distribution, the AD test can for certain values of the parameter h not be performed.

Value

A list with elements

simulation	A data frame with time information, observations, and number_sim columns containing the simulated runoff.
pars	A matrix containing the estimated parameters of the marginal distribution (if marginalpar).
p_val	A vector containing the p-values of ks.test or ad.test applied to the daily detrended data (if GoFtest is not NULL)

Author(s)

Manuela Brunner

References

Brunner, M. I., and E. Gilleland (2024). Future changes in floods, droughts, and their extents in the Alps: a sensitivity analysis with a non-stationary stochastic streamflow generator, Earth's Future.

See Also

ks.test

---

pRsim.weather *Weather simulation (temperature and precipitation) for multiple stations*

Description

Applies the wavelet-based weather simulation algorithm to multiple sites (single site possible as well)

Usage

```
prsim.weather(data_p, data_t, station_id_p="Precip",
station_id_t="Temp", number_sim=1, win_h_length=15,
n_wave=100,verbose=TRUE,t_margin='sep',p_margin='egpd',...)
```

Arguments

data_p	list of precipitation data frames. One list entry, i.e. data frame, corresponds to one station/grid cell. Each data frame contains the time indications and precipitation of one station. See 'Details'.
data_t	list of temperature data frames. One list entry, i.e. data frame, corresponds to one station/grid cell. Each data frame contains the time indications and temperature of one station. See 'Details'.
station_id_p	identifies the precipitation variable name in case several time series are present in data_p. See 'Details'.
station_id_t	identifies the temperature variable name in case several time series are present in data_t. See 'Details'.

number_sim	number of simulations to be carried out.
win_h_length	(half-)length of moving window size.
t_margin	marginal distribution to be used for the backtransformation of temperature. Can be either "sep" or any type of CDF (see 'Details'). "sep" uses the four-parameter skewed-exponential power for backtransformation. CDF allows for specifying any distribution 'Examples'.
p_margin	marginal distribution to be used for the backtransformation of precipitation. Can be either "egpd" or any type of CDF (see 'Details'). "egpd" uses the extended GPD for backtransformation. CDF allows for specifying any distribution 'Examples'.
verbose	logical. Should progress be reported?
n_wave	number of scales to be considered in the continuous wavelet transform.
...	any other argument passed to the sub-function specifying the cdf for fitting. See 'Details' and 'Examples'.

Details

Time can be given with three columns named "YYYY", "MM", "DD", or as in POSIXct format YYYY-MM-DD. All leap days (Feb 29th) will be omitted from the analysis, but no missing observations are allowed.

Stations are identified by list index.

Alternative distributions can be specified by providing three functions: (1) a function fitting the parameters of a distributions and providing a vector of these parameters as output (CDF_fit), (2) a function simulating random numbers from this distribution (rCDF), and (3) a function specifying the distribution (pCDF). See 'Examples' for the generalized beta for the second kind and for the Generalized Extreme Values (GEV) distribution.

Value

A list with elements temperature and precipitation of

simulation	data frames with time information, observations, and number_sim columns containing the simulated data.

Author(s)

Manuela Brunner

References

Brunner, M. I., and E. Gilleland (2021). Spatial compound hot-dry events in the United States: assessment using a multi-site multi-variable weather generator, in preparation.

runoff *Sample runoff of a catchment*

Description

Artifical runoff data based on actual and simulated observations.

Usage

```
data("runoff")
```

Format

A data frame with 15695 observations of the following 4 variables.

`YYYY` a numeric vector, year

`MM` a numeric vector, month

`DD` a numeric vector, day

`Qobs` a numeric vector, synthetic observed runoff

Details

The data mimiks the runoff of the river Plessur at the gauging station Chur, Switzerland. The the flow regime of the river is melt dominated. More information is given in the reference below.

Source

The provided data is a weighted average of the acutually observed values and a particular simulated runoff. The actual discharge data can be ordered from http://www.bafu.admin.ch/wasser/13462/13494/15076/index.

References

Brunner, M. I., A. Bárdossy, and R. Furrer (2019). Technical note: Stochastic simulation of streamflow time series using phase randomization. Hydrology and Earth System Sciences, 23, 3175-3187, https://doi.org/10.5194/hess-23-3175-2019.

Examples

```
data(runoff)
str(runoff)
runoff$timestamp <- paste(runoff$YYYY, runoff$MM, runoff$DD, sep=" ")
runoff$timestamp <- as.POSIXct(strptime(runoff$timestamp,
                                format="%Y %m %d", tz="GMT"))
plot(runoff$timestamp[1:1000], runoff$Qobs[1:1000], type="l",
  xlab="Time [d]", ylab=expression(paste("Discharge [m"^3,"/s]")))
```

runoff_multi_sites *Sample runoff of four catchments with a similar discharge regime*

Description

Observed runoff data from four USGS sites.

Usage

```
data("runoff_multi_sites")
```

Format

A list of four data frames (one list per station) of the following 4 variables.

`YYYY` a numeric vector, year

`MM` a numeric vector, month

`DD` a numeric vector, day

`Qobs` a numeric vector, observed runoff

Details

The data contains runoff for four USGS gages: (i) Calawah River near Forks, WA (USGS 12043000), (ii) NF Stillaguamish River near Arlington, WA (USGS 12167000), (iii) Nehalem River near Foss, OR (USGS 14301000), and (iv) Steamboat Creek near Glide, OR (USGS 14316700).

Source

The actual discharge data were downloaded from https://waterdata.usgs.gov/nwis.

References

Brunner, M. I., A. Bárdossy, and R. Furrer (2019). Technical note: Stochastic simulation of streamflow time series using phase randomization. Hydrology and Earth System Sciences, 23, 3175-3187, https://doi.org/10.5194/hess-23-3175-2019.

Examples

```
data(runoff_multi_sites)
str(runoff_multi_sites)
runoff_multi_sites[[1]]$timestamp <- paste(runoff_multi_sites[[1]]$YYYY,
runoff_multi_sites[[1]]$MM, runoff_multi_sites[[1]]$DD, sep=" ")
runoff_multi_sites[[1]]$timestamp <-
as.POSIXct(strptime(runoff_multi_sites[[1]]$timestamp,format="%Y %m %d", tz="GMT"))
plot(runoff_multi_sites[[1]]$timestamp[1:1000], runoff_multi_sites[[1]]$Qobs[1:1000], type="l",
  xlab="Time [d]", ylab=expression(paste("Discharge [m"^3,"/s]")))
```

runoff_multi_site_T *Sample runoff and temperature data of two catchments with a similar discharge regime*

Description

Observed runoff data from two catchments in Switzerland.

Usage

```
data("runoff_multi_site_T")
```

Format

A list of two data frames (one list per station) of the following 5 variables.

`YYYY` a numeric vector, year

`MM` a numeric vector, month

`DD` a numeric vector, day

`Qobs` a numeric vector, observed runoff

`T` a numeric vector, average temperature

Details

The data contains runoff for two Swiss gages: (i) Thur Andelfingen (FOEN 2044) and (ii) Alpbach Erstfeld (FOEN 2299).

Source

The actual discharge data were ordered from https://www.bafu.admin.ch/bafu/en/home/topics/water/state/data/obtaining-monitoring-data-on-the-topic-of-water/hydrological-data-service-for-wat html. Temperature averages were computed from E-OBS (https://cds.climate.copernicus.eu/cdsapp#!/dataset/insitu-gridded-observations-europe?tab=overview).

References

Brunner, M. I. and Eric Gilleland in preparation.

Examples

```
data(runoff_multi_site_T)
str(runoff_multi_site_T)
runoff_multi_site_T[[1]]$timestamp <- paste(runoff_multi_site_T[[1]]$YYYY,
runoff_multi_site_T[[1]]$MM, runoff_multi_site_T[[1]]$DD, sep=" ")
runoff_multi_site_T[[1]]$timestamp <-
as.POSIXct(strptime(runoff_multi_site_T[[1]]$timestamp,format="%Y %m %d", tz="GMT"))
plot(runoff_multi_site_T[[1]]$timestamp[1:1000], runoff_multi_site_T[[1]]$Qobs[1:1000], type="l",
  xlab="Time [d]", ylab=expression(paste("Discharge [m"^3,"/s]")))
```

`simulations` *Simulated runoff*

Description

The dataset is generated with the package own routines and represent 5 series of 18 years of runoff

Usage

```
data("simulations")
```

Format

A list of three elements, containing (i) a data frame with 6570 observations of the following variables

`YYYY` a numeric vector, year

`MM` a numeric vector, month

`DD` a numeric vector, day

`timestamp` `POSIXct` vector of the daily runoff

`deseasonalized` deseasonalized time series

`Qobs` observed runoff

`r1,...,r5` 5 simulated runoff series

(ii) a data frame with the daily fitted kappa parameters and (iii) p-values of the daily `ks.test`.

Details

The data is included to illustrate the validation and visualization routines in `demo("PRSim-validate")`.

Source

The data has been generated with

```
set.seed(14); prsim( runoff[ runoff$YYYY>1999,], number_sim=5, KStest=TRUE)
```

(default values for all other arguments).

References

Brunner, M. I., A. Bárdossy, and R. Furrer (2019). Technical note: Stochastic simulation of streamflow time series using phase randomization. Hydrology and Earth System Sciences, 23, 3175-3187, https://doi.org/10.5194/hess-23-3175-2019.

Examples

```
data(simulations)
names(simulations)
sim <- simulations$simulation
dim(sim)
sim$day_id <- rep(seq(1:365), times=length(unique(sim$YYYY)))
mean_obs <- aggregate(sim$Qobs, by=list(sim$day_id), FUN=mean, simplify=FALSE)
plot(unlist(mean_obs[,2]),lty=1,lwd=1,col="black", ylab="Discharge [m3/s]",
     xlab="Time [d]", main="Mean hydrographs", ylim=c(0,22), type="l")

for(r in 7:(length(names(sim))-1)){
  mean_hydrograph <- aggregate(sim[,r], by=list(sim$day_id), FUN=mean, simplify=FALSE)
  lines(mean_hydrograph, lty=1, lwd=1, col="gray")
}
lines( mean_obs, lty=1, lwd=1, col="black")
```

---

simulations_multi_sites

Simulated runoff for four catchments

Description

The dataset is generated with the package own routines and represent 5 series of 38 years of runoff for four catchments

Usage

```
data("simulations_multi_sites")
```

Format

A list of four elements (one per catchment), containing a data frame each holding information about the observed time series and the stochastic simulations

`YYYY` a numeric vector, year

`MM` a numeric vector, month

`DD` a numeric vector, day

`timestamp` `POSIXct` vector of the daily runoff

`Qobs` observed runoff

`r1,...,r5` 5 simulated runoff series

Details

The data is included to illustrate the validation and visualization routines in `demo("PRSim_wave-validate")`.

Source

The data has been generated with

`prsim.wave(data=runoff_multi_sites, number_sim=5, marginal="kappa", GoFtest = NULL,pars=NULL, p_val=NULL)`

(default values for all other arguments).

References

Brunner, M. I., A. Bárdossy, and R. Furrer (2019). Technical note: Stochastic simulation of streamflow time series using phase randomization. Hydrology and Earth System Sciences, 23, 3175-3187, https://doi.org/10.5194/hess-23-3175-2019.

Examples

```
oldpar <- par(mfrow = c(2, 1), mar = c(3, 3, 2, 1))
### greys
col_vect_obs <- c('#cccccc','#969696','#636363','#252525')
### oranges
col_vect_sim <- c('#fdbe85','#fd8d3c','#e6550d','#a63603')
data(simulations_multi_sites)
sim <- simulations_multi_sites
dim(sim[[1]])
### plot time series for multiple sites
par(mfrow=c(2,1),mar=c(3,3,2,1))
### determine ylim
ylim_max <- max(sim[[1]]$Qobs)*1.5
### observed
plot(sim[[1]]$Qobs[1:1000],
    ylab=expression(bold(
        paste("Specific discharge [mm/d]"))),
    xlab="Time [d]",type="l",col=col_vect_obs[1],
    ylim=c(0,ylim_max),main='Observations')
for(l in 2:4){
  lines(sim[[l]]$Qobs[1:1000],col=col_vect_obs[l])
}
legend('topleft',legend=c('Station 1','Station 2',
        'Station 3','Station 4'),
    lty=1,col=col_vect_obs[1:4])
### simulated (one run)
plot(sim[[1]]$r1[1:1000],
    ylab=expression(bold(paste("Specific discharge [mm/d]"))),
    xlab="Time [d]",type="l",col=col_vect_sim[1],
    ylim=c(0,ylim_max),
    main='Stochastic simulations')
for(l in 2:4){
  lines(sim[[l]]$r1[1:1000],col=col_vect_sim[l])
}
par(oldpar)
```

`weather_multi_sites` *Sample temperature and precipitation of four catchments derived from the ERA5-Land gridded dataset*

Description

Reanalysis data of four grid cells from ERA5-Land.

Usage

```
data("weather_multi_sites")
```

Format

Contains two lists data_p and data_t containing precipitation and temperature data, respectively. Each list consists of four data frames (one list per station/grid cell) of the following 4 variables.

`YYYY` a numeric vector, year

`MM` a numeric vector, month

`DD` a numeric vector, day

`Precip/Temp` a numeric vector, observed precipitation/temperature

Details

The data contains data for four grid cells in the Pacific Northwest.

Source

The preciptation data were downloaded from ERA5-Land https://cds.climate.copernicus.eu/cdsapp#!/dataset/reanalysis-era5-land?tab=overview.

References

Brunner, M. I., and E. Gilleland (2021). Spatial compound hot-dry events in the United States: assessment using a multi-site multi-variable weather generator, in preparation.

Examples

```
data(weather_multi_sites)
weather_multi_sites[[1]][[1]]$timestamp <- paste(weather_multi_sites[[1]][[1]]$YYYY,
weather_multi_sites[[1]][[1]]$MM, weather_multi_sites[[1]][[1]]$DD, sep=" ")
weather_multi_sites[[1]][[1]]$timestamp <-
as.POSIXct(strptime(weather_multi_sites[[1]][[1]]$timestamp,
format="%Y %m %d", tz="GMT"))
plot(weather_multi_sites[[1]][[1]]$timestamp[1:1000],
weather_multi_sites[[1]][[1]]$Qobs[1:1000], type="l",
xlab="Time [d]", ylab=expression(paste("Temperature [degrees]")))
```

`weather_sim_multi_sites` *Simulated temperature and precipitation for two grid cells*

Description

The dataset is generated with the package own routines and represent 5 series of 38 years of meteorological data for two grid cells

Usage

```
data("weather_sim_multi_sites")
```

Format

Two lists (one per variable) of four elements (one per catchment), containing a data frame each holding information about the observed time series and the stochastic simulations

`YYYY` a numeric vector, year

`MM` a numeric vector, month

`DD` a numeric vector, day

`timestamp` `POSIXct` vector of the daily runoff

`Prec/Temp` observed precipitation/temperature

`r1,...,r5` 5 simulated data series

Details

The data is included to illustrate the validation and visualization routines in `demo("PRSim_weather-validate")`.

Source

The data has been generated with

```
prsim.weather(data_p=data_p, data_t=data_t, number_sim=5, p_margin='egpd',t_margin='sep')
```

(default values for all other arguments).

References

Brunner, M. I., and E. Gilleland (2021). Spatial compound hot-dry events in the United States: assessment using a multi-site multi-variable weather generator, in preparation.

Examples

```
oldpar <- par(mfrow = c(2, 1), mar = c(3, 3, 2, 1))
data(weather_sim_multi_sites)
sim <- weather_sim_multi_sites
### define plotting colors
col_sim <- adjustcolor("#fd8d3c",alpha=0.8)
col_sim_tran <- adjustcolor("#fd8d3c",alpha=0.2)
col_obs <- adjustcolor( "black", alpha.f = 0.2)
### greys
col_vect_obs <- c('#cccccc','#969696','#636363','#252525')
### oranges
col_vect_sim <- c('#fdbe85','#fd8d3c','#e6550d','#a63603')

### plot time series for multiple sites

### Temperature (first list entry)
par(mfrow=c(2,1),mar=c(3,3,2,1))
### determine ylim
ylim_max <- max(sim[[1]][[1]]$Temp)*1.5
### observed
plot(sim[[1]][[1]]$Temp[1:1000],
ylab=expression(bold(paste("Temperature [degrees]"))),
xlab="Time [d]",type="l",col=col_vect_obs[1],
ylim=c(0,ylim_max),main='Observations')
for(l in 2){
  lines(sim[[l]][[1]]$Temp[1:1000],col=col_vect_obs[l])
}
# legend('topleft',legend=c('Station 1','Station 2'
# ),lty=1,col=col_vect_obs[1:2])
### simulated (one run)
plot(sim[[1]][[1]]$r1[1:1000],
ylab=expression(bold(paste("Temperature [degrees]"))),
xlab="Time [d]",type="l",col=col_vect_sim[1],
ylim=c(0,ylim_max),main='Stochastic simulations')
for(l in 2){
  lines(sim[[l]][[1]]$r1[1:1000],col=col_vect_sim[l])
}


### precipitation (second list entry)
ylim_max <- max(sim[[1]][[2]]$Prec)*1
### observed
plot(sim[[1]][[2]]$Prec[1:1000],
ylab=expression(bold(paste("Precipitation [mm/d]"))),
xlab="Time [d]",type="l",col=col_vect_obs[1],
ylim=c(0,ylim_max),main='Observations')
for(l in 2){
  lines(sim[[l]][[2]]$Prec[1:1000],col=col_vect_obs[l])
}
# legend('topleft',legend=c('Station 1','Station 2'
# ),lty=1,col=col_vect_obs[1:2])
### simulated (one run)
```

```
plot(sim[[1]][[2]]$r1[1:1000],
ylab=expression(bold(paste("Precipitation [mm/d]"))),
xlab="Time [d]",type="l",col=col_vect_sim[1],
ylim=c(0,ylim_max),main='Stochastic simulations')
for(l in 2){
  lines(sim[[l]][[2]]$r1[1:1000],col=col_vect_sim[l])
}
par(oldpar)
```

Index