Package: hyfo (via r-universe)

November 17, 2026

Type Package

Title Hydrology and Climate Forecasting

Version 1.4.6

Date 2023-8-15

Description Focuses on data processing and visualization in hydrology and climate forecasting. Main function includes data extraction, data downscaling, data resampling, gap filler of precipitation, bias correction of forecasting data, flexible time series plot, and spatial map generation. It is a good pre-processing and post-processing tool for hydrological and hydraulic modellers.

License GPL-2

Depends R (>= 3.1.0), stats (>= 3.1.3), utils(>= 3.1.3),

Imports ggplot2 (>= 1.0.1), reshape2 (>= 1.4.1), zoo (>= 1.7-12), sf (>= 1.0-12), plyr (>= 1.8.3), moments (>= 0.14), lmom (>= 2.5), maps(>= 2.3-9), sp (>= 2.0-0), ncdf4 (>= 1.14.1), MASS (>= 7.3-39), methods, data.table

Suggests gridExtra, knitr, rmarkdown

VignetteBuilder knitr

LazyData true

URL https://bnycx.github.io/hyfo/

BugReports https://github.com/bnycx/hyfo/issues

RoxygenNote 7.2.3

Config/pak/sysreqs libabsl-dev cmake libgdal-dev gdal-bin libgeos-dev libicu-dev libnetcdf-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev

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

Date/Publication 2026-11-16 16:41:34 UTC

RemoteUrl https://github.com/bnycx/hyfo

RemoteRef HEAD

RemoteSha 77e1599f9373d8b1b13ff4f40cde936d50ca8bd6

Contents

applyBiasFactor . . . 3
biasCorrect . . . 6
biasFactor-class . . . 11
biasFactor.hyfo-class . . . 12
checkBind . . . 12
collectData . . . 13
collectData_csv_anarbe . . . 13
collectData_excel_anarbe . . . 14
collectData_txt_anarbe . . . 15
coord2cell . . . 16
downscaleNcdf . . . 16
extractPeriod . . . 17
fillGap . . . 20
getAnnual . . . 22
getAnnual_dataframe . . . 23
getBiasFactor . . . 24
getEnsem_comb . . . 28
getFrcEnsem . . . 29
getHisEnsem . . . 31
getLMom . . . 34
getMeanPreci . . . 35
getMoment . . . 36
getNcdfVar . . . 36
getPreciBar . . . 37
getPreciBar_comb . . . 40
getSpatialMap . . . 41
getSpatialMap_comb . . . 42
getSpatialMap_mat . . . 43
list2Dataframe . . . 45
loadNcdf . . . 46
monthlyPreci . . . 47
plotTS . . . 48
plotTS_comb . . . 49
resample . . . 51
shp2cat . . . 52
testCat . . . 53
testdl . . . 53
tgridData . . . 54
writeNcdf . . . 55

Index **57**

`applyBiasFactor` *Apply bias factor to different forecasts for multi/operational/real time bias correction.*

Description

When you do multi/operational/real time bias correction. It's too expensive to input hindcast and obs every time. Especially when you have a long period of hindcast and obs, but only a short period of frc, it's too unecessary to read and compute hindcast and obs everytime. Therefore, biasFactor is designed. Using `getBiasFactor`, you can get the biasFactor with hindcast and observation, then you can use `applyBiasFactor` to apply the biasFactor to different forecasts.

Usage

```
applyBiasFactor(frc, biasFactor, obs = NULL)

## S4 method for signature 'data.frame,biasFactor'
applyBiasFactor(frc, biasFactor, obs = NULL)

## S4 method for signature 'list,biasFactor.hyfo'
applyBiasFactor(frc, biasFactor, obs = NULL)
```

Arguments

`frc`	a hyfo grid data output or a dataframe(time series) consists of Date column and one or more value columns, representing the frc data. Check details for more information.
`biasFactor`	a file containing all the information of the calibration, will be applied to different forecasts.
`obs`	for some methods, observation input is necessary. obs is a hyfo grid data output or a dataframe (time series) consists of Date column and one or more value columns, representing the observation data. Default value is NULL.

Details

Information about the method and how biasCorrect works can be found in `biasCorrect`

why use biasFactor

As for forecasting, for daily data, there is usually no need to have different bias factor every different day. You can calculate one bisa factor using a long period of hindcast and obs, and apply that factor to different frc.

For example,

You have 10 years of hindcast and observation. you want to do bias correction for some forecasting product, e.g. system 4. For system 4, each month, you will get a new forecast about the future 6 months. So if you want to do the real time bias correction, you have to take the 10 years of hindcast and observation data with you, and run `biasCorrect` every time you get a new forecast. That's too expensive.

For some practical use in forecasting, there isn't a so high demand for accuracy. E.g., Maybe for February and March, you can use the same biasFactor, no need to do the computation again.

It is a generic function, if in your case you need to debug, please see `?debug()` for how to debug S4 method.

Author(s)

Yuanchao Xu <xuyuanchao37@gmail.com >

References

Bias correction methods come from `biasCorrection` from `dowscaleR`

- Santander Meteorology Group (2015). downscaleR: Climate data manipulation and statistical downscaling. R package version 0.6-0. https://github.com/SantanderMetGroup/downscaleR/wiki
- R.A.I. Wilcke, T. Mendlik and A. Gobiet (2013) Multi-variable error correction of regional climate models. Climatic Change, 120, 871-887
- A. Amengual, V. Homar, R. Romero, S. Alonso, and C. Ramis (2012) A Statistical Adjustment of Regional Climate Model Outputs to Local Scales: Application to Platja de Palma, Spain. J. Clim., 25, 939-957
- C. Piani, J. O. Haerter and E. Coppola (2009) Statistical bias correction for daily precipitation in regional climate models over Europe, Theoretical and Applied Climatology, 99, 187-192
- O. Gutjahr and G. Heinemann (2013) Comparing precipitation bias correction methods for high-resolution regional climate simulations using COSMO-CLM, Theoretical and Applied Climatology, 114, 511-529

See Also

`biasCorrect` for method used in bias correction. `getBiasFactor`, for the first part.

Examples

```
######## hyfo grid file biascorrection
########

# If your input is obtained by \code{loadNcdf}, you can also directly biascorrect
# the file.

# First load ncdf file.
filePath <- system.file("extdata", "tnc.nc", package = "hyfo")
varname <- getNcdfVar(filePath)
nc <- loadNcdf(filePath, varname)

data(tgridData)
#' # Since the example data, has some NA values, the process will include some warning #message,
# which can be ignored in this case.
```

```
# Then we will use nc data as forecasting data, and use itself as hindcast data,
# use tgridData as observation.

biasFactor <- getBiasFactor(nc, tgridData)
newFrc <- applyBiasFactor(nc, biasFactor)

biasFactor <- getBiasFactor(nc, tgridData, method = 'eqm', extrapolate = 'constant',
preci = TRUE)
# This method needs obs input.
newFrc <- applyBiasFactor(nc, biasFactor, obs = tgridData)

biasFactor <- getBiasFactor(nc, tgridData, method = 'gqm', preci = TRUE)
newFrc <- applyBiasFactor(nc, biasFactor)


######## Time series biascorrection
########

# Use the time series from testdl as an example, we take frc, hindcast and obs from testdl.
data(testdl)

# common period has to be extracted in order to better train the forecast.

datalist <- extractPeriod(testdl, startDate = '1994-1-1', endDate = '1995-10-1')

frc <- datalist[[1]]
hindcast <- datalist[[2]]
obs <- datalist[[3]]


# The data used here is just for example, so there could be negative data.

# default method is scaling
biasFactor <- getBiasFactor(hindcast, obs)
frc_new <- applyBiasFactor(frc, biasFactor)

# for precipitation data, extra process needs to be executed, so you have to tell
# the program to it is a precipitation data.

biasFactor <- getBiasFactor(hindcast, obs, preci = TRUE)
frc_new1 <- applyBiasFactor(frc, biasFactor)

# You can use other methods to biascorrect, e.g. delta method.
biasFactor <- getBiasFactor(hindcast, obs, method = 'delta')
# delta method needs obs input.
frc_new2 <- applyBiasFactor(frc, biasFactor, obs = obs)

#
biasFactor <- getBiasFactor(hindcast, obs, method = 'eqm', preci = TRUE)
# eqm needs obs input
frc_new3 <- applyBiasFactor(frc, biasFactor, obs = obs)

biasFactor <- getBiasFactor(hindcast, obs, method = 'gqm', preci = TRUE)
```

```
frc_new4 <- applyBiasFactor(frc, biasFactor)

plotTS(obs, frc, frc_new, frc_new1, frc_new2, frc_new3, frc_new4, plot = 'cum')

# You can also give name to this input list.
TSlist <- list(obs, frc, frc_new, frc_new1, frc_new2, frc_new3, frc_new4)
names(TSlist) <- c('obs', 'frc', 'delta', 'delta_preci', 'scale', 'eqm', 'gqm')
plotTS(list = TSlist, plot = 'cum')


# If the forecasts you extracted only has incontinuous data for certain months and years, e.g.,
# for seasonal forecasting, forecasts only provide 3-6 months data, so the case can be
# for example Dec, Jan and Feb of every year from year 1999-2005.
# In such case, you need to extract certain months and years from observed time series.
# extractPeriod() can be then used.




# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

`biasCorrect` *Biascorrect the input timeseries or hyfo dataset*

Description

Biascorrect the input time series or dataset, the input time series or dataset should consist of observation, hindcast, and forecast. observation and hindcast should belong to the same period, in order to calibrate. Then the modified forecast will be returned. If the input is a time series, first column should be date column and rest columns should be the value column. If the input is a hyfo dataset, the dataset should be the result of `loadNcdf`, or a list file with the same format.

Usage

```
biasCorrect(
  frc,
  hindcast,
  obs,
  method = "scaling",
  scaleType = "multi",
  preci = FALSE,
  prThreshold = 0,
  extrapolate = "no"
)

## S4 method for signature 'data.frame,data.frame,data.frame'
```

```
biasCorrect(
  frc,
  hindcast,
  obs,
  method = "scaling",
  scaleType = "multi",
  preci = FALSE,
  prThreshold = 0,
  extrapolate = "no"
)

## S4 method for signature 'list,list,list'
biasCorrect(
  frc,
  hindcast,
  obs,
  method = "scaling",
  scaleType = "multi",
  preci = FALSE,
  prThreshold = 0,
  extrapolate = "no"
)
```

Arguments

`frc`	a hyfo grid data output or a dataframe (time series) consists of Date column and one or more value columns, representing the forecast to be calibrated.
`hindcast`	a hyfo grid data output or a dataframe(time series) consists of Date column and one or more value columns, representing the hindcast data. This data will be used in the calibration of the forecast, so it's better to have the same date period as observation data. Check details for more information.
`obs`	a hyfo grid data output or a dataframe (time series) consists of Date column and one or more value columns, representing the observation data.
`method`	bias correct method, including 'delta', 'scaling'..., default is 'scaling'
`scaleType`	only when the method "scaling" is chosen, scaleType will be available. Two different types of scaling method, 'add' and 'multi', which means additive and multiplicative scaling method. More info check details. Default scaleType is 'multi'.
`preci`	If the precipitation is biascorrected, then you have to assign `preci = TRUE`. Since for precipitation, some biascorrect methods may not apply to, or some methods are specially for precipitation. Default is FALSE, refer to details.
`prThreshold`	The minimum value that is considered as a non-zero precipitation. Default to 0 (assuming mm). If you want to use precipitation biascorrect, you should consider carefully how to set this threshold, usually is 1. But you can try with different numbers to see the results.
`extrapolate`	When use 'eqm' method, and 'no' is set, modified frc is bounded by the range of obs. If 'constant' is set, modified frc is not bounded by the range of obs. Default

is 'no'.

Details

Since climate forecast is based on global condition, when downscaling to different regions, it may include some bias, biascorrection is used then to fix the bias.

Hindcast

In order to bias correct, we need to pick up some data from the forecast to train with the observation, which is called hindcast in this function. Using hindcast and observation, the program can analyze the bias and correct the bias in the forecast.

Hindcast should have **EVERY** attributes that forecast has.

Hindcast is also called re-forecast, is the forecast of the past. E.g. you have a forecast from year 2000-2010, assuming now you are in 2005. So from 2000-2005, this period is the hindcast period, and 2005-2010, this period is the forecast period.

Hindcast can be the same as forecast, i.e., you can use forecast itself as hindcast to train the bias correction.

How it works

Forecast product has to be calibrated, usually the system is doing forecast in real time. So, e.g., if the forecast starts from year 2000, assuming you are in year 2003, then you will have 3 years' hindcast data (year 2000-2003), which can be used to calibrate. And your forecast period is (2003-2004)

E.g. you have observation from 2001-2002, this is your input obs. Then you can take the same period (2001-2002) from the forecast, which is the hindcast period. For forecast, you can take any period. The program will evaluate the obs and hindcast, to get the modification of the forecast, and then add the modification to the forecast data.

The more categorized input, the more accurate result you will get. E.g., if you want to bias correct a forecast for winter season. So you'd better to extract all the winter period in the hindcast and observation to train. `extractPeriod` can be used for this purpose.

method

Different methods used in the bias correction. Among which, delta, scaling can be applied to different kinds of parameters, with no need to set `preci`; eqm has two conditions for rainfall data and other data, it needs user to input `preci = TRUE/FALSE` to point to different conditions; gqm is designed for rainfall data, so `preci = TRUE` needs to be set.

delta

This method consists on adding to the observations the mean change signal (delta method). This method is applicable to any kind of variable but it is preferable to avoid it for bounded variables (e.g. precipitation, wind speed, etc.) because values out of the variable range could be obtained (e.g. negative wind speeds...)

scaling

This method consists on scaling the simulation with the difference (additive) or quotient (multiplicative) between the observed and simulated means in the train period. The `additive` or `multiplicative` correction is defined by parameter `scaling.type` (default is `additive`). The additive version is preferably applicable to unbounded variables (e.g. temperature) and the multiplicative to variables with a lower bound (e.g. precipitation, because it also preserves the frequency).

eqm

Empirical Quantile Mapping. This is a very extended bias correction method which consists on calibrating the simulated Cumulative Distribution Function (CDF) by adding to the observed quantiles both the mean delta change and the individual delta changes in the corresponding quantiles. This method is applicable to any kind of variable.

It can keep the extreme value, if you choose constant extrapolation method. But then you will face the risk that the extreme value is an error.

gqm

Gamma Quantile Mapping. This method is described in Piani et al. 2010 and is applicable only to precipitation. It is based on the initial assumption that both observed and simulated intensity distributions are well approximated by the gamma distribution, therefore is a parametric q-q map that uses the theorical instead of the empirical distribution.

It can somehow filter some extreme values caused by errors, while keep the extreme value. Seems more reasonable. Better have a long period of training, and the if the forecast system is relatively stable.

It is a generic function, if in your case you need to debug, please see `?debug()` for how to debug S4 method.

Author(s)

Yuanchao Xu <xuyuanchao37@gmail.com >

References

Bias correction methods come from `biasCorrection` from `dowscaleR`

- Santander Meteorology Group (2015). downscaleR: Climate data manipulation and statistical downscaling. R package version 0.6-0. https://github.com/SantanderMetGroup/downscaleR/wiki
- R.A.I. Wilcke, T. Mendlik and A. Gobiet (2013) Multi-variable error correction of regional climate models. Climatic Change, 120, 871-887
- A. Amengual, V. Homar, R. Romero, S. Alonso, and C. Ramis (2012) A Statistical Adjustment of Regional Climate Model Outputs to Local Scales: Application to Platja de Palma, Spain. J. Clim., 25, 939-957
- C. Piani, J. O. Haerter and E. Coppola (2009) Statistical bias correction for daily precipitation in regional climate models over Europe, Theoretical and Applied Climatology, 99, 187-192
- O. Gutjahr and G. Heinemann (2013) Comparing precipitation bias correction methods for high-resolution regional climate simulations using COSMO-CLM, Theoretical and Applied Climatology, 114, 511-529

Examples

```
######## hyfo grid file biascorrection
########

# If your input is obtained by \code{loadNcdf}, you can also directly biascorrect
# the file.

# First load ncdf file.
```

```
filePath <- system.file("extdata", "tnc.nc", package = "hyfo")
varname <- getNcdfVar(filePath)
nc <- loadNcdf(filePath, varname)

data(tgridData)
# Since the example data, has some NA values, the process will include some warning #message,
# which can be ignored in this case.



# Then we will use nc data as forecasting data, and use itself as hindcast data,
# use tgridData as observation.
newFrc <- biasCorrect(nc, nc, tgridData)
newFrc <- biasCorrect(nc, nc, tgridData, scaleType = 'add')
newFrc <- biasCorrect(nc, nc, tgridData, method = 'eqm', extrapolate = 'constant',
preci = TRUE)
newFrc <- biasCorrect(nc, nc, tgridData, method = 'gqm', preci = TRUE)


######## Time series biascorrection
########

# Use the time series from testdl as an example, we take frc, hindcast and obs from testdl.
data(testdl)

# common period has to be extracted in order to better train the forecast.

datalist <- extractPeriod(testdl, startDate = '1994-1-1', endDate = '1995-10-1')

frc <- datalist[[1]]
hindcast <- datalist[[2]]
obs <- datalist[[3]]


# The data used here is just for example, so there could be negative data.

# default method is scaling, with 'multi' scaleType
frc_new <- biasCorrect(frc, hindcast, obs)

# for precipitation data, extra process needs to be executed, so you have to tell
# the program that it is a precipitation data.

frc_new1 <- biasCorrect(frc, hindcast, obs, preci = TRUE)

# You can use other scaling methods to biascorrect.
frc_new2 <- biasCorrect(frc, hindcast, obs, scaleType = 'add')

#
frc_new3 <- biasCorrect(frc, hindcast, obs, method = 'eqm', preci = TRUE)
frc_new4 <- biasCorrect(frc, hindcast, obs, method = 'gqm', preci = TRUE)

plotTS(obs, frc, frc_new, frc_new1, frc_new2, frc_new3, frc_new4, plot = 'cum')
```

```
# You can also give name to this input list.
TSlist <- list(obs, frc, frc_new, frc_new1, frc_new2, frc_new3, frc_new4)
names(TSlist) <- c('obs', 'frc', 'delta', 'delta_preci', 'scale', 'eqm', 'gqm')
plotTS(list = TSlist, plot = 'cum')


# If the forecasts you extracted only has incontinuous data for certain months and years, e.g.,
# for seasonal forecasting, forecasts only provide 3-6 months data, so the case can be
# for example Dec, Jan and Feb of every year from year 1999-2005.
# In such case, you need to extract certain months and years from observed time series.
# extractPeriod() can be then used.




# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

`biasFactor-class` *An S4 class, representing the biasFactor of single time series biasCorrection.*

Description

An S4 class, representing the biasFactor of single time series biasCorrection.

Slots

`biasFactor` list of biasFactor, containing all the information for computing.

`method` the biascorrection method

`preci` if the data is precipitation

`scaleType` 'Valid when 'scaling' method is selected, 'multi' or 'add'.

`extrapolate` Valid when 'eqm' method is selected, 'constant' or 'no'

`memberDim` members contained.

`prThreshold` precipitation threshold, under which the precipitation is considered as 0.

biasFactor.hyfo-class *An S4 class, representing the biasFactor of hyfo file.*

Description

An S4 class, representing the biasFactor of hyfo file.

Slots

`lonLatDim` lists of biasFactor

checkBind *Check data for bind function.*

Description

check if the data is available for `rbind()` or `cbind()`

Usage

```
checkBind(data, bind)
```

Arguments

`data`	A list containing different sublists ready to be processed by `do.call('rbind')` or `do.call('cbind')`
`bind`	A string showing which bind you are going to use can be 'rbind' or 'cbind'

Value

data can be processed by bind function; data cannot be processed by bind function

Examples

```
data <- list(c(1,1,1),c(2,2,2))
bind <- 'rbind'
checkBind(data,bind)

data(testdl)
## Not run:
checkBind(testdl, 'rbind')

## End(Not run)
# Since the colnames in testdl are not the same, so it cannot be bound.
#
```

collectData *Collect data from different csv files.*

Description

Collect data from different csv files.

Usage

```
collectData(folderName, fileType = NULL, range = NULL, sheetIndex = 1)
```

Arguments

folderName	A string showing the path of the folder holding different csv files.
fileType	A string showing the file type, e.g. "txt", "csv", "excel".
range	A vector containing startRow, endRow, startColumn, endColumn, e.g., c(2,15,2,3)
sheetIndex	A number showing the sheetIndex in the excel file, if fileType is excel, sheetIndex has to be provided, default is 1.

Value

The collected data from different files in the folder.

Examples

```
#use internal data as an example.
folder <- file.path(path.package("hyfo"), 'extdata')
# file may vary with different environment, it if doesn't work, use local way to get
# folder path.

a <- collectData(folder, fileType = 'csv', range = c(10, 20, 1,2))

# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

collectData_csv_anarbe *Collect data from csv for Anarbe case.*

Description

Collect data from the gauging stations in spain, catchement Anarbe

Usage

```
collectData_csv_anarbe(folderName, output = TRUE)
```

Arguments

folderName	A string showing the path of the folder holding different csv files.
output	A boolean showing whether the output is given, default is T.

Value

The collected data from different csv files.

Source

http://meteo.navarra.es/estaciones/mapadeestaciones.cfm

References

- http://meteo.navarra.es/estaciones/mapadeestaciones.cfm
- R Core Team (2015). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Examples

```
#use internal data as an example.
file <- system.file("extdata", "1999.csv", package = "hyfo")
folder <- strsplit(file, '1999')[[1]][1]
a <- collectData_csv_anarbe(folder)

# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

---

collectData_excel_anarbe

Collect data from different excel files

Description

Collect data from different excel files

Usage

```
collectData_excel_anarbe(folderName, keyword = NULL, output = TRUE)
```

Arguments

folderName	A string showing the folder path.
keyword	A string showing the extracted column, e.g., waterLevel, waterBalance.
output	A boolean showing whether the output is given.

Value

The collected data from different excel files.

References

- R Core Team (2015). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/.

`collectData_txt_anarbe`
collect data from different txt.

Description

collect data from different txt.

Usage

```
collectData_txt_anarbe(
  folderName,
  output = TRUE,
  rangeWord = c("Ene       ", -1, "Total     ", -6)
)
```

Arguments

`folderName`	A string showing the folder path.
`output`	A boolean showing whether the result is given.
`rangeWord`	A list containing the keyword and the shift. defaut is set to be used in spain gauging station.

Value

The collected data from different txt files.

Source

http://www4.gipuzkoa.net/oohh/web/esp/02.asp

References

- http://www4.gipuzkoa.net/oohh/web/esp/02.asp
- R Core Team (2015). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Examples

```
#use internal data as an example.

## Not run:
file <- system.file("extdata", "1999.csv", package = "hyfo")
folder <- strsplit(file, '1999')[[1]][1]
a <- collectData_txt_anarbe(folder)

## End(Not run)


# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

`coord2cell` *Change lon lat coordinates to cell coordinates*

Description

Change lon lat coordinates to cell coordinates

Usage

```
coord2cell(coord, lon, lat)
```

Arguments

`coord`	input lon lat coordinate
`lon`	dataset lon array
`lat`	dataset lat array

Value

A cell coordinate

`downscaleNcdf` *Downscale NetCDF file*

Description

Downscale NetCDF file

Usage

```
downscaleNcdf(gridData, year = NULL, month = NULL, lon = NULL, lat = NULL)
```

Arguments

gridData	A hyfo list file from loadNcdf
year	A vector of the target year. e.g. year = 2000, year = 1980:2000
month	A vector of the target month. e.g. month = 2, month = 3:12
lon	A vector of the range of the downscaled longitude, should contain a max value and a min value. e.g. lon = c(-1.5, 2,5)
lat	A vector of the range of the downscaled latitude, should contain a max value and a min value. e.g. lat = c(32,2, 36)

Value

A downscaled hyfo list file.

References

- Santander MetGroup (2015). ecomsUDG.Raccess: R interface to the ECOMS User Data Gateway. R package version 2.2-6. http://meteo.unican.es/ecoms-udg

Examples

```
# First open the test NETcDF file.
filePath <- system.file("extdata", "tnc.nc", package = "hyfo")


# Then if you don't know the variable name, you can use \code{getNcdfVar} to get variable name
varname <- getNcdfVar(filePath)

nc <- loadNcdf(filePath, varname)

# Then write to your work directory

nc1 <- downscaleNcdf(nc, year = 2006, lon = c(-2, -0.5), lat = c(43.2, 43.7))
nc2 <- downscaleNcdf(nc, year = 2005, month = 3:8, lon = c(-2, -0.5), lat = c(43.2, 43.7))

# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

---

extractPeriod *Extract period from list or dataframe.*

Description

Extract common period or certain period from a list of different dataframes of time series, or from a dataframe. NOTE: all the dates in the datalist should follow the format in ?as.Datebase.

Usage

```
extractPeriod(
  data,
  startDate = NULL,
  endDate = NULL,
  commonPeriod = FALSE,
  year = NULL,
  month = NULL
)

## S4 method for signature 'data.frame'
extractPeriod(
  data,
  startDate = NULL,
  endDate = NULL,
  commonPeriod = FALSE,
  year = NULL,
  month = NULL
)

## S4 method for signature 'list'
extractPeriod(
  data,
  startDate = NULL,
  endDate = NULL,
  commonPeriod = FALSE,
  year = NULL,
  month = NULL
)
```

Arguments

`data`	A list of different dataframes of time series, or a dataframe with first column Date, the rest columns value.
`startDate`	A Date showing the start of the extract period, default as NULL, check details.
`endDate`	A Date showing the end of the extract period, default as NULL, check details.
`commonPeriod`	A boolean showing whether the common period is extracted. If chosen, startDate and endDate should be NULL.
`year`	extract certain year in the entire time series. if you want to extract year 2000, set `year = 2000`
`month`	extract certain months in a year. e.g. if you want to extract Jan, Feb of each year, set `month = c(1, 2)`.

Details

startDate and endDate

If startDate and endDate are assigned, then certain period between startDate and endDate will be returned, for both datalist input and dataframe input.

If startDate and endDate are NOT assigned, then,

if input is a datalist, the startDate and endDate of the common period of different datalists will be assigned to the startDate and endDate.

if input is a dataframe, the startDate and endDate of the input dataframe will be assigned to the startDate and endDate . Since different value columns share a common Date column in a dataframe input.

year and month

For year crossing month input, hyfo will take from the year before. E.g. if `month = c(10, 11, 12, 1)`, and `year = 1999`, hyfo will take month 10, 11 and 12 from year 1998, and month 1 from 1999.You DO NOT have to set `year = 1998 : 1999`.

Well, if you set `year = 1998 : 1999`, hyfo will take month 10, 11 and 12 from year 1997, and month 1 from 1998, then, take month 10, 11 and 12 from year 1998, month 1 from 1999. So you only have to care about the latter year.

It is a generic function, if in your case you need to debug, please see `?debug()` for how to debug S4 method.

Value

A list or a dataframe with all the time series inside containing the same period.

References

- Achim Zeileis and Gabor Grothendieck (2005). zoo: S3 Infrastructure for Regular and Irregular Time Series. Journal of Statistical Software, 14(6), 1-27. URL https://www.jstatsoft.org/v14/i06/

Examples

```
# Generate timeseries datalist. Each data frame consists of a Date and a value.

AAA <- data.frame(
# date column
Date = seq(as.Date('1990-10-28'),as.Date('1997-4-1'),1),
 # value column
AAA = sample(1:100,length(seq(as.Date('1990-10-28'),as.Date('1997-4-1'),1)), repl = TRUE))

BBB <- data.frame(
Date = seq(as.Date('1993-3-28'),as.Date('1999-1-1'),1),
BBB = sample(1:100,length(seq(as.Date('1993-3-28'),as.Date('1999-1-1'),1)), repl = TRUE))

CCC <- data.frame(
Date = seq(as.Date('1988-2-2'),as.Date('1996-1-1'),1),
CCC = sample(1:100,length(seq(as.Date('1988-2-2'),as.Date('1996-1-1'),1)), repl = TRUE))

list <- list(AAA, BBB, CCC)# dput() and dget() can be used to save and load list file.

list_com <- extractPeriod(list, commonPeriod = TRUE)
```

```
# list_com is the extracted datalist.
str(list_com)

# If startDate and endDate is provided, the record between them will be extracted.
# make sure startDate is later than any startDate in each dataframe and endDate is
# earlier than any endDate in each dataframe.

data(testdl)
datalist_com1 <- extractPeriod(testdl, startDate = '1994-1-1', endDate = '1995-10-1')


dataframe <- list2Dataframe(datalist_com1)
# now we have a dataframe to extract certain months and years.
dataframe_new <- extractPeriod(dataframe, month = c(1,2,3))
dataframe_new <- extractPeriod(dataframe, month = c(12,1,2), year = 1995)


# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

`fillGap` *Fill gaps in the rainfall time series.*

Description

Fill gaps in the rainfall time series.

Usage

```
fillGap(dataset, corPeriod = "daily")
```

Arguments

`dataset`	A dataframe with first column the time, the rest columns are rainfall data of different gauges
`corPeriod`	A string showing the period used in the correlation computing, e.g. daily, monthly, yearly.

Details

the gap filler follows the rules below:

1. The correlation coefficient of every two columns (except time column) is calculated. the correlation coefficient calculation can be based on 'daily', 'monthly', 'annual', in each case, the daily data, the monthly mean daily data and annual mean daily data of each column will be taken in the correlation calculation.

Then the correlation matrix is got, then based on the matrix, for each column, the 1st, 2nd, 3rd,... correlated column will be got. So if there is missing value in the column, it will get data from orderly 1st, 2nd, 3rd column.

2. The simple linear regress is calculated between every two columns. When generating the linear coefficient, the incept should be force to 0. i.e. y = a*x + b should be forec to y = a*x.

3. Gap filling. E.g., on a certain date, there is a missing value in column A, then the correlation order is column B, column C, column D, which means A should take values from B firstly, if B is also missing data, then C, then D.

Assuming finally value from column C is taken. Then according to step 2, A = a*C, then the final value filled in column A is missing_in_A = a*value_in_C, a is the linear coeffcient.

Value

The filled dataframe

References

Gap fiiling method based on correlation and linear regression.

- Hirsch, Robert M., et al. "Statistical analysis of hydrologic data." Handbook of hydrology. (1992): 17-1. Salas, Jose D. "Analysis and modeling of hydrologic time series." Handbook of hydrology 19 (1993): 1-72.

Examples

```
b <- read.table(text = '        Date  AAA  BBB  CCC  DDD  EEE
49 1999-12-15 24.8 21.4 25.6 35.0 17.4
50 1999-12-16   NA  0.6  1.5  6.3  2.5
51 1999-12-17   NA 16.3 20.3  NA 19.2
52 1999-12-18   13  1.6 NA  6.3  0.0
53 1999-12-19   10 36.4 12.5 26.8 24.9
54 1999-12-20   NA  0.0  0.0  0.2  0.0
55 1999-12-21  0.2  0.0  0.0  0.0  0.0
56 1999-12-22  0.0  0.0  0.0  0.0  0.0')

b1 <- fillGap(b) # if corPeriod is missing, 'daily' is taken as default.

data(testdl)
a <- extractPeriod(testdl, commonPeriod = TRUE)
a1 <- list2Dataframe(a)
a2 <- fillGap(a1)
a3 <- fillGap(a1, corPeriod = 'monthly')


# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

`getAnnual` *Get annual rainfall of different rainfall time series*

Description

Get annual rainfall of different raninfall time series.

Usage

```
getAnnual(data, output = "series", minRecords = 355, ...)

## S4 method for signature 'data.frame'
getAnnual(data, output = "series", minRecords = 355, ...)

## S4 method for signature 'list'
getAnnual(data, output = "series", minRecords = 355, ...)
```

Arguments

`data`	A list containing different time series of different rainfall gauges. Or a dataframe with first column Date and the rest columns the value of different gauging stations. Usually an output of `list2Dataframe`.
`output`	A string showing the output output.
`minRecords`	A number showing the minimum accept record number, e.g. for a normal year(365 days), if `minRecords = 360`, it means if a year has less than 360 records of a year, it will be ignored in the mean annual value calculation. Only valid when `output = "mean"`, default is 355.
`...`	`title, x, y` showing the title and x and y axis of the plot. e.g. `title = 'aaa'`

Details

It is a generic function, if in your case you need to debug, please see `?debug()` for how to debug S4 method.

Value

The annual rainfall and the number of missing data of each year and each rainfall gauge, which will also be plotted. If output "mean" is seleted, the mean annual rainfall will be returned.

References

- H. Wickham. ggplot2: elegant graphics for data analysis. Springer New York, 2009.
- Hadley Wickham (2007). Reshaping Data with the reshape Package. Journal of Statistical Software, 21(12), 1-20. URL http://www.jstatsoft.org/v21/i12/.
- R Core Team (2015). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Examples

```
#datalist is provided by the package as a test.
data(testdl)
a <- getAnnual(testdl)
#set minRecords to control the calculation of annual rainfall.
b <- getAnnual(testdl, output = 'mean', minRecords = 350)
c <- getAnnual(testdl, output = 'mean', minRecords = 365)

a1 <- extractPeriod(testdl, comm = TRUE)
a2 <- list2Dataframe(a1)
getAnnual(a2)

a3 <- fillGap(a2)
getAnnual(a3)


# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

`getAnnual_dataframe` *Get annual rainfall of the input time series.*

Description

Get annual rainfall of the input time series.

Usage

```
getAnnual_dataframe(dataset)
```

Arguments

`dataset`	A dataframe containing one time series, e.g., rainfall from one gauging station. the time should follow the format : "1990-1-1"

Value

The annual rainfall of each year of the input station.

getBiasFactor *Get bias factor for multi/operational/real time bias correction.*

Description

When you do multi/operational/real time bias correction. It's too expensive to input hindcast and obs every time. Especially when you have a long period of hindcast and obs, but only a short period of frc, it's too unecessary to read and compute hindcast and obs everytime. Therefore, biasFactor is designed. Using `getBiasFactor`, you can get the biasFactor with hindcast and observation, then you can use `applyBiasFactor` to apply the biasFactor to different forecasts.

Usage

```
getBiasFactor(
  hindcast,
  obs,
  method = "scaling",
  scaleType = "multi",
  preci = FALSE,
  prThreshold = 0,
  extrapolate = "no"
)

## S4 method for signature 'data.frame,data.frame'
getBiasFactor(
  hindcast,
  obs,
  method = "scaling",
  scaleType = "multi",
  preci = FALSE,
  prThreshold = 0,
  extrapolate = "no"
)

## S4 method for signature 'list,list'
getBiasFactor(
  hindcast,
  obs,
  method = "scaling",
  scaleType = "multi",
  preci = FALSE,
  prThreshold = 0,
  extrapolate = "no"
)
```

Arguments

hindcast	a hyfo grid data output or a dataframe(time series) consists of Date column and one or more value columns, representing the hindcast data. This data will be used in the calibration of the forecast, so it's better to have the same date period as observation data. Check details for more information.
obs	a hyfo grid data output or a dataframe (time series) consists of Date column and one or more value columns, representing the observation data.
method	bias correct method, including 'delta', 'scaling'...,default method is 'scaling'.
scaleType	only when the method "scaling" is chosen, scaleType will be available. Two different types of scaling method, 'add' and 'multi', which means additive and multiplicative scaling method, default is 'multi'. More info check details.
preci	If the precipitation is biascorrected, then you have to assign `preci = TRUE`. Since for precipitation, some biascorrect methods may not apply to, or some methods are specially for precipitation. Default is FALSE, refer to details.
prThreshold	The minimum value that is considered as a non-zero precipitation. Default to 1 (assuming mm).
extrapolate	When use 'eqm' method, and 'no' is set, modified frc is bounded by the range of obs. If 'constant' is set, modified frc is not bounded by the range of obs. Default is 'no'.

Details

Information about the method and how biasCorrect works can be found in `biasCorrect`

why use biasFactor

As for forecasting, for daily data, there is usually no need to have different bias factor every different day. You can calculate one bisa factor using a long period of hindcast and obs, and apply that factor to different frc.

For example,

You have 10 years of hindcast and observation. you want to do bias correction for some forecasting product, e.g. system 4. For system 4, each month, you will get a new forecast about the future 6 months. So if you want to do the real time bias correction, you have to take the 10 years of hindcast and observation data with you, and run `biasCorrect` every time you get a new forecast. That's too expensive.

For some practical use in forecasting, there isn't a so high demand for accuracy. E.g., Maybe for February and March, you can use the same biasFactor, no need to do the computation again.

It is a generic function, if in your case you need to debug, please see `?debug()` for how to debug S4 method.

Author(s)

Yuanchao Xu <xuyuanchao37@gmail.com >

References

Bias correction methods come from `biasCorrection` from `dowscaleR`

- Santander Meteorology Group (2015). downscaleR: Climate data manipulation and statistical downscaling. R package version 0.6-0. https://github.com/SantanderMetGroup/downscaleR/wiki
- R.A.I. Wilcke, T. Mendlik and A. Gobiet (2013) Multi-variable error correction of regional climate models. Climatic Change, 120, 871-887
- A. Amengual, V. Homar, R. Romero, S. Alonso, and C. Ramis (2012) A Statistical Adjustment of Regional Climate Model Outputs to Local Scales: Application to Platja de Palma, Spain. J. Clim., 25, 939-957
- C. Piani, J. O. Haerter and E. Coppola (2009) Statistical bias correction for daily precipitation in regional climate models over Europe, Theoretical and Applied Climatology, 99, 187-192
- O. Gutjahr and G. Heinemann (2013) Comparing precipitation bias correction methods for high-resolution regional climate simulations using COSMO-CLM, Theoretical and Applied Climatology, 114, 511-529

See Also

`biasCorrect` for method used in bias correction. `applyBiasFactor`, for the second part.

Examples

```
######## hyfo grid file biascorrection
########

# If your input is obtained by \code{loadNcdf}, you can also directly biascorrect
# the file.

# First load ncdf file.
filePath <- system.file("extdata", "tnc.nc", package = "hyfo")
varname <- getNcdfVar(filePath)
nc <- loadNcdf(filePath, varname)

data(tgridData)
# Since the example data, has some NA values, the process will include some warning #message,
# which can be ignored in this case.



# Then we will use nc data as forecasting data, and use itself as hindcast data,
# use tgridData as observation.

biasFactor <- getBiasFactor(nc, tgridData)
newFrc <- applyBiasFactor(nc, biasFactor)

biasFactor <- getBiasFactor(nc, tgridData, method = 'eqm', extrapolate = 'constant',
preci = TRUE)
# This method needs obs input.
newFrc <- applyBiasFactor(nc, biasFactor, obs = tgridData)
```

```
biasFactor <- getBiasFactor(nc, tgridData, method = 'gqm', preci = TRUE)
newFrc <- applyBiasFactor(nc, biasFactor)


######## Time series biascorrection
########

# Use the time series from testdl as an example, we take frc, hindcast and obs from testdl.
data(testdl)

# common period has to be extracted in order to better train the forecast.

datalist <- extractPeriod(testdl, startDate = '1994-1-1', endDate = '1995-10-1')

frc <- datalist[[1]]
hindcast <- datalist[[2]]
obs <- datalist[[3]]


# The data used here is just for example, so there could be negative data.

# default method is scaling
biasFactor <- getBiasFactor(hindcast, obs)
frc_new <- applyBiasFactor(frc, biasFactor)

# for precipitation data, extra process needs to be executed, so you have to tell
# the program to it is a precipitation data.

biasFactor <- getBiasFactor(hindcast, obs, preci = TRUE)
frc_new1 <- applyBiasFactor(frc, biasFactor)

# You can use other methods to biascorrect, e.g. delta method.
biasFactor <- getBiasFactor(hindcast, obs, method = 'delta')
# delta method needs obs input.
frc_new2 <- applyBiasFactor(frc, biasFactor, obs = obs)

#
biasFactor <- getBiasFactor(hindcast, obs, method = 'eqm', preci = TRUE)
# eqm needs obs input
frc_new3 <- applyBiasFactor(frc, biasFactor, obs = obs)

biasFactor <- getBiasFactor(hindcast, obs, method = 'gqm', preci = TRUE)
frc_new4 <- applyBiasFactor(frc, biasFactor)

plotTS(obs, frc, frc_new, frc_new1, frc_new2, frc_new3, frc_new4, plot = 'cum')

# You can also give name to this input list.
TSlist <- list(obs, frc, frc_new, frc_new1, frc_new2, frc_new3, frc_new4)
names(TSlist) <- c('obs', 'frc', 'delta', 'delta_preci', 'scale', 'eqm', 'gqm')
plotTS(list = TSlist, plot = 'cum')


# If the forecasts you extracted only has incontinuous data for certain months and years, e.g.,
```

```
# for seasonal forecasting, forecasts only provide 3-6 months data, so the case can be
# for example Dec, Jan and Feb of every year from year 1999-2005.
# In such case, you need to extract certain months and years from observed time series.
# extractPeriod() can be then used.




# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

`getEnsem_comb` *Combine ensembles together*

Description

Combine ensembles together

Usage

```
getEnsem_comb(
  ...,
  list = NULL,
  nrow = 1,
  legend = TRUE,
  x = "",
  y = "",
  title = "",
  output = FALSE
)
```

Arguments

`...`	different ensembles generated by `getHisEnsem(, output = 'ggplot')` or `getFrcEnsem(, output = 'ggplot')`, see details.
`list`	If input is a list containing different ggplot data, use `list = inputlist`.
`nrow`	A number showing the number of rows.
`legend`	A boolean representing whether you want the legend. Sometimes when you combine plots, there will be a lot of legends, if you don't like it, you can turn it off by setting `legend = FALSE`, default is TRUE.
`x`	A string of x axis name.
`y`	A string of y axis name.
`title`	A string of the title.
`output`	A boolean, if chosen TRUE, the output will be given.

Value

A combined ensemble plot.

References

- H. Wickham. ggplot2: elegant graphics for data analysis. Springer New York, 2009.
- Santander Meteorology Group (2015). downscaleR: Climate data manipulation and statistical downscaling. R package version 0.6-0. https://github.com/SantanderMetGroup/downscaleR/wiki

Examples

```
data(testdl)

a <- testdl[[1]]

# Choose example from "1994-2-4" to "1996-1-4"


b1<- getHisEnsem(a, example = c('1995-2-4', '1996-1-4'), plot = 'cum', output = 'ggplot',
                 name = 1)

b2 <- getHisEnsem(a, example = c('1995-4-4', '1996-3-4'), plot = 'cum', output = 'ggplot',
                 name = 2)

getEnsem_comb(b1, b2)
getEnsem_comb(list = list(b1, b2), nrow = 2)

# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

---

getFrcEnsem *Extract time series from forecasting data.*

Description

getFrcEnsem extract timeseries from forecasting data, if forecasting data has a member session an ensemble time sereis will be returned, if forecasting data doesn't have a member session, a singe time series will be returned.

Usage

```
getFrcEnsem(
  dataset,
  cell = "mean",
  plot = "norm",
  output = "data",
  name = NULL,
```

```
  mv = 0,
  coord = NULL,
  ...
)
```

Arguments

dataset	A list containing different information, should be the result of loadNcdf
cell	A vector containing the locaton of the cell, e.g. c(2, 3), default is "mean", representing the spatially averaged value. Check details for more information.
plot	A string showing whether the plot will be shown, e.g., 'norm' means normal plot (without any process), 'cum' means cummulative plot, default is 'norm'. For other words there will be no plot.
output	A string showing which type of output you want. Default is "data", if "ggplot", the data that can be directly plotted by ggplot2 will be returned, which is easier for you to make series plots afterwards. NOTE: If output = 'ggplot', the missing value in the data will be replaced by mv, if assigned, default mv is 0.
name	If output = 'ggplot', name has to be assigned to your output, in order to differentiate different outputs in the later multiplot using getEnsem_comb.
mv	A number showing representing the missing value. When calculating the cumulative value, missing value will be replaced by mv, default is 0.
coord	A coordinate of longitude and latitude. e.g. corrd = c(lon, lat). If coord is assigned, cell argument will no longer be used.
...	title, x, y showing the title and x and y axis of the plot. e.g. title = 'aaa'

Details

cell representing the location of the cell, NOTE: this location means the index of the cell, IT IS NOT THE LONGITUDE AND LATITUDE. e.g., cell = c(2, 3), the program will take the 2nd longitude and 3rd latitude, by the increasing order. Longitude comes first.

name Assuming you have two ggplot outputs, you want to plot them together. In this situation, you need a name column to differentiate one ggplot output from the other. You can assigne this name by the argument directly, If name is not assigned and output = 'ggplot' is selected, then the system time will be selected as name column.

Value

A ensemble time series extracted from forecating data.

References

- H. Wickham. ggplot2: elegant graphics for data analysis. Springer New York, 2009.
- Hadley Wickham (2007). Reshaping Data with the reshape Package. Journal of Statistical Software, 21(12), 1-20. URL http://www.jstatsoft.org/v21/i12/.
- Santander Meteorology Group (2015). downscaleR: Climate data manipulation and statistical downscaling. R package version 0.6-0. https://github.com/SantanderMetGroup/downscaleR/wiki

Examples

```
filePath <- system.file("extdata", "tnc.nc", package = "hyfo")
# Then if you don't know the variable name, you can use \code{getNcdfVar} to get variable name
varname <- getNcdfVar(filePath)
nc <- loadNcdf(filePath, varname)
a <- getFrcEnsem(nc)

# If there is no member session in the dataset, a single time sereis will be extracted.
a1 <- getFrcEnsem(tgridData)


# The default output is spatially averaged, if there are more than one cells in the dataset,
# the mean value of the cells will be calculated. While if you are interested in special cell,
# you can assign the cell value. You can also directly use longitude and latitude to extract
# time series.

getSpatialMap(nc, 'mean')
a <- getFrcEnsem(nc, cell = c(6,2))

# From the map, cell = c(6, 2) means lon = -1.4, lat = 43.2, so you can use corrd to locate
# your research area and extract time series.
b <- getFrcEnsem(nc, coord = c(-1.4, 43.2))


# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

`getHisEnsem` *Get ensemble forecast from historical data.*

Description

getHisEnsem use historical data as the forecasting input time series.

Usage

```
getHisEnsem(
  TS,
  example,
  interval = 365,
  buffer = 0,
  plot = "norm",
  output = "data",
  name = NULL,
  mv = 0,
  ...
)
```

Arguments

TS	A time series dataframe, with first column Date, and second column value.
example	A vector containing two strings showing the start and end date, which represent the forecasting period. Check details for more information. the program will extract every possible period in TS you provided to generate the ensemble. Check details for more information.
interval	A number representing the interval of each ensemble member. NOTE: "interval" takes 365 as a year, and 30 as a month, regardless of leap year and months with 31 days. So if you want the interval to be 2 years, set `interval = 730`, which equals 2 * 365 ; if two months, set `interval = 60`; 2 days, `interval = 2`, for other numbers that cannot be divided by 365 or 30 without remainder, it will treat the number as days.By defualt interval is set to be 365, a year.
buffer	A number showing how many days are used as buffer period for models. Check details for more information.
plot	A string showing whether the plot will be shown, e.g., 'norm' means normal plot (without any process), 'cum' means cummulative plot, default is 'norm'. For other words there will be no plot.
output	A string showing which type of output you want. Default is "data", if "ggplot", the data that can be directly plotted by ggplot2 will be returned, which is easier for you to make series plots afterwards. NOTE: If `output = 'ggplot'`, the missing value in the data will be replaced by `mv`, if assigned, default mv is 0.
name	If `output = 'ggplot'`, name has to be assigned to your output, in order to differentiate different outputs in the later multiplot using `getEnsem_comb`.
mv	A number showing representing the missing value. When calculating the cumulative value, missing value will be replaced by mv, default is 0.
...	`title, x, y` showing the title and x and y axis of the plot. e.g. `title = 'aaa'`

Details

`example` E.g., if you have a time series from 2000 to 2010. Assuming you are in 2003, you want to forecast the period from 2003-2-1 to 2003-4-1. Then for each year in your input time series, every year from 1st Feb to 1st Apr will be extracted to generate the ensemble forecasts. In this case your input example should be `example = c('2003-2-1', '2003-4-1')`

`interval` doesn't care about leap year and the months with 31 days, it will take 365 as a year, and 30 as a month. e.g., if the interval is from 1999-2-1 to 1999-3-1, you should just set interval to 30, although the real interval is 28 days.

`example` and `interval` controls how the ensemble will be generated. e.g. if the time series is from 1990-1-1 to 2001-1-1.

if `example = c('1992-3-1', '1994-1-1')` and `interval = 1095`, note, 1095 = 365 * 3, so the program treat this as 3 years.

Then you are supposed to get the ensemble consisting of following part:

1. 1992-3-1 to 1994-1-1 first one is the example, and it's NOT start from 1990-3-1. 2. 1995-3-1 to 1997-1-1 second one starts from 1993, because "interval" is 3 years. 3. 1998-3-1 to 2000-1-1

because the last one "2000-3-1 to 2002-1-1", 2002 exceeds the original TS range, so it will not be included.

Sometimes, there are leap years and months with 31 days included in some ensemble part, in which case the length of the data will be different, e.g., 1999-1-1 to 1999-3-1 is 1 day less than 2000-1-1 to 2000-3-1. In this situation, the data will use example as a standard. If the example is 1999-1-1 to 1999-3-1, then the latter one will be changed to 2001-1-1 to 2000-2-29, which keeps the start Date and change the end Date.

If the end date is so important that cannot be changed, try to solve this problem by resetting the example period, to make the event included in the example.

Good set of example and interval can generate good ensemble.

`buffer` Sometimes the model needs to run for a few days to warm up, before the forecast. E.g., if a forecast starts at '1990-1-20', for some model like MIKE NAM model, the run needs to be started about 14 days. So the input timeseries should start from '1990-1-6'.

Buffer is mainly used for the model hotstart. Sometimes the hot start file cannot contain all the parameters needed, only some important parameters. In this case, the model needs to run for some time, to make other parameters ready for the simulation.

`name` Assuming you have two ggplot outputs, you want to plot them together. In this situation, you need a name column to differentiate one ggplot output from the other. You can assigne this name by the argument directly, name has to be assigned if `output = 'ggplot'` is selected,

Value

A ensemble time series using historical data as forecast.

References

- Hadley Wickham (2007). Reshaping Data with the reshape Package. Journal of Statistical Software, 21(12), 1-20. URL http://www.jstatsoft.org/v21/i12/.
- H. Wickham. ggplot2: elegant graphics for data analysis. Springer New York, 2009.

Examples

```
data(testdl)

a <- testdl[[1]]

# Choose example from "1994-2-4" to "1996-1-4"
b <- getHisEnsem(a, example = c('1994-2-4', '1996-1-4'))

# Default interval is one year, can be set to other values, check help for information.

# Take 7 months as interval
b <- getHisEnsem(a, example = c('1994-2-4', '1996-1-4'), interval = 210, plot = 'cum')
# Take 30 days as buffer
b <- getHisEnsem(a, example = c('1994-2-4', '1996-1-4'), interval = 210, buffer = 30)


# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

`getLMom` *get L moment analysis of the input distribution*

Description

get L moment analysis of the input distribution

Usage

```
getLMom(dis)
```

Arguments

`dis`	A distribution, for hydrology usually a time series with only data column without time.

Value

The mean, L-variation, L-skewness and L-kurtosis of the input distribution

References

- J. R. M. Hosking (2015). L-moments. R package, version 2.5. URL: https://CRAN.R-project.org/package=lmom.

Examples

```
dis <- seq(1, 100)
getLMom(dis)

# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

getMeanPreci *Get mean rainfall data.*

Description

Get mean rainfall data, e.g. mean annual rainfall, mean monthly rainfall and mean winter rainfall.

Usage

```
getMeanPreci(
  inputTS,
  method = NULL,
  yearIndex = NULL,
  monthIndex = NULL,
  fullResults = FALSE,
  omitNA = TRUE,
  plot = FALSE,
  ...
)
```

Arguments

inputTS	A time series with only data column (1 column).
method	A string showing the method used to calculate mean value, e.g., "annual". more information please refer to details.
yearIndex	A NUMERIC ARRAY showing the year index of the time series.
monthIndex	A NUMERIC ARRAY showing the month index of the time series.
fullResults	A boolean showing whether the full results are shown, default is FALSE. If FALSE, only mean value will be returned, if TRUE, the sequence of values will be returned.
omitNA	A boolean showing in the calculation, whether NA is omitted, default is FALSE.
plot	A boolean showing whether the results will be plotted.
...	title, x, y showing the title and x and y axis of the plot, shoud be a string.

Details

There are following methods to be selected, "annual": annual rainfall of each year is plotted. "winter", "spring", "autumn", "summer": seasonal rainfall of each year is plotted. Month(number 1 to 12): month rainfall of each year is plotted, e.g. march rainfall of each year. "meanMonthly": the mean monthly rainfall of each month over the whole period.

Since "winter" is a crossing year, 12, 1, 2, 12 is in former year, and 1, 2 are in latter year. so winter belongs to the latter year.

Value

The mean value of the input time series or the full results before calculating mean.

getMoment *get moment analysis of the input distribution*

Description

get moment analysis of the input distribution

Usage

```
getMoment(dis)
```

Arguments

dis	A distribution, for hydrology usually a time series with only data column without time.

Value

The mean, variation, skewness and kurtosis of the input distribution

References

- Lukasz Komsta and Frederick Novomestky (2015). moments: Moments, cumulants, skewness, kurtosis and related tests. R package version 0.14. https://CRAN.R-project.org/package=moments
- R Core Team (2015). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Examples

```
dis <- seq(1, 100)
getMoment(dis)

# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

getNcdfVar *Get variable name of the NetCDF file.*

Description

Get variable name in the NetCDF file. After knowning the name, you can use `loadNcdf` to load the target variable.

Usage

```
getNcdfVar(filePath)
```

Arguments

filePath	A path pointing to the netCDF file.

Value

The names of the varialbes in the file.

References

- David Pierce (2015). ncdf4: Interface to Unidata netCDF (Version 4 or Earlier) Format Data Files. R package version 1.14.1. https://CRAN.R-project.org/package=ncdf4

Examples

```
# First open the test NETcDF file.
filePath <- system.file("extdata", "tnc.nc", package = "hyfo")

# Then if you don't know the variable name, you can use \code{getNcdfVar} to get variable name
varname <- getNcdfVar(filePath)

# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

---

getPreciBar *get mean rainfall bar plot of the input dataset or time series.*

Description

get mean rainfall bar plot of the input dataset or time series.

Usage

```
getPreciBar(
  data,
  method,
  cell = "mean",
  output = "data",
  name = NULL,
  plotRange = TRUE,
  member = NULL,
  omitNA = TRUE,
  info = FALSE,
  ...
)

## S4 method for signature 'list'
getPreciBar(
  data,
```

```
  method,
  cell = "mean",
  output = "data",
  name = NULL,
  plotRange = TRUE,
  member = NULL,
  omitNA = TRUE,
  info = FALSE,
  ...
)

## S4 method for signature 'data.frame'
getPreciBar(
  data,
  method,
  cell = "mean",
  output = "data",
  name = NULL,
  plotRange = TRUE,
  member = NULL,
  omitNA = TRUE,
  info = FALSE,
  ...
)
```

Arguments

data	A list containing different information, should be the result of reading netcdf file using loadNcdf, or a time series, with first column the Date, second the value. Time series can be an ENSEMBLE containning different members. Than the mean value will be given and the range will be given.
method	A string showing the calculating method of the input time series. More information please refer to the details.
cell	A vector containing the locaton of the cell, e.g. c(2, 3), default is "mean", representing the spatially averaged value. Check details for more information.
output	A string showing the type of the output, if output = 'ggplot', the returned data can be used in ggplot and getPreciBar_comb(); if output = 'plot', the returned data is the plot containing all layers' information, and can be plot directly or used in grid.arrange; if not set, the data will be returned.
name	If output = 'ggplot', name has to be assigned to your output, in order to differentiate different outputs in the later multiplot using getSpatialMap_comb.
plotRange	A boolean showing whether the range will be plotted.
member	A number showing which member is selected to get, if the dataset has a "member" dimension. Default is NULL, if no member assigned, and there is a "member" in dimensions, the mean value of the members will be taken.
omitNA	A boolean showing whether the missing value is omitted.

info	A boolean showing whether the information of the map, e.g., max, mean ..., default is FALSE.
...	title, x, y showing the title and x and y axis of the plot. e.g. title = 'aaa'

Details

There are following methods to be selected, "annual": annual rainfall of each year is plotted. "winter", "spring", "autumn", "summer": seasonal rainfall of each year is plotted. Month(number 1 to 12): month rainfall of each year is plotted, e.g. march rainfall of each year. "meanMonthly": the mean monthly rainfall of each month over the whole period.

#Since "winter" is a crossing year, 12, 1, 2, 12 is in former year, and 1, 2 are in latter year. #so winter belongs to the latter year.

cell representing the location of the cell, NOTE: this location means the index of the cell, IT IS NOT THE LONGITUDE AND LATITUDE. e.g., cell = c(2, 3), the program will take the 2nd longitude and 3rd latitude, by the increasing order. Longitude comes first.

It is a generic function, if in your case you need to debug, please see ?debug() for how to debug S4 method.

Value

The calculated mean value of the input time series and the plot of the result.

References

- Hadley Wickham (2007). Reshaping Data with the reshape Package. Journal of Statistical Software, 21(12), 1-20. URL http://www.jstatsoft.org/v21/i12/.
- H. Wickham. ggplot2: elegant graphics for data analysis. Springer New York, 2009.
- R Core Team (2015). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Examples

```
#gridData provided by package is the result of \code{loadNcdf()}
data(tgridData)
b1 <- getPreciBar(tgridData, method = 'annual')
b2 <- getPreciBar(tgridData, method = 'meanMonthly')

data(testdl)
TS  <- testdl[[1]]
a <- getPreciBar(TS, method = 'spring')
# if info = T, the information will be given at the bottom.
a <- getPreciBar(TS, method = 'spring', info = TRUE)


# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

`getPreciBar_comb` *Combine bars together*

Description

Combine bars together

Usage

```
getPreciBar_comb(
  ...,
  list = NULL,
  nrow = 1,
  x = "",
  y = "",
  title = "",
  output = FALSE
)
```

Arguments

`...`	different barplots generated by `getPreciBar(, output = 'ggplot')`, refer to details.
`list`	If input is a list containing different ggplot data, use `llist = inputlist`. NOTE: yOU HAVE TO PUT A `list =` , before your list.
`nrow`	A number showing the number of rows.
`x`	A string of x axis name.
`y`	A string of y axis name.
`title`	A string of the title.
`output`	A boolean, if chosen TRUE, the output will be given.

Details

..., representing different ouput generated by `getPreciBar(, output = 'ggplot')`, they have to be of the same type, e.g., 1. Jan precipitation of different years, Feb precipitation of different years, and... They are both monthly precipitation, and they share x axis.

2. Mean monthly precipitation of different dataset. e.g., long term mean monthly precipitation and short term mean monthly precipitation. They are both mean monthly precipitation.

Value

A combined barplot.

References

- H. Wickham. ggplot2: elegant graphics for data analysis. Springer New York, 2009.

Examples

```
data(tgridData)# the result of \code{\link{loadNcdf}}
#output type of getPreciBar() has to be 'ggplot'.
b1 <- getPreciBar(tgridData, method = 2, output = 'ggplot', name = 'b1')
b2 <- getPreciBar(tgridData, method = 3, output = 'ggplot', name = 'b2')

getPreciBar_comb(b1, b2)

# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

`getSpatialMap` *Get spatial map of the input dataset.*

Description

Get spatial map of the input dataset.

Usage

```
getSpatialMap(dataset, method = NULL, member = "mean", ...)
```

Arguments

`dataset`	A list containing different information, should be the result of reading netcdf file using `loadNcdf`.
`method`	A string showing different calculating method for the map. More information please refer to details.
`member`	A number showing which member is selected to get, if the dataset has a "member" dimension. Default is NULL, if no member assigned, and there is a "member" in dimensions, the mean value of the members will be taken.
`...`	several arguments including x, y, title, catchment, point, output, name, info, scale, color, type in `?getSpatialMap_mat` for details.

Details

There are following methods to be selected, "meanAnnual": annual rainfall of each year is plotted. "winter", "spring", "autumn", "summer": MEAN seasonal rainfall of each year is plotted. Month(number 1 to 12): MEAN month rainfall of each year is plotted, e.g. MEAN march rainfall of each year. "mean", "max", "min": mean daily, maximum daily, minimum daily precipitation.

Value

A matrix representing the raster map is returned, and the map is plotted.

Examples

```
## Not run:
#gridData provided in the package is the result of \code {loadNcdf}
data(tgridData)
getSpatialMap(tgridData, method = 'meanAnnual')
getSpatialMap(tgridData, method = 'winter')


getSpatialMap(tgridData, method = 'winter', catchment = testCat)

file <- system.file("extdata", "point.txt", package = "hyfo")
point <- read.table(file, header = TRUE, sep = ',' )
getSpatialMap(tgridData, method = 'winter', catchment = testCat, point = point)

## End(Not run)


# More examples can be found in the user manual on http://yuanchao-xu.github.io/hyfo/
```

`getSpatialMap_comb` *Combine maps together*

Description

Combine maps together

Usage

```
getSpatialMap_comb(
  ...,
  list = NULL,
  nrow = 1,
  x = "",
  y = "",
  title = "",
  output = FALSE
)
```

Arguments

`...`	different maps generated by `getSpatialMap(, output = 'ggplot')`, see details.
`list`	If input is a list containing different ggplot data, use `list = inputlist`.
`nrow`	A number showing the number of rows.
`x`	A string of x axis name.

y	A string of y axis name.
title	A string of the title.
output	A boolean, if chosen TRUE, the output will be given.

Details

For getSpatialMap_comb, the maps to be compared should be with same size and resolution, in other words, they should be fully overlapped by each other.

If they have different resolutions, use interpGridData{ecomsUDG.Raccess} to interpolate.

Value

A combined map.

References

- H. Wickham. ggplot2: elegant graphics for data analysis. Springer New York, 2009.

Examples

```
## Not run:
data(tgridData)# the result of \code{\link{loadNcdf}}
#The output should be 'ggplot'
a1 <- getSpatialMap(tgridData, method = 'summer', output = 'ggplot', name = 'a1')
a2 <- getSpatialMap(tgridData, method = 'winter', output = 'ggplot', name = 'a2')
a3 <- getSpatialMap(tgridData, method = 'mean', output = 'ggplot', name = 'a3')
a4 <- getSpatialMap(tgridData, method = 'max', output = 'ggplot', name = 'a4')
getSpatialMap_comb(a1, a2)

# or you can put them into a list.
getSpatialMap_comb(list = list(a1, a2), nrow = 2)

## End(Not run)



# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

---

getSpatialMap_mat *Replot raster matrix*

Description

replot the matrix output from getSpatialMap, when output = 'data' or output is default value.

Usage

```
getSpatialMap_mat(
  matrix,
  title_d = NULL,
  catchment = NULL,
  point = NULL,
  output = "data",
  name = NULL,
  info = FALSE,
  scale = "identity",
  color = NULL,
  ...
)
```

Arguments

matrix	A matrix raster, should be the result of `getSpatialMap()`, output should be default or 'data'
title_d	A string showing the title of the plot, defaut is NULL.
catchment	A catchment file geting from `shp2cat()` in the package, if a catchment is available for background.
point	A dataframe, showing other information, e.g., location of the gauging stations. The the data.frame should be with columes "name, lon, lat, z, value".
output	A string showing the type of the output, if `output = 'ggplot'`, the returned data can be used in ggplot and `getSpatialMap_comb()`; if `output = 'plot'`, the returned data is the plot containing all layers' information, and can be plot directly or used in grid.arrange; if not set, the raster matrix data will be returned.
name	If `output = 'ggplot'`, name has to be assigned to your output, in order to differentiate different outputs in the later multiplot using `getSpatialMap_comb`.
info	A boolean showing whether the information of the map, e.g., max, mean ..., default is FALSE.
scale	A string showing the plot scale, 'identity' or 'sqrt'.
color	Most of time you don't have to set this, but if you are not satisfied with the default color, you can set your own palette here. e.g., `color = c('red', 'blue')`, then the value from lowest to highest, will have the color from red to blue. More info about color, please check ?palette().
...	`title, x, y` showing the title and x and y axis of the plot. e.g. `title = 'aaa'` default is about precipitation.

Value

A matrix representing the raster map is returned, and the map is plotted.

References

- R Core Team (2015). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.
- Hadley Wickham (2007). Reshaping Data with the reshape Package. Journal of Statistical Software, 21(12), 1-20. URL http://www.jstatsoft.org/v21/i12/.
- Hadley Wickham (2011). The Split-Apply-Combine Strategy for Data Analysis. Journal of Statistical Software, 40(1), 1-29. URL http://www.jstatsoft.org/v40/i01/.
- Original S code by Richard A. Becker and Allan R. Wilks. R version by Ray Brownrigg. Enhancements by Thomas P Minka <tpminka at media.mit.edu> (2015). maps: Draw Geographical Maps. R package version 2.3-11. https://CRAN.R-project.org/package=maps
- Pebesma, Edzer, and Roger Bivand. 2023a. Sp: Classes and Methods for Spatial Data. https://CRAN.R-project.org/package=sp.
- Roger Bivand and Colin Rundel (2015). rgeos: Interface to Geometry Engine - Open Source (GEOS). R package version 0.3-11. https://CRAN.R-project.org/package=sf

Examples

```
## Not run:
data(tgridData)# the result of \code{loadNcdf}
#the output type of has to be default or 'data'.
a1 <- getSpatialMap(tgridData, method = 'mean')
a2 <- getSpatialMap(tgridData, method = 'max')
a3 <- getSpatialMap(tgridData, method = 'winter')
a4 <- getSpatialMap(tgridData, method = 'summer')
#For example, if we want to investigate the difference between mean value and max.

a5 <- a2 - a1
getSpatialMap_mat(a4)

#Or to investigate the difference between winter value and summer value.
a6 <- a3 - a4
getSpatialMap_mat(a6)


## End(Not run)


# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

`list2Dataframe` *Convert a list to a dataframe.*

Description

Convert a list of different time series to a dataframe. Usually the list is the output of `extractPeriod` NOTE: Since it's dataframe, so the dataframes in the input datalist should have the same date, if not, please use `extractPeriod` to process.

Usage

```
list2Dataframe(datalist)
```

Arguments

datalist	A list containing different time series, each sub list has to have the same length.

Value

The converted dataframe

Examples

```
# open file attached in the package.
file <- system.file("extdata", "testdl.txt", package = "hyfo")
datalist <- dget(file) # read list file.
datalist_new <- extractPeriod(datalist, commonPeriod = TRUE)

dataframe <- list2Dataframe(datalist_new)

# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

---

loadNcdf *Load NetCDF file*

Description

Load NetCDF file

Usage

```
loadNcdf(filePath, varname, tz = "GMT", ...)
```

Arguments

filePath	A path pointing to the NetCDF file, version3.
varname	A character representing the variable name, you can use getNcdfVar to get the basic information about the variables and select the target.
tz	A string representing the time zone, default is GMT, if you know what time zone is you can assign it in the argument. If tz = '', current time zone will be taken.
...	Several arguments including Year, month, lon, lat type in ?downscaleNcdf for details.You can load while downscale, and also first load than use downscaleNcdf to downscale.

Value

A list object from hyfo containing the information to be used in the analysis, or biascorrection.

References

- David Pierce (2015). ncdf4: Interface to Unidata netCDF (Version 4 or Earlier) Format Data Files. R package version 1.14.1. https://CRAN.R-project.org/package=ncdf4
- Santander MetGroup (2015). ecomsUDG.Raccess: R interface to the ECOMS User Data Gateway. R package version 2.2-6. http://meteo.unican.es/ecoms-udg

Examples

```
# First open the test NETcDF file.
filePath <- system.file("extdata", "tnc.nc", package = "hyfo")

# Then if you don't know the variable name, you can use \code{getNcdfVar} to get variable name
varname <- getNcdfVar(filePath)

nc <- loadNcdf(filePath, varname)

# you can directly add your downscale information to the argument.
nc1 <- loadNcdf(filePath, varname, year = 2006, lon = c(-2, -0.5), lat = c(43.2, 43.7))
nc2 <- loadNcdf(filePath, varname, year = 2005, month = 3:8, lon = c(-2, -0.5),
lat = c(43.2, 43.7))

# More examples can be found in the user manual on http://yuanchao-xu.github.io/hyfo/
```

`monthlyPreci` *Get monthly rainfall*

Description

Get monthly rainfall

Usage

```
monthlyPreci(TS, year, mon)
```

Arguments

`TS`	A rainfall time series.
`year`	A list showing the year index of the time series.
`mon`	A list showing the mon index of the time series.

Value

the monthly rainfall matrix of the rainfall time series.

plotTS *plot time series, with marks on missing value.*

Description

plot time series, with marks on missing value.

Usage

```
plotTS(
  ...,
  type = "line",
  output = "data",
  plot = "norm",
  name = NULL,
  showNA = TRUE,
  x = NULL,
  y = NULL,
  title = NULL,
  list = NULL
)
```

Arguments

...	input time series.
type	A string representing the type of the time series, e.g. 'line' or 'bar'.
output	A string showing which type of output you want. Default is "data", if "ggplot", the data that can be directly plotted by ggplot2 will be returned, which is easier for you to make series plots afterwards.
plot	representing the plot type, there are two types, "norm" and "cum", "norm" gives an normal plot, and "cum" gives a cumulative plot. Default is "norm".
name	If `output = 'ggplot'`, name has to be assigned to your output, in order to differentiate different outputs in the later multiplot using `plotTS_comb`.
showNA	A boolean representing whether the NA values should be marked, default is TRUE.
x	label for x axis.
y	label for y axis.
title	plot title.
list	If your input is a list of time series, then use `list = your time sereis list`

Details

If your input has more than one time series, the program will only plot the common period of different time series.

Value

A plot of the input time series.

References

- H. Wickham. ggplot2: elegant graphics for data analysis. Springer New York, 2009.

Examples

```
plotTS(testdl[[1]])
plotTS(testdl[[1]], x = 'xxx', y = 'yyy', title = 'aaa')

# If input is a datalist
plotTS(list = testdl)

# Or if you want to input time series one by one
# If plot = 'cum' then cumulative curve will  be plotted.
plotTS(testdl[[1]], testdl[[2]], plot = 'cum')

# You can also directly plot multicolumn dataframe
dataframe <- list2Dataframe(extractPeriod(testdl, commonPeriod = TRUE))
plotTS(dataframe, plot = 'cum')

# Sometimes you may want to process the dataframe and compare with the original one
dataframe1 <- dataframe
dataframe1[, 2:4] <- dataframe1[, 2:4] + 3
plotTS(dataframe, dataframe1, plot = 'cum')
# But note, if your input is a multi column dataframe, it's better to plot one using plotTS,
# and compare them using plotTS_comb. If all data are in one plot, there might be too messy.


# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

---

plotTS_comb *Combine time seires plot together*

Description

Combine time seires plot together

Usage

```
plotTS_comb(
  ...,
  nrow = 1,
  type = "line",
  list = NULL,
  x = "Date",
```

```
  y = "",
  title = "",
  output = FALSE
)
```

Arguments

...	different time series plots generated by `plotTS(, output = 'ggplot')`, refer to details.
nrow	A number showing the number of rows.
type	A string showing 'line' or 'bar'.
list	If input is a list containing different ggplot data, use `llist = inputlist`.
x	A string of x axis name.
y	A string of y axis name.
title	A string of the title.
output	A boolean, if chosen TRUE, the output will be given. NOTE: yOU HAVE TO PUT A `list =` , before your list.

Details

..., representing different ouput file generated by `plotTS(, output = 'ggplot'), name = yourname`, different names must be assigned when generating different output.

e.g. a1, a2, a3 are different files generated by `plotTS(, output = 'ggplot'), name = yourname`, you can set `plotTS(a1,a2,a3)` or `plotTS(list = list(a1,a2,a3))`

Value

A combined time series plot.

References

- H. Wickham. ggplot2: elegant graphics for data analysis. Springer New York, 2009.

Examples

```
a1 <- plotTS(testdl[[1]], output = 'ggplot', name = 1)
a2 <- plotTS(testdl[[2]], output = 'ggplot', name = 2)

plotTS_comb(a1, a2)
plotTS_comb(list = list(a1, a2), y = 'y axis', nrow = 2)

# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

`resample` *Resample your time series or ncdf files.*

Description

Resameple your time series or ncdf files, more info pleae see details.

Usage

```
resample(data, method)

## S4 method for signature 'data.frame'
resample(data, method)

## S4 method for signature 'list'
resample(data, method)
```

Arguments

`data`	a hyfo grid data or a time series, with first column date, and second column value. The date column should follow the format in `as.Date`, i.e. seperate with "-" or "/". Check details for more information.
`method`	A string showing whether you want to change a daily data to monthly data or monthly data to daily data.e.g. "mon2day" and "day2mon".

Details

Note, when you want to change daily data to monthly data, a new date column will be generated, usually the date column will be the middle date of each month, 15th, or 16th. However, if your time series doesn't start from the beginning of a month or ends to the end of a month, e.g. from 1999-3-14 to 2008-2-2, the first and last generated date could be wrong. Not only the date, but also the data, because you are not calculating based on a intact month.

It is a generic function, if in your case you need to debug, please see `?debug()` for how to debug S4 method.

Value

converted time series.

References

- R Core Team (2015). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Examples

```
# Daily to monthly
data(testdl)
TS <- testdl[[2]] # Get daily data
str(TS)
TS_new <- resample(TS, method = 'day2mon')

# Monthly to daily
TS <- data.frame(Date = seq(as.Date('1999-9-15'), length = 30, by = '1 month'),
runif(30, 3, 10))
TS_new <- resample(TS, method = 'mon2day')

#' # First load ncdf file.
filePath <- system.file("extdata", "tnc.nc", package = "hyfo")
varname <- getNcdfVar(filePath)
nc <- loadNcdf(filePath, varname)

nc_new <- resample(nc, 'day2mon')


# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

---

shp2cat *Get a catchment object from selected shape file.*

Description

Get a catchment object from selected shape file.

Usage

```
shp2cat(filePath)
```

Arguments

filePath	A string representing the path of the shape file.

Details

This function is based on the package sf and sp, and the output comes from the package sp

Value

A catchment object can be used in getSpatialMap().

References

- Roger Bivand, Tim Keitt and Barry Rowlingson (2015). rgdal: Bindings for the Geospatial Data Abstraction Library. R package version 1.0-4. https://CRAN.R-project.org/package=sf
- R Core Team (2015). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Examples

```
#open internal file
file <- system.file("extdata", "testCat.shp", package = "hyfo")
catchment <- shp2cat(file)

# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

`testCat` *testCat*

Description

testCat

Usage

```
testCat
```

Format

A catchment file generated by library sf.

class Formal class 'SpatialPolygonsDataFrame' [package "sp"] with 5 slots ...

`testdl` *testdl*

Description

A list containing different precipitation time series.

Usage

```
testdl
```

Format

A list consists of 3 different lists.

AAA AAA, a dataframe containing a date column and a value column.

BBB BBB, a dataframe containing a date column and a value column.

CCC CCC, a dataframe containing a date column and a value column. ...

Source

http://meteo.navarra.es/estaciones/mapadeestaciones.cfm http://www4.gipuzkoa.net/oohh/web/esp/02.asp

References

- http://meteo.navarra.es/estaciones/mapadeestaciones.cfm
- #' http://www4.gipuzkoa.net/oohh/web/esp/02.asp

---

tgridData *tgridData*

Description

A list containing different information getting from grid data file, e.g., netcdf file.

Usage

```
tgridData
```

Format

A list containing different information.

Variables variable information.

Data Data.

xyCoords longitude and latitude of the data.

Dates Date information. ...

Source

http://www.meteo.unican.es/datasets/spain02

References

- Herrera, S., Ancell, R., Gutierrez, J. M., Pons, M. R., Frias, M. D., & Fernandez, J. (2012). Development and analysis of a 50-year high-resolution daily gridded precipitation dataset over Spain (Spain02). International Journal of Climatology (http://www.meteo.unican.es/datasets/spain02), 10.1002/joc.2256.

`writeNcdf` *Write to NetCDF file using hyfo list file*

Description

Write to NetCDF file using hyfo list file

Usage

```
writeNcdf(
  gridData,
  filePath,
  missingValue = 1e+20,
  tz = "GMT",
  units = NULL,
  version = 3
)
```

Arguments

`gridData`	A hyfo list file from `loadNcdf`
`filePath`	A path of the new NetCDF file, should end with ".nc"
`missingValue`	A number representing the missing value in the NetCDF file, default is 1e20 #' @param tz A string representing the time zone, default is GMT, if you know what time zone is you can assign it in the argument. If `tz = ''`, current time zone will be taken.
`tz`	time zone, default is "GMT"
`units`	A string showing in which unit you are putting in the NetCDF file, it can be seconds or days and so on. If not specified, the function will pick up the possible largest time units from `c('weeks', 'days', 'hours', 'mins', 'secs')`
`version`	ncdf file versions, default is 3, if 4 is chosen, output file will be foreced to version 4.

Value

An NetCDF version 3 file.

References

- David Pierce (2015). ncdf4: Interface to Unidata netCDF (Version 4 or Earlier) Format Data Files. R package version 1.14.1. https://CRAN.R-project.org/package=ncdf4
- Santander MetGroup (2015). ecomsUDG.Raccess: R interface to the ECOMS User Data Gateway. R package version 2.2-6. http://meteo.unican.es/ecoms-udg

Examples

```
# First open the test NETcDF file.
filePath <- system.file("extdata", "tnc.nc", package = "hyfo")


# Then if you don't know the variable name, you can use \code{getNcdfVar} to get variable name
varname <- getNcdfVar(filePath)

nc <- loadNcdf(filePath, varname)

# Then write to your work directory

## Not run:
writeNcdf(nc, 'test.nc')

## End(Not run)


# More examples can be found in the user manual on https://yuanchao-xu.github.io/hyfo/
```

Index

∗ **datasets**
testCat, 53
testdl, 53
tgridData, 54

applyBiasFactor, 3, *26*
applyBiasFactor,data.frame,biasFactor-method *(applyBiasFactor)*, 3
applyBiasFactor,list,biasFactor.hyfo-method *(applyBiasFactor)*, 3

biasCorrect, *3, 4*, 6, *25, 26*
biasCorrect,data.frame,data.frame,data.frame-method *(biasCorrect)*, 6
biasCorrect,list,list,list-method *(biasCorrect)*, 6
biasFactor-class, 11
biasFactor.hyfo-class, 12

checkBind, 12
collectData, 13
collectData_csv_anarbe, 13
collectData_excel_anarbe, 14
collectData_txt_anarbe, 15
coord2cell, 16

downscaleNcdf, 16

extractPeriod, 17
extractPeriod,data.frame-method *(extractPeriod)*, 17
extractPeriod,list-method *(extractPeriod)*, 17

fillGap, 20

getAnnual, 22
getAnnual,data.frame-method *(getAnnual)*, 22
getAnnual,list-method *(getAnnual)*, 22
getAnnual_dataframe, 23
getBiasFactor, *4*, 24
getBiasFactor,data.frame,data.frame-method *(getBiasFactor)*, 24
getBiasFactor,list,list-method *(getBiasFactor)*, 24
getEnsem_comb, 28
getFrcEnsem, 29
getHisEnsem, 31
getLMom, 34
getMeanPreci, 35
getMoment, 36
getNcdfVar, 36
getPreciBar, 37
getPreciBar,data.frame-method *(getPreciBar)*, 37
getPreciBar,list-method *(getPreciBar)*, 37
getPreciBar_comb, 40
getSpatialMap, 41
getSpatialMap_comb, 42
getSpatialMap_mat, 43

list2Dataframe, 45
loadNcdf, *17*, *30*, *38*, 46, *55*

monthlyPreci, 47

plotTS, 48
plotTS_comb, 49

resample, 51
resample,data.frame-method *(resample)*, 51
resample,list-method *(resample)*, 51

shp2cat, 52

testCat, 53
testdl, 53
tgridData, 54

writeNcdf, 55