Package: grwat (via r-universe)

October 29, 2026

Type Package

Title River Hydrograph Separation and Analysis

Version 0.1.0.9999

Date 2025-10-02

Description River hydrograph separation and daily runoff time series analysis. Provides various filters to separate baseflow and quickflow. Implements advanced separation technique by Rets et al. (2022) <doi:10.1134/S0097807822010146> which involves meteorological data to reveal genetic components of the runoff: ground, rain, thaw and spring (seasonal thaw). High-performance C++17 computation, annually aggregated variables, statistical testing and numerous plotting functions for high-quality visualization.

License MIT + file LICENSE

URL https://github.com/tsamsonov/grwat, https://tsamsonov.github.io/grwat/

BugReports https://github.com/tsamsonov/grwat/issues

Encoding UTF-8

Language en-US

LinkingTo Rcpp

SystemRequirements pandoc

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

VignetteBuilder knitr

Depends R (>= 4.1)

Imports cli, Rcpp, dplyr, tidyr, lubridate, stringr, rlang, grid, ggplot2, zoo, trend, mblm, R.utils

Suggests ggridges, ggHoriPlot, ggthemes, kableExtra, knitr, ncdf4, rmarkdown, sf, testthat, stringi

Config/pak/sysreqs libicu-dev

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

Date/Publication 2026-09-29 16:25:21 UTC

RemoteUrl https://github.com/tsamsonov/grwat

RemoteRef HEAD

RemoteSha 4306331d3113ab0251363b859f3e078428c40774

Contents

gr_baseflow . . . 3
gr_buffer_geo . . . 5
gr_check_data . . . 6
gr_check_params . . . 7
gr_fill_gaps . . . 8
gr_from_pardf . . . 9
gr_get_gaps . . . 11
gr_get_params . . . 12
gr_help_params . . . 13
gr_help_vars . . . 13
gr_join_rean . . . 14
gr_kable_tests . . . 15
gr_plot_acf . . . 16
gr_plot_hori . . . 17
gr_plot_matrix . . . 18
gr_plot_minmonth . . . 19
gr_plot_periods . . . 20
gr_plot_ridge . . . 21
gr_plot_sep . . . 22
gr_plot_tests . . . 24
gr_plot_vars . . . 25
gr_read_era5_daily . . . 26
gr_read_rean . . . 27
gr_report . . . 28
gr_separate . . . 29
gr_set_locale . . . 32
gr_set_param . . . 33
gr_summarize . . . 34
gr_test_vars . . . 35
gr_to_pardf . . . 36
spas . . . 37

Index **39**

`gr_baseflow` *Extract baseflow*

Description

Extract baseflow from hydrological series using the filtering approach

Usage

```
gr_baseflow(
  Q,
  a = 0.925,
  k = 0.975,
  C = 0.05,
  aq = -0.5,
  passes = 3,
  padding = 30,
  method = "lynehollick"
)
```

Arguments

`Q`	Numeric runoff vector.
`a`	Numeric value of a filtering parameter used in `'chapman'`, `'jakeman'` and `'lynehollick'` methods. Defaults to `0.925`.
`k`	Numeric value of a filtering parameter used in `'boughton'` and `'maxwell'` methods. Defaults to `0.975`.
`C`	Numeric value of a separation shape parameter used in `'boughton'`, `'jakeman'` and `'maxwell'` methods
`aq`	Numeric value of a filtering parameter used in `'jakeman'` method. Defaults to `-0.5`.
`passes`	Integer number of filtering iterations. The first iteration is forward, second is backward, third is forward and so on. Defaults to `3`.
`padding`	Integer number of elements padded at the beginning and ending of runoff vector to reduce boundary effects. Defaults to `30`.
`method`	Character string to set baseflow filtering method. Available methods are `'boughton'`, `'chapman'`, `'jakeman'`, `'lynehollick'` and `'maxwell'`. Default is `'lynehollick'`, which corresponds to Lyne-Hollick (1979) hydrograph separation method.

Value

Numeric baseflow vector with length equal to `Q`

Examples

```
library(grwat)
library(ggplot2)
library(dplyr)
library(tidyr)
library(lubridate)

data(spas) # example Spas-Zagorye data is included with grwat package

# Calculate baseflow using Line-Hollick approach
hdata = spas |>
  mutate(Qbase = gr_baseflow(Q, method = 'lynehollick',
                             a = 0.925, passes = 3))

# Visualize for 1980 year
ggplot(hdata) +
  geom_area(aes(Date, Q), fill = 'steelblue', color = 'black') +
  geom_area(aes(Date, Qbase), fill = 'orangered', color = 'black') +
  scale_x_date(limits = c(ymd(19800101), ymd(19801231)))

# Compare various approaches
hdata = spas |>
  mutate(lynehollick = gr_baseflow(Q, method = 'lynehollick', a = 0.9),
         boughton = gr_baseflow(Q, method = 'boughton', k = 0.9),
         jakeman = gr_baseflow(Q, method = 'jakeman', k = 0.9),
         maxwell = gr_baseflow(Q, method = 'maxwell', k = 0.9)) |>
  pivot_longer(lynehollick:maxwell, names_to = 'Method', values_to = 'Qbase')

# Visualize for 1980 year
ggplot(hdata) +
  geom_area(aes(Date, Q), fill = 'steelblue', color = 'black') +
  geom_area(aes(Date, Qbase), fill = 'orangered', color = 'black') +
  scale_x_date(limits = c(ymd(19810101), ymd(19811231))) +
  facet_wrap(~Method)

# Compare Lyne to Kudelin
p = gr_get_params('center')
p$filter = 'kudelin'

hdata = spas |>
  mutate(lynehollick = gr_baseflow(Q, method = 'lynehollick',
                                   a = 0.925, passes = 3),
         kudelin = gr_separate(spas, p)$Qbase) |>
  pivot_longer(lynehollick:kudelin, names_to = 'Method', values_to = 'Qbase')

# Visualize for 1980 year
ggplot(hdata) +
  geom_area(aes(Date, Q), fill = 'steelblue', color = 'black') +
  geom_area(aes(Date, Qbase), fill = 'orangered', color = 'black') +
  scale_x_date(limits = c(ymd(19800101), ymd(19801231))) +
  facet_wrap(~Method)
```

gr_buffer_geo *Quasi-geographic buffering*

Description

Generate the buffer of spatial object in geographic coordinates. The function transforms the object into Azimuthal equidistant projection, then buffers it by the specified radius and then reprojects into geographical coordinate system (WGS84)

Usage

```
gr_buffer_geo(g, bufsize)
```

Arguments

g	sf or sfg object with known coordinate system.
bufsize	Numeric value of a buffer distance, in meters.

Value

sf or sfg object, buffered to bufsize and projected into geographic coordinates (WGS84).

Examples

```
if (require("sf")) {

  library(grwat)
  library(ggplot2)
  path = system.file("extdata", "spas-zagorye.gpkg", package = "grwat")
  basin = sf::st_read(path, layer = 'basin') # read basin region

  basin_buffer = gr_buffer_geo(basin, 25000)

  ggplot() +
    geom_sf(data = basin_buffer, fill = 'orangered', color = 'black') +
    geom_sf(data = basin, fill = 'steelblue', color = 'black')

}
```

gr_check_data *Check the correctness of data frame for separating*

Description

This function is called inside gr_separate(), but can be used explicitly inside your code.

Usage

```
gr_check_data(df)
```

Arguments

df	data.frame with four columns: date, runoff, temperature, precipitation, as required by gr_separate().

Value

stops execution if df contains the wrong number of columns, or the columns have the wrong types, or the data in columns is incorrect (e.g. runoff or precipitation are negative).

Examples

```
library(grwat)

# example Spas-Zagorye data is included with grwat package
data(spas)
head(spas)

gr_check_data(spas)

# raw Spas-Zagorye data represents date components
# in columns and does not contain meteorologgical variables
path = system.file("extdata", "spas-zagorye.txt",
                   package = "grwat")

hdata_raw = read.delim(path, header = FALSE,
                       sep = ' ', na.strings = c('-999', '-999.0', '-'),
                       col.names = c('d', 'm', 'y', 'q'))

print(hdata_raw)

try(gr_check_data(hdata_raw))
```

gr_check_params *Check the correctness of parameters list for separating*

Description

Check the correctness of parameters list for separating

Usage

```
gr_check_params(params, df = NULL)
```

Arguments

params	list of separation parameters, as returned by gr_get_params() function
df	data.frame with four columns: date, runoff, temperature, precipitation, as required by gr_separate(). Required when params is a list of parameter lists. Defaults to NULL.

Value

stops the execution if anything is wrong and prints the exact reason of the error. Otherwise prints the message that everything is OK

Examples

```
library(grwat)

# example Spas-Zagorye data is included with grwat package
data(spas)

params = gr_get_params(reg = 'center')

gr_check_params(params)

# set the unknown parameter
params$new = -2

# use try if you do not want to stop at error
try(gr_check_params(params))

# remove wrong parameter
params$new = NULL

# remove right parameter
params$grad1 = NULL
try(gr_check_params(params))

# reset
params = gr_get_params(reg = 'center')
```

```
sep = gr_separate(spas, params, debug = TRUE)
parlist = attributes(sep)$params

parlist[['2002']]$grad1 = 4

# if the parlist is used for separation
# then data frame must be specified
try(gr_check_params(parlist))

gr_check_params(parlist, spas)

# grad parameter is not known
parlist[['2002']]$grad = 4
try(gr_check_params(parlist, spas))

# remove wrong parameter
parlist[['2002']]$grad = NULL

# remove year
parlist[['2002']] = NULL
try(gr_check_params(parlist, spas))

parlist[['2002']] = parlist[['2001']]
gr_check_params(parlist, spas)
```

`gr_fill_gaps` *Fill missing daily data*

Description

Use the function to fill the missing daily data by linear interpolation. These can be both missing dates and missing runoff or temperature values. A preliminary summary of missing data can be viewed by `gr_get_gaps()`

Usage

```
gr_fill_gaps(hdata, autocorr = 0.7, nobserv = NULL)
```

Arguments

`hdata`	`data.frame` with at least two columns, where the first column is `Date`, and the remaining columns have numeric type.
`autocorr`	Autocorrelation value that defines possible length of the period that can be filled. Defaults to 0.7. If `nobserv` parameter is set, then this parameter is ignored. If both parameters are `NULL`, then all gaps are filled disregard of their lengths (not recommended).

nobserv — Maximum number of contiguous observations that can be interpolated. Defaults to NULL. If this parameter is set, then autocorr parameter is ignored. If both parameters are NULL, then all gaps are filled disregard of their lengths (not recommended).

Value

data.frame which is a filled version of hdata

Examples

```
library(grwat)
library(dplyr)

# example Spas-Zagorye data is included with grwat package
path = system.file("extdata", "spas-zagorye.txt",
                   package = "grwat")

hdata_raw = read.delim(path, header = FALSE,
                       sep = ' ', na.strings = c('-999', '-999.0', '-'),
                       col.names = c('d', 'm', 'y', 'q'))

hdata = hdata_raw |>
  transmute(Date = lubridate::make_date(y, m, d),
            Q = q)

head(hdata)

# identify gaps
gr_get_gaps(hdata)

# fill gaps
fhdata = gr_fill_gaps(hdata, autocorr = 0.8)

# check the results
gr_get_gaps(fhdata)

# fill gaps
fhdata = gr_fill_gaps(hdata, nobserv = 7)

# check the results
gr_get_gaps(fhdata)
```

---

gr_from_pardf *Convert data frame of parameters to list*

Description

Convert data frame of parameters to list

Usage

```
gr_from_pardf(pardf)
```

Arguments

pardf	tibble data frame with tabular representation of hydrograph separation parameters as returned by gr_to_pardf().

Value

list of lists of hydrograph separation parameters as returned in params attribute by gr_separate() with debug = TRUE.

Examples

```
library(grwat)

data(spas) # example Spas-Zagorye data is included with grwat package
head(spas)

# separate
sep = gr_separate(spas, params = gr_get_params(reg = 'center'))

# Visualize
gr_plot_sep(sep, c(1978, 1989))

# Debug mode gives access to additional information
sep_debug = gr_separate(spas,
                        params = gr_get_params(reg = 'center'),
                        debug = TRUE)

# a vector of years with jittered params
jit = attributes(sep_debug)$jittered
print(jit)

# actual params used for each year
parlist = attributes(sep_debug)$params

# tweak parameters for selected year
parlist[['1989']]$grad1 = 3
parlist[['1989']]$grad2 = 6

# tabular representation of parameters
partab = gr_to_pardf(parlist)

head(partab)

partab |>
  dplyr::filter(year == 1989) |>
  head()

parlist_back = gr_from_pardf(partab)
```

```
head(parlist_back[['1989']])
```

---

gr_get_gaps *Get gaps in the daily data*

Description

Use the function to detect periods of missing data. The first column must be of Date type. The data is considered to be a gap if any value in the row is missing.

Usage

```
gr_get_gaps(hdata)
```

Arguments

hdata	data.frame with at least two columns, where the first column is Date

Value

data.frame with periods of data and periods of gaps, containing five columns: number of the period (num), start of the period (start_date), end of the period (end_date), duration of the period (duration) and type of the period (type).

Examples

```
library(grwat)
library(dplyr)

# example Spas-Zagorye data is included with grwat package
path = system.file("extdata", "spas-zagorye.txt",
                   package = "grwat")

hdata_raw = read.delim(path, header = FALSE,
                       sep = ' ', na.strings = c('-999', '-999.0', '-'),
                       col.names = c('d', 'm', 'y', 'q'))

hdata = hdata_raw |>
  transmute(Date = lubridate::make_date(y, m, d),
            Q = q)

head(hdata)

# identify gaps
gr_get_gaps(hdata)

# fill gaps
fhdata = gr_fill_gaps(hdata, autocorr = 0.8)
```

```
# check the results
gr_get_gaps(fhdata)

# fill gaps
fhdata = gr_fill_gaps(hdata, nobserv = 7)

# check the results
gr_get_gaps(fhdata)
```

`gr_get_params` *Get hydrograph separation parameters*

Description

The function returns the list of parameters that can be used by `gr_separate()`. Since the parameters are region-specific, the location must be selected. It can be identified by region name or geographic coordinates. If both are specified, then region have a higher priority

Usage

```
gr_get_params(reg = "center", lon = NULL, lat = NULL)
```

Arguments

`reg`	Character string — the name of the region. Defaults to `'center'`.
`lon`	Numeric value of the longitude. Ignored if `reg` is specified.
`lat`	Numeric value of the latitude. Ignored if `reg` is specified.

Value

List of separation parameters that can be used in `gr_separate()` function.

Examples

```
library(grwat)

params = gr_get_params(reg = 'center')

print(params)
```

gr_help_params *Get the information about parameters used to separate the hydrograph*

Description

Get the information about parameters used to separate the hydrograph

Usage

```
gr_help_params()
```

Value

`data.frame` with description of hydrograph separation parameters that are used in `gr_separate()`.

Examples

```
library(grwat)

gr_help_params()
```

gr_help_vars *Hydrograph separation variables*

Description

Use this function to learn the meaning of the variables that are calculated by `gr_summarize()`.

Usage

```
gr_help_vars()
```

Value

`data.frame` of hydrograph separation variables

Examples

```
library(grwat)

gr_help_vars()
```

gr_join_rean *Join reanalysis data*

Description

The function performs spatial join of meteorological variables (temperature and precipitation) from grwat reanalysis to the daily runoff time series. Reanalysis covers the East European Plain with 0.75 degrees spatial resolution and is obtained based on CIRES-DOE (1880-1949) and ERA5 (1950-2021) data. This function is useful when the data from meteorological stations are missing inside the basin.

Usage

```
gr_join_rean(hdata, rean, buffer)
```

Arguments

hdata	data.frame containing 2 columns: Date and runoff
rean	list as returned by gr_read_rean() or gr_read_era5_daily()
buffer	sf object containing the region to select reanalysis data. Usually a river basin is used to select the meteorological data. Use gr_buffer_geo() to buffer the basin by specified distance and get more data, if needed.

Details

Download the reanalysis archive from here.

Value

data.frame with four columns: date, runoff, temperature, precipitation.

Examples

```
if (require("sf") && require("ncdf4")) {

  library(grwat)
  library(dplyr)

  # example Spas-Zagorye daily runoff data is included with grwat package
  data_path = system.file("extdata", "spas-zagorye.txt",
                          package = "grwat")

  hdata_raw = read.delim(data_path, header = FALSE,
                         sep = ' ', na.strings = c('-999', '-999.0', '-'),
                         col.names = c('d', 'm', 'y', 'q'))

  hdata = hdata_raw |>
    transmute(Date = lubridate::make_date(y, m, d),
```

```
              Q = q)

  head(hdata)

  # read basin
  basin_path = system.file("extdata", "spas-zagorye.gpkg",
                           package = "grwat")
  basin = sf::st_read(basin_path, layer = 'basin') # read basin region
  basin_buffer = gr_buffer_geo(basin, 25000)

  ## Not run:
    # read reanalysis data
    rean = gr_read_rean(
      '/Volumes/Data/Spatial/Reanalysis/grwat/pre_1880-2021.nc',
      '/Volumes/Data/Spatial/Reanalysis/grwat/temp_1880-2021.nc'
    )

    # spatial join of reanalysis data to runoff data
    hdata_rean = gr_join_rean(hdata, rean, basin_buffer)

    head(hdata_rean)

## End(Not run)

}
```

`gr_kable_tests` *Tabular representation of tests*

Description

This function is used to represent the results of `gr_test_vars()` in a tabular form. Used mainly in `gr_report()`, but can be used for your own purposes.

Usage

```
gr_kable_tests(tests, format = "html")
```

Arguments

`tests`	`list` of tests as returned by `gr_test_vars()` function.
`format`	Character string encoding the type of output. Currently `'html'` only is supported.

Value

HTML table as returned by `knitr::kable()` function.

Examples

```
if (require("kableExtra")) {

  library(grwat)

  data(spas) # example Spas-Zagorye data is included with grwat package

  # separate
  sep = gr_separate(spas, params = gr_get_params(reg = 'center'))

  # summarize from 1965 to 1990
  vars = gr_summarize(sep, 1965, 1990)

  # test all variables
  tests = gr_test_vars(vars)

  # kable tests
  gr_kable_tests(tests)

}
```

`gr_plot_acf` *Plot runoff ACF*

Description

The function plots the autocorrelation function (ACF) for daily runoff time series. A number of days corresponding to the specified `autocorr` value is highlighted.

Usage

```
gr_plot_acf(hdata, autocorr = 0.7, maxlag = 30, print = TRUE)
```

Arguments

`hdata`	`data.frame` with first column as `Date` and the second column as runoff
`autocorr`	Numeric value of the autocorrelation for which the time period will be highlighted. Defaults to `0.7`.
`maxlag`	Integer value of the maximum daily lag used to calculate the correlation. Defaults to `30`.
`print`	Boolean. Print plot? Defaults to `TRUE`. Use `FALSE` if you want to tweak the plot aesthetics before plotting.

Value

`ggplot2` object representing the autocorrelation function (ACF) for daily runoff time series

Examples

```
library(grwat)

# example Spas-Zagorye data is included with grwat package
data(spas)
head(spas)

# plot ACF
gr_plot_acf(spas, 0.65)
```

---

gr_plot_hori *Horizon hydrograph plot*

Description

A convenient wrapper around ggHoriPlot::geom_horizon() to visualize multiple river hydrographs at once.

Usage

```
gr_plot_hori(df, years, pal = "Blues", rev = TRUE, scale = 6, print = TRUE)
```

Arguments

df	data.frame with date (1st) and runoff (2nd) columns.
years	Integer vector of years to be plotted.
pal	Numeric or character string. Color palette identifier passed to ggplot2::scale_fill_distiller().
rev	Boolean. Reverse the palette? Defaults to FALSE.
scale	Numeric scale factor passed to ggHoriPlot::geom_horizon(). Defaults to 6.
print	Boolean. Print plot? Defaults to TRUE. Use FALSE if you want to tweak the plot aesthetics before plotting.

Value

ggplot2 object representing multiple river hydrographs at once using the horizon plot approach

Examples

```
if (require("ggHoriPlot") && require("ggthemes")) {

  library(grwat)

  data(spas) # example Spas-Zagorye data is included with grwat package

  # separate
  sep = gr_separate(spas, params = gr_get_params(reg = 'center'))
```

```
  # horizon plot for selected years
  gr_plot_hori(sep, years = 1960:1980)

}
```

`gr_plot_matrix` *Runoff matrix plot*

Description

The function plots runoff values, components and seasons using the matrix-based approach. The X axis corresponds to the day of the year, and the Y axis corresponds to the year. The function is useful when the whole picture of river runoff needs to be assessed.

Usage

```
gr_plot_matrix(df, years = NULL, type = "runoff", print = TRUE)
```

Arguments

`df`	`data.frame` of hydrograph separation produced by `gr_separate()`.
`years`	Integer vector of years to be plotted. Defaults to `NULL`.
`type`	Character string. Supported options are `'runoff'`, `'component'`, and `'season'`. Defaults to `'runoff'`.
`print`	Boolean. Print plot? Defaults to `TRUE`. Use `FALSE` if you want to tweak the plot aesthetics before plotting.

Value

`ggplot2` object representing the runoff values, components or seasons using the matrix-based approach

Examples

```
library(grwat)

data(spas) # example Spas-Zagorye data is included with grwat package

# separate
sep = gr_separate(spas, params = gr_get_params(reg = 'center'))

# matrix plot for runoff
gr_plot_matrix(sep, type = 'runoff')

# matrix plot for seasons
gr_plot_matrix(sep, type = 'season')

# matrix plot for genetic components
gr_plot_matrix(sep, type = 'component')
```

gr_plot_minmonth *Plot minimum runoff month*

Description

Generate a histogram of a minimum runoff month for two periods: before and after the change year set by `year` parameter.

Usage

```
gr_plot_minmonth(
  df,
  year = NULL,
  exclude = NULL,
  tests = NULL,
  pagebreak = FALSE,
  print = TRUE
)
```

Arguments

`df`	`data.frame` of hydrograph and meteorological variables as produced by `gr_summarize()`.
`year`	Integer. Change year value to separate two periods.
`exclude`	Integer vector of years to be excluded from plotting.
`tests`	Tests list for the same variables (generated by `gr_test_vars()` function)
`pagebreak`	Logical. Whether to break page between plots (needed for reporting). Defaults to `FALSE`.
`print`	Boolean. Print plot? Defaults to `TRUE`. Use `FALSE` if you want to tweak the plot aesthetics before plotting.

Value

`list` of two `ggplot2` objects, representing the histogram of a minimum runoff month for two periods: before and after the change year

Examples

```
library(grwat)

data(spas) # example Spas-Zagorye data is included with grwat package

# separate
sep = gr_separate(spas, params = gr_get_params(reg = 'center'))

# summarize from 1965 to 1990
vars = gr_summarize(sep, 1965, 1990)
```

```
# plot minimum runoff month for two periods divided by Pettitt test
gr_plot_minmonth(vars, tests = gr_test_vars(vars))

# plot minimum runoff month for two periods divided by fixed year
gr_plot_minmonth(vars, year = 1978)
```

`gr_plot_periods` *Plot long-term hydrograph variable changes*

Description

This function generates boxplots of the hydrograph separation variables produced by `gr_summarize()`. The data for each variable is divided into two samples: before and after the change year either set by `year` parameter or extracted from `tests` (statistically estimated). Different background fill colors are used to differentiate seasons types.

Usage

```
gr_plot_periods(
  df,
  ...,
  year = NULL,
  exclude = NULL,
  tests = NULL,
  layout = as.matrix(1),
  pagebreak = FALSE,
  print = TRUE
)
```

Arguments

`df`	`data.frame` of hydrograph and meteorological variables produced by `gr_summarize()`.
`...`	Quoted sequence of variable names.
`year`	Integer. Change year value to separate two periods (overridden by tests if it is supplied).
`exclude`	Integer vector of years to be excluded from plotting.
`tests`	Tests list for the same variables (generated by `gr_test_vars()` function)
`layout`	`matrix` that encodes the order of plotting.
`pagebreak`	Logical. Whether to break page between plots (needed for reporting). Defaults to `FALSE`.
`print`	Boolean. Print plot? Defaults to `TRUE`. Use `FALSE` if you want to tweak the plot aesthetics before plotting.

Value

`list` of `ggplot2` objects, one for each variable, representing its long-term changes

Examples

```
library(grwat)

data(spas) # example Spas-Zagorye data is included with grwat package

# separate
sep = gr_separate(spas, params = gr_get_params(reg = 'center'))

# summarize from 1965 to 1990
vars = gr_summarize(sep, 1965, 1990)

# plot periods with fixed change year
gr_plot_periods(vars, Qygr, year = 1978)

# plot periods with change year from Pettitt test
gr_plot_periods(vars, Qygr, tests = TRUE)

# calculate test beforehand
tests = gr_test_vars(vars)
gr_plot_periods(vars, Qspmax, tests = tests)

# use matrix layout to plot multiple variables
gr_plot_periods(vars, Qygr, Qspmax, D10w1, Wsprngr,
                layout = matrix(1:4, nrow = 2),
                tests = tests)
```

---

gr_plot_ridge *Ridgeline hydrograph plot*

Description

A convenient wrapper around ggridges::geom_ridgeline() to visualize multiple river hydrographs at once.

Usage

```
gr_plot_ridge(
  df,
  years,
  pal = 4,
  rev = FALSE,
  scale = 0.01,
  alpha = 0.8,
  print = TRUE
)
```

Arguments

df	data.frame with date (1st) and runoff (2nd) columns.
years	Integer vector of years to be plotted.
pal	Numeric or character string. Color palette identifier passed to ggplot2::scale_fill_distiller().
rev	Boolean. Reverse the palette? Defaults to FALSE.
scale	Numeric scale factor passed to ggridges::geom_ridgeline(). Defaults to 0.01.
alpha	Numeric opacity value of the ridgeline plot. Defaults to 0.8.
print	Boolean. Print plot? Defaults to TRUE. Use FALSE if you want to tweak the plot aesthetics before plotting.

Value

ggplot2 object representing the multiple river hydrographs at once using the ridgeline plot approach

Examples

```
if (require("ggridges")) {

  library(grwat)

  data(spas) # example Spas-Zagorye data is included with grwat package

  # separate
  sep = gr_separate(spas, params = gr_get_params(reg = 'center'))

  # ridgline plot for selected years
  gr_plot_ridge(sep, years = c(1960, 1965, 1989, 2001, 2012))

}
```

---

gr_plot_sep *Plot hydrograph separation*

Description

The function plots river hydrograph by filling the different flow types using colors. Matrix layouts can be used if multiple plots are needed. Temperature and precipitation can be overlaid.

Usage

```
gr_plot_sep(
  df,
  years = NULL,
  layout = as.matrix(1),
```

```
  pagebreak = FALSE,
  temp = FALSE,
  prec = FALSE,
  span = 5,
  print = TRUE,
  yrange = "uniform"
)
```

Arguments

df	data.frame of hydrograph separation as produced by gr_separate().
years	Integer vector of years to be plotted.
layout	matrix that encodes the order of plotting.
pagebreak	Logical. Whether to break page between plots (used by gr_report()). Defaults to FALSE.
temp	Boolean. Add temperature curve to the plot? Defaults to FALSE. If both temp = TRUE and prec = TRUE, then the axis is drawn for precipitation.
prec	Boolean. Add precipitation curve to the plot? Defaults to FALSE. If both temp = TRUE and prec = TRUE, then the axis is drawn for precipitation.
span	Integer number of days to accumulate precipitation for plotting.
print	Boolean. Print plot? Defaults to TRUE. Use FALSE if you want to tweak the plot aesthetics before plotting.
yrange	Boolean. Y range for each plot. Defaults to 'uniform' which means that the highest value of Y axis is the same on all plots and is determined by the highest discharge for all years to be plotted. Use 'each' if you want each plot to have its own maximum along Y axis determined by the highest discharge for this year. You can also use any numeric value ymax in a form yrange = ymax to determine your own Y scale maximum for all plots.

Value

list of ggplot2 objects, one for each year, representing the hydrograph separation

Examples

```
library(grwat)

data(spas) # example Spas-Zagorye data is included with grwat package

# separate
sep = gr_separate(spas, params = gr_get_params(reg = 'center'))

# One year
gr_plot_sep(sep, 1978)

# Two years
gr_plot_sep(sep, c(1978, 1989))
```

```
# Two years in a matrix layout
gr_plot_sep(sep, 1987:1988, layout = matrix(1:2, nrow = 2, byrow = TRUE))

# Four years in a matrix layout with free Y scale
gr_plot_sep(sep, 1987:1990, layout = matrix(1:4, nrow = 2, byrow = TRUE), yrange = 'each')

# Add temperature and precipitation
gr_plot_sep(sep, 1991, temp = TRUE, prec = TRUE)
```

`gr_plot_tests` *Plot change year density*

Description

The function extracts change years from results of `gr_test_vars()` and plots their probability density. Since for every variable the change year is individual, this procedure allows finding the one most probable year, which is the mode of the distribution. This year is highlighted by the line and labeled on the plot.

Usage

```
gr_plot_tests(tests, type = "year", print = TRUE)
```

Arguments

`tests`	`list` of tests generated by `gr_test_vars()`.
`type`	Character string type of the plot. Currently only `'year'` is supported, which means that the distribution density of the change year detected by Pettitt test is visualized. Ignored until other types are implemented.
`print`	Boolean. Print plot? Defaults to `TRUE`. Use `FALSE` if you want to tweak the plot aesthetics before plotting.

Value

`ggplot2` object representing the selected type of the tested variable

Examples

```
library(grwat)

data(spas) # example Spas-Zagorye data is included with grwat package

# separate
sep = gr_separate(spas, params = gr_get_params(reg = 'center'))

# summarize from 1965 to 1990
vars = gr_summarize(sep, 1965, 1990)
```

```
# test all variables
tests = gr_test_vars(vars)

# plot change year from Pettitt test
gr_plot_tests(tests, type = 'year')
```

`gr_plot_vars` *Plot interannual hydrograph variable changes*

Description

This function plots the hydrograph separation variables produced by `gr_summarize()`. Different background fill colors and line types are used to differentiate seasons and variable types.

Usage

```
gr_plot_vars(
  df,
  ...,
  tests = NULL,
  exclude = NULL,
  smooth = TRUE,
  layout = as.matrix(1),
  pagebreak = FALSE,
  print = TRUE
)
```

Arguments

`df`	`data.frame` of hydrograph and meteorological variables produced by `gr_summarize()`.
`...`	Quoted sequence of variable names.
`tests`	`list` of tests for the same variables (generated by `gr_test_vars()` function). If tests are specified, then they are added to the plot.
`exclude`	Integer vector of years to be excluded from plotting.
`smooth`	Logical. If `TRUE` then local smoothing regression is plotted. Defaults to `TRUE`.
`layout`	`matrix` that encodes the order of plotting.
`pagebreak`	Logical. Whether to break page between plots (`gr_report()`). Defaults to `FALSE`.
`print`	Boolean. Print plot? Defaults to `TRUE`. Use `FALSE` if you want to tweak the plot aesthetics before plotting.

Value

`list` of `ggplot2` objects, one for each variable, representing its interannual changes

Examples

```
library(grwat)

data(spas) # example Spas-Zagorye data is included with grwat package

# separate
sep = gr_separate(spas, params = gr_get_params(reg = 'center'))

# summarize from 1965 to 1990
vars = gr_summarize(sep, 1965, 1990)

# plot one selected variable
gr_plot_vars(vars, Qygr)

# plot two variables sequentially
gr_plot_vars(vars, D10w1, Wsprngr)

# four variables in matrix layout with tests calculated on the fly
gr_plot_vars(vars, Qspmax, Qygr, D10w1, Wsprngr,
             layout = matrix(1:4, nrow = 2, byrow = TRUE),
             tests = TRUE)
```

---

gr_read_era5_daily *Read ERA5 daily reanalysis data*

Description

The function reads meteorological variables (temperature and precipitation) from ERA 5 post-processed daily statistics on single levels for using with `gr_join_rean()`.

Usage

```
gr_read_era5_daily(file_prec, file_temp)
```

Arguments

`file_prec`	Character string path to daily sum precipitation NetCDF file.
`file_temp`	Character string path to daily mean temperature NetCDF file.

Value

`list` containing time series, precipitation series, temperature series and spatial points (sf)

Examples

```
if (require("sf") && require("ncdf4")) {

  library(grwat)

  prec_path = system.file("extdata", "era5_daily_prec.nc",
                          package = "grwat")

  temp_path = system.file("extdata", "era5_daily_temp.nc",
                          package = "grwat")

  # read reanalysis data
  rean = gr_read_era5_daily(prec_path, temp_path)

  str(rean)
}
```

`gr_read_rean` *Read reanalysis data*

Description

The function reads meteorological variables (temperature and precipitation) from grwat reanalysis for using with `gr_join_rean()`. Reanalysis covers the East European Plain with 0.75 degrees spatial resolution and is obtained based on CIRES-DOE (1880-1949) and ERA5 (1950-2021) data.

Usage

```
gr_read_rean(file_prec, file_temp)
```

Arguments

`file_prec`	Character string path to precipitation NetCDF file.
`file_temp`	Character string path to temperature NetCDF file.

Details

Download the reanalysis archive from here.

Value

`list` containing time series, precipitation series, temperature series and spatial points (sf)

Examples

```
if (require("sf") && require("ncdf4")) {

  library(grwat)

  # read reanalysis data
  ## Not run:
    rean = gr_read_rean(
      '/Volumes/Data/Spatial/Reanalysis/grwat/pre_1880-2021.nc',
      '/Volumes/Data/Spatial/Reanalysis/grwat/temp_1880-2021.nc'
    )

    str(rean)

## End(Not run)

}
```

`gr_report` *Report hydrograph separation and variables*

Description

This function generates a graphical HTML report that summarizes separation of hydrograph, its variables and their statistical properties. See example report generated by this command for `spas` dataset included in grwat package.

Usage

```
gr_report(
  sep,
  vars,
  output = "Report.html",
  year = NULL,
  exclude = NULL,
  temp = FALSE,
  prec = FALSE,
  span = 5,
  locale = "EN"
)
```

Arguments

`sep`	`data.frame` of hydrograph separation as returned by `gr_separate()` function.
`vars`	`data.frame` of hydrograph variables as returned by `gr_summarize()` function.
`output`	Character string path to the output file. Must have `.html` extension.

year	Integer value of year used to divide series in two samples compared by Student and Fisher tests. Defaults to NULL which means that the year is calculated automatically by Pettitt test. Defaults to NULL.
exclude	Integer vector of years to be excluded from reporting. Defaults to NULL.
temp	Boolean. Plot temperature on the top of hydrograph? Defaults to FALSE. If both temp = TRUE and prec = TRUE, then the axis is drawn for precipitation.
prec	Boolean. Plot precipitation on the top of hydrograph? Defaults to FALSE. If both temp = TRUE and prec = TRUE, then the axis is drawn for precipitation.
span	Integer number of days to accumulate precipitation for plotting. Defaults to 5.
locale	Character string locale. Currently only English ('EN') and Russian ('RU') locales are supported. Defaults to 'EN'.

Value

No return value, called for side effects

Examples

```
## Not run:
  if (require("knitr") && require("rmarkdown") && require("kableExtra")) {
    library(grwat)

    data(spas) # example Spas-Zagorye data is included with grwat package

    # separate
    sep = gr_separate(spas, params = gr_get_params(reg = 'center'))

    # summarize
    vars = gr_summarize(sep)

    # report
    report = '~/Spas-Zagorye.html'

    gr_report(sep, vars, output = report)
    browseURL(report)
  }

## End(Not run)
```

---

gr_separate *Advanced hydrograph separation*

Description

Separates the runoff into genetic components: groundwater, thaw, rain and spring.

Usage

```
gr_separate(df, params = gr_get_params(), debug = FALSE)
```

Arguments

df	data.frame with four columns: date, runoff, temperature, precipitation.
params	list of separation parameters, as returned by gr_get_params() function. Can also be a list of such lists if modified parameters are required for some years. In this case the length of params must be equal to the number of calendar years in df or be equal to 1.
debug	Boolean. If TRUE then additional attributes jittered and params are written to the output data.frame. jittered is an integer vector of years for which the separation parameters were randomly jittered. params is a list of separation parameter lists used for each year (some o those may have been jittered). Defaults to FALSE.

Value

A data.frame with 11 columns:

Column	Description
Date	date
Q	total runoff
Temp	temperature
Prec	precipitation
Qbase	baseflow
Quick	quickflow
Qspri	spring flood
Qrain	rain floods
Qthaw	thaw floods
Season	a season of the year
Year	a water-resources year

Examples

```
library(grwat)

data(spas) # example Spas-Zagorye data is included with grwat package
head(spas)

# separate
sep = gr_separate(spas, params = gr_get_params(reg = 'center'))

# Visualize
gr_plot_sep(sep, c(1978, 1989))

# Debug mode gives access to additional information
sep_debug = gr_separate(spas,
```

```
                        params = gr_get_params(reg = 'center'),
                        debug = TRUE)

# a vector of years with jittered params
jit = attributes(sep_debug)$jittered
print(jit)

# actual params used for each year
parlist = attributes(sep_debug)$params

# tabular representation of parameters
partab = gr_to_pardf(parlist)
head(partab)

parlist2 = partab |>
  dplyr::select(-year) |>
  apply(1, as.list) |>
  lapply(\(X) {
    n = length(X)
    X[1:(n - 1)] <- lapply(X[1:(n - 1)], as.numeric)
    return(X)
  }) |>
  setNames(partab$year)

# extract and tweak parameters for selected year
p = parlist[['1989']]
p$grad1 = 1
p$grad2 = 2.5

# use tweaked parameters for all years
sep_debug = gr_separate(spas, params = p, debug = TRUE)

# Visualize
gr_plot_sep(sep_debug, c(1978, 1989))

# actual params used for each year
parlist = attributes(sep_debug)$params

# tweak parameters for selected year
parlist[['1989']]$grad1 = 3
parlist[['1989']]$grad2 = 6

# set the sprecdays parameter for multiple years
parlist = gr_set_param(parlist, sprecdays,
                       years = c(1978, 1989:1995),
                       value = 15)

# set the spcomp parameter for all years
parlist = gr_set_param(parlist, spcomp, value = 2.5)

# use the list of parameters for separation
sep_debug = gr_separate(spas, params = parlist, debug = TRUE)
```

```
# Visualize
gr_plot_sep(sep_debug, c(1978, 1989))
```

`gr_set_locale` *Set the language that is used for plotting*

Description

Run this function once at the beginning of the session. All plots will be labeled using the selected language.

Usage

```
gr_set_locale(locale = "EN")
```

Arguments

`locale`	Character string locale. Currently only English (`'EN'`), Russian (`'RU'`) and Ukrainian (`'UA'`) locales are supported. More locales can be requested at issue on GitHub. Defaults to `'EN'`.

Details

Note to Linux users: the desired locale may not be installed on the system. A list of available locales can be obtained in bash terminal:

`locale -a`

Russian locale is `ru_RU.UTF-8`, and Ukrainian locale is `uk_UA.UTF-8`. If absent in the list, then install the desired locales by:

```
sudo locale-gen ru_RU.UTF-8
sudo locale-gen uk_UA.UTF-8
sudo update-locale
```

Then restart R session, and localization should work as expected.

Value

No return value, called for side effects

Examples

```
library(grwat)

data(spas) # example Spas-Zagorye data is included with grwat package

# separate
sep = gr_separate(spas, params = gr_get_params(reg = 'center'))
```

```
# Default is  English
gr_set_locale('EN')
gr_plot_sep(sep, 1978)
```

`gr_set_param` *Set the value of hydrograph separation parameter*

Description

The value is set for selected years in parameter list. Such list is returned by `gr_separate()` with `debug = TRUE` set.

Usage

```
gr_set_param(params, p, value, years = NULL)
```

Arguments

`params`	`list` of `lists` of hydrograph separation parameters as returned in `params` attribute by `gr_separate()` with `debug = TRUE`.
`p`	Name of the parameter.
`value`	Numeric value to set.
`years`	Integer vector of years to modify. Defaults to `NULL`, which means that all years will be modified.

Value

`list` of `lists` — a modified version of `params`

Examples

```
library(grwat)

data(spas) # example Spas-Zagorye data is included with grwat package

# Debug mode gives access to additional information
sep = gr_separate(spas,
                  params = gr_get_params(reg = 'center'),
                  debug = TRUE)

# Visualize
gr_plot_sep(sep, c(1978, 1989))

# actual params used for each year
parlist = attributes(sep)$params

# set the sprecdays parameter for multiple years
```

```
parlist = gr_set_param(parlist, sprecdays,
                       years = c(1978, 1989:1995),
                       value = 15)

# use the list of parameters for separation
sep_new = gr_separate(spas, params = parlist, debug = TRUE)

# Visualize
gr_plot_sep(sep_new, c(1978, 1989))
```

`gr_summarize` *Summarize hydrograph separation*

Description

Use this function to get meaningful summary statistics for hydrograph separation. Resulting variables are described by `gr_help_vars()`. This function is a convenient wrapper around dplyr's `df |> group_by |> summarize` idiom.

Usage

```
gr_summarize(df, year_min = NULL, year_max = NULL)
```

Arguments

`df`	`data.frame` of hydrograph separation resulting from `gr_separate()` function
`year_min`	`integer` first year to summarise
`year_max`	`integer` last year to summarise

Value

`data.frame` with one row for each water-resources year and multiple columns of statistics explained by `gr_help_vars()`.

Examples

```
library(grwat)

data(spas) # example Spas-Zagorye data is included with grwat package

# separate
sep = gr_separate(spas, params = gr_get_params(reg = 'center'))

# summarize
vars = gr_summarize(sep)

head(vars)

gr_plot_vars(vars, Qygr, tests = TRUE)
```

`gr_test_vars` *Test hydrograph changes*

Description

Use this function to test interannual changes or hydrograph separation variables returned by `gr_summarize()`. Pettitt test is used to detect the change year — i.e. the year which divides the time series into the statistically most differing samples. Student (Welch) and Fisher tests are used to estimate the significance of mean and variance differences of these samples. Theil-Sen test calculates the trend slope value. Mann-Kendall test is performed to reveal the significance of the trend.

Usage

```
gr_test_vars(df, ..., year = NULL, exclude = NULL)
```

Arguments

`df`	`data.frame` as produced by `gr_summarize()` function.
`...`	Names of the tested variables (quoted).
`year`	Integer value of year used to divide series in two samples compared by Student and Fisher tests. Defaults to `NULL` which means that the year is calculated automatically by Pettitt test.
`exclude`	Integer vector of years to be excluded from tests.

Details

Number of observations formally required for various tests: Pettitt > 0, Mann-Kendall > 2, Theil-Sen > 1, Student > 1, Fisher > 1.

Value

`list` of testing results with following elements:

Element	Description
`ptt`	Pettitt tests for change year
`mkt`	Mann-Kendall test for trend significance
`tst`	Theil-Sen test for slope estimation
`ts_fit`	Theil-Sen linear model fit
`tt`	Student (Welch) test for significance of mean differences between two periods
`ft`	Fisher test for significance of variance differences between two periods
`year`	Integer value of year used to divide series in two samples compared by Student and Fisher tests
`maxval`	Maximum value for the variable along the full time series
`fixed_year`	Boolean `TRUE` or `FALSE` value indicating if the year was fixed
`pvalues`	p-values of all tests summarized as a single table for all variables

Examples

```
library(grwat)

data(spas) # example Spas-Zagorye data is included with grwat package

# separate
sep = gr_separate(spas, params = gr_get_params(reg = 'center'))

# summarize from 1965 to 1990
vars = gr_summarize(sep, 1965, 1990)

# test all variables
tests = gr_test_vars(vars)

# view Pettitt test for Qygr
tests$ptt$Qygr

# view Fisher test for Q30s
tests$ft$Q30s

# test only Qygr and Q30s using 1978 as fixed year and excluding 1988-1991 yrs
gr_test_vars(vars, Qygr, Q30s, year = 1978, exclude = 1981:1983)
```

---

gr_to_pardf *Convert list of parameters to data frame*

Description

Convert list of parameters to data frame

Usage

```
gr_to_pardf(params)
```

Arguments

params	list of lists of hydrograph separation parameters as returned in params attribute by gr_separate() with debug = TRUE.

Value

tibble data frame with tabular representation of hydrograph separation parameters

Examples

```
library(grwat)

data(spas) # example Spas-Zagorye data is included with grwat package
head(spas)
```

```
# separate
sep = gr_separate(spas, params = gr_get_params(reg = 'center'))

# Visualize
gr_plot_sep(sep, c(1978, 1989))

# Debug mode gives access to additional information
sep_debug = gr_separate(spas,
                        params = gr_get_params(reg = 'center'),
                        debug = TRUE)

# a vector of years with jittered params
jit = attributes(sep_debug)$jittered
print(jit)

# actual params used for each year
parlist = attributes(sep_debug)$params

# tweak parameters for selected year
parlist[['1989']]$grad1 = 3
parlist[['1989']]$grad2 = 6

# tabular representation of parameters
partab = gr_to_pardf(parlist)

head(partab)

partab |>
  dplyr::filter(year == 1989) |>
  head()
```

---

spas *Spas-Zagorye daily runoff data*

Description

A dataset containing the daily runoff data for Spas-Zagorye gauge on Protva river in Central European plane. The dataset is supplemented by meteorological variables (temperature and precipitation) obtained from CIRES-DOE (1880-1949) and ERA5 (1950-2021) data.

Usage

```
spas
```

Format

A data frame with 23742 rows and 4 variables:

Date date, in dates

Q daily runoff, in m3/s

Temp daily temperature, in Celsius degrees

Prec daily precipitation, in mm

Source

https://allrivers.info/gauge/protva-obninsk

https://gmvo.skniivh.ru

https://www.ecmwf.int/en/forecasts/dataset/ecmwf-reanalysis-v5

https://psl.noaa.gov/data/gridded/data.20thC_ReanV3.html

Index

∗ **datasets**
spas, 37

ggHoriPlot::geom_horizon(), *17*
ggplot2::scale_fill_distiller(), *17*, *22*
ggridges::geom_ridgeline(), *21, 22*
gr_baseflow, 3
gr_buffer_geo, 5
gr_buffer_geo(), *14*
gr_check_data, 6
gr_check_params, 7
gr_fill_gaps, 8
gr_from_pardf, 9
gr_get_gaps, 11
gr_get_gaps(), *8*
gr_get_params, 12
gr_get_params(), *7*, *30*
gr_help_params, 13
gr_help_vars, 13
gr_help_vars(), *34*
gr_join_rean, 14
gr_join_rean(), *26, 27*
gr_kable_tests, 15
gr_plot_acf, 16
gr_plot_hori, 17
gr_plot_matrix, 18
gr_plot_minmonth, 19
gr_plot_periods, 20
gr_plot_ridge, 21
gr_plot_sep, 22
gr_plot_tests, 24
gr_plot_vars, 25
gr_read_era5_daily, 26
gr_read_era5_daily(), *14*
gr_read_rean, 27
gr_read_rean(), *14*
gr_report, 28
gr_report(), *15*, *23*, *25*
gr_separate, 29
gr_separate(), *6, 7*, *10*, *12, 13*, *18*, *23*, *28*, *33, 34*, *36*
gr_set_locale, 32
gr_set_param, 33
gr_summarize, 34
gr_summarize(), *13*, *19, 20*, *25*, *28*, *35*
gr_test_vars, 35
gr_test_vars(), *15*, *19, 20*, *24, 25*
gr_to_pardf, 36
gr_to_pardf(), *10*

knitr::kable(), *15*

spas, 37