Package: HASP (via r-universe)

November 3, 2026

Type Package

Title Hydrologic AnalySis Package

Version 1.0.4.9000

Description Data aggregation and plotting of USGS groundwater level data.

License CC0

Copyright This software is in the public domain because it contains materials that originally came from the United States Geological Survey, an agency of the United States Department of Interior. For more information, see the official USGS copyright policy at https://www.usgs.gov/visual-id/credit_usgs.html#copyright

Depends R (>= 4.1.0)

Imports dataRetrieval (>= 2.7.21), dplyr, tidyr, ggplot2, ggpmisc, ggpp, scales, rkt, checkmate, whisker, xfun, usethis, EnvStats, stats

Suggests testthat, knitr, kableExtra, covr, readxl, leaflet, shiny, DT, shinyAce, shinycssloaders, shinydashboard, openxlsx, rmarkdown, targets

BugReports https://code.usgs.gov/water/stats/hasp/-/issues

VignetteBuilder knitr

BuildVignettes true

LazyLoad yes

LazyData yes

Encoding UTF-8

RoxygenNote 7.3.3

Config/pak/sysreqs libabsl-dev cmake libgdal-dev gdal-bin libgeos-dev git make libgit2-dev libicu-dev libuv1-dev libxml2-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev libx11-dev

Repository https://ldecicco-usgs.r-universe.dev

Date/Publication 2026-11-03 14:58:33 UTC

RemoteUrl https://code.usgs.gov/water/stats/HASP

RemoteRef HEAD

RemoteSha 63eefc4a8be5b92eff7776e4047e58e8d5345770

Contents

aquifer_data . . . 3
composite_data . . . 3
create_groundwater_report . . . 4
create_targets_script . . . 5
daily_frequency_table . . . 6
daily_gwl_plot . . . 7
daily_gwl_summary . . . 9
data_available . . . 10
explore_aquifers . . . 11
filter_sites . . . 12
first_day . . . 13
get_aquifer_data . . . 13
get_range . . . 14
get_state_data . . . 15
gwl_plot_field . . . 15
L2701_example_data . . . 18
last_day . . . 19
map_hydro_data . . . 19
mid_month . . . 20
monthly_frequency_plot . . . 21
monthly_frequency_table . . . 24
monthly_mean . . . 26
nat_aqfr_state . . . 26
normalized_data . . . 27
plot_composite_data . . . 28
plot_normalized_data . . . 29
Sc_Cl_plot . . . 30
site_data_summary . . . 31
site_summary . . . 32
stats_by_interval . . . 32
theme_gwl . . . 34
trend_plot . . . 35
trend_test . . . 36
water_year . . . 38
weekly_frequency_plot . . . 39
weekly_frequency_table . . . 41

Index **43**

aquifer_data *Example aquifer data*

Description

Example data representing Basin and Range basin-fill aquifers (N100BSNRGB).

Examples

```
head(aquifer_data)
head(site_info)
```

composite_data *Composite hydrograph data*

Description

Create composite data

Usage

```
composite_data(x, num_years, parameter_cd)
```

Arguments

x	aquifer data
num_years	integer number of years required
parameter_cd	character, 5-digit parameter code, default is "72019".

Value

data frame with year, name, and value

Examples

```
aquifer_data <- aquifer_data
num_years <- 30

comp_data <- composite_data(aquifer_data, num_years, "72019")
```

`create_groundwater_report` *Generate HASP report*

Description

Opens a template of the blanks sample report. Running the function will create a Rmd file. The file can be "knit" as-is, or adjusted before rendering.

Usage

```
create_groundwater_report(
  siteID,
  report_name,
  report_folder,
  output_type = "word"
)
```

Arguments

`siteID`	character. USGS site ID, should be a groundwater site.
`report_name`	character. Base name of report.
`report_folder`	character. Report folder, can be full path or partial.
`output_type`	should be either "word", "html", or "pdf".

Examples

```
## Not run:
create_groundwater_report("USGS-253029080295601",
                          report_name = "example_report",
                          report_folder = "reports",
                          output_type = "word")

create_groundwater_report("USGS-253029080295601",
                          report_name = "example_report",
                          report_folder = "reports",
                          output_type = "html")

## End(Not run)
```

`create_targets_script` *Generate targets script for national aquifer data pull*

Description

Opens a template of the blanks sample report. Running the function will create a Rmd file.

Usage

```
create_targets_script(
  national_aquifer_cd,
  parameter_cd = "72019",
  statistic_cd = "00003",
  start_date,
  end_date,
  file_name = "_targets.R"
)
```

Arguments

national_aquifer_cd	character.
parameter_cd	character.
statistic_cd	character.
start_date	character.
end_date	character.
file_name	description

Examples

```
## Not run:
create_targets_script(national_aquifer_cd = "N100BSNRGB",
                      parameter_cd = "72019",
                      statistic_cd = "00003",
                      start_date = "1988-10-01",
                      end_date = "2021-01-01",
                      file_name = "_targets_test.R")

## End(Not run)
```

daily_frequency_table *Daily frequency table*

Description

Calculates the historical max, mean, minimum, and number of available points for each day of the year

Usage

```
daily_frequency_table(
  gw_level_dv,
  gwl_data,
  parameter_cd = NA,
  date_col = c("time", "time"),
  value_col = c("value", "value"),
  approved_col = c("approval_status", "approval_status")
)
```

Arguments

gw_level_dv	data frame, daily groundwater level data. Often obtained from read_waterdata_daily. Use NULL for no daily data.
gwl_data	data frame returned from read_waterdata_field_measurements, or data frame with a date, value, and approval columns. Use NULL for no discrete data.
parameter_cd	Can be used to filter data if the data frame has a "parameter_code" column. The default is NA, which will not do any filtering. If the gwl_data and gw_level_dv need different parameter code filtering, use a vector of 2 parameter codes. The first one will filter the gw_level_dv data frame, the second will filter the gwl_data data frame.
date_col	the name of the time columns. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is c("time", "time").
value_col	the name of the value columns. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is c("value", "value").
approved_col	the name of the column to get provisional/approved status. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is c("approval_status", "approval_status"). It is expected that these columns will have only "Approved" or "Provisional".

Value

a data frame giving the max, mean, min, and number of available days of data for each day of the year.

Examples

```
site <- "USGS-263819081585801"
p_code_dv <- "62610"
statCd <- "00001"
# gw_level_dv <- dataRetrieval::read_waterdata_daily(monitoring_location_id = site,
#                                                    parameter_code = p_code_dv,
#                                                    statistic_id = statCd)

gw_level_dv <- L2701_example_data$Daily

daily_frequency_table(gw_level_dv,
                      NULL,
                      parameter_cd = "62610")

# gwl_data <- dataRetrieval::read_waterdata_field_measurements(monitoring_location_id = site)
gwl_data <- L2701_example_data$Discrete

daily_frequency_table(gw_level_dv,
                      gwl_data,
                      parameter_cd = "62610")
```

`daily_gwl_plot` *Plot daily data*

Description

Calculates daily statistics based on all approved data. Daily, discrete, or both types are included. Historic median or mean are plotted based on all of the approved data.

Usage

```
daily_gwl_plot(
  gw_level_dv,
  gwl_data,
  parameter_cd = NA,
  date_col = c("time", "time"),
  value_col = c("value", "value"),
  approved_col = c("approval_status", "approval_status"),
  historical_stat = "mean",
  month_breaks = TRUE,
  plot_title = "",
  plot_range = "Past year",
  subtitle = "U.S. Geological Survey",
  y_axis_label = "",
  flip = FALSE
)
```

Arguments

gw_level_dv	data frame, daily groundwater level data. Often obtained from read_waterdata_daily. Use NULL for no daily data.
gwl_data	data frame returned from read_waterdata_field_measurements, or data frame with a date, value, and approval columns. Use NULL for no discrete data.
parameter_cd	Can be used to filter data if the data frame has a "parameter_code" column. The default is NA, which will not do any filtering. If the gwl_data and gw_level_dv need different parameter code filtering, use a vector of 2 parameter codes. The first one will filter the gw_level_dv data frame, the second will filter the gwl_data data frame.
date_col	the name of the time columns. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is c("time", "time").
value_col	the name of the value columns. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is c("value", "value").
approved_col	the name of the column to get provisional/approved status. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is c("approval_status", "approval_status"). It is expected that these columns will have only "Approved" or "Provisional".
historical_stat	the summary statistic to use for middle line of the plot. Either "mean" or "median."
month_breaks	a logical indicating whether to use monthly breaks for the plot
plot_title	the title to use on the plot
plot_range	the time frame to use for the plot. Use "Past year" (default) to see the last year of data, or "Calendar year" to use the current calendar year, beginning in January. Or specify two dates representing the start and end of the plot. If the first date is NA, it will start at the earliest record, if the second date is NA, it will end at the latest record.
subtitle	character. Sub-title for plot, default is "U.S. Geological Survey".
y_axis_label	the label used for the y-axis of the plot.
flip	logical. If TRUE, flips labels so that the lower numbers are in the higher percentages. Default is TRUE.

Examples

```
site <- "USGS-263819081585801"
p_code_dv <- "62610"
statCd <- "00001"
# gw_level_dv <- dataRetrieval::read_waterdata_daily(monitoring_location_id = site,
#                                                    parameter_code = p_code_dv,
#                                                    statistic_id = statCd)

gw_level_dv <- L2701_example_data$Daily
```

```
# gwl_data <- dataRetrieval::read_waterdata_field_measurements(monitoring_location_id = site)
gwl_data <- L2701_example_data$Discrete

daily_gwl_plot(gw_level_dv,
               NULL,
               plot_title = "Groundwater Level",
               historical_stat = "median")

daily_gwl_plot(gw_level_dv,
               gwl_data,
               parameter_cd = "62610",
               plot_title = "Groundwater Level",
               historical_stat = "median")

daily_gwl_plot(gw_level_dv,
               gwl_data,
               parameter_cd = "62610",
               plot_title = "Groundwater Level",
               month_breaks = TRUE,
               plot_range = c("2020-10-01", "2021-06-01"),
               historical_stat = "median")
```

`daily_gwl_summary` *Summary table of daily data*

Description

Summary table of daily data

Usage

```
daily_gwl_summary(
  gw_level_dv,
  gwl_data,
  parameter_cd = NA,
  date_col = c("time", "time"),
  value_col = c("value", "value"),
  approved_col = c("approval_status", "approval_status")
)
```

Arguments

`gw_level_dv`	data frame, daily groundwater level data. Often obtained from `read_waterdata_daily`. Use `NULL` for no daily data.
`gwl_data`	data frame returned from `read_waterdata_field_measurements`, or data frame with a date, value, and approval columns. Use `NULL` for no discrete data.

parameter_cd	Can be used to filter data if the data frame has a "parameter_code" column. The default is NA, which will not do any filtering. If the gwl_data and gw_level_dv need different parameter code filtering, use a vector of 2 parameter codes. The first one will filter the gw_level_dv data frame, the second will filter the gwl_data data frame.
date_col	the name of the time columns. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is c("time", "time").
value_col	the name of the value columns. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is c("value", "value").
approved_col	the name of the column to get provisional/approved status. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is c("approval_status", "approval_status"). It is expected that these columns will have only "Approved" or "Provisional".

Value

a summary table giving the period of record, completeness and percentile values

Examples

```
site <- "USGS-263819081585801"
p_code_dv <- "62610"
statCd <- "00001"
# gw_level_dv <- dataRetrieval::read_waterdata_daily(monitoring_location_id = site,
#                                                    parameter_code = p_code_dv,
#                                                    statistic_id = statCd)

gw_level_dv <- L2701_example_data$Daily

daily_gwl_summary(gw_level_dv,
                  gwl_data = NULL,
                  parameter_cd = p_code_dv)

# gwl_data <- dataRetrieval::read_waterdata_field_measurements(monitoring_location_id = site)
gwl_data <- L2701_example_data$Discrete

daily_gwl_summary(gw_level_dv,
                  gwl_data = gwl_data,
                  parameter_cd = p_code_dv)
```

---

data_available *site_summary*

Description

Get station summary information

Usage

```
data_available(siteID)
```

Arguments

siteID	character. USGS site ID for a groundwater site.

Examples

```
siteID <- "USGS-263819081585801"
site_data_available <- data_available(siteID)
```

---

explore_aquifers *Explore data in shiny applications*

Description

Open an interactive app in a browser. See the "Shiny App" vignette: `vignette("shinyApp", package = "toxEval")` for more details. Using this function is a quick and convenient way to explore data. For more customization, the R-code to produce each graph and table is displayed in the app. That is a good starting-point for a custom analysis.

Usage

```
explore_aquifers(browse = TRUE)

explore_site(browse = TRUE)
```

Arguments

browse	Logical. Use browser for running Shiny app.

Examples

```
## Not run:
explore_aquifers()

## End(Not run)
## Not run:
explore_site()

## End(Not run)
```

filter_sites *filter_sites*

Description

Filter down to sites with num_years of data

Usage

```
filter_sites(
  x,
  parameter_cd = "72019",
  num_years = NA,
  start_year = NA,
  end_year = NA
)
```

Arguments

x	aquifer data
parameter_cd	character, 5-digit parameter code, default is "72019".
num_years	integer number of years required. This can be NA, in which case the filter will use the full range of the data.
start_year	integer the first year to filter from. If NA, the filter will use the minimum from the data.
end_year	integer the last year to filter from. If NA, the filter will use the last year.

Value

data frame filter of x

Examples

```
aquifer_data <- aquifer_data
num_years <- 30

aq_data <- filter_sites(aquifer_data,
                        parameter_cd = "72019",
                        num_years = num_years)
```

first_day *Find the first day of the month for a given date*

Description

Find the first day of the month for a given date

Usage

```
first_day(date)
```

Arguments

date	a vector of dates

Value

the first day of the month that given dates fall in

Examples

```
date <- as.Date("2020-12-28")
first_day(date)
```

get_aquifer_data *get_aquifer_data*

Description

Get USGS data based on aquiferCd

Usage

```
get_aquifer_data(aquiferCd, startDate, endDate, parameter_cd = "72019")
```

Arguments

aquiferCd	character. To see valid aquifer codes, see the included data frame local_aqfr.
startDate	date or string. Beginning date of when to pull data.
endDate	date of string Ending date to pull data.
parameter_cd	5-digit character USGS parameter code.

Examples

```
end_date <- "2021-01-01"
start_date <- "1989-12-31"

aquiferCd <- "S100CSLLWD"

# aq_data <- get_aquifer_data(aquiferCd, start_date, end_date)
```

`get_range` *Get range*

Description

Get range

Usage

```
get_range(plot_range, full_range)
```

Arguments

`plot_range`	the time frame to use for the plot. Use "Past year" (default) to see the last year of data, or "Calendar year" to use the current calendar year, beginning in January. Or specify two dates representing the start and end of the plot.
`full_range`	the full time range of the data.

Value

list of plot_start, plot_end, label

Examples

```
full_range <- c("2000-01-01", "2025-10-01")
get_range("Past year", full_range)
get_range("Calendar year", full_range)
get_range(c("2024-10-01", "2025-06-01"), full_range)
get_range(c(NA, "2020-01-01"), full_range)
get_range(c("2020-01-01", NA), full_range)
```

get_state_data *get_state_data*

Description

Get USGS data based for a single state with specific aquifer codes.

Usage

```
get_state_data(state, aquiferCd, startDate, endDate, parameter_cd = "72019")
```

Arguments

state	character. Can be state abbreviation, long name, or numeric code.
aquiferCd	character. To see valid aquifer codes, see the included data frame local_aqfr.
startDate	date or string. Beginning date of when to pull data.
endDate	date of string Ending date to pull data.
parameter_cd	5-digit character USGS parameter code. Default is "72019".

gwl_plot_field *Single site groundwater level plots and tables*

Description

Function to create the field groundwater level data plot.

Usage

```
gwl_plot_field(
  gwl_data,
  plot_title = "",
  parameter_cd = NA,
  date_col = "time",
  value_col = "value",
  approved_col = "approval_status",
  flip = TRUE,
  y_label = "",
  subtitle = "U.S. Geological Survey"
)

gwl_plot_all(
  gw_level_dv,
  gwl_data,
  parameter_cd = NA,
```

```
  date_col = c("time", "time"),
  value_col = c("value", "value"),
  approved_col = c("approval_status", "approval_status"),
  y_label = "",
  subtitle = "U.S. Geological Survey",
  plot_title = "",
  add_trend = FALSE,
  n_years = 10,
  POR_trend = TRUE,
  flip = FALSE,
  days_required_per_month = 14
)
```

Arguments

gwl_data	data frame returned from read_waterdata_field_measurements, or data frame with a date, value, and approval columns. Use NULL for no discrete data.
plot_title	character, title for plot.
parameter_cd	Can be used to filter data if the data frame has a "parameter_code" column. The default is NA, which will not do any filtering. If the gwl_data and gw_level_dv need different parameter code filtering, use a vector of 2 parameter codes. The first one will filter the gw_level_dv data frame, the second will filter the gwl_data data frame.
date_col	the name of the time columns. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is c("time", "time").
value_col	the name of the value columns. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is c("value", "value").
approved_col	the name of the column to get provisional/approved status. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is c("approval_status", "approval_status"). It is expected that these columns will have only "Approved" or "Provisional".
flip	logical. If TRUE, flips labels so that the lower numbers are in the higher percentages. Default is TRUE.
y_label	character for y-axis label. Consider using read_waterdata_parameter_codes for USGS parameter_name.
subtitle	character. Sub-title for plot, default is "U.S. Geological Survey".
gw_level_dv	data frame, daily groundwater level data. Often obtained from read_waterdata_daily. Use NULL for no daily data.
add_trend	logical. Uses trend_test.
n_years	integer. This is the number of years to calculate the trend on. Default is 10. This can be a vector of years.
POR_trend	a logical indicating whether to include a trend test for the full period of record. Default is TRUE.

days_required_per_month
: integer. Number of days required per month. Default is 14. Only used if add_trend is TRUE using daily data.

Examples

```
site <- "USGS-263819081585801"
# gwl_data <- dataRetrieval::read_waterdata_field_measurements(monitoring_location_id = site)
gwl_data <- L2701_example_data$Discrete

pcodes <- dataRetrieval::read_waterdata_parameter_codes(
            parameter_code = unique(gwl_data$parameter_code)
          )

gwl_plot_field(gwl_data,
               plot_title = "L2701_example",
               parameter_cd = "62610",
               flip = FALSE,
               y_label = pcodes$parameter_name[pcodes$parameter_code == "62610"])

gwl_plot_field(gwl_data,
               plot_title = "L2701_example",
               y_label = pcodes$parameter_name[pcodes$parameter_code == "62611"],
               parameter_cd = "62611",
               flip = FALSE)

gwl_plot_field(gwl_data,
               plot_title = "L2701_example",
               y_label = pcodes$parameter_name[pcodes$parameter_code == "72019"],
               parameter_cd = "72019",
               flip = TRUE)

site <- "USGS-263819081585801"
p_code_dv <- "62610"
statCd <- "00001"
# gw_level_dv <- dataRetrieval::read_waterdata_daily(monitoring_location_id = site,
#                                                    parameter_code = p_code_dv,
#                                                    statistic_id = statCd)

gw_level_dv <- L2701_example_data$Daily

site_info <- dataRetrieval::read_waterdata_monitoring_location(monitoring_location_id = site)

plot_title <- site_info$monitoring_location_name
pcodes <- dataRetrieval::read_waterdata_parameter_codes(parameter_code = p_code_dv)

gwl_plot_all(gw_level_dv,
             NULL,
             plot_title = plot_title,
             y_label = pcodes$parameter_name,
             flip = TRUE)

# gwl_data <- dataRetrieval::read_waterdata_field_measurements(monitoring_location_id = site)
```

```
gwl_data <- L2701_example_data$Discrete

gwl_plot_all(gw_level_dv,
             gwl_data,
             parameter_cd = "62610",
             plot_title = plot_title,
             y_label = pcodes$parameter_name,
             add_trend = FALSE,
             flip = FALSE)

gwl_plot_all(gw_level_dv,
             gwl_data,
             parameter_cd = "62610",
             n_years = c(5, 20),
             POR_trend = TRUE,
             y_label = pcodes$parameter_name,
             plot_title = plot_title,
             add_trend = TRUE)

gwl_plot_all(NULL,
             gwl_data,
             parameter_cd = "62610",
             plot_title = plot_title,
             y_label = pcodes$parameter_name)

gwl_plot_all(NULL,
             gwl_data,
             parameter_cd = "62610",
             plot_title = plot_title,
             y_label = pcodes$parameter_name,
             add_trend = TRUE)
```

`L2701_example_data` *Example groundwater level data*

Description

Example data from site 263819081585801 L-2701. Data is a named list of "Daily" for daily groundwater level at the site and "Discrete" for discrete groundwater level measurements from the site. Updated April 9, 2021.

Examples

```
head(L2701_example_data$Daily)
head(L2701_example_data$Discrete)
head(L2701_example_data$QW)
```

last_day *Find the last day of the month for a given date*

Description

Find the last day of the month for a given date

Usage

```
last_day(date)
```

Arguments

date	a vector of dates

Value

the last day of the month that given dates fall in

Examples

```
date <- as.Date("2020-12-28")
last_day(date)
last_day("2020-02-15")
last_day("2019-02-15")
last_day(c("2020-12-28", "2020-02-15", "2019-02-15"))
```

map_hydro_data *Map data used in composite hydrographs*

Description

Map data used in composite hydrographs

Usage

```
map_hydro_data(x, site_info, num_years, parameter_cd = "72019")
```

Arguments

x	aquifer data frame. Requires at least 3 columns. Two are required "site_no", "year", and "value".
site_info	data frame from read_waterdata_monitoring_location
num_years	integer number of years required
parameter_cd	character, 5-digit parameter code, default is "72019".

Value

leaflet object

Examples

```
aquifer_data <- aquifer_data
site_info <- site_info
num_years <- 30

map_data <- map_hydro_data(aquifer_data, site_info, num_years)
map_data
```

---

mid_month *Find the middle of the month for a given date*

Description

Find the middle of the month for a given date

Usage

```
mid_month(date)
```

Arguments

date	a vector of dates

Value

the middle day of the month the given dates fall in

Examples

```
date <- as.Date("2020-12-28")
mid_month(date)
mid_month(c("2019-02-15", "2020-03-08", "2010-06-01"))
```

monthly_frequency_plot *Plot monthly frequency plot*

Description

This plot uses calculations from `monthly_frequency_table`. Daily, discrete, or both types of data can be included.

Usage

```
monthly_frequency_plot(
  gw_level_dv,
  gwl_data,
  parameter_cd = NA,
  date_col = c("time", "time"),
  value_col = c("value", "value"),
  approved_col = c("approval_status", "approval_status"),
  plot_title = "",
  subtitle = "U.S. Geological Survey",
  plot_range = c("Past year"),
  y_axis_label = "",
  flip = FALSE,
  percentile_colors = NA,
  include_edges = FALSE,
  median_point_size = 2.5,
  data_point_size = 2.5
)
```

Arguments

`gw_level_dv`	data frame, daily groundwater level data. Often obtained from `read_waterdata_daily`. Use `NULL` for no daily data.
`gwl_data`	data frame returned from `read_waterdata_field_measurements`, or data frame with a date, value, and approval columns. Use `NULL` for no discrete data.
`parameter_cd`	Can be used to filter data if the data frame has a "parameter_code" column. The default is `NA`, which will not do any filtering. If the gwl_data and gw_level_dv need different parameter code filtering, use a vector of 2 parameter codes. The first one will filter the gw_level_dv data frame, the second will filter the gwl_data data frame.
`date_col`	the name of the time columns. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is `c("time", "time")`.
`value_col`	the name of the value columns. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is `c("value", "value")`.

approved_col — the name of the column to get provisional/approved status. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is c("approval_status", "approval_status"). It is expected that these columns will have only "Approved" or "Provisional".

plot_title — the title to use on the plot

subtitle — character. Sub-title for plot, default is "U.S. Geological Survey".

plot_range — the time frame to use for the plot. Use "Past year" (default) to see the last year of data, or "Calendar year" to use the current calendar year, beginning in January. Or specify two dates representing the start and end of the plot. If the first date is NA, it will start at the earliest record, if the second date is NA, it will end at the latest record.

y_axis_label — the label used for the y-axis of the plot.

flip — logical. If TRUE, flips labels so that the lower numbers are in the higher percentages. Default is TRUE.

percentile_colors — Optional argument to provide a vector of 5 colors used to fill the percentile bars in order from lowest percentile bin to the highest percentile bin. Default behavior (NA) is to use legacy plot colors.

include_edges — Optional argument to toggle on the "edge bins" min-5 and 95-max on the plot. Default is FALSE which does not plot those bins.

median_point_size — Optional argument to specify the size of the median point markers which are shown as black triangles on the plot. The default size is 2.5.

data_point_size — Optional argument to specify the size of the data point markers which are shown as red diamonds on the plot. The default size is 2.5.

Value

a ggplot with rectangles representing the historical monthly percentile, black triangles representing the historical monthly median, and red diamonds showing the last year of groundwater level measurements.

Examples

```
site <- "USGS-263819081585801"
p_code_dv <- "62610"
statCd <- "00001"
# gw_level_dv <- dataRetrieval::read_waterdata_daily(monitoring_location_id = site,
#                                                    parameter_code = p_code_dv,
#                                                    statistic_id = statCd)

gw_level_dv <- L2701_example_data$Daily

label <- dataRetrieval::read_waterdata_parameter_codes(parameter_code = p_code_dv)$parameter_name

monthly_frequency <- monthly_frequency_plot(gw_level_dv,
                                            gwl_data = NULL,
```

```
                                          plot_title = "L2701 Groundwater Level",
                                          y_axis_label = label,
                                          flip = FALSE)
monthly_frequency

# gwl_data <- dataRetrieval::read_waterdata_field_measurements(monitoring_location_id = site)
gwl_data <- L2701_example_data$Discrete

monthly_frequency_plot(gw_level_dv,
                       gwl_data = gwl_data,
                       parameter_cd = "62610",
                       plot_title = "L2701 Groundwater Level",
                       y_axis_label = label,
                       flip = FALSE)

monthly_frequency_plot(gw_level_dv,
                       gwl_data = gwl_data,
                       parameter_cd = "62610",
                       plot_title = "L2701 Groundwater Level",
                       y_axis_label = label,
                       plot_range = c("2023-10-01", "2024-09-30"),
                       flip = FALSE)

monthly_frequency_flip <- monthly_frequency_plot(gw_level_dv,
                                  gwl_data,
                                  parameter_cd = "62610",
                                  y_axis_label = label,
                                  plot_title = "L2701 Groundwater Level",
                                  flip = TRUE)
monthly_frequency_flip

monthly_frequency_custom_colors <- monthly_frequency_plot(
                                     gw_level_dv,
                                     gwl_data,
                                     parameter_cd = "62610",
                                     y_axis_label = label,
                                     plot_title = "L2701 Groundwater Level",
                                     flip = TRUE,
                                     percentile_colors = c(
                                                  "red",
                                                  "yellow",
                                                  "green",
                                                  "blue",
                                                  "orange"
                                     ))

monthly_frequency_custom_colors

monthly_frequency_edge_bins <- monthly_frequency_plot(gw_level_dv,
                                    gwl_data,
                                    parameter_cd = "62610",
                                    y_axis_label = label,
                                    plot_title = "L2701 Groundwater Level",
```

```
                                    flip = FALSE,
                                    include_edges = TRUE)
monthly_frequency_edge_bins

monthly_frequency_custom_point_sizes <- monthly_frequency_plot(gw_level_dv,
                                    gwl_data = gwl_data,
                                    parameter_cd = "62610",
                                    plot_title = "L2701 Groundwater Level",
                                    y_axis_label = label,
                                    median_point_size = 0.5,
                                    data_point_size = 3)
monthly_frequency_custom_point_sizes
```

`monthly_frequency_table`

Create a table of monthly frequency analysis

Description

The table will accept daily, discrete, or a both types of data. The median of each year/month is calculated. Then using that median, monthly stats are calculated. Percentiles are calculated using the `quantile` function with "type=6".

Usage

```
monthly_frequency_table(
  gw_level_dv,
  gwl_data,
  parameter_cd = NA,
  date_col = c("time", "time"),
  value_col = c("value", "value"),
  approved_col = c("approval_status", "approval_status"),
  flip = FALSE
)
```

Arguments

`gw_level_dv`	data frame, daily groundwater level data. Often obtained from `read_waterdata_daily`. Use `NULL` for no daily data.
`gwl_data`	data frame returned from `read_waterdata_field_measurements`, or data frame with a date, value, and approval columns. Use `NULL` for no discrete data.
`parameter_cd`	Can be used to filter data if the data frame has a "parameter_code" column. The default is `NA`, which will not do any filtering. If the gwl_data and gw_level_dv need different parameter code filtering, use a vector of 2 parameter codes. The first one will filter the gw_level_dv data frame, the second will filter the gwl_data data frame.

date_col	the name of the time columns. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is c("time", "time").
value_col	the name of the value columns. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is c("value", "value").
approved_col	the name of the column to get provisional/approved status. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is c("approval_status", "approval_status"). It is expected that these columns will have only "Approved" or "Provisional".
flip	logical. If TRUE, flips labels so that the lower numbers are in the higher percentages. Default is TRUE.

Value

a data frame of monthly groundwater level statistics including the 5th, 10th, 25th, 75th, 90th, and 95th percentiles; the number of years of data; and the lowest monthly median and the highest monthly median.

Examples

```
site <- "USGS-263819081585801"
p_code_dv <- "62610"
statCd <- "00001"
# gw_level_dv <- dataRetrieval::read_waterdata_daily(monitoring_location_id = site,
#                                                    parameter_code = p_code_dv,
#                                                    statistic_id = statCd)

gw_level_dv <- L2701_example_data$Daily

monthly_frequency <- monthly_frequency_table(gw_level_dv,
                                             NULL)
head(monthly_frequency)
# gwl_data <- dataRetrieval::read_waterdata_field_measurements(monitoring_location_id = site)
gwl_data <- L2701_example_data$Discrete

monthly_frequency_combo <- monthly_frequency_table(gw_level_dv = gw_level_dv,
                                             gwl_data = gwl_data,
                                             parameter_cd = "62610")
head(monthly_frequency_combo)
monthly_flip <- monthly_frequency_table(gw_level_dv,
                                        gwl_data,
                                        parameter_cd = "62610",
                                        flip = TRUE)
head(monthly_flip)
```

monthly_mean *Monthly mean*

Description

Take the mean of each month, filter if there's not at least 15 days, and create a new date that is the midpoint of each month.

Usage

```
monthly_mean(x, date_col = "time", value_col = "value")
```

Arguments

x	data.frame
date_col	character name of date column
value_col	character name of value column

Examples

```
# site <- "USGS-263819081585801"
p_code_dv <- "62610"
# statCd <- "00001"
# gw_level_dv <- dataRetrieval::read_waterdata_daily(monitoring_location_id = site,
#                                                    parameter_code = p_code_dv,
#                                                    statistic_id = statCd,
#                                                    skipGeometry = TRUE)
gw_level_dv <- L2701_example_data$Daily

site_statistics <- monthly_frequency_table(gw_level_dv,
                                           NULL,
                                           parameter_cd = p_code_dv)

gw_monthly <- monthly_mean(gw_level_dv)
```

nat_aqfr_state *Aquifer data included*

Description

National aquifer data

Local aquifer data codes

markerTable raw data

Value

data frame with columns Country, State, nat_aqfr_cd, long_name, and other_cd

data frame with columns long_name, nat_aqfr_cd, state_indexes, and states

data frame with aqfr_cd, Aqfr_Name_prpr

data frame with columns Trend and MarkerDescription

data frame with columns trendType, trend, markerDescription, r_lwd

Examples

```
head(nat_aqfr_state)
head(summary_aquifers)
head(local_aqfr)
```

---

normalized_data *Composite normalized hydrograph data*

Description

Create normalized composite data

Usage

```
normalized_data(x, num_years, parameter_cd = "72019")
```

Arguments

x	aquifer data
num_years	integer number of years required
parameter_cd	character, 5-digit parameter code, default is "72019".

Value

data frame with year, name, and value

Examples

```
aquifer_data <- aquifer_data
num_years <- 30

norm_data <- normalized_data(aquifer_data, num_years, "72019")
```

plot_composite_data *Plot composite hydrograph data*

Description

Create composite hydrograph plot

Usage

```
plot_composite_data(
  x,
  num_years = NA,
  parameter_cd = "72019",
  plot_title = "",
  subtitle = "U.S. Geological Survey"
)
```

Arguments

x	aquifer data frame. Requires at least 3 columns: "monitoring_location_id", "year", and "value".
num_years	integer number of years required. If NA, the analysis will default to the range of the data in x.
parameter_cd	character, 5-digit parameter code, default is "72019".
plot_title	character title included on plot.
subtitle	character. Sub-title for plot, default is "U.S. Geological Survey".

Value

ggplot2 object

Examples

```
aquifer_data <- aquifer_data

comp_data <- plot_composite_data(aquifer_data,
                                 num_years = 30,
                                 parameter_cd = "72019",
                                 plot_title = "Calendar Year")
comp_data
# Do it on a water year:
aquifer_data$cal_year <- aquifer_data$year
aquifer_data$year <- aquifer_data$water_year
plot_composite_data(aquifer_data,
                    parameter_cd = "72019",
                    num_years = 30,
                    plot_title = "Water Year")
```

plot_normalized_data *Plot normalized composite hydrograph data*

Description

Create composite hydrograph plot

Usage

```
plot_normalized_data(
  x,
  num_years = NA,
  parameter_cd = "72019",
  plot_title = "",
  subtitle = "U.S. Geological Survey"
)
```

Arguments

x	aquifer data frame. Requires at least 3 columns. Two are required "site_no", "year", and "value".
num_years	integer number of years required to the analysis. If NA, the analysis will default to the range of the data in x.
parameter_cd	character, 5-digit parameter code, default is "72019".
plot_title	character title of plot.
subtitle	character. Sub-title for plot, default is "U.S. Geological Survey".

Value

ggplot2 object

Examples

```
aquifer_data <- aquifer_data

norm_data <- plot_normalized_data(aquifer_data,
                                  parameter_cd = "72019",
                                  num_years = 30)
norm_data

aquifer_data$cal_year <- aquifer_data$year
aquifer_data$year <- aquifer_data$water_year
plot_normalized_data(aquifer_data,
                     parameter_cd = "72019",
                     num_years = 30)
```

Sc_Cl_plot *Specific conductance and chloride*

Description

Functions to create the individual chloride, specific conductance, and combination plots and tables for a single site.

Usage

```
Sc_Cl_plot(qw_data, plot_title, subtitle = "U.S. Geological Survey")

Sc_Cl_table(qw_data)

qw_plot(
  qw_data,
  plot_title,
  y_label = NA,
  CharacteristicName = "Chloride",
  start_date = NA,
  end_date = NA,
  subtitle = "U.S. Geological Survey"
)

qw_summary(qw_data, CharacteristicName, norm_range = NA)
```

Arguments

qw_data	data frame returned from readWQPqw, must include columns "ActivityStartDateTime", "CharacteristicName", and "ResultMeasureValue"
plot_title	character title for plot
subtitle	character. Sub-title for plot, default is "U.S. Geological Survey".
y_label	character label for y axis. If left as NA, the function will attempt to use the "variableInfo" attribute of qw_data. This is attached to dataRetrieval output.
CharacteristicName	character CharacteristicName to filter to.
start_date	Date to start plot. If NA (which is the default), the plot will start at the earliest measurement.
end_date	Date to end plot. If NA (which is the default), the plot will end with the latest measurement.
norm_range	a numerical range to potentially group the data. If NA, no grouping is shown.

Examples

```
# site <- "USGS-263819081585801"
# parameterCd <- c("00095","90095","00940","99220")
# site_data <- dataRetrieval::readWQPqw(site,
#                                       parameterCd)
# Using package example data:
qw_data <- L2701_example_data$QW
plot_title <- paste("USGS-263819081585801",
 ": Specific Conductance vs Chloride")
Sc_Cl_plot(qw_data, plot_title)

sc_cl <- Sc_Cl_table(qw_data)
plot_title <- paste("USGS-263819081585801",
 ": Specific Conductance vs Chloride")
qw_plot(qw_data, plot_title, CharacteristicName = "Chloride")
qw_plot(qw_data, plot_title, CharacteristicName = "Specific conductance")
qw_plot(qw_data,
        plot_title,
        CharacteristicName = "Specific conductance",
        start_date = "1990-01-01")

site <- "USGS-01491000"
qw_data_phos <- dataRetrieval::readWQPqw(site, "Orthophosphate")
qw_plot(qw_data_phos ,
        CharacteristicName = "Orthophosphate",
        plot_title = "Choptank: Orthophosphate")

qw_summary(qw_data, CharacteristicName = "Chloride",
 norm_range = c(230, 860))
qw_summary(qw_data, CharacteristicName = "Specific conductance",
 norm_range = NA)
```

`site_data_summary` *site_data_summary*

Description

Get summaries of data by site. Requires a column site_no, and will take the summaries

Usage

```
site_data_summary(x, value_col = "value", site_col = "monitoring_location_id")
```

Arguments

`x`	data frame
`value_col`	name of value column. The default is `"value"`.
`site_col`	name of site column. This is the column we are grouping by.

Value

data frame with 10 columns

Examples

```
aquifer_data <- aquifer_data
aquifer_data <- aquifer_data[aquifer_data$parameter_code == "72019", ]
summary_info <- site_data_summary(aquifer_data)
```

`site_summary` *site_summary*

Description

Get station summary information

Usage

```
site_summary(siteID, markdown = FALSE)
```

Arguments

`siteID`	character
`markdown`	logical. Use markdown formating or console-friendly.

Examples

```
siteID <- "USGS-263819081585801"
site_metadata <- site_summary(siteID)
```

`stats_by_interval` *Create a frequency analysis based on custom interval*

Description

The table will accept daily, discrete, or a both types of data. The median of each year/month is calculated. Then using that median, monthly stats are calculated. Percentiles are calculated using the `quantile` function with "type=6".

Usage

```
stats_by_interval(
  interval,
  gw_level_dv,
  gwl_data,
  parameter_cd = NA,
  date_col = NA,
  value_col = NA,
  approved_col = NA,
  flip = FALSE
)
```

Arguments

gw_level_dv	data frame, daily groundwater level data. Often obtained from read_waterdata_daily. Use NULL for no daily data.
gwl_data	data frame returned from read_waterdata_field_measurements, or data frame with a date, value, and approval columns. Use NULL for no discrete data.
parameter_cd	Can be used to filter data if the data frame has a "parameter_code" column. The default is NA, which will not do any filtering. If the gwl_data and gw_level_dv need different parameter code filtering, use a vector of 2 parameter codes. The first one will filter the gw_level_dv data frame, the second will filter the gwl_data data frame.
date_col	the name of the time columns. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is c("time", "time").
value_col	the name of the value columns. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is c("value", "value").
approved_col	the name of the column to get provisional/approved status. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is c("approval_status", "approval_status"). It is expected that these columns will have only "Approved" or "Provisional".
flip	logical. If TRUE, flips labels so that the lower numbers are in the higher percentages. Default is TRUE.

Examples

```
site <- "USGS-263819081585801"
p_code_dv <- "62610"
statCd <- "00001"
# gw_level_dv <- dataRetrieval::read_waterdata_daily(monitoring_location_id = site,
#                                                    parameter_code = p_code_dv,
#                                                    statistic_id = statCd)

gw_level_dv <- L2701_example_data$Daily
monthly_frequency <- stats_by_interval("month",
```

```
                                        gw_level_dv,
                                        NULL)
head(monthly_frequency)
# gwl_data <- dataRetrieval::read_waterdata_field_measurements(monitoring_location_id = site)
gwl_data <- L2701_example_data$Discrete
gwl_data <- gwl_data[gwl_data$parameter_code == p_code_dv, ]

monthly_frequency2 <- stats_by_interval("month",
                                        gw_level_dv,
                                        gwl_data)

annual_frequency2 <- stats_by_interval("year",
                                        gw_level_dv,
                                        gwl_data)

week_frequency2 <- stats_by_interval("week",
                                     gw_level_dv,
                                     gwl_data)
```

---

theme_gwl *ggplot2 themes and accesories for package*

Description

Themes and other useful graphing utilities for the HASP package

Usage

```
theme_gwl(base_family = "", ...)

hasp_framework(
  x_label,
  y_label,
  plot_title,
  include_y_scale = FALSE,
  zero_on_top = TRUE,
  subtitle = "U.S. Geological Survey"
)
```

Arguments

base_family	character base font family
...	additional arguments
x_label	character. Label for x-axis.
y_label	character. Label for y-axis.
plot_title	character. Title for plot.

include_y_scale
: logical. If TRUE, include groundwater type style for y-axis. Default is FALSE.

zero_on_top
: logical. If zero_on_top is TRUE, there is no padding at the top of y axis. If FALSE, no padding at the bottom. If NA, padding on both top and bottom.

subtitle
: character. Sub-title for plot, default is "U.S. Geological Survey".

Examples

```
ggplot2::ggplot() + theme_gwl()



x_label <- "Date"
y_label <- "Level"
plot_title <- "Super Site"

ggplot2::ggplot() +
     hasp_framework(x_label,
                    y_label,
                    plot_title)
```

---

trend_plot *Chloride plot with trend*

Description

Function creates the cloride over time plot with trends.

Usage

```
trend_plot(
  qw_data,
  plot_title,
  y_label = NA,
  n_years = 10,
  POR_trend = TRUE,
  CharacteristicName = c("Chloride"),
  norm_range = c(230, 860),
  subtitle = "U.S. Geological Survey",
  include_table = TRUE
)
```

Arguments

qw_data
: data frame returned from readWQPqw must include columns ActivityStartDateTime, CharacteristicName, result_va

plot_title
: character title for plot

y_label	character label for y axis. If left as NA, the function will attempt to use the "variableInfo" attribute of qw_data. This is attached to dataRetrieval output.
n_years	integer. This is the number of years to calculate the trend on. Default is 10. This can be a vector of years.
POR_trend	a logical indicating whether to include a trend test for the full period of record. Default is TRUE.
CharacteristicName	character CharacteristicName to plot
norm_range	a numerical range to potentially group the data. If NA, no grouping is shown.
subtitle	character. Sub-title for plot, default is "U.S. Geological Survey".
include_table	logical whether or not to include the trend table in the upper left corner.

Examples

```
# site <- "USGS-263819081585801"
# parameterCd <- c("00095","90095","00940","99220")
# site_data <- dataRetrieval::readWQPqw(site,
#                                       parameterCd)
# Using package example data:
qw_data <- L2701_example_data$QW
plot_title <- paste(attr(qw_data, "siteInfo")[["station_nm"]], ": Chloride")
trend_plot(qw_data, plot_title)
```

---

trend_test *Trend Test*

Description

Test for period of record and user-specified ranges. The default trends are calculated for 10 year and the full period of record.

Usage

```
trend_test(
  gw_level_dv,
  gwl_data,
  n_years = 10,
  parameter_cd = NA,
  date_col = c("time", "time"),
  value_col = c("value", "value"),
  approved_col = c("approval_status", "approval_status"),
  pctComplete = 0.5,
  days_required_per_month = 14,
  POR_trend = TRUE
)
```

Arguments

gw_level_dv	data frame, daily groundwater level data. Often obtained from read_waterdata_daily. Use NULL for no daily data.
gwl_data	data frame returned from read_waterdata_field_measurements, or data frame with a date, value, and approval columns. Use NULL for no discrete data.
n_years	integer. This is the number of years to calculate the trend on. Default is 10. This can be a vector of years.
parameter_cd	Can be used to filter data if the data frame has a "parameter_code" column. The default is NA, which will not do any filtering. If the gwl_data and gw_level_dv need different parameter code filtering, use a vector of 2 parameter codes. The first one will filter the gw_level_dv data frame, the second will filter the gwl_data data frame.
date_col	the name of the time columns. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is c("time", "time").
value_col	the name of the value columns. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is c("value", "value").
approved_col	the name of the column to get provisional/approved status. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is c("approval_status", "approval_status"). It is expected that these columns will have only "Approved" or "Provisional".
pctComplete	number percentage complete. This is a fraction that represents the amount of data that must be included overall in order to calculate a trend. The default is 0.5, which means if gaps in the data span more than 50 total record, a trend will not be calculated.
days_required_per_month	integer. Number of days required per month to include in the trend test. Default is 14.
POR_trend	a logical indicating whether to include a trend test for the full period of record. Default is TRUE.

Details

For data that is at least on a daily interval, the rkt function is used. For periodic data, the kendallTrendTest is used.

Value

a data frame of test results from 5 and 20 year Kendall Seasonal Trend test

Examples

```
site <- "USGS-263819081585801"
p_code_dv <- "62610"
statCd <- "00001"
```

```
# Using package example data:
# gwl_data <- dataRetrieval::read_waterdata_field_measurements(monitoring_location_id = site)
gwl_data <- L2701_example_data$Discrete

# gw_level_dv <- dataRetrieval::read_waterdata_daily(monitoring_location_id = site,
#                                                    parameter_code = p_code_dv,
#                                                    statistic_id = statCd)

gw_level_dv <- L2701_example_data$Daily

trend_test(gw_level_dv,
           gwl_data,
           parameter_cd = "62610")

trend_test(gw_level_dv,
           gwl_data,
           POR_trend = FALSE,
           parameter_cd = "62610")

trend_test(gw_level_dv,
           gwl_data,
           parameter_cd = "62610",
           n_years = 5)

trend_test(gw_level_dv,
           gwl_data,
           parameter_cd = "62610",
           n_years = c(5, 10, 20))

# Only periodic data:
trend_test(NULL,
           gwl_data,
           parameter_cd = "62610")
```

`water_year` *Convert to water year*

Description

This function is a little more robust than `calcWaterYear`

Usage

```
water_year(x)
```

Arguments

x	character vector

Examples

```
x <- c("2010-01-01", "1994-02", "1980", "2009-11-01")
water_year(x)
```

`weekly_frequency_plot` *Plot weekly frequency plot*

Description

Weekly statistics are calculated using the `weekly_frequency_table` function. Daily, discrete, or both types of data can be used.

Usage

```
weekly_frequency_plot(
  gw_level_dv,
  gwl_data,
  parameter_cd = NA,
  date_col = c("time", "time"),
  value_col = c("value", "value"),
  approved_col = c("approval_status", "approval_status"),
  plot_range = "Past year",
  plot_title = "",
  subtitle = "U.S. Geological Survey",
  y_axis_label = "",
  flip = FALSE,
  percentile_colors = NA
)
```

Arguments

`gw_level_dv`	data frame, daily groundwater level data. Often obtained from `read_waterdata_daily`. Use `NULL` for no daily data.
`gwl_data`	data frame returned from `read_waterdata_field_measurements`, or data frame with a date, value, and approval columns. Use `NULL` for no discrete data.
`parameter_cd`	Can be used to filter data if the data frame has a "parameter_code" column. The default is `NA`, which will not do any filtering. If the gwl_data and gw_level_dv need different parameter code filtering, use a vector of 2 parameter codes. The first one will filter the gw_level_dv data frame, the second will filter the gwl_data data frame.
`date_col`	the name of the time columns. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is `c("time", "time")`.
`value_col`	the name of the value columns. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is `c("value", "value")`.

approved_col — the name of the column to get provisional/approved status. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is c("approval_status", "approval_status"). It is expected that these columns will have only "Approved" or "Provisional".

plot_range — the time frame to use for the plot. Use "Past year" (default) to see the last year of data, or "Calendar year" to use the current calendar year, beginning in January. Or specify two dates representing the start and end of the plot. If the first date is NA, it will start at the earliest record, if the second date is NA, it will end at the latest record.

plot_title — the title to use on the plot

subtitle — character. Sub-title for plot, default is "U.S. Geological Survey".

y_axis_label — the label used for the y-axis of the plot.

flip — logical. If TRUE, flips labels so that the lower numbers are in the higher percentages. Default is TRUE.

percentile_colors — Optional argument to provide a vector of 5 colors used to fill the percentile bars in order from lowest percentile bin to the highest percentile bin. Default behavior (NA) is to use legacy plot colors.

Value

a ggplot object with rectangles representing the historical weekly percentiles, and points representing the historical median and daily values

Examples

```
site <- "USGS-263819081585801"
p_code_dv <- "62610"
statCd <- "00001"

# gwl_data <- dataRetrieval::read_waterdata_field_measurements(monitoring_location_id = site)
gwl_data <- L2701_example_data$Discrete

# gw_level_dv <- dataRetrieval::read_waterdata_daily(monitoring_location_id = site,
#                                                    parameter_code = p_code_dv,
#                                                    statistic_id = statCd)

gw_level_dv <- L2701_example_data$Daily

weekly_frequency_plot(gw_level_dv,
                      gwl_data = NULL)

weekly_frequency_plot(gw_level_dv,
                      gwl_data = gwl_data,
                      parameter_cd = "62610",
                      plot_range = c("2023-10-01", "2024-09-30"))

weekly_frequency_plot(gw_level_dv,
                      gwl_data = gwl_data,
```

```
                      parameter_cd = "62610")

weekly_frequency_plot(gw_level_dv,
                      gwl_data = gwl_data,
                      parameter_cd = "62610",
                      flip = TRUE)
```

`weekly_frequency_table` *Create a table of weekly frequency analysis*

Description

The weekly frequency analysis is based on daily, discrete, or both types of data. The median of each year/week combo is calculated, then overall weekly statistics are calculated off of that median.

Usage

```
weekly_frequency_table(
  gw_level_dv,
  gwl_data,
  parameter_cd = NA,
  date_col = c("time", "time"),
  value_col = c("value", "value"),
  approved_col = c("approval_status", "approval_status"),
  flip = FALSE
)
```

Arguments

`gw_level_dv`	data frame, daily groundwater level data. Often obtained from `read_waterdata_daily`. Use `NULL` for no daily data.
`gwl_data`	data frame returned from `read_waterdata_field_measurements`, or data frame with a date, value, and approval columns. Use `NULL` for no discrete data.
`parameter_cd`	Can be used to filter data if the data frame has a "parameter_code" column. The default is `NA`, which will not do any filtering. If the gwl_data and gw_level_dv need different parameter code filtering, use a vector of 2 parameter codes. The first one will filter the gw_level_dv data frame, the second will filter the gwl_data data frame.
`date_col`	the name of the time columns. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is `c("time", "time")`.
`value_col`	the name of the value columns. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is `c("value", "value")`.

approved_col	the name of the column to get provisional/approved status. The first value is associated with the gw_level_dv input, and the second value is associated with the gwl_data input. The default is c("approval_status", "approval_status"). It is expected that these columns will have only "Approved" or "Provisional".
flip	logical. If TRUE, flips labels so that the lower numbers are in the higher percentages. Default is TRUE.

Value

a data frame of weekly frequency analysis

Examples

```
site <- "USGS-263819081585801"
p_code_dv <- "62610"
statCd <- "00001"
# gw_level_dv <- dataRetrieval::read_waterdata_daily(monitoring_location_id = site,
#                                                    parameter_code = p_code_dv,
#                                                    statistic_id = statCd)

gw_level_dv <- L2701_example_data$Daily

weekly_frequency <- weekly_frequency_table(gw_level_dv, NULL)
head(weekly_frequency)

# gwl_data <- dataRetrieval::read_waterdata_field_measurements(monitoring_location_id = site)
gwl_data <- L2701_example_data$Discrete

weekly_frequency <- weekly_frequency_table(gw_level_dv,
                                           gwl_data,
                                           parameter_cd = "62610")
head(weekly_frequency)
weekly_flip <- weekly_frequency_table(gw_level_dv,
                                      gwl_data,
                                      parameter_cd = "62610",
                                      flip = TRUE)
head(weekly_flip)
```

Index

∗ **datasets**
    nat_aqfr_state, 26
∗ **data**
    aquifer_data, 3
∗ **quality**
    aquifer_data, 3
∗ **water**
    aquifer_data, 3

aquifer_data, 3

calcWaterYear, *38*
composite_data, 3
create_groundwater_report, 4
create_targets_script, 5

daily_frequency_table, 6
daily_gwl_plot, 7
daily_gwl_summary, 9
data_available, 10

explore_aquifers, 11
explore_site *(*explore_aquifers*)*, 11

filter_sites, 12
first_day, 13

get_aquifer_data, 13
get_range, 14
get_state_data, 15
gwl_plot_all *(*gwl_plot_field*)*, 15
gwl_plot_field, 15

hasp_framework *(*theme_gwl*)*, 34

kendallTrendTest, *37*

L2701_example_data, 18
last_day, 19
local_aqfr *(*nat_aqfr_state*)*, 26

map_hydro_data, 19
markerTable *(*nat_aqfr_state*)*, 26
markerTable2 *(*nat_aqfr_state*)*, 26
mid_month, 20
monthly_frequency_plot, 21
monthly_frequency_table, 24
monthly_mean, 26

nat_aqfr_state, 26
normalized_data, 27

plot_composite_data, 28
plot_normalized_data, 29

qw_plot *(*Sc_Cl_plot*)*, 30
qw_summary *(*Sc_Cl_plot*)*, 30

read_waterdata_daily, *6*, *8*, *9*, *16*, *21*, *24*, *33*, *37*, *39*, *41*
read_waterdata_field_measurements, *6*, *8*, *9*, *16*, *21*, *24*, *33*, *37*, *39*, *41*
read_waterdata_monitoring_location, *19*
read_waterdata_parameter_codes, *16*
readWQPqw, *30*, *35*
rkt, *37*

Sc_Cl_plot, 30
Sc_Cl_table *(*Sc_Cl_plot*)*, 30
site_data_summary, 31
site_info *(*aquifer_data*)*, 3
site_summary, 32
stats_by_interval, 32
summary_aquifers *(*nat_aqfr_state*)*, 26

theme_gwl, 34
trend_plot, 35
trend_test, 36

water_year, 38
weekly_frequency_plot, 39
weekly_frequency_table, 41