Package 'SWTools'

May 7, 2026

Type Package

Title Helper Tools for Australian Hydrologists

Version 1.1.0

Author Matt Gibbs [aut, cre]

Maintainer Matt Gibbs <gibbs.ms@gmail.com>

URL https://github.com/matt-s-gibbs/SWTools

BugReports https://github.com/matt-s-gibbs/swtools/issues

Description Functions to speed up work flow for hydrological analysis.
Focused on Australian climate data (SILO climate data), hydrological models (eWater Source) and in particular South Australia (<https://water.data.sa.gov.au> hydrological data).

Depends R (>= 4.0.0)

License GPL-3

Encoding UTF-8

Imports httr, zoo, dplyr, tidyr, readr, tibble, forcats, stringr, rlang, rmarkdown, ggplot2, ggpubr, ggspatial, prettymapr, magrittr, hydroTSM, jsonlite, sf, lubridate, segmented

RoxygenNote 7.3.2

NeedsCompilation no

Repository CRAN

Date/Publication 2024-10-14 06:30:02 UTC

Contents

SWTools-package 2
AQWPDownload 3
AQWPLoad 5
HydstraSiteDetails 6
read_res.csv 7
SILOCheckConsistency 8

SILOCorrectSite . . . 9
SILOCumulativeDeviation . . . 11
SILODoubleMass . . . 12
SILODownload . . . 12
SILOImport . . . 14
SILOLoad . . . 15
SILOMap . . . 16
SILOMonthlyRainfall . . . 16
SILOMortonQualityCodes . . . 17
SILOQualityCodes . . . 18
SILOReport . . . 19
SILOSitesfromPolygon . . . 20
SILOSiteSummary . . . 20
SILOThiessenShp . . . 21
SILOWriteforSource . . . 22
SILOWriteFunctionsforSource . . . 23
VeneerGetInputSets . . . 25
VeneerGetNodesbyType . . . 25
VeneerGetPiecewise . . . 26
VeneerGetTS . . . 27
VeneerGetTSbyNode . . . 28
VeneerGetTSbyVariable . . . 28
VeneerGetTSVariables . . . 29
VeneerlatestRunNumber . . . 30
VeneerRunSource . . . 30
VeneerSetFunction . . . 31
VeneerSetPiecewise . . . 32
WritepwtoIS . . . 33

Index **34**

`SWTools-package` *SWTools: Helper Tools for Australian Hydrologists*

Description

Functions to speed up workflow for hydrological analysis. Focused on Australian climate data (SILO climate data), hydrological models (eWater Source) and South Australian hydrological data (from Water Data SA).

SILO functions

SILO is a database of Australian climate data from 1889 to the present. It provides daily meteorological datasets for a range of climate variables in ready-to-use formats suitable for biophysical modelling, research and climate applications SILO Website.

These functions allow SILO data to be downloaded from the SILO Website, imported into R, calculate some basic statistics and undertake some quality assurance tests to easily visualise how

much data has been interpolated, and to compare nearby sites to identify potential data issues. SILODownload,SILOLoad and SILOReport functions allow a vector of SILO sites to be downloaded and summarised in a Microsoft Word report.

Source and Veneer functions

eWater Source is the Australia's national hydrological modelling platform, and is increasing in use around the world. Functions are included to write SILO climate data to the format expected for Source SILOWriteforSource, and reading in model outputs, read_res.csv.

Veneer is a RESTful API for interacting with Source models. Functions are included that are wrappers for Veneer, to build URLs to get or set data in the Source model, and process the json object returned.

Aquarius functions

South Australia's hydrological data is hosted on Water Data SA. The Export link creates URLs that enable multiple datasets to be downloaded. AQWPDownload builds these URLs to download data in json format, and AQWPLoad loads this json file into the R interface.

Hydstra functions

New South Wales, Queensland and Victoria use a Hydstra database, with the site information available over a Kisters API. HydstraSiteDetails will use this API to download streamflow site information, such as the cross section, rating curve, gaugings as well as the daily (9am-9am) time series of discharge data.

Author(s)

Maintainer: Matt Gibbs <gibbs.ms@gmail.com>

See Also

Useful links:

- https://github.com/matt-s-gibbs/SWTools
- Report bugs at https://github.com/matt-s-gibbs/swtools/issues

---

AQWPDownload *Function to download data from https://water.data.sa.gov.au*

Description

For most inputs, valid options will be returned if an unexpected input is provided. The exception are **Location** and **Dataset**, if the location, or dataset for that location, don't exist no data will be returned. Browse the Export tab on https://water.data.sa.gov.au to find **Location** and **Dataset** that exists.

Usage

```
AQWPDownload(
  Location,
  Dataset,
  Unit,
  file = "AQWP.json",
  Interval = "Daily",
  Calculation = "Aggregate",
  Calendar = "CALENDARYEAR",
  Step = 1,
  DateRange = "EntirePeriodOfRecord",
  StartTime = NULL,
  EndTime = NULL
)
```

Arguments

Location	A string or vector of strings, with site numbers, e.g. "A4261001"
Dataset	A string or vector of strings, with dataset names, as expected by AWQP, e.g. "Tide Height.Best Available–Continuous"
Unit	A string or vector of strings, with units, e.g. "Metres" or "mg/L". If only 1 is string is provided it will be used for each site in Location
file	Location and name of json file to download. Defaults to "AQWP.json".
Interval	Interval of output, e.g. "PointsAsRecorded", or "Daily"
Calculation	For larger intervals, what calculation to do, e.g. "Aggregate" (average) or "Maximum"
Calendar	When to start the periods, e.g. "WATERDAY9AM"
Step	How many intervals e.g. 15 with Interval="Minutely" returns 15 minute data.
DateRange	Period of data to return, e.g. "EntirePeriodOfRecord" or "Custom". "Years1" seems to not work on AWQP.
StartTime	Start Date and Time if DateRange="Custom", in a format that as.POSIXct will convert, e.g 2000-01-01 00:00
EndTime	End Date and Time if DateRange="Custom", in a format that as.POSIXct will convert, e.g 2001-01-02 00:00

Value

The lnk created to download the data, which is useful for debugging. The data is saved to "file", that can then be read in with AQWPLoad()

Examples

```
## Not run:
Location=c("A4260633","A4261209","A4260572")
Dataset=rep("Tide Height.Best Available--Continuous",3)
Unit=rep("Metres",3)
```

```
S="2020-01-01 00:00"
E="2020-01-02 00:00"
AQWPDownload(Location,Dataset,Unit,DateRange="Custom",
StartTime=S,EndTime=E,file=tempfile("AQWP",fileext=".json"))

## End(Not run)
```

AQWPLoad *Function to load in an Aquarius json file, downloaded from https://water.data.sa.gov.au, possibly using AWQPDownload()*

Description

Function to load in an Aquarius json file, downloaded from https://water.data.sa.gov.au, possibly using AWQPDownload()

Usage

```
AQWPLoad(filename, qual_codes = TRUE, long_format = TRUE)
```

Arguments

filename	A file downloaded from the Export Data tab on https://water.data.sa.gov.au, or using AQWPDownload()
qual_codes	TRUE/FALSE to return quality codes. Defaults to true
long_format	TRUE/FALSE to return data in long format, rather than wide (e.g. a spreadsheet). Long is useful for plotting with ggplot

Value

A tibble with the data in the file

Examples

```
## Not run:
AQWPLoad("AQWP.json")

## End(Not run)
```

`HydstraSiteDetails` *Get streamflow station information from Hydstra API*

Description

This function use the API associated with states' Hydstra databases to return useful site information.

Usage

```
HydstraSiteDetails(site, state, out_folder, flood_level = NA)
```

Arguments

`site`	station number, e.g. "425018"
`state`	relevant state database foro the station, e.g. "NSW"
`out_folder`	path to folder to save outputs
`flood_level`	optional, water level in stage datum to plot on the cross section data

Details

Currently, the relevant websites are useful for site discovery:

- https://water-monitoring.information.qld.gov.au
- https://realtimedata.waternsw.com.au
- https://data.water.vic.gov.au/WMIS

The Bureau of Meteorology's Water Data Online site is also useful, which can also be queried using `get_station_list()` from BomWater package.

The function will save a number of files to `out_folder` that have a file name starting with the station number followed by:

- site_info.csv: general site information returned, e.g. site name, coordinates, length of record, elevation
- x_sec.csv: chainage `chain` and elevation (`rl` in gauge datum) of the control section
- rating.csv: current rating curve, the discharge (`vf` in ML/d) for a gauge height `vf`, and also as above the cease to flow level (`above_ctf`)
- gaugings.csv: record of streamflow gaugings available
- discharge.csv: daily time series of streamflow, over a 9am - 9am period
- plot.png: summary plot of the above data

Quality codes shown on the plot are those used by the Bureau of Meteorology defined here

Value

a vector of length 3, with the number of cross sections, rating curves and streamflow gaugings found, respectively.

Examples

```
## Not run:
HydstraSiteDetails("425018","NSW","c:/Temp")

## End(Not run)
```

`read_res.csv` *Function to import a Source .res.csv File Returns data (as a Data Frame, Zoo, or tibble) as a time Series with all Results Read Source .res.csv file into a data table or zoo time series*

Description

Function to import a Source .res.csv File Returns data (as a Data Frame, Zoo, or tibble) as a time Series with all Results Read Source .res.csv file into a data table or zoo time series

Usage

```
read_res.csv(resFile, returnType = "df")
```

Arguments

`resFile`	A character string representing the full file path of the .res.csv file
`returnType`	A character string to set the return type: "z", "t", "df". If not matching "t" (tibble) or "z" (zoo), data frame returned.

Value

Data in the format selected with all data read in from the Source .res.csv file

Examples

```
## Not run:
X = read_res.csv("./SWTools/extdata/Scenario1.res.csv",returnType="t")

## End(Not run)
```

`SILOCheckConsistency` *Check for homogeneity between SILO rainfall station data*

Description

Compute tests on rainfall double mass curves and cumulative deviation in annual rainfall totals to test for consistency between a rainfall station and the average of another group of stations. Non-homogeneity can occur for a number of reasons, such as interception from vegetation or buildings over time, moving of a station location, or due to interpolation of missing data or station closure

Usage

```
SILOCheckConsistency(X, folder = NA, pvallim = 0.05, changelim = 0.025)
```

Arguments

`X`	A list of SILO station data, in the format created by `SILOLoad`
`folder`	Path to folder to save resulting images to. Will be created if it doesn't exist
`pvallim`	p value limit of the break point detection to display the double mass break point. Defaults to p=0.05
`changelim`	significant slope limit display the double mass break point. Defaults to a slope change of 0.025

Details

Two tests are calculated by `SILOCheckConsistency`, which are outlined in Annex 4 of Allen et al. (1998).

The first considers the residual errors in annual rainfall at a station, compared to the straight line (intercept=0) regression with the average annual rainfall from the other sites in `X`. The residuals should follow a normal distribution with mean zero and standard deviation s_y,x. The annual rainfall data is plotted to visually assess the homoscedasticity requirement (constant variance). Ellipses for 80\

The second test tests for a break point in the plot of cumulative annual rainfall, commonly referred to as a double-mass analysis. This analysis is outlined in Allen et al. (1998) and also Chang and Lee (1974). A bootstraped estimate of any breakpoint in the double-mass plot, indicating a change in the relationship between rainfall at the station and the average of all other stations, is assessed using the method of Muggeo (2003), as provided in segmented.

Value

If folder is not specified (or `NA`) the plots are shown in the R environment. If folder is specified, a figure for each station in `X` is saved to `folder`. There are 4 panels on the figure:

- Annual rainfall for a given station, against the average across all stations in X (except the station presented).

- Cumulative residuals of the annual rainfall from the straight line regression shown in the first panel. Assuming the residuals are are independent random variables, this figure include ellipses representing 80th and 95th percentile that the hypothesis that there is no change in slope can be rejected.
- double mass curve, plotting the cumulative annual rainfall for the station against the station average. If a breakpoint is identified, this is displayed on the plot.The colours represent the median quality code for each year, with the same colour palette as SILOQualityCodes
- Residuals of the cumulative rainfall from the straight line fitted to the double mass curve.

References

Chang, M., and Lee, R. (1974) Objective double-mass analysis, Water Resour. Res., 10(6), 1123-1126, doi:10.1029/WR010i006p01123.

Allan, R., Pereira, L. and Smith, M. (1998) Crop evapotranspiration-Guidelines for computing crop water requirements-FAO Irrigation and drainage paper 56.

Muggeo, V.M.R. (2003) Estimating regression models with unknown break-points. Statistics in Medicine 22, 3055-3071.

See Also

SILOLoad, SILOSiteSummary, SILOQualityCodes,SILOCorrectSite

Examples

```
## Not run:
X<-SILOLoad(c("24001","24002","24003"),path="./SWTools/extdata")
SILOCheckConsistency(X,tempdir())

## End(Not run)
```

SILOCorrectSite *Correct a slope change in a rainfall data set based on another site*

Description

If the break point of a non-homogenous rainfall station has been identified (potentially using SILOCheck-Consistency), correct the data on one side of the breakpoint

Usage

```
SILOCorrectSite(
  X,
  s_correct,
  s_reference,
  year_break,
  year_start = NULL,
  year_end = NULL,
```

```
  after = TRUE,
  plot = NA
)
```

Arguments

X	A list of SILO station data, in the format created by SILOLoad
s_correct	Station number that exists in X to correct
s_reference	Station number that exists in X to used for the correction
year_break	year in the time series that the break points exists
year_start	first year of data (before year_break) to to develop the first linear regression between s_correct and s_reference. Defaults to the start of the dataset
year_end	last year of data (after year_break) to to develop the second linear regression between s_correct and s_reference. Defaults to the end of the dataset
after	TRUE/FALSE value, indicating if the homogeneous data to develop the relationship to correct the non-homogeneous data is after the breakpoint (TRUE) or before (FALSE).
plot	if specified, the file (including path if necessary) to save a scatter plot of the annual rainfall totals, including regression equations used to correct the non-homogeneous data.

Details

The method of cumulative residuals outlined in Annex 4 of Allen et al. (1998) has been used. That is, two linear regressions between the annual rainfall totals are calculated P_s_correct~P_s_reference over the periods year_start:year_break and year_break:year_end For the period to correct (after the breakpoint if after=TRUE) an annual scaling factor is calculated from the ratio of the predicted rainfall total from the two regression equations, based on the rainfall total for each year at the reference site. This scaling factor is then applied to the daily rainfall data for that year.

Value

A list with the same structure as X, with the element for s_correct updated with the corrections on one side of the breakpoint year.

References

Chang, M., and Lee, R. (1974) Objective double-mass analysis, Water Resour. Res., 10(6), 1123–1126, doi:10.1029/WR010i006p01123.

See Also

SILOLoad, SILOSiteSummary, SILOQualityCodes,SILOCorrectSite

Examples

```
## Not run:
stations<-c("23313","23302","23300","23317","23725","23705")
SILODownload(stations)
X<-SILOLoad(stations,startdate="1891-01-01",enddate="2020-12-31")
X<-SILOCorrectSite(X,"23313","23705",1970,after=FALSE)

## End(Not run)
```

`SILOCumulativeDeviation` *Plot the cumulative deviation from the mean for each silo station on one plot*

Description

Plot the cumulative deviation from the mean for each silo station on one plot

Usage

```
SILOCumulativeDeviation(SILO, filename = NULL, cols = pkg.env$cols)
```

Arguments

`SILO`	a list of sites with SILO data, as created by SILOLoad()
`filename`	optional, filename to write the plot to, including extension (e.g. .png). Filename can include full path or sub folders.
`cols`	optional, a vector of colours to use for the plotting

Value

a ggplot plot of the cumulative deviation from the mean.

Examples

```
## Not run:
X<-SILOLoad(c("24001","24002","24003"),path="./SWTools/extdata")
p<-SILOCumulativeDeviation(X)

## End(Not run)
```

SILODoubleMass *Plot double mass curves of each rainfall site against each other*

Description

Plot double mass curves of each rainfall site against each other

Usage

```
SILODoubleMass(SILO, filename = NULL, plotsperpage = 4)
```

Arguments

SILO	a list of sites with SILO data, as created by SILOLoad()
filename	optional, filename to write the plot to, including extension. Filename can include full path or sub folders.
plotsperpage	optional, number of plots to output per element of the list returned. Defaults to 4

Value

a list of ggplot objects that plot of the double mass curves of each station in the SILO list against each other. The double mass plot is on the bottom diagonal, and the slope of the line for each case in the upper diagonal. Each list element contains plotsperpage (default to 4) double mass plots, to allow them to be plotted on multiple pages

Examples

```
## Not run:
X<-SILOLoad(c("24001","24002","24003"),path="./SWTools/extdata")
p<-SILODoubleMass(X)

## End(Not run)
```

SILODownload *Download SILO data*

Description

Download SILO data

Usage

```
SILODownload(
  SiteList,
  username = "noemail@net.com",
  password = "gui",
  path = getwd(),
  startdate = "18890101",
  enddate = NULL,
  ssl = FALSE
)
```

Arguments

SiteList	A station number or vector of station numbers, as a string (e.g. "24001")
username	SILO user name. Defaults to credentials used by https://www.longpaddock.qld.gov.au/silo/point-data/
password	SILO password
path	Where to save the output. Will default to getwd() if not specified
startdate	First day of data, in the format "YYYYMMDD". Will default to the first day of the record "18890101" if not specified
enddate	Last day of data, in the format "YYYYMMDD". Will default to yesterday if not specified
ssl	if true set ssl_cipher_list to "RC4-SHA" for file download. Seems to be necessary on some machines. default to FALSE

Value

A file for each station will be saved to path, named station number.txt. Nothing is returned to the R environment.

Examples

```
## Not run:
SILODownload(c("24001","24002","24003"),
path=tempdir(),
startdate="20180101",enddate="20200101")

## End(Not run)
```

`SILOImport` *Import a SILO file*

Description

Import a SILO file

Usage

```
SILOImport(station, path, startdate, enddate)
```

Arguments

`station`	Station number (e.g. "24001") to import. The function expects the file to be called "24001.txt".
`path`	Location where the file is located. Use "/" or "\\" for folders. Defaults to getwd() if not specified.
`startdate`	Start date of data to load, in format "YYYY-MM-DD". Defaults to start of the file if not provided
`enddate`	End date of data to load, in format "YYYY-MM-DD". Defaults to end of the file if not provided

Value

a list of data from the file, with members:

tsd the raw data as a daily zoo object

Site the name of the site

Station the station number

Lon Longitude

Lat Latitude

start the first date with good quality rainfall data

end the last date with good quality rainfall data

goodpct the percentage of good quality coded rainfall data between start and end

SILOLoad *Import multiple SILO files*

Description

Import multiple SILO files

Usage

```
SILOLoad(sites, path = getwd(), startdate, enddate)
```

Arguments

sites	a vector of Station numbers (e.g. c("24001","24002","24003")) to import. The function expects the file to be called "24001.txt".
path	Location where the file is located. Use "/" or "\\" for folders. Defaults to getwd() if not specified.
startdate	Start date of data to load, in format "YYYY-MM-DD". Defaults to start of the file if not provided
enddate	End date of data to load, in format "YYYY-MM-DD". Defaults to end of the file if not provided

Value

a list of data from the file, with members:

tsd the raw data as a daily zoo object

Site the name of the site

Station the station number

Lon Longitude

Lat Latitude

start the first date with good quality rainfall data

end the last date with good quality rainfall data

goodpct the percentage of good quality coded rainfall data between start and end

Examples

```
## Not run:
X<-SILOLoad(c("24001","24002","24003"),path="./SWTools/extdata")

## End(Not run)
```

SILOMap *Plot a map of the SILO station locations*

Description

Plot a map of the SILO station locations

Usage

```
SILOMap(SILO, filename = NULL)
```

Arguments

SILO	a list of sites with SILO data, as created by SILOLoad()
filename	optional, filename to write the plot to, including extension. Filename can include full path or sub folders.

Value

a google map of the SILO station locations

Examples

```
## Not run:
X<-SILOLoad(c("24001","24002","24003"),path="./SWTools/extdata")
p<-SILOMap(X)

## End(Not run)
```

SILOMonthlyRainfall *Plot a boxplot of monthly rainfall with mean monthly evaporation*

Description

Plot a boxplot of monthly rainfall with mean monthly evaporation

Usage

```
SILOMonthlyRainfall(
  SILO,
  evapcol = "Mwet",
  filename = NULL,
  cols = pkg.env$cols
)
```

Arguments

SILO	a list of sites with SILO data, as created by SILOLoad()
evapcol	name of an evaporation column to plot, defaults to "MWet".
filename	optional, filename to write the plot to, including extension. Filename can include full path or sub folders.
cols	optional, a vector of colours to use for the plotting

Value

a ggplot of the monthly rainfall and evaporation.

Examples

```
## Not run:
X<-SILOLoad(c("24001","24002","24003"),path="./SWTools/extdata")
p<-SILOMonthlyRainfall(X,"Span",cols=c("black","red","#124734"))

## End(Not run)
```

SILOMortonQualityCodes

Plot the quality codes of the input data for Morton's Evap calculations

---

Description

Produces a tile plot displaying the quality codes for variables that are input to the calculation of Morton's evaporation equations, being maximum and minimum temperature, solar radiation and vapor pressure (derived from wet bulb temperature). Evaporation is also plotted, if the site has pan observations.

Usage

```
SILOMortonQualityCodes(SILO, filename = NULL)
```

Arguments

SILO	a list of sites with SILO data, as created by SILOLoad()
filename	optional, filename to write a plot of the rainfall quality codes to, including extension (e.g. png). Filename can include full path or sub folders.

Value

a ggplot geom_tile plot of the rainfall quality codes

Examples

```
## Not run:
X<-SILOLoad(c("24001","24002","24003"),path="./SWTools/extdata")
p<-SILOMortonQualityCodes(X)

## End(Not run)
```

`SILOQualityCodes` *Plot the quality codes of the SILO rainfall data*

Description

Plot the quality codes of the SILO rainfall data

Usage

```
SILOQualityCodes(SILO, filename = NULL)
```

Arguments

`SILO`	a list of sites with SILO data, as created by SILOLoad()
`filename`	optional, filename to save a plot of the rainfall quality codes to, including extension (e.g. .png). Filename can include full path or sub folders.

Value

a ggplot geom_tile plot of the rainfall quality codes

Examples

```
## Not run:
X<-SILOLoad(c("24001","24002","24003"),path="./SWTools/extdata")
p<-SILOQualityCodes(X)

## End(Not run)
```

SILOReport *Write SILO data report to word document. The report includes output from SILOSiteSummary(), SILOQualityCodes(), SILOMortonQualityCodes(). SILOMap(), SILOMonthlyRainfall(), SILOCumulativeDeviation() and SILODoubleMass().*

Description

Write SILO data report to word document. The report includes output from SILOSiteSummary(), SILOQualityCodes(), SILOMortonQualityCodes(). SILOMap(), SILOMonthlyRainfall(), SILOCumulativeDeviation() and SILODoubleMass().

Usage

```
SILOReport(SILO, filename, path = getwd(), cols = pkg.env$cols)
```

Arguments

SILO	a list of sites with SILO data, as created by SILOLoad()
filename	filename to write the report to.
path	Optional. Folder to save the report to, defaults to current working directory
cols	Optional. vector of colours to use for the monthly rainfall and cumulative deviation plots. Must be at least as long as the number of sites in the SILO list.

Value

Nothing to the environment. A word document report is written to "filename".

Examples

```
## Not run:
X<-SILOLoad(c("24001","24002","24003"),path="./SWTools/extdata")
SILOReport(X,"MyReport.docx") #requires pandoc installed

## End(Not run)
```

SILOSitesfromPolygon *Find SILO sites within a polygon*

Description

Find SILO sites within a polygon

Usage

```
SILOSitesfromPolygon(shpFile, ssl = FALSE, buffer = 0)
```

Arguments

shpFile	location to a shapefile to search within for SILO sites
ssl	See SILODownload, if true if true sets ssl_cipher_list="RC4-SHA" for httr::GET()
buffer	distance in km to buffer the shapefile to look for sites outside the catchment The buffer distance is approximate for a couple of reasons: the shapefile is projected to match SILO site coordinates, WGS84 and sf::st_buffer does not correctly buffer longitude/latitude data. sdaf Also the input distance in km is converted to degrees using the conversion at the equator of 0.008.

Value

a table of site information including site numbers found within the polygon

Examples

```
## Not run:
Sites=SILOSitesfromPolygon("path/to/shapefile.shp")
SILODownload(Sites$Number,
path=tempdir(),
startdate="20180101",enddate="20200101")
X<-SILOLoad(Sites$Number,path=tempdir())

## End(Not run)
```

SILOSiteSummary *Produce a table summarising SILO sites*

Description

Produce a table summarising SILO sites

Usage

```
SILOSiteSummary(SILO)
```

Arguments

SILO	a list of sites with SILO data, as created by SILOLoad()

Value

a dataframe with the following columns

Site site name

Station station number

StartDate date of the first good quality rainfall data

EndDate date of the last good quality rainfall data

PctMissing percentage of days that do not have good quality code between StartDate and EndDate

AnnualRainfall Mean annual rainfall in mm

Latitute Latitude

Longitude Longitude

Elevation Elevation

Examples

```
## Not run:
X<-SILOLoad(c("24001","24002","24003"),path="./SWTools/extdata")
d<-SILOSiteSummary(X)

## End(Not run)
```

SILOThiessenShp *Function to generate Thiessen polygons from SILO sites*

Description

Function to generate Thiessen polygons from SILO sites

Usage

```
SILOThiessenShp(SILOdata, path, shpname, boundary = NULL)
```

Arguments

SILOdata	• data loaded from SILO based on site list
path	• file path to save Thiessen polygon shapefile
shpname	• filename to save ESRI shapefile (no extension)
boundary	• optional either a filename(including path) of a boundary, e.g. catchment boundary, to apply, or the boundary as a sfc_MULTIPOLYGON object

Value

A simple feature geometry (sf::sfc object) of the polgyons created. Shape file saved to path \ shpname

If boundary is specified weights are written to the attribute table of the polygon return, which can be extracted with `st_drop_geometry(returnedfeature[c("Station","weights")])`

Examples

```
## Not run:
X<-SILOLoad(c("24001","24002","24003"),path="./SWTools/extdata")
p<-SILOThiessenShp(X,tempdir(),"Theissens")
a<-SILOSiteSummary(X)
ggplot(p)+geom_sf(aes(fill=AnnualRainfall))+
geom_point(data=a,aes(Longitude,Latitude))+
geom_text(data=a,aes(Longitude,Latitude,label=Site),nudge_y = 0.02)

## End(Not run)
```

SILOWriteforSource *Write a SILO time series to a csv file in the format expected by eWater Source*

Description

Write a SILO time series to a csv file in the format expected by eWater Source

Usage

```
SILOWriteforSource(SILO, col, filename, scalefactor = 1)
```

Arguments

SILO	a list of sites with SILO data, as created by SILOLoad()
col	Name of a column in a silo file to write out, e.g. Rain
filename	file to write to.
scalefactor	factor to scale the data by. Defaults to 1. Useful for Pan evap or rainfall scaling. Could also be a vector, with a value for each station in SILO

Value

Nothing to the R environment. SILO data is written to "filename".

Examples

```
## Not run:
X<-SILOLoad(c("24001","24002","24003"),path="./SWTools/extdata")
SILOWriteforSource(X,"Rain",tempfile("Rainfall",fileext=".csv"))

## End(Not run)
```

SILOWriteFunctionsforSource
Function to bulk create functions for SILO data in Source.

Description

Function to bulk create functions for SILO data in Source.

Usage

```
SILOWriteFunctionsforSource(
  X,
  boundary,
  shpColumn,
  functionsfile,
  RRfile,
  RainfallDatasourcesFolder,
  PETDatasourcesFolder,
  RainfallDatafile,
  PETDatafile,
  fus
)
```

Arguments

X	List of SILO station data, loaded into R using SILOLoad.
boundary	path to a subcatchment shapefile containing the subcatchments in the Source catchment model
shpColumn	column in the shapefile attribute table that corresponds to the catchment numbering.
functionsfile	filename to create with functions to import into Source
RRfile	filename to create to be imported into the Source Rainfall Runoff feature table
RainfallDatasourcesFolder	Name to use when creating a folder in the Source function editor for the rainfall functions and time series variables
PETDatasourcesFolder	Name to use when creating a folder in the Source function editor for the PET functions and time series variables

`RainfallDatafile`
: Filename of data source loaded in Source for rainfall, in formatting used by Source (e.g. for a file called Rain.csv from a relative folder called TimeseriesData is TimeSeriesData_Rain_csv).

`PETDatafile`
: Filename of data source loaded in Source for PET, in formatting used by Source

`fus`
: character vector of function unit names in the model.

  It is assumed that the Source rainfall-runoff scenario was created using the Geographic wizard, using the 'draw network' method (as opposed to DEM based) This allows a raster to be loaded into Source, with an integer in each cell representing the different subcatchments. `boundary` should be the path to a polygon shapefile with these catchment boundaries, and an attribute column that represent the catchment numbers. Typically this shapefile will be used to generate the raster that Source requires.

  This function will create two files:

  - `functionsfile` one to be imported into Source using the Function Import/Export plugin. The functions in the file that will be imported are a time series function for each SILO site in `X` and a function weighting these SILO sites using Thiessen polygon areas for each subcatchment in `boundary`
  - `RRFile` This file points each subcatchment and functional unit to the relevant function created for rainfall and PET, to be Imported in the Rainfall Runoff feature table (Edit-Rainfall Runoff Models and Import button)

Value

Nothing to the R environment. Files `functionsfile` and `RRfile` are created.

Examples

```
## Not run:
X<-SILOLoad(sites)
shpColumn<-"OBJECTID"
functionsfile<-"functions.csv"
RRfile<-"RRFile.csv"
RainfallDatasourcesFolder<-"Rainfall"
PETDatasourcesFolder<-"PET"
RainfallDatafile<-"TimeSeriesData_Rain_csv"
PETDatafile<-"TimeSeriesData_MWet_csv"
fus<-c("regolith","igneous","carbonate","sedimentary")
SILOWriteFunctionsforSource(X,boundary,shpColumn,functionsfile,RRfile,
                            RainfallDatasourcesFolder,PETDatasourcesFolder,
                           RainfallDatafile,PETDatafile,fus)

## End(Not run)
```

VeneerGetInputSets *Get vector of InputSets*

Description

Get vector of InputSets

Usage

```
VeneerGetInputSets(baseURL = "http://localhost:9876")
```

Arguments

baseURL	URL of the Veneer server. Defaults to the veneer default.

Value

vector containing info on Input Sets in the model

Examples

```
## Not run:
VeneerGetInputSets()

## End(Not run)
```

VeneerGetNodesbyType *Get a vector of node names for a given type*

Description

Get a vector of node names for a given type

Usage

```
VeneerGetNodesbyType(NodeType, baseURL = "http://localhost:9876")
```

Arguments

NodeType	The node to return the names of. The icon in /network is searched for this name
baseURL	URL of the Veneer server. Defaults to the veneer default.

Value

vector of node names matching the specified node type

Examples

```
## Not run:
VeneerGetNodesbyType("Weir")

## End(Not run)
```

`VeneerGetPiecewise` *Get data from a Source piecewise table using Veneer*

Description

Get data from a Source piecewise table using Veneer

Usage

```
VeneerGetPiecewise(pw_table, baseURL = "http://localhost:9876")
```

Arguments

`pw_table`	The name of the piecewise linear variable, without the $
`baseURL`	URL of the Veneer server. Defaults to the veneer default.

Value

a matrix with the data from the piecewise table.

Examples

```
## Not run:
VeneerGetPiecewise(data,"pw_table")

## End(Not run)
```

VeneerGetTS *Get a time series result from Source using Veneer*

Description

Get a time series result from Source using Veneer

Usage

```
VeneerGetTS(TSURL, baseURL = "http://localhost:9876")
```

Arguments

TSURL	the URL of the time series to retrieve
baseURL	URL of the Veneer server. Defaults to the veneer default.

Value

a zoo time series of the data

The URL of the time series must be specified, by interrogation using a browser or other analysis. By default Source returns SI units. Some conversion is undertaken:

- Flow converted to ML/d
- Volume converted to ML
- Area converted to ha

Spaces are OK, like in the example below (dont need to insert %20 for example).

Examples

```
## Not run:
VeneerGetTS("/runs/latest/location/EndofSystem/element/Downstream Flow/variable/Flow")

## End(Not run)
```

VeneerGetTSbyNode *Get all time series recorded in Source for a given node*

Description

Get all time series recorded in Source for a given node

Usage

```
VeneerGetTSbyNode(Node, run = "latest", baseURL = "http://localhost:9876")
```

Arguments

Node	Name of node to retrieve Time Series for
run	Which run to retrieve from. Defaults to the latest
baseURL	URL of the Veneer server. Defaults to the veneer default.

Value

a zoo time series, with each variable as a column

Examples

```
## Not run:
VeneerGetTSbyNode("Storage 1")

## End(Not run)
```

VeneerGetTSbyVariable *Get all time series recorded in Source of a given variable type*

Description

Get all time series recorded in Source of a given variable type

Usage

```
VeneerGetTSbyVariable(
  variable = "Flow",
  run = "latest",
  baseURL = "http://localhost:9876"
)
```

Arguments

variable	Which variable to retrieve. Defaults to Flow.
run	Which run to retrieve from. Defaults to the latest
baseURL	URL of the Veneer server. Defaults to the veneer default.

Value

a zoo time series, with each output as a column

Examples

```
## Not run:
VeneerGetTSbyVariable() #returns all flow outputs recorded in the latest run
VeneerGetTSbyVariable("Water Surface Elevation",1)

## End(Not run)
```

VeneerGetTSVariables *Get a vector of the type of time series variables recorded*

Description

Get a vector of the type of time series variables recorded

Usage

```
VeneerGetTSVariables(run = "latest", baseURL = "http://localhost:9876")
```

Arguments

run	Which run to retrieve from. Defaults to the latest
baseURL	URL of the Veneer server. Defaults to the veneer default.

Value

a vector of variable types (e.g. Downstream flow, Downstream Flow Concentration, water surface elevation)

Examples

```
## Not run:
 VeneerGetTSVariables()

## End(Not run)
```

VeneerlatestRunNumber *Get the number of the latest run*

Description

Get the number of the latest run

Usage

```
VeneerlatestRunNumber(baseURL = "http://localhost:9876")
```

Arguments

baseURL	URL of the Veneer server. Defaults to the veneer default.

Value

integer of the latest run number

Examples

```
## Not run:
VeneerlatestRunNumber()

## End(Not run)
```

VeneerRunSource *Run Source using Veneer*

Description

Run Source using Veneer

Usage

```
VeneerRunSource(
  StartDate = NULL,
  EndDate = NULL,
  InputSet = NULL,
  baseURL = "http://localhost:9876"
)
```

Arguments

StartDate	Optional. Start date for simulation. Must be dd/mm/yyyy
EndDate	Optional. End date for simulation. Must be dd/mm/yyyy
InputSet	Optional. Input set to use
baseURL	URL of the Veneer server. Defaults to the veneer default.

Value

Nothing to the R environment.

If not set, the configuration parameters (StartDate, EndDate, InputSet), was is specified in the Source configuration in the GUI will be used.

The console will show any errors returned by Veneer.

Examples

```
## Not run:
VeneerRunSource()
VeneerRunSource("01/07/2017","01/02/2018","NoDams")

## End(Not run)
```

VeneerSetFunction *Update a function value or expression. Function must exist before being updated.*

Description

Update a function value or expression. Function must exist before being updated.

Usage

```
VeneerSetFunction(Name, Expression, baseURL = "http://localhost:9876")
```

Arguments

Name	Name of the function without the "$", e.g. f_ScaleFactor
Expression	Expression to change it to, e.g. 1.2
baseURL	URL of the Veneer server. Defaults to the veneer default.

Value

Nothing to the R environment.

Examples

```
## Not run:
 VeneerSetFunction("f_ScaleFactor",1.2)
 VeneerSetFunction("f_TargetLevel","if($m_Flow<1000,3.2,3.5)")

## End(Not run)
```

`VeneerSetPiecewise` *Change a Source piecewise table using Veneer*

Description

Change a Source piecewise table using Veneer

Usage

```
VeneerSetPiecewise(data, pw_table, baseURL = "http://localhost:9876")
```

Arguments

`data`	A 2 column data.frame or matrix with the data to load into the piecewise table.
`pw_table`	The name of the piecewise linear variable, without the "$".
`baseURL`	URL of the Veneer server. Defaults to the veneer default.

Value

Nothing to the R environment.

Examples

```
## Not run:
 data<-data.frame(X=seq(1,5),Y=seq(1,5))
 VeneerSetPiecewise(data,"pw_table")

## End(Not run)
```

WritepwtoIS *Write an input set line for a piecewise lookup table from a csv file*

Description

Write an input set line for a piecewise lookup table from a csv file

Usage

```
WritepwtoIS(folder, csvfiles, outputfile)
```

Arguments

folder	Folder for where are the csv files with the lookup tables
csvfiles	vector of files to turn into an input set line. File name should be the name of the pw table in Source, including the folder name if necessary, separated by "." (see example). The first row in the file should be column names, the same as used in Source, i.e. XValue and YValue
outputfile	text file to save the lines to

Value

Nothing to the R environment. Input set lines are written to "outputfile".

Examples

```
## Not run:
folder<-"C:/Source/tables"
csvfiles<-c("LowerLakesOps.pw_LakeTarget.csv","Operations.pw_NA_Lock5_16p8.csv")
outputfile<-"inputset.txt"
WritepwtoIS(folder,csvfiles,outputfile)

## End(Not run)
```

Index

AQWPDownload, *3*, 3
AQWPLoad, *3*, 5

HydstraSiteDetails, *3*, 6

read_res.csv, *3*, 7

segmented, *8*
SILOCheckConsistency, 8, *9*
SILOCorrectSite, *9*, 9, *10*
SILOCumulativeDeviation, 11
SILODoubleMass, 12
SILODownload, *3*, 12
SILOImport, 14
SILOLoad, *3*, *8–10*, 15
SILOMap, 16
SILOMonthlyRainfall, 16
SILOMortonQualityCodes, 17
SILOQualityCodes, *9, 10*, 18
SILOReport, *3*, 19
SILOSitesfromPolygon, 20
SILOSiteSummary, *9, 10*, 20
SILOThiessenShp, 21
SILOWriteforSource, *3*, 22
SILOWriteFunctionsforSource, 23
SWTools *(*SWTools-package*)*, 2
SWTools-package, 2

VeneerGetInputSets, 25
VeneerGetNodesbyType, 25
VeneerGetPiecewise, 26
VeneerGetTS, 27
VeneerGetTSbyNode, 28
VeneerGetTSbyVariable, 28
VeneerGetTSVariables, 29
VeneerlatestRunNumber, 30
VeneerRunSource, 30
VeneerSetFunction, 31
VeneerSetPiecewise, 32

WritepwtoIS, 33