Package: USGSHydroTools (via r-universe)

October 26, 2024

Type Package

Title Collection of functions for hydrological analysis

Version 1.3.0

Date 2022-02-07

Author Steve Corsi

Maintainer Steve Corsi <srcorsi@usgs.gov>

Description Collection of functions for storm volume computation (Hydrovol), time series data aggregation, baseflow determination, load computations, and spatial data visualization.

License CC0

Depends R (>= 3.0)

Imports dataRetrieval (>= 2.0.1), smwrBase, methods, stats, graphics

Suggests knitr

LazyLoad yes

LazyData yes

RoxygenNote 7.1.2

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

RemoteUrl https://github.com/USGS-R/USGSHydroTools

RemoteRef HEAD

RemoteSha 737725aeb3ea5342476833e63d60ad681f6d1b8f

Contents

`GSHydroTools-package` *Collection of functions for hydrological analysis.*

Description

Package:	GSHydroTools
Type:	Package
Version:	1.0.0
Date:	2014-01-10
License:	Unlimited for this package, dependencies have more restrictive licensing.
Copyright:	This software is in the public domain because it contains materials that originally came from the United States G
LazyLoad:	yes

Details

Collection of functions for hydrological analysis.

Author(s)

Steve Corsi <srcorsi@usgs.gov>

`computeSeasonal` *Compute sine and cosine terms for dates to use as seasonal terms in data analysis*

Description

Function to compute seasonal sin and cosine terms from POSIXlt variable

Usage

```
computeSeasonal(df, date, return.var)
```

Arguments

`df`	dataframe with date included
`date`	string column name of date to convert in POSIXlt format
`return.var`	string suffix for variable names to return

Value

df

Examples

```
sampleData <- sampleData
sampleData$bpdate <- as.POSIXlt(sampleData$Hbpdate) #convert from POSIXct to POSIXlt
computeSeasonal(df=sampleData,date="bpdate",return.var="bdate")
```

`determineHYSEPEvents` *Determine baseflow and events from HYSEP output*

Description

Function to find the longest continuous start and end dates from the Daily dataframe. Primary use case is to find input value to use in a call to HYSEP (from package DVstats). If there are gaps in the data, the function will look for the largest continous gap.

Usage

```
determineHYSEPEvents(HYSEPReturn, sampleDates, percent = 0.8, value = "Flow")
```

Arguments

`HYSEPReturn`	dataframe returned from hysep function (in DVstats package)
`sampleDates`	dataframe with two columns "Discharge_cubic_feet_per_second" and "maxSampleTime"
`percent`	number to use to determine event conditions. This number will be multiplied by the flow, and if that product is greater than the calculated baseflow, the sample time will be labeled an event.
`value`	character name of discharge column.

Value

sampleDates dataframe

Examples

```
site <- "04085427"
sampleDates <- sampleDates
Start_extend <- as.character(as.Date(min(sampleDates$ActivityStartDateGiven, na.rm=TRUE))-60)
End_extend <- as.character(as.Date(max(sampleDates$ActivityStartDateGiven, na.rm=TRUE))+60)
Daily <- dataRetrieval::readNWISdv(site,'00060', Start_extend, End_extend)
Daily <- dataRetrieval::renameNWISColumns(Daily)
sampleDates <- findSampleQ(site, sampleDates, Daily)
startEnd <- getMaxStartEnd(Daily)
Start <- startEnd$Start
End <- startEnd$End
naFreeDaily <- Daily[!is.na(Daily$Flow),]
INFO <- dataRetrieval::readNWISsite(site)
DA_mi <- INFO$drain_area_va
HYSEPReturn <- exampleHYSEP
sampleDates <- determineHYSEPEvents(HYSEPReturn, sampleDates,0.8)
```

dfOptical *Example data for correlations*

Description

Example data with response and predictor variables

Author(s)

Steve Corsi <srcorsi@usgs.gov>

events *Example event dates and times.*

Description

Example event begin and end dates and times to define a sampled hydrograph

Author(s)

Steve Corsi <srcorsi@usgs.gov>

exampleHYSEP *Example HYSEP output data.*

Description

Example HYSEP output data. Needs more info.

Author(s)

Steve Corsi <srcorsi@usgs.gov>

FIBdata *Example water quality data included in GSHydroTools*

Description

Example data representing composite fecal indicator bacteria from the Menomonee River at Wauwatosa, Wisconsin

Author(s)

Steve Corsi <srcorsi@usgs.gov>

findSampleQ *Find flow for sample times*

Description

Function to find flows values for given sample times. If instantaneous data is available, this function will retrieve that data, otherwise the Daily streamflow data will be used. If the sample times have a start and end time, the flow is the maximum flow in the range of the sample.

Usage

```
findSampleQ(site, sampleDates, localDaily, value = "Flow")
```

Arguments

site	string USGS identification number
sampleDates	dataframe with two columns "ActivityStartDateGiven" and "ActivityEndDateGiven"
localDaily	dataframe returned from dataRetrieval
value	character name of discharge column

Value

sampleDates

Examples

```
site <- "04085427"
sampleDates <- sampleDates
Start_extend <- as.character(as.Date(min(sampleDates$ActivityStartDateGiven, na.rm=TRUE))-60)
End_extend <- as.character(as.Date(max(sampleDates$ActivityStartDateGiven, na.rm=TRUE))+60)
Daily <- dataRetrieval::readNWISdv(site,'00060', Start_extend, End_extend)
Daily <- dataRetrieval::renameNWISColumns(Daily)
sampleDates <- findSampleQ(site, sampleDates, Daily)
```

flowData *Example flow data.*

Description

Example instantaneous (unit value) flow data from Menomonee River at Wauwatosa, WI.

Author(s)

Steve Corsi <srcorsi@usgs.gov>

getMaxStartEnd *Find maximum start and end dates from Daily dataframe*

Description

Function to find the longest continuous start and end dates from the Daily dataframe. Primary use case is to find input value to use in a call to HYSEP (from package DVStats). If there are gaps in the data, the function will look for the largest continous gap.

Usage

```
getMaxStartEnd(localDaily, value = "Flow", date = "Date")
```

Arguments

localDaily	dataframe returned from dataRetrieval
value	character name of discharge column
date	character name of date column

Value

named list with Start and End values

Examples

```
site <- "04085427"
sampleDates <- sampleDates
Start_extend <- as.character(as.Date(min(sampleDates$ActivityStartDateGiven, na.rm=TRUE))-60)
End_extend <- as.character(as.Date(max(sampleDates$ActivityStartDateGiven, na.rm=TRUE))+60)
Daily <- dataRetrieval::readNWISdv(site,'00060', Start_extend, End_extend)
Daily <- dataRetrieval::renameNWISColumns(Daily)
startEnd <- getMaxStartEnd(Daily)
Start <- startEnd$Start
End <- startEnd$End
```

Hydrovol *Hydrovol*

Description

Computes volumes and max discharge for hydrographs given the discharge time series and the begin and end dates and times of the hydrographs. Dates must be in POSIXct format.

Usage

```
Hydrovol(
  dfQ,
  Q = "Q",
  time = "pdate",
  df.dates,
  bdate = "bpdate",
  edate = "epdate",
  volume = "event.vol",
  Qmax = "Qmax",
  duration = "Eduration"
)
```

Arguments

dfQ	dataframe with Q and time
Q	string name of column in dfQ with Q, defaults to "Q"
time	string name of column in dfQ with POSIXct time, defaults to "pdate"
df.dates	dataframe with begin and end dates/times in POSIXct format
bdate	string begin date in POSIXct column name, defaults to "bpdate"
edate	string end date in POSIXct column name, defaults to "epdate"
volume	string name of resulting volume variable, defaults to "event.vol"
Qmax	string name of Qmax variable, defaults to "Qmax"
duration	string name of resulting duration variable, defaults to "Eduration"

Value

df.dates2 dataframe

Examples

```
sampleData <- sampleData
flowData <- flowData
Hydrovol(dfQ=flowData,Q="Q",time="pdate",
         df.dates=sampleData,bdate="Hbpdate",edate="Hepdate")
```

LoadCompEvent *LoadCompEvent*

Description

Computation of loadings for event periods using individual discrete samples. Results in added columns to the event data frame that represent the event loadings in the original mass units from the concentration variable and the flow-weighted event mean concentration. maximum flow in the original flow units, volumes in the original volume units from the flow variable, and loadings in the original mass units from the concentration variable.

Usage

```
LoadCompEvent(
  df.samples,
  Conc,
  sample.time,
  Conc2liters,
  df.Q,
  Q,
  Q.time,
  Q2liters,
  df.events,
  event.bdate,
  event.edate
)
```

Arguments

df.samples	dataframe with discrete sample results and dates/times
Conc	string column name in df.samples with the concentration results
sample.time	string column name in df.samples with sample dates/times in POSIXct format
Conc2liters	numeric conversion factor that converts the concentrations to a units/liter
df.Q	dataframe with Q and date/time
Q	string name of column in dfQ with Q
Q.time	string name of column in dfQ with date/time in POSIXct format
Q2liters	numeric conversion factor that converts flow to rate per liters
df.events	dataframe with begin and end dates defining the event period
event.bdate	character string with name of variable defining beginning date for events in POSIXct format
event.edate	character string with name of variable defining end date for events in POSIXct format

Value

df.load

Examples

```
WQdata <- WQdata
flowData <- flowData
events <- events
LoadCompEvent(df.samples=WQdata,Conc="Total_P",sample.time="dateTime",Conc2liters=1,
df.Q=flowData,Q="Q",Q.time="pdate",Q2liters=28.3168466,
df.events=events,event.bdate="pbdate",event.edate="pedate")
```

LoadInstantaneous *LoadInstantaneous*

Description

Computation of loadings for individual discrete samples. Results in added columns to the concentration data frame that represent the maximum flow in the original flow units, volumes in the original volume units from the flow variable, and loadings in the original mass units from the concentration variable.

Usage

```
LoadInstantaneous(
  df.samples,
  Conc,
  sample.time,
  Conc2liters,
  df.Q,
  Q,
  Q.time,
  Q2liters
)
```

Arguments

df.samples	dataframe with discrete sample results and dates/times
Conc	string column name in df.samples with the concentration results
sample.time	string column name in df.samples with sample dates/times in POSIXct format
Conc2liters	numeric conversion factor that converts the concentrations to a units/liter
df.Q	dataframe with Q and date/time
Q	string name of column in dfQ with Q
Q.time	string name of column in dfQ with date/time in POSIXct format
Q2liters	numeric conversion factor that converts flow to rate per liters

Value

df.load

Examples

```
WQdata <- WQdata
flowData <- flowData
LoadInstantaneous(df.samples=WQdata, Conc="Total_P",
         sample.time="dateTime", Conc2liters=1,
         df.Q=flowData, Q="Q", Q.time="pdate", Q2liters=28.3168466)
```

multiCor *multiCor compute correlation coefficients for one response variable vs multiple predictor (independent) variables. The output is a dataframe ordered by highest to lowest correlation*

Description

multiCor compute correlation coefficients for one response variable vs multiple predictor (independent) variables. The output is a dataframe ordered by highest to lowest correlation

Usage

```
multiCor(df, response, IVs, method = "spearman")
```

Arguments

df	is dataframe with response variable and predictor variables
response	is a character string that is the name of the response variable in df
IVs	is a vector of character strings that are the independent variables in df
method	is either "spearman" (nonparametric) or "pearson" (parametric)

Value

z dataframe with the variable name in column 1 and correlation coefficient in column 2. The dataframe is ordered from greatest correlation to least correlation.

Examples

```
data <- dfOptical
multiCor(data,"logEColi",names(data)[-1],"spearman")
```

plotBaseflow *Plot baseflow/event plot*

Description

Plot output of flow, with daily and instantaneous flow (when available).

Usage

```
plotBaseflow(
  sampleDates,
  Daily,
  INFO,
  site,
  HYSEPReturn,
  baseflowColumns = "flowConditionHYSEP_localMin",
  HYSEPcolNames = "LocalMin",
  xlabel = TRUE,
  showLegend = TRUE,
  plotTitle = TRUE,
  instantFlow = NA,
  whatDischarge,
  value = "Flow",
  valueInst = "Flow_Inst"
)
```

Arguments

sampleDates	dataframe with two columns "Discharge_cubic_feet_per_second" and "maxSampleTime"
Daily	dataframe from getNWISDaily function in the dataRetrieval package
INFO	dataframe from getNWISInfo function in dataRetrieval package. Alternatively, a dataframe with a column "station.nm"
site	string USGS site identification
HYSEPReturn	dataframe with one column Dates, and at least 1 column of baseflow
baseflowColumns	string. Names of columns in the sampleDates dataframe with "Baseflow" or "Event" indicators.
HYSEPcolNames	string. Name of column in HYSEPReturn.
xlabel	logical. Whether or not to print x label
showLegend	logical. Whether or not to print legend
plotTitle	logical. Whether or not to print title
instantFlow	dataframe returned from retrieveUnitNWISData. If none available, NA.
whatDischarge	dataframe returned from whatNWISdata
value	character name of discharge column in Daily
valueInst	character name of discharge column in instantFlow

Examples

```
site <- "04085427"
sampleDates <- sampleDates
Start_extend <- as.character(as.Date(min(sampleDates$ActivityStartDateGiven, na.rm=TRUE))-60)
End_extend <- as.character(as.Date(max(sampleDates$ActivityStartDateGiven, na.rm=TRUE))+60)
```

```
Daily <- dataRetrieval::readNWISdv(site,'00060', Start_extend, End_extend)
Daily <- dataRetrieval::renameNWISColumns(Daily)
sampleDates <- findSampleQ(site, sampleDates, Daily)
startEnd <- getMaxStartEnd(Daily)
Start <- startEnd$Start
End <- startEnd$End
naFreeDaily <- Daily[!is.na(Daily$Flow),]
INFO <- dataRetrieval::readNWISsite(site)
DA_mi <- as.numeric(INFO$drain_area_va)
HYSEPReturn <- exampleHYSEP
sampleDates <- determineHYSEPEvents(HYSEPReturn, sampleDates,0.8)
whatDischarge <- dataRetrieval::whatNWISdata(siteNumber = site)
whatDischarge <-  whatDischarge[whatDischarge$parm_cd == "00060", ]
Start <- as.character(as.Date(min(sampleDates$ActivityStartDateGiven, na.rm=TRUE)))
End <- as.character(as.Date(max(sampleDates$ActivityStartDateGiven, na.rm=TRUE)))

if ("uv" %in% whatDischarge$data_type_cd){
 if(any(whatDischarge$begin_date[whatDischarge$data_type_cd == "uv"] < End, na.rm = TRUE)){
    instantFlow <- dataRetrieval::readNWISuv(site,"00060",Start,End)
    instantFlow <- dataRetrieval::renameNWISColumns(instantFlow)
  }
}
plotBaseflow(sampleDates,Daily,INFO,site,HYSEPReturn,
             baseflowColumns="flowConditionHYSEP_localMin",
             HYSEPcolNames = "LocalMin",plotTitle=TRUE,
             instantFlow=instantFlow,whatDischarge=whatDischarge,xlabel=FALSE)
```

---

plotHydroConc *plotHydroConc*

Description

Function to generate a two panel graph with hydrograph(s) in the top panel and concentration or other water quality parameter (e.g. flux) in the lower panel with corresponding date axes for the top and bottom panels

Usage

```
plotHydroConc(
  Q,
  QVars,
  QDateVars,
  smooth,
  sites,
  Conc,
  CVars,
  CDateVar,
  CVarsDisplay,
  dates,
```

```
  sampDates,
  eventDates,
  Qcols,
  Ccols,
  leftBuffer,
  rightBuffer,
  concLines = TRUE,
  Qylab = "Discharge (cfs)",
  Cylab = "Concentration",
  title1 = "Flow",
  title2 = "and concentration"
)
```

Arguments

Q	dataframe list with flow variables and POSIXct date variables. The list contains one file per Q record (usually one dataframe per site).
QVars	vector of strings that signify the column names that represent the flow variables in the dataframes defined in Q
QDateVars	vector of strings that signify the column names that represent the POSIXct date variables in the dataframes defined in Q
smooth	Boolean vector to trigger lowess smooth in graphing rather than direct flow graphing (useful for sites impacted by seiche)
sites	Sites for Q data. This will be used in the legend of the Q panel
Conc	dataframe with variables to be plotted in the second panel
CVars	vector of strings that represent the variables in Conc to include in the graphs in the second panel
CDateVar	column name in Conc for sample dates and times in POSIXct
CVarsDisplay	variable names from CVars to be displayed on the legend
dates	dataframe with beginning and ending sample dates and times and beginning and ending hydrograph dates and times
sampDates	vector of length two that contains strings representing the column names for beginning and ending dates and times for samples
eventDates	vector of length two that contains strings representing the column names for beginning and ending dates and times for the sampling event. This is the hydrograph segment that the samples are intended to represent (often a runoff hydrograph or a baseflow period)
Qcols	vector of colors for ploting hydrographs from the Q dataframe
Ccols	vector of colors for plotting from the Conc dataframe
leftBuffer	time in days to plot before the beginning of the event period
rightBuffer	time in days to plot after the end of the event period
concLines	Boolean variable to signify whether a line should be drawn between consecutive data points from the Conc dataframe in the second graph panel
Qylab	y-axis label for the first graph panel

Cylab	y-axis label for the second graph panel
title1	Line 1 of plot title
title2	Line 2 of plot title

Examples

```
#Add example
```

`plotHYSEPOverview` *Plot 3 baseflow/event plots from hysep output*

Description

Plot sliding, fixed, and local_min output of flow, with daily and instantaneous flow (when available).

Usage

```
plotHYSEPOverview(
  sampleDates,
  Daily,
  INFO,
  site,
  HYSEPReturn,
  baseflowColumns = c("flowConditionHYSEP_localMin", "flowConditionHYSEP_Fixed",
    "flowConditionHYSEP_Sliding"),
  HYSEPcolNames = c("LocalMin", "Fixed", "Sliding")
)
```

Arguments

sampleDates	dataframe with two columns "Discharge_cubic_feet_per_second" and "maxSampleTime"
Daily	dataframe from getNWISDaily function in the dataRetrieval package
INFO	dataframe from getNWISSiteInfo function in dataRetrieval package. Alternatively, a dataframe with a column "station.nm"
site	string USGS site identification
HYSEPReturn	dataframe with one column Dates, and 3 columns of baseflow as defined by HYSEPcolNames
baseflowColumns	sting vector length of 3. Names of columns with "Baseflow" or "Event" indicators.
HYSEPcolNames	sting vector length of 3. Names of columns in HYSEPReturn

Value

sampleDates dataframe

Examples

```
site <- "04085427"
sampleDates <- sampleDates
Start_extend <- as.character(as.Date(min(sampleDates$ActivityStartDateGiven, na.rm=TRUE))-60)
End_extend <- as.character(as.Date(max(sampleDates$ActivityStartDateGiven, na.rm=TRUE))+60)
Daily <- dataRetrieval::readNWISdv(site,'00060', Start_extend, End_extend)
Daily <- dataRetrieval::renameNWISColumns(Daily)
sampleDates <- findSampleQ(site, sampleDates, Daily)
startEnd <- getMaxStartEnd(Daily)
Start <- startEnd$Start
End <- startEnd$End
naFreeDaily <- Daily[!is.na(Daily$Flow),]
INFO <- dataRetrieval::readNWISsite(siteNumber = site)
DA_mi <- as.numeric(INFO$drain_area_va)
HYSEPReturn <- exampleHYSEP
sampleDates <- determineHYSEPEvents(HYSEPReturn, sampleDates,0.8)
plotHYSEPOverview(sampleDates,Daily,INFO,site,HYSEPReturn)
```

---

sampleData *Example sample data.*

Description

Example sample data. Needs more info.

Author(s)

Steve Corsi <srcorsi@usgs.gov>

---

sampleDates *Example sample dates.*

Description

Example sample dates data.

Author(s)

Steve Corsi <srcorsi@usgs.gov>

shape_hydroline *Rivers in US.*

Description

River shapefiles from http://dds.cr.usgs.gov/pub/data/nationalatlas/hydro0m_shp_nt00300.tar.gz

shape_hydropoly *Lakes in US*

Description

Lake shapefiles from http://dds.cr.usgs.gov/pub/data/nationalatlas/hydro0m_shp_nt00300.tar.gz

shape_polibounds *Political boundaries.*

Description

Political boundary shapefiles from http://dds.cr.usgs.gov/pub/data/nationalatlas/bound0m_shp_nt00298.tar.gz

SI *Example USGS site information*

Description

Station name, coordinates, label offsets, and line offsets for positioning the labels on a map and lines from the data points to the labels

stationINFO *Example USGS site information*

Description

Raw data pulled from NWIS

summarizedataDF *Summarize dataDF by group*

Description

Mapping routine that displays spatial dataDF variability by color differences. over layers with political boundaries, hydrologic polygons, and hydrologic lines.

Usage

```
summarizedataDF(dataDF, colGroup, colValue, colDate)
```

Arguments

dataDF	dataframe with columns defined by colGroup (grouping column, such as site ID), colValue (value column), and optionally colDate (date columns)
colGroup	string defines grouping column in dataDF
colValue	string defines value column in dataDF
colDate	string defines date column in dataDF. If colDate = NA, the calculations for start and end date are ignored.

Value

dataframe with count, mean, median, min, max, start(date/time), end(date/time), and number of non-detects (nd) defined as number of NA's grouped by colGroup

Examples

```
df <- data.frame(site=c("1","x","2","1","x","2"),
                 conc=c(2,3,4,5,NA,7),
                 dates=as.Date(c("2011-01-01","2011-01-01",
                   "2011-01-01","2011-01-02","2011-01-02","2011-01-02")))
sumDF <- summarizedataDF(df, "site", "conc", "dates")
```

TSstats *Compute various antecedent summary statistics from time series data for specified windows of time*

Description

Compute various stats for time series data over a period of time Originally scripted for NOAA Great Lakes model from GDP for given set of dates and time periods, but could be used for any time series. File format must include the POSIX formatted date (yyyy-mm-ddThh:mm:ssZ), and then columns of values with the time series data

read date with format mm/dd/yy hh:mm (use koepkeSM$date <- as.POSIXct(koepkeSM$Date,"

read date with format mm/dd/yyyy hh:mm cedardates$psdate <- as.POSIXct(cedardates$Startdate," cedardates$parfdate <- as.POSIXct(cedardates$Enddate,"

Subset the data by begin and end date (can also assign to a df if you like) then define min mean median and max for the subset. Do this for all date periods in the file.

Usage

```
TSstats(
  df,
  date = "date",
  varnames,
  dates,
  starttime = "psdate",
  times = c(1, 2),
  units = "hours",
  stats.return = c("mean"),
  subdfvar = "",
  subdfvalue = "",
  subdatesvar = "",
  subdatesvalue = "",
  out.varname = ""
)
```

Arguments

df	dataframe Unit values file
date	string Date column in POSIX format in unit values file
varnames	string Column name with unit values
dates	dataframe File with sample dates
starttime	string Column in sample dates file with dates in POSIX format, defaults to "psdate"
times	vector to define desired processing times. Zero indicates then nearest or nearest previous value. Default is hours, but can be specified using "units" variable
units	string Units of times vector. Can be any of the following: "minutes","min","mins","hours","hr","hrs","day
stats.return	string Options include "mean","max","min","median","sum","sd","maxdiff","difference",nearest","nearpr maxdiff is the maximum value minus the minimum value for the time period, difference is the latest minus the first value, nearest is the closest value in time, nearprev is the closest value previous to the specified time, nearest and nearprev require a 0 in the times vector,

subdfvar	string column name in UVdf with names of parameters, default is ""
subdfvalue	string Optional: value of varname to use in subsetting df, default is ""
subdatesvar	string Optional: subset dates data frame by a value in this column, default is ""
subdatesvalue	string Optional: value to use in subsetting
out.varname	string

Value

dates dataframe

Examples

```
flowData <- flowData
sampleData <- sampleData
TSstats(df=flowData,date="pdate",varnames="Q",
        dates=sampleData,starttime="Hbpdate",times=c(1,3,6,12,24),
        units="hrs",stats.return=c("mean","max","sd"),out.varname="Q")
```

TSstormstats *Compute various time-series summary statistics between specified time periods*

Description

Compute various stats for time series data over a period of time Can be used for time series data with equally spaced time increments. File format must include the POSIXct formatted date and columns of values with the time series data

Usage

```
TSstormstats(
  df,
  date = "pdate",
  varname,
  dates,
  starttime = "Ebpdate",
  endtime = "Eepdate",
  stats.return = c("mean"),
  subdfvar = "",
  subdfvalue = "",
  subdatesvar = "",
  subdatesvalue = "",
  out.varname = ""
)
```

Arguments

df	dataframe with unit values values and date/time in POSIX
date	string name of POSIX date column
varname	string column with unit values in df
dates	dataframe with sample dates
starttime	string Column in sample dates data fram with dates in POSIX format used for extracting summary data from dates dataframe This date serves as the beginning date of the summary period, default is "psdate"
endtime	string Column in sample dates data fram with dates in POSIX format used for extracting summary data from dates dataframe This date serves as the ending date of the summary period, default is "pedate"
stats.return	string vector Options include = c("mean","max","min","median","sum") specification of stats to apply to the time series data. Current options include mean, max, min, median, sum, difference, nearest, and nearprev. difference is the latest minus the first value, nearest is the closest value in time to starttime, nearprev is the closest value previous to starttime, nearest and nearprev require a 0 in the times vector.
subdfvar	string subset df data frame by a value in this column
subdfvalue	string value to use in subsetting df
subdatesvar	string subset dates data frame by a value in this column
subdatesvalue	string value to use in subsetting
out.varname	string variable name for resulting column

Value

dates dataframe

Examples

```
flowData <- flowData
sampleData <- sampleData
TSstormstats(df=flowData,date="pdate",varname="Q",
      dates=sampleData,starttime="Hbpdate",endtime="Hepdate",
      stats.return=c("mean","max","sd"),out.varname="Q")
```

WQcompos *Combine multiple samples into one composite sample (loads and concentrations)*

Description

function to composite samples weighted by the associated volume the result is a volume-weighted concentration and summation of volumes

Usage

```
WQcompos(df.samples, sampleID, parms, volume = "Evolume", bdate, edate, codes)
```

Arguments

df.samples	dataframe with sample results and volumes
sampleID	character variable name for the IDs for compositing samples (multiple samples will have the same ID)
parms	vector Parameters to composite
volume	character variable name for the volume, defaults to "Evolume"
bdate	character variable name for the beginning of event times for each sample
edate	character variable name for the ending of event times for each sample
codes	a vector of character variable names for the values that should be pasted together into one string when combining samples (lab IDs are common here)

Value

IDdf dataframe

Examples

```
flowData <- flowData
FIBdata <- FIBdata
FIBcomposData <- Hydrovol(dfQ=flowData,Q="Q",time="pdate",
       df.dates=FIBdata,bdate="SSdate",edate="SEdate")
WQcompos(df.samples=FIBcomposData,sampleID="SampleID",
       parms=c("Ecoli","Enterococci"), volume="event.vol",
       bdate="SSdate",edate="SEdate",codes="SampleID")
```

WQdata *Example water quality data included in GSHydroTools*

Description

Example data representing discrete total phosphorus concentrtions from the Menomonee River at Wauwatosa, Wisconsin

Author(s)

Steve Corsi <srcorsi@usgs.gov>

Index