Package: EGRET (via r-universe)

October 23, 2026

Type Package

Title Exploration and Graphics for RivEr Trends

Version 3.0.13

Description Statistics and graphics for streamflow history, water quality trends, and the statistical modeling algorithm: Weighted Regressions on Time, Discharge, and Season (WRTDS).

License CC0

Depends R (>= 3.5)

Imports dataRetrieval (>= 2.7.19), survival, fields, methods, utils, grDevices, graphics, stats, truncnorm, foreach, MASS, sf, zoo

Suggests EGRETci, knitr, rmarkdown, extrafont, testthat, rkt, doParallel, parallel, pkgdown, png, dplyr, zyp, lubridate, covr

LazyLoad yes

LazyData yes

BugReports https://github.com/DOI-USGS/EGRET/issues

VignetteBuilder knitr

BuildVignettes true

URL https://pubs.usgs.gov/tm/04/a10/

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/information-policies-and-instructions/copyrights-and-credits

Roxygen list(old_usage = TRUE)

Encoding UTF-8

Config/roxygen2/version 8.1.1

Config/pak/sysreqs libabsl-dev cmake libgdal-dev gdal-bin libgeos-dev libxml2-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev libx11-dev

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

Date/Publication 2026-10-23 17:18:43 UTC

RemoteUrl https://code.usgs.gov/water/EGRET

RemoteRef HEAD

RemoteSha 8ee2a442f4a5a03b4d95ad985f9893f46ecff9d1

Contents

as.egret . . . 4
blankTime . . . 5
boxConcMonth . . . 6
boxConcThree . . . 8
boxQTwice . . . 9
boxResidMonth . . . 11
calculateMonthlyResults . . . 12
censoredSegments . . . 13
checkStartEndDate . . . 14
checkSurfaceSpan . . . 14
Choptank_eList . . . 15
cleanUp . . . 15
compressData . . . 16
concUnit-class . . . 17
Constants . . . 17
cumQdate . . . 18
dataOverview . . . 19
dateFormatCheck . . . 19
decimalDate . . . 20
decimalHighLow . . . 20
errorStats . . . 21
estCrossVal . . . 22
estDailyFromSurfaces . . . 23
estSurfaces . . . 24
fill_missing_daily . . . 26
fillUpElists . . . 27
fixSampleFrame . . . 28
flexFN . . . 29
flexPlotAddOn . . . 30
flowDuration . . . 31
fluxBiasMulti . . . 32
fluxBiasStat . . . 33
fluxUnit-class . . . 34
generalAxis . . . 34
genericEGRETDotPlot . . . 35
genmissing . . . 38
getDaily . . . 38
getInfo . . . 39
getSample . . . 40
getSurfaces . . . 41

INFOdataframe . . . 41
is.egret . . . 43
jitterSam . . . 44
logPretty1 . . . 44
logPretty3 . . . 45
makeAnnualSeries . . . 46
makeAugmentedSample . . . 47
makeDateInfo . . . 48
mergeReport . . . 48
modelEstimation . . . 50
monthLabel-class . . . 51
multiPlotDataOverview . . . 51
plot15 . . . 53
plot1of15 . . . 54
plotConcHist . . . 54
plotConcPred . . . 56
plotConcQ . . . 58
plotConcQSmooth . . . 60
plotConcTime . . . 62
plotConcTimeDaily . . . 65
plotConcTimeSmooth . . . 66
plotContours . . . 69
plotDiffContours . . . 71
plotFlowSingle . . . 74
plotFluxHist . . . 76
plotFluxPred . . . 77
plotFluxQ . . . 79
plotFluxTimeDaily . . . 80
plotFour . . . 82
plotFourStats . . . 83
plotMonthTrend . . . 84
plotQTimeDaily . . . 85
plotResidPred . . . 87
plotResidQ . . . 88
plotResidTime . . . 90
plotSDLogQ . . . 91
plotTimeSlice . . . 92
plotWRTDSKalman . . . 93
populateConcentrations . . . 94
populateDaily . . . 95
populateDailySamp . . . 97
populateDateColumns . . . 98
populateParameterINFO . . . 99
populateSampleColumns . . . 99
populateSiteINFO . . . 100
precompute_L . . . 101
print.egret . . . 101
printFluxUnitCheatSheet . . . 102

printGroups . . . 103
printPairs . . . 104
printqUnitCheatSheet . . . 104
printSeries . . . 105
processQWData . . . 106
qUnit-class . . . 107
randomSubset . . . 107
readDataFromFile . . . 108
readNWISDaily . . . 109
readNWISSample . . . 110
readUserDaily . . . 111
readUserSample . . . 113
readWQPSample . . . 114
removeDuplicates . . . 116
runGroups . . . 117
runPairs . . . 120
runSeries . . . 123
runSurvReg . . . 126
saveResults . . . 128
selectDays . . . 128
setPA . . . 129
setSeasonLabel . . . 130
setSeasonLabelByUser . . . 131
setUpEstimation . . . 131
setupYears . . . 132
startEnd . . . 133
stitch . . . 134
surfaceIndex . . . 136
surfaceStartEnd . . . 137
tableChange . . . 137
tableFlowChange . . . 139
tableResults . . . 140
triCube . . . 141
WRTDSKalman . . . 141
yPretty . . . 142

Index **143**

`as.egret` *Create named list for EGRET analysis*

Description

Create a named list with the INFO, Daily, and Sample dataframes, and surface matrix. If any of these are not available, an NA should be

Usage

```
as.egret(INFO, Daily, Sample = NA, surfaces = NA)
```

Arguments

INFO	dataframe containing the INFO dataframe
Daily	dataframe containing the daily data
Sample	dataframe containing the sample data
surfaces	matrix returned from modelEstimation. Default is NA.

Value

eList named list with Daily, Sample, and INFO dataframes, along with the surfaces matrix. Any of these values can be NA, not all EGRET functions will work with missing parts of the named list eList.

See Also

readNWISDaily, readNWISSample

Examples

```
eList <- Choptank_eList
Daily <- getDaily(eList)
INFO <- getInfo(eList)
eList_flowHistory <- as.egret(INFO, Daily)
plotFlowSingle(eList_flowHistory, 1)
Sample <- getSample(eList)
surfaces <- getSurfaces(eList)
eList_full <- as.egret(INFO, Daily, Sample, surfaces)
plotFluxQ(eList_full)
```

---

blankTime — *Deletes the computed values during periods of time when there are no sample data*

Description

This function is used when the data analyst believes that a gap in the sample data record is so long that estimates during that period are not reliable. This is only used for periods of several years in duration. For this period, the values of Conc, Flux, FNConc and FNFlux are all converted to NA.

Usage

```
blankTime(eList, startBlank, endBlank)
```

Arguments

eList	named list with at least the Daily dataframe
startBlank	character specifying starting date of blank period, input in quotes in yyyy-mm-dd format
endBlank	character specifying the ending date of blank period, input in quotes in yyyy-mm-dd format

Details

The startBlank and endBlank arguments should generally coincide with the starting and ending date of the period of analysis that is being used. startBlank should be placed fairly close to the start of the period of no data and endBlank should be placed fairly close to the end of the period of no data. They do not eliminate any water quality data from the set of data being used to estimate the model, they only eliminate results computed for the specified blank period. If the data set has more than one large data gap the `blankTime()` function can be used multiple times to blank out multiple sets of results.

Value

eList named list with modified Daily data frame.

Examples

```
startBlank = "2004-10-01"
endBlank = "2006-09-30"
eList <- Choptank_eList
eList <- blankTime(eList, startBlank, endBlank)
```

---

boxConcMonth *Box plot of the water quality data by month*

Description

Data come from named list, which contains a Sample dataframe with the sample data, and an INFO dataframe with metadata.

Box widths are proportional to the square root of the number of samples in the month.

Although there are a lot of optional arguments to this function, most are set to a logical default.

Usage

```
boxConcMonth(eList, printTitle = TRUE, cex = 0.8, cex.axis = 1.1,
  cex.main = 1.1, las = 1, logScale = FALSE, tcl = 0.5,
  tinyPlot = FALSE, customPar = FALSE, showYLabels = TRUE, concLab = 1,
  showXLabels = TRUE, showXAxis = TRUE, showYAxis = TRUE, monthLab = 1,
  ...)
```

Arguments

eList	named list with at least the Sample and INFO dataframes
printTitle	logical variable if TRUE title is printed, if FALSE not printed (this is best for a multi-plot figure)
cex	numerical value giving the amount by which plotting symbols should be magnified
cex.axis	magnification to be used for axis annotation relative to the current setting of cex
cex.main	magnification to be used for main titles relative to the current setting of cex
las	numeric in c(0,1,2,3); the style of axis labels, see ?par
logScale	logical if TRUE y plotted in log axis
tcl	number defaults to 0.5, specifies length of tick marks as fraction of height of a line of text
tinyPlot	logical variable, if TRUE plot is designed to be plotted small as part of a multi-plot figure, default is FALSE.
customPar	logical defaults to FALSE. If TRUE, par() should be set by user before calling this function
showYLabels	logical defaults to TRUE. If FALSE, the y axis label is not plotted
concLab	object of concUnit class, or numeric represented the short code, or character representing the descriptive name. By default, this argument sets concentration labels to use either Concentration or Conc (for tiny plots). Units are taken from the eList$INFO$param.units. To use any other words than "Concentration" see `vignette(topic = "units", package = "EGRET")`.
showXLabels	logical defaults to TRUE. If FALSE, the x axis label is not plotted
showXAxis	logical defaults to TRUE. If FALSE, the x axis is not plotted
showYAxis	logical defaults to TRUE. If FALSE, the y axis is not plotted
monthLab	object of monthLabel class, or numeric represented the short code, or character representing the descriptive name.
...	arbitrary graphical parameters that will be passed to genericEGRETDotPlot function (see ?par for options)

See Also

boxplot

Examples

```
eList <- Choptank_eList
# Water year:
boxConcMonth(eList)
# Graphs consisting of Jun-Aug
eList <- setPA(eList, paStart=6,paLong=3)
boxConcMonth(eList)
spanish_month <- new("monthLabel",
                  monthAbbrev = c("enero", "feb",  "marzo", "abr",
```

```
                                  "mayo", "jun", "jul", "agosto", "set",
                                  "oct", "nov", "dic"),
                    monthFull = c("enero", "febrero",  "marzo", "abril",
                                  "mayo", "junio", "julio", "agosto", "septiembre",
                                  "octubre", "noviembre", "diciembre"),
                    monthSingle = c("E", "F", "M", "A", "M", "J", "J",
                                  "A", "S", "O", "N", "D"))
boxConcMonth(eList, monthLab = spanish_month,
             showXLabels = FALSE, printTitle = FALSE)
```

---

boxConcThree *Three box plots side-by-side*

Description

This function is used to compare the distribution of concentration in the sample and predicted data set. It shows three boxplots. One for the sample, one for the predictions on days with sample values, and one for all days (whether or not they had sample values). Box widths are proportional to the square root of the number of observations represented by the box.

Data come from named list, which contains a Sample dataframe with the sample data, a Daily dataframe with the daily flow data, and an INFO dataframe with metadata.

Although there are a lot of optional arguments to this function, most are set to a logical default.

Usage

```
boxConcThree(eList, tinyPlot = FALSE, printTitle = TRUE,
  moreTitle = "WRTDS", customPar = FALSE, font.main = 2, cex = 0.8,
  cex.main = 1.1, cex.axis = 1.1, concLab = 1, ...)
```

Arguments

eList	named list with at least the Daily, Sample, and INFO dataframes
tinyPlot	logical variable, if TRUE plot is designed to be plotted small as part of a multi-plot figure, default is FALSE.
printTitle	logical variable if TRUE title is printed, if FALSE not printed (this is best for a multi-plot figure)
moreTitle	character specifying some additional information to go in figure title, typically some information about the specific estimation method used, default is no additional information
customPar	logical defaults to FALSE. If TRUE, par() should be set by user before calling this function
font.main	font to be used for plot main titles
cex	numerical value giving the amount by which plotting symbols should be magnified
cex.main	magnification to be used for main titles relative to the current setting of cex

cex.axis	magnification to be used for axis annotation relative to the current setting of cex
concLab	object of concUnit class, or numeric represented the short code, or character representing the descriptive name. By default, this argument sets concentration labels to use either Concentration or Conc (for tiny plots). Units are taken from the eList$INFO$param.units. To use any other words than "Concentration" see `vignette(topic = "units", package = "EGRET")`.
...	arbitrary graphical parameters that will be passed to genericEGRETDotPlot function (see ?par for options)

See Also

boxplot

Examples

```
eList <- Choptank_eList
# Water year:
boxConcThree(eList)
# Graphs consisting of Jun-Aug
eList <- setPA(eList, paStart=6,paLong=3)
boxConcThree(eList)
```

---

boxQTwice *Two box plots side-by-side, discharge on sample days, and discharge on all days*

Description

This function is used to compare the distribution of discharges in the sample data set and the discharges in the full daily data set. Note that discharge is plotted on a logarithmic axis. The boxplot is created using the log values but the scale is presented in the original units. An ideal situation would show the two boxes roughly similar to each other or the sample boxplot having median, upper quartile, and higher values being slightly greater than in the boxplot of all days.

Box widths are proportional to the square root of the number of observations (left box based on number of sampled days, right box based on total number of days in the record).

Data come from named list, which contains a Sample dataframe with the sample data, a Daily dataframe with the daily flow data, and an INFO dataframe with metadata.

Although there are a lot of optional arguments to this function, most are set to a logical default.

Usage

```
boxQTwice(eList, printTitle = TRUE, qUnit = 2, cex = 0.8,
  cex.main = 1.1, logScale = TRUE, cex.axis = 1.1, tcl = 0.5,
  las = 1, tinyPlot = FALSE, usgsStyle = FALSE, customPar = FALSE, ...)
```

Arguments

eList	named list with at least the Daily, Sample, and INFO dataframes
printTitle	logical variable if TRUE title is printed, if FALSE not printed (this is best for a multi-plot figure)
qUnit	object of qUnit class printqUnitCheatSheet, or numeric represented the short code, or character representing the descriptive name.
cex	numerical value giving the amount by which plotting symbols should be magnified
cex.main	magnification to be used for main titles relative to the current setting of cex
logScale	logical if TRUE y plotted in log axis. Defaults to TRUE.
cex.axis	magnification to be used for axis annotation relative to the current setting of cex
tcl	number defaults to 0.5, specifies length of tick marks as fraction of height of a line of text
las	numeric in c(0,1,2,3); the style of axis labels, see ?par
tinyPlot	logical variable, if TRUE plot is designed to be plotted small as part of a multi-plot figure, default is FALSE.
usgsStyle	logical option to use USGS style guidelines. Setting this option to TRUE does NOT guarantee USGS compliance. It will only change automatically generated labels.
customPar	logical defaults to FALSE. If TRUE, par() should be set by user before calling this function
...	arbitrary graphical parameters that will be passed to genericEGRETDotPlot function (see ?par for options)

See Also

boxplot

Examples

```
eList <- Choptank_eList
# Water year:
boxQTwice(eList)
boxQTwice(eList, qUnit=1)
boxQTwice(eList, qUnit='cfs')
# Graphs consisting of Jun-Aug
eList <- setPA(eList, paStart=6,paLong=3)
boxQTwice(eList)
```

boxResidMonth *A box plot of WRTDS residuals by month*

Description

This function produces a boxplot of the residuals from WRTDS, expressed in natural log concentration units. It provides an alternative for viewing the standardized residuals, where the each residual is divided by its estimated standard error. The monthly boxplot widths are proportional to the square root of the sample size. The residuals for a censored value are determined as the difference between the natural log of the average of the upper and lower. bounds on the sample value, minus the log space estimate of concentration.

Although there are a lot of optional arguments to this function, most are set to a logical default.

Data come from named list, which contains a Sample dataframe with the sample data, and an INFO dataframe with metadata

Usage

```
boxResidMonth(eList, stdResid = FALSE, las = 1, printTitle = TRUE,
  cex = 0.8, cex.axis = 1.1, cex.main = 1.1, font.main = 2,
  tinyPlot = FALSE, customPar = FALSE, monthLab = 1,
  randomCensored = FALSE, ...)
```

Arguments

eList	named list with at least the Sample and INFO dataframes
stdResid	logical variable, if TRUE it uses the standardized residual, if FALSE it uses the actual, default is FALSE
las	numeric in c(0,1,2,3); the style of axis labels
printTitle	logical variable if TRUE title is printed, if FALSE not printed (this is best for a multi-plot figure)
cex	numerical value giving the amount by which plotting symbols should be magnified
cex.axis	magnification to be used for axis annotation relative to the current setting of cex
cex.main	magnification to be used for main titles relative to the current setting of cex
font.main	font to be used for plot main titles
tinyPlot	logical variable, if TRUE plot is designed to be plotted small, as a part of a multipart figure, default is FALSE
customPar	logical defaults to FALSE. If TRUE, par() should be set by user before calling this function
monthLab	object of monthLabel class, or numeric represented the short code, or character representing the descriptive name.
randomCensored	logical. Show censored residuals as randomized. Default = FALSE.
...	arbitrary graphical parameters that will be passed to genericEGRETDotPlot function (see ?par for options)

See Also

boxplot

Examples

```
eList <- Choptank_eList
# Water year:
boxResidMonth(eList)
# Graphs consisting of Jun-Aug
eList <- setPA(eList, paStart = 6, paLong = 3)
boxResidMonth(eList)
```

`calculateMonthlyResults`

Calculates monthly mean values of Q, Conc, FNConc, Flux, and FNFlux for the entire record. If WRTDSKalman has been run it also includes the monthly mean values of GenConc and GenFlux.

Description

Computes the monthly mean values of discharge, concentration, flux, flow-normalized concentration and flow-normalized flux (Q, Conc, FNConc, Flux, and FNFlux) in SI units If WRTDSKalman has been run the outputs are averages for Q, Conc, GenConc, FNConc, Flux, GenFlux, and FNFlux. Note that the Flux, GenFlux, and FNFlux values are average flux values (not totals). For discharge the units are in m3/s, concentration is mg/L, and flux is kg/day. It returns a data frame containing month, year, decimal year, and mean values of DecYear, Q, Conc, GenConc, FNConc, Flux, GenFlux, and FNFlux.

Usage

```
calculateMonthlyResults(eList)
```

Arguments

`eList`	named list with at least the Daily dataframes

Value

MonthlyResults data frame of numeric values describing the monthly average values

Examples

```
eList <- Choptank_eList
monthlyResults <- calculateMonthlyResults(eList)
```

censoredSegments *Generic plotting function to create censored line segments*

Description

Basic plotting framework for EGRET dot plots. Graphical parameters default to values that work well with most plots, but all can be re-assigned. See ?par for complete definitions of most optional input variables.

Usage

```
censoredSegments(yBottom, yLow, yHigh, x, Uncen, col = "black", lwd = 1)
```

Arguments

yBottom	number specifying minimum flux (required)
yLow	vector specifying the x data (required), such as ConcLow
yHigh	vector specifying the x data (required), such as ConcHigh
x	vector x data (required)
Uncen	vector that defines whether the values are censored (0) or not (1)
col	color of points on plot, see ?par 'Color Specification'
lwd	number line width

See Also

segments

Examples

```
x <- c(1,2,3,4,5,6)
y <- c(1,3,4,3.3,4.4,7)
xlim <- c(min(x)*.75,max(x)*1.25)
ylim <- c(0,1.25*max(y))
xlab <- "Date"
ylab <- "Concentration"
xTicks <- pretty(xlim)
yTicks <- pretty(ylim)
genericEGRETDotPlot(x=x, y=y,
                    xlim=xlim, ylim=ylim,
                    xlab=xlab, ylab=ylab,
                    xTicks=xTicks, yTicks=yTicks,
                    plotTitle="Test"
)
yBottom <- 0
yLow <- c(NA,3,4,3.3,4,7)
yHigh <- c(1,3,4,3.3,5,NA)
Uncen <- c(0,1,1,1,0,0)
censoredSegments(yBottom=yBottom,yLow=yLow,yHigh=yHigh,x=x,Uncen=Uncen)
```

checkStartEndDate *checkStartEndDate*

Description

Checks that the start date is before the end date. If not, it will give the user the opportunity to correct, otherwise will create a warning.

Usage

```
checkStartEndDate(startDate, endDate, interactive = TRUE)
```

Arguments

startDate	character
endDate	character
interactive	logical Option for interactive mode. If true, there is user interaction for error handling and data checks.

Value

vector where first value is startDate, second is endDate

Examples

```
startDate <- '1985-01-01'
endDate <- '1990-01-01'
checkStartEndDate(startDate, endDate)
```

checkSurfaceSpan *checkSurfaceSpan*

Description

checkSurfaceSpan

Usage

```
checkSurfaceSpan(eList)
```

Arguments

eList	named list with at least the Daily, Sample, and INFO dataframes

Examples

```
eList <- Choptank_eList
checkSurfaceSpan(eList)
```

Choptank_eList *Example eList*

Description

Example data representing data from the Choptank River at Greensboro, MD, USGS data Data is a named list of the Daily, Sample, INFO dataframes, and the surface matrix.

Examples

```
head(Choptank_eList$Daily)
head(Arkansas_eList$Daily)
```

cleanUp *cleanUp eList*

Description

Takes an eList as the input. If there are duplicated dates in the Sample data frame, will randomly select one value for that date. If there are censored values in the data set they will be replaced by random censored values. If there are no days with duplicate samples and no censored valued then the eList returned by the function will be identical to the eList that is passed to it.

Usage

```
cleanUp(eList, seed = NA)
```

Arguments

eList	named list with the INFO, Daily, and Sample dataframes and surfaces matrix.
seed	integer value. Defaults to NA, which will not change the current seed. Setting the seed to any given value can be used to create repeatable output.

Details

This function is run before each iteration of generating a random sequence in the WRTDSKalman function

Value

eList with duplicated dates in the Sample data frame randomly sampled and censored values are replaced by random values.

Examples

```
eList <- Choptank_eList

eList <- cleanUp(eList)
```

---

compressData *Compress sample data frame*

Description

Using raw data that has at least dateTime, value, code, populates the measured data portion of the Sample dataframe used in EGRET. ConcLow = Lower bound for an observed concentration ConcHigh = Upper bound for an observed concentration Uncen = 1 if uncensored, 0 if censored

Usage

```
compressData(data, verbose = TRUE)
```

Arguments

data	dataframe contains at least dateTime, code, value, columns in that order. If there are more than the initial date, code, value...it combines the data is combined as using interval censored concepts.
verbose	logical specifying whether or not to display progress message

Value

data frame returnDataFrame data frame containing dateTime, ConcHigh, ConcLow, Uncen

Examples

```
dateTime <- c('1985-01-01', '1985-01-02', '1985-01-03')
comment1 <- c("","","")
value1 <- c(1,2,3)
comment2 <- c("","<","")
value2 <- c(2,3,4)
comment3 <- c("","","<")
value3 <- c(3,4,5)
dataInput <- data.frame(dateTime, comment1, value1,
      comment2, value2,
      comment3, value3, stringsAsFactors=FALSE)
compressData(dataInput)
```

concUnit-class *concUnit class*

Description

Some details about the concUnit class

Details

longPrefix A character specifying the long name for concentration labels.

shortPrefix A character specifying the short name for concentration labels.

Constants *Constants included with EGRET*

Description

fluxConst	Flux conversion object
qConst	Flow conversion object
monthInfo	Month object

Examples

```
fluxConst
fluxConst[['kgDay']]
fluxConst[['kgDay']]@unitName
qConst
qConst[['cfs']]
qConst[['cfs']]@qUnitName
concConst[['concentration']]
concConst[['concentration']]@shortPrefix
```

cumQdate *Cumulative flow calculation*

Description

This function computes the first day of the calendar year at which a specific fraction of the cumulative flow for that year has been exceeded. Typically one looks for the point where half the cumulative flow has happened (fract = 0.5). The portion of the year being considered is set by paStart and paLong. The matrix returned has 2 columns: the first is the year (integer when the period of analysis ends), the second is the day of the year when the fraction has been exceeded. None of the rows will have any NA values.

Usage

```
cumQdate(eList, paStart = 10, paLong = 12, fract = 0.5)
```

Arguments

eList	named list with at least the Sample and INFO dataframes
paStart	numeric integer specifying the starting month for the period of analysis, 1 <= paStart <= 12, default is 10
paLong	numeric integer specifying the length of the period of analysis, in months, 1 <= paLong <= 12, default is 12
fract	numeric fraction of the flow

Details

It is common to use this type of analysis on the snowmelt period of the year. If (for example) we assume that snowmelt starts with the month of March and ends in July then we would set paStart = 3 and paLong = 5

Value

annualSeries an integer matrix of two columns. The first column is the calendar year for the end of the period The second column is day of the year when the flow has exceeded the specified fraction of the entire period being considered

Examples

```
eList <- Choptank_eList
annualFlow <- cumQdate(eList)
head(annualFlow)
plot(annualFlow)
mod1 <- stats::lm(annualFlow[,2] ~ annualFlow[,1])
summary(mod1)
```

dataOverview *Data Overview for WRTDS*

Description

Gives a summary of data to be used for WRTDS analysis

Usage

```
dataOverview(Daily, Sample)
```

Arguments

Daily	dataframe
Sample	dataframe

See Also

mergeReport

Examples

```
eList <- Choptank_eList
exDaily <- getDaily(eList)
exSample <- getSample(eList)
dataOverview(Daily = exDaily, Sample = exSample)
```

dateFormatCheck *Check date format*

Description

Checks to see if format is YYYY-MM-DD. Also performs a few other date checks.

Usage

```
dateFormatCheck(date)
```

Arguments

date	character

Value

condition logical TRUE or FALSE if checks passed or failed

Examples

```
date <- '1985-01-01'
dateFormatCheck(date)
dateWrong <- '1999/1/7'
dateFormatCheck(dateWrong)
```

`decimalDate` *decimalDate*

Description

Create a decimal date or date/time from a vector.

Usage

```
decimalDate(rawData)
```

Arguments

`rawData`	vector of dates or dateTimes.

Examples

```
dateTime <- c('1984-02-28 13:56',
              '1984-03-01 00:00',
              '1986-03-01 00:00',
              '1986-10-15 00:00')
decimalDate(dateTime)

dateTime <- c('1984-02-28',
              '1984-03-01',
              '1986-03-01',
              '1986-10-15')
decimalDate(dateTime)
```

`decimalHighLow` *decimalHighLow*

Description

decimalHighLow figures out the highest and lowest decimal year based on water year. The input is a data frame with columns Month and DecYear.

Usage

```
decimalHighLow(df)
```

Arguments

df	data.frame with Month, DecYear, and Month columns

Value

list with DecHigh and DecLow (water year high/low decimal values)

Examples

```
eList <- Choptank_eList
highLow <- decimalHighLow(eList$Sample)

DecHigh <- highLow[["DecHigh"]]
DecLow <- highLow[["DecLow"]]
```

---

errorStats *Error statistics*

Description

This function takes a fitted WRTDS model and computes error statistics the residuals used here are cross-validation residuals, which will be slightly larger than regular regression residuals in the case of censored data, the residuals are computed from random residuals computed from makeAugmentedSample(), the function returns a list of error statistics and also prints them to the console

Usage

```
errorStats(eList)
```

Arguments

eList	named list with at least the Daily, Sample, and INFO dataframes

Value

erStats a numeric vector consisting of the following statistics RsqLC the R squared value for predictions of ln(Concentration) RsqLF the R squared value for predictions of ln(Flux) rmse the root mean squared error for ln(Concentration), same value would apply for Flux sepPercent the standard error of prediction for Concentration, expressed in percent same value would apply for Flux

Examples

```
eList <- Choptank_eList
errorStats(eList)
```

`estCrossVal` *Jack-Knife cross validation of the WRTDS (Weighted Regressions on Time, Discharge, and Season)*

Description

This function fits the WRTDS model n times (where n is the number of observations). For each fit, the data value being estimated is eliminated from the record. This gives predictions that do not depend on knowing the actual result for that day. Thus it provides for a more "honest" estimate of model performance than a traditional error analysis that uses all the data.

Usage

```
estCrossVal(DecLow, DecHigh, Sample, windowY = 7, windowQ = 2,
  windowS = 0.5, minNumObs = 100, minNumUncen = 50, edgeAdjust = TRUE,
  verbose = TRUE)
```

Arguments

`DecLow`	number specifying minimum decimal year
`DecHigh`	number specifying maximum decimal year
`Sample`	data frame containing the sample values, default is Sample
`windowY`	numeric specifying the half-window width in the time dimension, in units of years, default is 7
`windowQ`	numeric specifying the half-window width in the discharge dimension, units are natural log units, default is 2
`windowS`	numeric specifying the half-window with in the seasonal dimension, in units of years, default is 0.5
`minNumObs`	numeric specifying the miniumum number of observations required to run the weighted regression, default is 100
`minNumUncen`	numeric specifying the minimum number of uncensored observations to run the weighted regression, default is 50
`edgeAdjust`	logical specifying whether to use the modified method for calculating the windows at the edge of the record. The modified method tends to reduce curvature near the start and end of record. Default is TRUE.
`verbose`	logical specifying whether or not to display progress message

Value

SampleCrossV data frame containing the sample data augmented by the results of the cross-validation exercise

Examples

```
eList <- Choptank_eList
Sample <- getSample(eList)
Daily <- getDaily(eList)
numDays <- length(Daily$DecYear)
DecLow <- Daily$DecYear[1]
DecHigh <- Daily$DecYear[numDays]

SampleCrossV <- estCrossVal(DecLow,DecHigh,Sample)
```

---

estDailyFromSurfaces *Estimates all daily values of Concentration, Flux, Flow-Normalized Concentration, and Flow Normalized Flux*

Description

Uses the surfaces matrix estimated in estSurfaces to estimate 6 daily time series and appends them to the Daily data frame. The time series are (in order): yHat, the estimated natural log of concentration, dimensionless SE, the standard error of the natural log of concentration ConcDay, the estimated concentration in mg/L FluxDay, the estimated flux in kg/day FNConc, the flow-normalized concentration in mg/L FNFlux, the flow-normalized flux in kg/day

Bin the LogQ values by day-of-year.

Usage

```
estDailyFromSurfaces(eList, localsurfaces = NA, localDaily = NA)

getConcFluxFromSurface(eList, allLogQsByDayOfYear, localDaily,
  localsurfaces = NA)

getSurfaceEstimates(eList, localsurfaces = NA, localDaily = NA)

bin_Qs(localDaily)
```

Arguments

`eList`	named list with at least the Daily and INFO dataframes, and the surface matrix
`localsurfaces`	surface over-riding the one stored in eList. Default is NA.
`localDaily`	data frame to override eList$Daily. Default is NA.
`allLogQsByDayOfYear`	list

Details

The results are stored in an augmented version of the Daily data frame, which is returned as part of an EGRET object.

Value

egret object with altered Daily dataframe

Daily dataframe with yHat, SE, ConcDay and FluxDay calulated

Examples

```
eList <- Choptank_eList
################################################
# This is usually done in modelEstimation:
Daily <- getDaily(eList)
surfaceIndexParameters<-surfaceIndex(Daily)
INFO <- eList$INFO
INFO$bottomLogQ<-surfaceIndexParameters[['bottomLogQ']]
INFO$stepLogQ<-surfaceIndexParameters[['stepLogQ']]
INFO$nVectorLogQ<-surfaceIndexParameters[['nVectorLogQ']]
INFO$bottomYear<-surfaceIndexParameters[['bottomYear']]
INFO$stepYear<-surfaceIndexParameters[['stepYear']]
INFO$nVectorYear<-surfaceIndexParameters[['nVectorYear']]
eList$INFO <- INFO
################################################

Daily <- estDailyFromSurfaces(eList)
```

`estSurfaces` *Estimate the three surfaces (for yHat, SE and ConcHat) as a function of DecYear and logQ and store in the three-dimensional object called surfaces*

Description

This function uses weighted survival regression to estimate three surfaces that cover the complete range of DecYear and log(Q) values in the Daily data set. These surfaces are: (1) is the estimated log concentration (yHat), (2) is the estimated standard error (SE), (3) is the estimated concentration (ConcHat). They are mapped as an array that covers the complete space of daily discharge and time. The first index is discharge, layed out in 14 equally spaced levels of log(Q). The second index is time, layed out as 16 increments of the calendar year, starting January 1. It returns the 3 dimensional array called surfaces. This array will be used to estimate these 3 quantities for any given day in the daily values record.

Usage

```
estSurfaces(eList, surfaceStart = NA, surfaceEnd = NA, localSample = NA,
  windowY = 7, windowQ = 2, windowS = 0.5, minNumObs = 100,
  minNumUncen = 50, edgeAdjust = TRUE, verbose = TRUE,
  interactive = NULL, run.parallel = FALSE)
```

Arguments

eList	named list with at least the Sample and Daily dataframes
surfaceStart	Date object for start of surface slice (or character starting date for data retrieval in the form YYYY-MM-DD). Default is NA .
surfaceEnd	Date object for end of surface slice (or character starting date for data retrieval in the form YYYY-MM-DD). Default is NA .
localSample	data frame to override eList$Sample. Default is NA .
windowY	numeric specifying the half-window width in the time dimension, in units of years, default is 7
windowQ	numeric specifying the half-window width in the discharge dimension, units are natural log units, default is 2
windowS	numeric specifying the half-window with in the seasonal dimension, in units of years, default is 0.5
minNumObs	numeric specifying the miniumum number of observations required to run the weighted regression, default is 100
minNumUncen	numeric specifying the minimum number of uncensored observations to run the weighted regression, default is 50
edgeAdjust	logical specifying whether to use the modified method for calculating the windows at the edge of the record. Default is TRUE.
verbose	logical specifying whether or not to display progress message
interactive	logical deprecated. Use 'verbose' instead
run.parallel	logical to run bootstrapping in parallel or not

Value

surfaces array containing the three surfaces estimated, array is 3 dimensional

Examples

```
eList <- Choptank_eList

surfaces <- estSurfaces(eList)

surfaceStart <- "1984-10-01"
surfaceEnd <- "1986-09-30"
surfaces_1 <- estSurfaces(eList, surfaceStart, surfaceEnd)

wall_sample <- head(eList$Sample, n=500)

surface_wall <- estSurfaces(eList, localSample = wall_sample)
```

`fill_missing_daily` *Fill missing daily data*

Description

Uses either linear interpolation, linear interpolation in the log scale, or fixed-interval smoothing on time series to fill missing data. This function gets used within the Daily functions if fill=TRUE. As a standalone function, the input can be data directly download from 'dataRetrieval::read_waterdata_daily'.

Usage

```
fill_missing_daily(df, fill_type, maxgap = 21, value_col = "value",
  qualifier_col = "qualifier")
```

Arguments

`df`	Data frame with at least value and qualifier columns. The names of those columns are defined by value_col and qualifier_col. The data frame is expected to be a complete and uniform time series. This could be one row per day, or one row per X interval. The data in the value_col will be filled in with the assumption that the data is uniform and any missing data is set as 'NA'.
`fill_type`	character to define what process to fill missing data. Options are "interpolation" - linear interpolation from the 'zoo::na.approx', or "log_interp" - linear interpolation in the log space. Only used if fill is set to TRUE.
`maxgap`	Maximum number of NA days allowed for interpolating gaps. Default is 21. Only used if fill is set to TRUE.
`value_col`	Character, name of value column.
`qualifier_col`	Character, name of qualifier column.

Examples

```
Date <- seq(from = as.Date("2001/1/1"),
            to = as.Date("2002/1/2"),
            by = "day")
Qualifier <- rep("",367)
Q <- 2+sin(seq(from = 0, to = 2*pi, length.out = 367))
Q <- jitter(Q, factor = 500)
plot(Q, ylim = c(0, 3.2))
dataInput <- data.frame(time = Date,
                        value = Q,
                        qualifier = Qualifier)
# Remove some rows to test missing:
dataInput$value[4:5] <- NA
dataInput$value[10:20] <- NA

# Linear interpolation:
interp1 <- fill_missing_daily(df = dataInput,
```

```
                                fill_type = "interpolation")
plot(interp1$time[1:30],
     interp1$value[1:30],
     col = as.factor(interp1$qualifier[1:30]),
     type = "b", pch = 16, ylim = c(0, 3.2),
     main = "Linear Interpolation")

# Add a gap that is too big do deal with:
dataInput$value[200:255] <- NA

df_interp <- fill_missing_daily(dataInput,
                                  fill_type = "interpolation")
plot(df_interp$time, df_interp$value,
     col = as.factor(df_interp$qualifier),
     main = "Linear Interpolation",
     type = "b", pch = 16, ylim = c(0, 3.2))
plot(df_interp$time[1:50], df_interp$value[1:50],
     col = as.factor(df_interp$qualifier[1:50]),
     main = "Linear Interpolation",
     type = "b", pch = 16, ylim = c(0, 3.2))

df_log_interp <- fill_missing_daily(dataInput,
                                    fill_type = "log_interp")
plot(df_log_interp$time, df_log_interp$value,
     col = as.factor(df_log_interp$qualifier),
     main = "Linear Interpolation in Log Scale",
     type = "b", pch = 16, ylim = c(0, 3.2))
plot(df_log_interp$time[1:50], df_log_interp$value[1:50],
     col = as.factor(df_log_interp$qualifier[1:50]),
     main = "Linear Interpolation in Log Scale",
     type = "b", pch = 16, ylim = c(0, 3.2))
```

`fillUpElists` *fillUpElists*

Description

Function used in 'runSeries' to populate the yHat, SE, and ConcHat columns into the Daily and Sample data frames within the eList.

Usage

```
fillUpElists(eList)
```

Arguments

`eList`	named list with the INFO, Daily, and Sample dataframes and surfaces matrix

Examples

```
eList <- Choptank_eList
eList$Daily$yHat <- NULL
eList$Daily$SE <- NULL
eList$Sample$SE <- NULL
eList$Sample$yHat <- NULL

eList <- fillUpElists(eList)
head(eList$Daily$yHat)
head(eList$Daily$SE)
head(eList$Sample$SE)
head(eList$Sample$yHat)
```

---

fixSampleFrame *Update Sample dataframe*

Description

Used for updating the Sample dataframe if ConcLow or ConcHigh is manually adjusted. Adjusts ConcAve and Uncen columns.

Usage

```
fixSampleFrame(eList)
```

Arguments

eList	named list with at least the Sample dataframes

Value

localSample data frame

Examples

```
eList <- Choptank_eList
Sample <- eList$Sample
Sample[1,c("ConcLow","ConcHigh")] <- c(NA, 0.01) # Adjusted to left-censored
Sample[2,c("ConcLow","ConcHigh")] <- c(1.1, 1.3) # Adjusted to interval-censored
Sample[3,c("ConcLow","ConcHigh")] <- c(1.3, 1.3) # Simple adjustment
eList$Sample <- Sample
eList <- fixSampleFrame(eList)
eList$Sample[1:3,]
```

flexFN *Flexible Flow Normalization*

Description

This function implements generalized flow normalization. This means that for determining the flow normalized concentration and flow normalized flux for any given year, there is a specified list of years from which to create the discharge record used in the flow-normalization process. That set of years is defined by the dateInfo object.

Usage

```
flexFN(eList, dateInfo, localsurfaces = NA, oldSurface = FALSE,
  flowNormStartCol = "flowNormStart", flowNormEndCol = "flowNormEnd",
  flowStartCol = "flowStart", flowEndCol = "flowEnd")
```

Arguments

eList	named list with at least the Daily, Sample, and INFO dataframes
dateInfo	data frame with 4 columns. The column names and descriptions are described below. Default is NA.
localsurfaces	surface (3-dimensional matrix) over-riding the one stored in eList Default = NA.
oldSurface	logical, if TRUE, use the surface object in eList. Default is FALSE.
flowNormStartCol	character, name of the column in dateInfo that starts the segment for the flow normalization
flowNormEndCol	character, name of the column in dateInfo that ends the segment for the flow normalization
flowStartCol	character, name of the column in dateInfo that starts the segment for the portion of the flow to be populated with flow-normalized values.
flowEndCol	character, name of the column in dateInfo that ends the segment for the portion of the flow to be populated with flow-normalized values.

Value

named list, eList, containing INFO, Daily, Sample, and surfaces objects

Examples

```
eList <- Choptank_eList
eList <- setUpEstimation(eList)
flowNormStart <- c("1979-10-01","1990-01-01","1992-10-10")
flowNormEnd <- c("1995-06-06","2004-03-03","2011-09-29")
flowStart <- c("1979-10-01","1995-06-07","2004-03-04")
flowEnd <- c("1995-06-06","2004-03-03","2011-09-29")
dateInfo <- data.frame(flowNormStart,
```

```
                      flowNormEnd,
                      flowStart,
                      flowEnd,
                      stringsAsFactors = FALSE)

newEList <- flexFN(eList, dateInfo)
plotFluxHist(newEList)
flexPlotAddOn(newEList)

wallSurface <- estSurfaces(eList, localSample = eList$Sample[1:500,])
wallEList <- flexFN(eList, dateInfo, localsurface = wallSurface)
plotFluxHist(wallEList)
```

`flexPlotAddOn` *Flexible Flow Normalization Plot Add On*

Description

Flexible Flow Normalization Plot Add On

Usage

```
flexPlotAddOn(eList, showArrows = TRUE, showRect = TRUE,
  customPalette = NULL)
```

Arguments

`eList`	named list with at least the Daily, Sample, and INFO dataframes
`showArrows`	logical whether or not to show arrows representing flow segments
`showRect`	logical whether or not to show rectangles representing sample segments
`customPalette`	character vector of colors as a hexadecimal string of the form "#rrggbb". Defaults to NULL, which indicates the use of a default palette (up to 21 segments).

Examples

```
eList <- Choptank_eList
eList <- setUpEstimation(eList)
flowNormStart <- c("1979-10-01","1990-01-01","1992-10-10")
flowNormEnd <- c("1995-06-06","2004-03-03","2011-09-29")
flowStart <- c("1979-10-01","1995-06-07","2004-03-04")
flowEnd <- c("1995-06-06","2004-03-03","2011-09-29")
dateInfo <- data.frame(flowNormStart,
                      flowNormEnd,
                      flowStart,
                      flowEnd,
                      stringsAsFactors = FALSE)

newEList <- flexFN(eList, dateInfo)
```

```
plotFluxHist(newEList)
flexPlotAddOn(newEList)

plotFluxHist(newEList)
flexPlotAddOn(newEList, customPalette=c("#d5ce48", "#fd300f", "#3e0289"))
```

`flowDuration` *Computes several values of the flow duration curve for streamflow centered on a specific date of the year*

Description

This function is useful for helping the analyst determine the empirical probability distribution of streamflow for a particular part of the year or for the whole year. This is particularly useful in setting up discharge scales for various other plots in this package.

Usage

```
flowDuration(eList, centerDate = "09-30", qUnit = 2, span = 365,
  monthLab = 1)
```

Arguments

`eList`	named list with at least Daily and INFO dataframes
`centerDate`	character specifying the center date of the part of the year for which the flow duration is to be calculated, it is in the form "mm-dd" (it must be in quotes). Default is "09-30"
`qUnit`	object of qUnit class `printqUnitCheatSheet`, or numeric represented the short code, or character representing the descriptive name. Default is qUnit = 2, which corresponds to cubic meters per second.
`span`	number this is the half-width of the window over which the discharge values are to be used in constructing the flow-duration curve. If the full year is desired any value greater than 182 will. Note that for a window of about 2-months width, a span value should be about 30. Default is 365.
`monthLab`	object of monthLabel class, or numeric represented the short code, or character representing the descriptive name.

Value

qDuration A named vector with flow duration information.

Examples

```
eList <- Choptank_eList
# for a window of 30 days either side of June 25 expressed in units
# of cfs:
flowDuration(eList, "06-25", qUnit = 1, span = 30)
# for a flow-duration curve covering the whole year,
# expressed in units of cms, and returning a data frame of results:
qDuration <- flowDuration(eList, qUnit = 2)
```

`fluxBiasMulti` *Produces 8-panel plot that is useful for determining if there is a flux bias problem*

Description

These plots use the jack-knife estimates from WRTDS to investigate the potential flux bias problem. It can also be used for estimates constructed by other methods (such as LOADEST) if the results are stored in a data frame organized like the Sample data frame. It allows additional label information to indicate what method is used. The use of this plot is described in Hirsch, Robert M., 2014. Large Biases in Regression-Based Constituent Flux Estimates: Causes and Diagnostic Tools. Journal of the American Water Resources Association (JAWRA) 1-24. DOI: 10.1111/jawr.12195

Although there are a lot of optional arguments to this function, most are set to a logical default.

Data come from named list, which contains a Sample dataframe with the sample data, a Daily dataframe with the daily flow data, and an INFO dataframe with metadata.

Usage

```
fluxBiasMulti(eList, qUnit = 2, fluxUnit = 3, moreTitle = "WRTDS",
  cex = 0.7, cex.axis = 1.1, cex.main = 1.1, randomCensored = FALSE,
  col = "black", lwd = 1, concLab = 1, monthLab = 1, ...)
```

Arguments

`eList`	named list with at least Sample, Daily, and INFO dataframes
`qUnit`	object of qUnit class. `printqUnitCheatSheet`, or numeric represented the short code, or character representing the descriptive name.
`fluxUnit`	object of fluxUnit class. `printFluxUnitCheatSheet`, or numeric represented the short code, or character representing the descriptive name.
`moreTitle`	character specifying some additional information to go in figure title, typically some information about the specific estimation method used, default is no additional information
`cex`	numerical value giving the amount by which plotting symbols should be magnified
`cex.axis`	magnification to be used for axis annotation relative to the current setting of cex
`cex.main`	magnification to be used for main titles relative to the current setting of cex

randomCensored	logical, if TRUE plot a random value for censored data. Default is FALSE.
col	color of points on plot, see ?par 'Color Specification'
lwd	number line width
concLab	object of concUnit class, or numeric represented the short code, or character representing the descriptive name. By default, this argument sets concentration labels to use either Concentration or Conc (for tiny plots). Units are taken from the eList$INFO$param.units. To use any other words than "Concentration" see `vignette(topic = "units", package = "EGRET")`.
monthLab	object of monthLabel class, or numeric represented the short code, or character representing the descriptive name.
...	arbitrary graphical parameters that will be passed to genericEGRETDotPlot function (see ?par for options)

Examples

```
eList <- Choptank_eList
# Water year:
fluxBiasMulti(eList)
fluxBiasMulti(eList, fluxUnit = 2)
# Graphs consisting of Jun-Aug
eList <- setPA(eList,paStart=6,paLong=3)
fluxBiasMulti(eList)
```

---

fluxBiasStat *Compute the flux bias statistic: (mean of estimated flux - mean of observed flux) / mean of estimated flux*

Description

Computes three versions of the flux bias: The first where all censored values are set to their miniumum. The second where all censored values are set to their maximum. The third which is the average of the other two. In practice there is rarely a noticable difference among them.

Usage

```
fluxBiasStat(localSample)
```

Arguments

localSample	data frame that contains the concentration data, default name is Sample

Value

fluxBias a vector of three numerical values, a lower bound, upper bound and an average estimate of the ratio of (mean estimated flux - mean observed flux) / mean estimated flux. Typically one should use fluxBias[3]

Examples

```
eList <- Choptank_eList
Sample <- getSample(eList)
fluxBias <- fluxBiasStat(Sample)
```

`fluxUnit-class` *fluxUnit class*

Description

Some details about the fluxUnit class

Details

shortName A character specifying the short name.

unitFactor A numeric representing the conversion factor

unitName A character specifying the full name.

unitExpress An expression specifying the full name starting with Observed.

unitExpressTiny An expression specifying the abbreviated name starting with Observed.

unitEstimate An expression specifying the full name starting with Estimated.

unitEstimateTiny An expression specifying the abbreviated name starting with Estimated.

unitUSGS A character specifying flux with full text.

shortCode A number for quick lookup

`generalAxis` *Axis generation for log discharge*

Description

Discharge axis tick generation

Usage

```
generalAxis(x, maxVal, minVal, units = NA, logScale = FALSE,
  tinyPlot = FALSE, padPercent = 5, concentration = TRUE, concLab = 1,
  usgsStyle = FALSE, prettyDate = TRUE)
```

Arguments

x	vector to create scale about
maxVal	number maximum value on returned scale
minVal	number minimum value on returned scale
units	character concentration units. Typically found in INFO$param.units.
logScale	logical whether or not to return a log scale
tinyPlot	logical
padPercent	number used to pad the max and min if not specified
concentration	logical if concentration=TRUE, labels returned as concentration units, otherwise flux units.
concLab	object of concUnit class, or numeric represented the short code, or character representing the descriptive name. By default, this argument sets concentration labels to use either Concentration or Conc (for tiny plots). Units are taken from the eList$INFO$param.units. To use any other words than "Concentration" see `vignette(topic = "units", package = "EGRET")`.
usgsStyle	logical option to use USGS style guidelines. Setting this option to TRUE does NOT guarantee USGS compliance. It will only change automatically generated labels
prettyDate	logical use 'pretty' limits for date axis if TRUE, or force the yearStart/yearEnd as limits if FALSE

Examples

```
eList <- Choptank_eList
Daily <- getDaily(eList)
INFO <- getInfo(eList)
x <- Daily$Q
max <- max(x)
min <- 0
units <- INFO$param.units
generalAxis(x, max, min, units)
min <- min(x)
generalAxis(x, max, min, units, log=TRUE)
generalAxis(Daily$ConcDay, 100, 0, concLab = "concentration")
```

---

genericEGRETDotPlot *Generic EGRET plotting function*

Description

Basic plotting framework for EGRET dot plots. Graphical parameters default to values that work well with most plots, but all can be re-assigned. See ?par for complete definitions of most optional input variables.

Usage

```
genericEGRETDotPlot(x, y, xlim, ylim, xTicks = pretty(xlim),
  yTicks = pretty(ylim), printTitle = TRUE, xaxs = "i", xlab = "",
  yaxs = "i", ylab = "", plotTitle = "", pch = 20, cex = 0.7,
  cex.main = 1.3, font.main = 2, cex.lab = 1.2, tcl = 0.5,
  cex.axis = 1, las = 1, xDate = FALSE, tinyPlot = FALSE,
  hLine = FALSE, oneToOneLine = FALSE, rmSciX = FALSE, rmSciY = FALSE,
  customPar = FALSE, col = "black", lwd = 1, showXLabels = TRUE,
  showYLabels = TRUE, showXAxis = TRUE, showYAxis = TRUE,
  removeFirstX = FALSE, removeLastX = FALSE, removeFirstY = FALSE,
  removeLastY = FALSE, ...)
```

Arguments

x	vector specifying the x data (required)
y	vector specifying the y data (required)
xlim	vector specifying the x plotting range (required)
ylim	vector specifying the y plotting range (required)
xTicks	vector specifying x axis tick placement (required)
yTicks	vector specifying y axis tick placement (required)
printTitle	logical defaults to TRUE, plotting parameter to control whether to have title
xaxs	character defaults to "i", defines the style of x-axis interval calculation. Possible values are i, r, e, s, d.
xlab	character defaults to "", defines the x label
yaxs	character defaults to "i", defines the style of y-axis interval calculation. Possible values are i, r, e, s, d.
ylab	character defaults to "", defines the y label
plotTitle	character defaults to "", defines the plot title
pch	number defaults to 20, specifies plot symbol
cex	number defaults to 0.7, specifies plotting text magnification
cex.main	number defaults to 1.3, specifies title text magnification
font.main	number defaults to 2, specifies which font to use for text
cex.lab	number defaults to 1.2 specifies label text magnification
tcl	number defaults to 0.5, specifies length of tick marks as fraction of height of a line of text.
cex.axis	number defaults to 1, specifies axis text magnification
las	number represents style of axis labels
xDate	logical defaults to FALSE, changes x label to "year-month" format if set to TRUE and total years less than 4.
tinyPlot	logical defaults to FALSE, if TRUE, changes defaults to be appropriate for multi-plot

hLine	logical defaults to FALSE, inserts horizontal line at zero
oneToOneLine	logical defaults to FALSE, inserts 1:1 line
rmSciX	logical defaults to FALSE, changes x label from scientific to fixed
rmSciY	logical defaults to FALSE, changes y label from scientific to fixed
customPar	logical defaults to FALSE. If TRUE, par() should be set by user before calling this function
col	color of points on plot, see ?par 'Color Specification'
lwd	number line width
showXLabels	logical defaults to TRUE. If FALSE, the x axis label is not plotted
showYLabels	logical defaults to TRUE. If FALSE, the y axis label is not plotted
showXAxis	logical defaults to TRUE. If FALSE, the x axis is not plotted
showYAxis	logical defaults to TRUE. If FALSE, the y axis is not plotted
removeFirstX	logical defaults to FALSE. If TRUE, removes the first x axis label. This can be handy for plotting mutliple plots.
removeLastX	logical defaults to FALSE. If TRUE, removes the last x axis label. This can be handy for plotting mutliple plots.
removeFirstY	logical defaults to FALSE. If TRUE, removes the first y axis label. This can be handy for plotting mutliple plots.
removeLastY	logical defaults to FALSE. If TRUE, removes the last y axis label. This can be handy for plotting mutliple plots.
...	additional graphical parameters can be adjusted

Examples

```
eList <- Choptank_eList
Daily <- getDaily(eList)
x <- Daily$Date
y <- Daily$Q
xlim <- c(min(x),max(x))
ylim <- c(min(y),1.05*max(y))
xlab <- "Date"
ylab <- "Flow"
genericEGRETDotPlot(x=x, y=y,
                    xlim=xlim, ylim=ylim,
                    xlab=xlab, ylab=ylab,
                    plotTitle="Test"
)
```

genmissing *genmissing*

Description

Generates a lag one auto-regressive time series, where the first and last values are fixed. Marginal expected value is zero and variance is one. Generated values have a normal conditional distribution. This version replaces genmissing and is much faster for long sample gaps.

Usage

```
genmissing(X1, XN, rho, N, Lfull)
```

Arguments

X1	value before the gap
XN	value after the gap
rho	the lag one autocorrelation
N	the length of the sequence including X1 and XN. It is two more than the gap length
Lfull	Cholesky factor created with precompute_L

Value

numeric vector of length N, conditioned on the first value (X1) and last value (XN) with the specified lag one autocorrelation in the limit (where N is large) the values are normal with mean 0 and variance 1

getDaily *Get Daily dataframe from EGRET object*

Description

From a named list or EGRET object, extract the Daily dataframe

Usage

```
getDaily(x, ...)

## S3 method for class 'egret'
getDaily(x, ...)

## Default S3 method:
getDaily(x, ...)
```

Arguments

x	EGRET object or named list
...	additional parameters

Value

Daily dataframe

See Also

readNWISDaily, readNWISSample

Examples

```
eList <- Choptank_eList
Daily <- getDaily(eList)
```

---

getInfo *Get INFO dataframe from EGRET object*

Description

From a named list or EGRET object, extract the INFO dataframe

Usage

```
getInfo(x, ...)

## S3 method for class 'egret'
getInfo(x, ...)

## Default S3 method:
getInfo(x, ...)
```

Arguments

x	EGRET object or named list
...	additional parameters

Value

INFO dataframe

See Also

readNWISDaily, readNWISSample

Examples

```
eList <- Choptank_eList
INFO <- getInfo(eList)
```

`getSample` *Get Sample dataframe from EGRET object*

Description

From a named list or EGRET object, extract the Sample dataframe

Usage

```
getSample(x, ...)

getSample(x, ...)

getSample.default(x, ...)
```

Arguments

`x`	EGRET object or named list
`...`	additional parameters

Value

Sample dataframe

See Also

`readNWISDaily`, `readNWISSample`

Examples

```
eList <- Choptank_eList
Sample <- getSample(eList)
```

getSurfaces *Get surfaces matrix from EGRET object*

Description

From a named list or EGRET object, extract the surfaces matrix

Usage

```
getSurfaces(x, ...)

## S3 method for class 'egret'
getSurfaces(x, ...)

## Default S3 method:
getSurfaces(x, ...)
```

Arguments

x	EGRET object or named list
...	additional parameters

Value

Sample dataframe

See Also

readNWISDaily, readNWISSample

Examples

```
eList <- Choptank_eList
surfaces <- getSurfaces(eList)
```

INFOdataframe *Import metadata to create INFO data frame*

Description

Populates INFO data frame from either NWIS (readNWISInfo), Water Quality Portal (readWQPInfo), or user-supplied files (readUserInfo).

Usage

```
readNWISInfo(siteNumber, parameterCd, interactive = TRUE)

readWQPInfo(siteNumber, parameterCd, interactive = TRUE)

readUserInfo(filePath, fileName, hasHeader = TRUE, separator = ",",
  interactive = TRUE)
```

Arguments

siteNumber	character site number. For readNWISInfo, this is usually an 8 digit number, for readWQPInfo, it is usually a longer code. For instance, a USGS site number in the Water Quality Portal would be in the form 'USGS-XXXXXXXX'. If the siteNumber is left blank (an empty string), the interactive option allows users to enter required information by hand, otherwise those fields are left blank.
parameterCd	character USGS parameter code (a 5 digit number) or characteristic name (if using readWQPInfo). If the parameterCd is left blank (an empty string), the interactive option allows users to enter required information by hand, otherwise those fields are left blank.
interactive	logical Option for interactive mode. If true, there is user interaction for error handling and data checks.
filePath	character specifying the path to the file (used in readUserInfo)
fileName	character name of file to open (used in readUserInfo)
hasHeader	logical true if the first row of data is the column headers (used in readUserInfo)
separator	character that separates data cells (used in readUserInfo)

Value

INFO data frame. Any metadata can be stored in INFO. However, there are 8 columns that EGRET uses by name in some functions:

Required column	Used in function	Description
param.units***	All concentration plotting functions	The units as listed in this field are used to create the concentration a
shortName	All plotting functions	Station short name, used to label plots
paramShortName	All plotting functions	Parameter short name, used to label plots
drainSqKm	plotFlowSingle, printSeries	Calculate runoff
constitAbbrev	saveResults	Parameter abbrieviation, used to auto-name workspace
staAbbrev	saveResults	Station abbrieviation, used to auto-name workspace
paStart	Most EGRET functions	Starting month of period of analysis. Defaults to 10
paLong	Most EGRET functions	Length in number of months of period of analysis. Defaults to 12

*** Additionally, EGRET assumes that all concentrations are saved in mg/l. If some variation of 'mg/l' is not found in INFO$param.units, functions that calculate flux will issue a warning. This is because the conversion from mg/l to the user-specified flux unit (e.g., kg/day) uses hard-coded conversion factors.

See Also

readNWISsite, readNWISpCode

whatWQPsites

Examples

```
# These examples require an internet connection to run
# Automatically gets information about site 05114000 and temperature

INFO <- readNWISInfo('05114000','00010',interactive = FALSE)



# These examples require an internet connection to run
# Automatically gets information about site 01594440 and temperature, no interaction with user
nameToUse <- 'Specific conductance'
pcodeToUse <- '00095'

INFO <- readWQPInfo('USGS-04024315',pcodeToUse, interactive = FALSE)

INFO2 <- readWQPInfo('WIDNR_WQX-10032762',nameToUse, interactive = FALSE)
# To adjust the label names:
INFO$shortName <- "Little"
INFO$paramShortName <- "SC"


filePath <- system.file("extdata", package="EGRET")
fileName <- 'infoTest.csv'
INFO <- readUserInfo(filePath,fileName, separator=",",interactive=FALSE)
```

---

is.egret *Check for EGRET object*

Description

Checks object to see if it is an EGRET object

Usage

```
is.egret(x)
```

Arguments

x	object to check

Value

logical

Examples

```
eList <- Choptank_eList
is.egret(eList)
```

`jitterSam` *jitter Sample*

Description

This function is used in cases where there are numerical problems with the estimation of the WRTDS model. This mostly happens during bootstrap estimation or when the data sets are very large. In order to reduce the collinearity in the explanatory variables, some random noise is added to the time and log discharge variables in the Sample data frame.

Usage

```
jitterSam(Sam, V = 0.2)
```

Arguments

`Sam`	data frame with at least columns DecYear and LogQ
`V`	a multiplier for the sd of the LogQ jitter. for example V = 0.02, means that the sd of the LnQ jitter is 0.02*sdLQ

Value

SamR a data frame structured like the Sam data frame but with the time and discharge variables modified by adding random jitter

Examples

```
eList <- Choptank_eList
Sample_jitter <- jitterSam(eList$Sample)
```

`logPretty1` *Sets up tick marks for an axis with a log scale, where the graph is small*

Description

Axis tick marks for a log scale for cases where the data cover many orders of magnitude and the graph is small. These tick marks are designed to progress by factors of 10.

Usage

```
logPretty1(xMin, xMax)
```

Arguments

xMin	A numeric value for the minimum value to be plotted, it must be > 0
xMax	A numeric value for the maximum value to be plotted, it must be > xMax

Value

xTicks A vector representing the values for each of the tick marks

Examples

```
xMin <- 0.7
xMax <- 990000
logPretty1(xMin, xMax)
xMin <- 3
xMax <- 15
logPretty1(xMin, xMax)
```

---

logPretty3 *Sets up tick marks for an axis with a log scale*

Description

Axis tick marks for a log scale. These tick marks are designed to progress with 3 tick marks for every factor of 10. For example: 2,5,10,20,50,100,200,500.

Usage

```
logPretty3(xMin, xMax)
```

Arguments

xMin	A numeric value for the minimum value to be plotted, it must be >0
xMax	A numeric value for the maximum value to be plotted, it must be >xMax

Value

xTicks A vector representing the values for each of the tick marks

Examples

```
logPretty3(0.7, 990000)
logPretty3(3, 15)
```

`makeAnnualSeries` *Produces annual series of 8 streamflow statistics (and a lowess smooth of them) from daily streamflow data*

Description

Part of the flowHistory system. The data come from Daily and INFO data frames. Note that the function setPA must be run before this to establish the period of analysis (e.g. water year).

Usage

```
makeAnnualSeries(eList, edgeAdjust = TRUE)
```

Arguments

`eList`	named list with at least Daily and INFO dataframes
`edgeAdjust`	logical specifying whether to use the modified method for calculating the windows at the edge of the record. The modified method tends to reduce curvature near the start and end of record. Default is TRUE, but a logical in INFO$edgeAdjust will override the default.

Details

istat	Name
1	minimum 1-day daily mean discharge
2	minimum 7-day mean of the daily mean discharges
3	minimum 30-day mean of the daily mean discharges
4	median of the daily mean discharges
5	mean of the daily mean discharges
6	maximum 30-day mean of the daily mean discharges
7	maximum 7-day mean of the daily mean discharges
8	maximum 1-day daily mean discharge

The smooth is a loess smooth computed on the log flow values and then transformed back to real space Smoothing window is a fixed number of years, the window width has a default value of 20 years but can be modified by changing its value in INFO data frame (using setPA function)

Value

annualSeries matrix that contains the annual series of streamflow statistics annualSeries is a matrix 3 * 8 * numYears, where numYears is the number of years in the data set in the first dimension 1 is the year, 2 is the actual value, 3 is the smoothed value in the second dimension, the index is the istat value (identifying the flow statistic) the third dimension is year

Examples

```
eList <- Choptank_eList
annualSeries <- makeAnnualSeries(eList)
```

`makeAugmentedSample` *Create randomized residuals and observations for data sets that have some censored data*

Description

This function is used to add two columns to the Sample data frame: rResid and rObserved. rResid is the randomized residual value computed in log concentration units, and rObserved is the randomized 'observed' value of concentration in concentration units. Both of these are computed for all censored samples ("less than values"). They are created for purposes of plotting and are not used in any computations in EGRET.

Usage

```
makeAugmentedSample(eList)
```

Arguments

`eList`	named list with at least the Sample dataframe

Details

The WRTDS model must be estimated before this function can be run. The random value that is generated lies between the reporting limit and zero and is distributed as a truncated log-normal distribution, with parameters derived from the fitted WRTDS model. These random values are never used in any computations in EGRET but are used for purposes of plotting the data set or residuals. When plotted in other functions they are shown as open circles.

Value

eList named list with modified Sample data frame.

Examples

```
choptankAugmented <- makeAugmentedSample(Choptank_eList)
```

makeDateInfo *makeDateInfo*

Description

Create a data frame that organizes date segmentations for runSeries.

Usage

```
makeDateInfo(windowSide, surfaceStart, surfaceEnd, firstQDate0, lastQDate0)
```

Arguments

windowSide	integer number of automatically generated span sections, default is 7. If NA, code will use
surfaceStart	character (or Date) in YYYY-MM-DD. Date on which we want the analysis to start, it must be at or after the
surfaceEnd	character (or Date) in YYYY-MM-DD. Date on which we want the analysis to end, it must be at or before the end of
firstQDate0	character (or Date) in YYYY-MM-DD. The first day used in flow normalizing distributions, default is the start of eList$Daily
lastQDate0	character (or Date) in YYYY-MM-DD. The last day used in flow normalizating distributions, default is the end of eList$Daily

Examples

```
windowSide <- 7
surfaceStart <- "1984-01-01"
surfaceEnd <- "2012-12-31"
firstQDate0 <- "1970-01-01"
lastQDate0 <- "2014-06-01"
dateInfo <- makeDateInfo(windowSide,
                         surfaceStart, surfaceEnd,
                         firstQDate0, lastQDate0)
```

mergeReport *mergeReport*

Description

This function does three things. 1) It transfers the daily discharge value from the Daily data frame to to Sample data frame for those days with samples. 2) It merges the INFO, Daily and Sample data frames to form an eList object, 3) and it prints out a "report" of basic information about the Daily and Sample data frames.

Usage

```
mergeReport(INFO, Daily, Sample = NA, surfaces = NA, verbose = TRUE,
  interactive = NULL)
```

Arguments

INFO	dataframe metadata about the Sample and Daily data frames.
Daily	dataframe containing the daily discharge data
Sample	dataframe containing the sample data
surfaces	matrix returned from modelEstimation. Default is NA.
verbose	logical specifying whether or not to display summary information on the Daily and Sample dataframes.
interactive	logical deprecated. Use 'verbose' instead

Details

There must be an INFO and a Daily data frame for this function to work. That would be the case for a study of flow only, with no consideration of water quality. If water quality is being considered then INFO, Daily, and Sample all need to be provided in the call to this function.

Note that the Sample dataframe in the global environment does not update with the flow information.

Value

eList named list with at least INFO, and Daily data frames. It can also include a Sample data frame.

See Also

readNWISDaily, readNWISSample

Examples

```
siteNumber <- '01491000'
pCode <- '00631'

Daily <- readNWISDaily(siteNumber,'00060', '1984-10-01', '')
Sample <- readNWISSample(siteNumber,pCode, '1984-10-01', '')
INFO <- readNWISInfo(siteNumber,pCode, interactive = FALSE)
eList <- mergeReport(INFO, Daily, Sample)
Sample <- eList$Sample
plot(eList)

# Create eList with no water quality data:

eList <- mergeReport(INFO, Daily, Sample = NA)
plotFour(eList)
```

`modelEstimation` *Estimation process for the WRTDS (Weighted Regressions on Time, Discharge, and Season)*

Description

This one function does three things. 1) a jack-knife cross-validation of a WRTDS model in which it augments the Sample data frame in the eList, 2) fits the WRTDS model creating the surfaces matrix and places it in the eList (the surfaces matrix expresses the estimated concentration as a function of discharge and time), and 3) estimates the daily values of concentration and flux, and flow normalized concentration and flux and places these in the Daily data frame in the eList. It returns a named list with the following dataframes: Daily, INFO, Sample, and the matrix: surfaces.

Usage

```
modelEstimation(eList, windowY = 7, windowQ = 2, windowS = 0.5,
  minNumObs = 100, minNumUncen = 50, edgeAdjust = TRUE, verbose = TRUE,
  run.parallel = FALSE)
```

Arguments

`eList`	named list with at least the INFO, Daily, and Sample dataframes
`windowY`	numeric specifying the half-window width in the time dimension, in units of years, default is 7
`windowQ`	numeric specifying the half-window width in the discharge dimension, units are natural log units, default is 2
`windowS`	numeric specifying the half-window with in the seasonal dimension, in units of years, default is 0.5
`minNumObs`	numeric specifying the minimum number of observations required to run the weighted regression, default is 100
`minNumUncen`	numeric specifying the minimum number of uncensored observations to run the weighted regression, default is 50
`edgeAdjust`	logical specifying whether to use the modified method for calculating the windows at the edge of the record. The edgeAdjust method tends to reduce curvature near the start and end of record. Default is TRUE.
`verbose`	logical specifying whether or not to display progress message
`run.parallel`	logical to run WRTDS in parallel or not

Value

eList named list with INFO, Daily, and Sample dataframes, along with the surfaces matrix.

Examples

```
eList <- Choptank_eList

eList <- modelEstimation(eList)
```

`monthLabel-class` *monthLabel class*

Description

Some details about the monthLabel class

Details

monthAbbrev A character specifying the abbreviated month name.

monthFull A character specifying the full month name

monthSingle A character specifying the single letter of the month.

`multiPlotDataOverview` *Produces a 4 panel plot that gives an overview of the data set prior to any processing*

Description

This function produces the 4 plots based only on the data stored in the eList. The four plots are 1) log concentration versus log discharge, 2) log concentration versus time 3) a boxplot of log concentration by month, and 4) a side-by-side boxplot of the sampled discharges and all daily discharges. To save space, the graphic is labeled only at the top of the 4 graph display.

Although there are a lot of optional arguments to this function, most are set to a logical default.

Usage

```
multiPlotDataOverview(eList, qUnit = 2, cex.main = 1.2,
  randomCensored = FALSE, logScaleConc = TRUE, logScaleQ = TRUE,
  concLab = 1)
```

Arguments

eList	named list with at least Daily, Sample, and INFO dataframes
qUnit	object of qUnit class printqUnitCheatSheet, or numeric represented the short code, or character representing the descriptive name.
cex.main	magnification to be used for main titles relative to the current setting of cex
randomCensored	logical. Show censored values as randomized. Default is FALSE. If TRUE, makeAugmentedSample must be run first.
logScaleConc	logical if TRUE y in concentration graphs plotted in log axis. Default is TRUE.
logScaleQ	logical if TRUE y in streamflow graphs plotted in log axis. Default is TRUE.
concLab	object of concUnit class, or numeric represented the short code, or character representing the descriptive name. By default, this argument sets concentration labels to use either Concentration or Conc (for tiny plots). Units are taken from the eList$INFO$param.units. To use any other words than "Concentration" see vignette(topic = "units", package = "EGRET").

See Also

plotConcQ, boxConcMonth, plotConcTime, boxQTwice

Examples

```
eList <- Choptank_eList
# Water year:
multiPlotDataOverview(eList, qUnit=1)
# Graphs consisting of Jun-Aug
eList <- setPA(eList, paStart=6,paLong=3)
multiPlotDataOverview(eList, qUnit=1)

# Custom axes:
eList$INFO$param.units <- "ng"
qConst_precip <- new("qUnit",
                     qShortName = "   mm  ",
                     qUnitFactor = 1,
                     qUnitName = "Millimeter",
                     qUnitExpress = expression(paste("Precipitation in ",mm)),
                     qUnitTiny = expression(paste("Precipitation ", "(", mm, ")")),
                     shortCode = 1,
                     unitUSGS = "Precipitation, in mm",
                     prefix = "Precipitation")

deposition <- new("concUnit",
                  longPrefix = "Deposition",
                  shortPrefix = "Dep")

multiPlotDataOverview(eList,
                      qUnit = qConst_precip,
                      concLab = deposition)
```

plot15 *Makes 15 graphs of streamflow statistics on a single page. These encompass the 7-day minimum, mean, and 1-day maximum for each of the following 5 Periods of Analysis: Annual, Fall, Winter, Spring and Summer.*

Description

Part of flowHistory system. All results are expressed as runoff (mm/day). The individual plots are constructed by the same method as used in plotFlowSingle. The annual results are based on the Water Year. The seasons are defined as the following groups of months: SON, DJF, MAM, JJA.

Usage

```
plot15(eList, yearStart, yearEnd)
```

Arguments

eList	named list with at least the Daily and INFO dataframes
yearStart	A numeric value for year in which the graph should start, default is NA, which indicates that the graph should start with first annual value
yearEnd	A numeric value for year in which the graph should end, default is NA, which indicates that the graph should end with last annual value

Details

For formatting purposes it is best to use the following commands before calling the plot15 function (savePath is the pathname for directory to store the output) # plotName <- paste(savePath, "plot15.", eList$INFO$shortName, ".ps", sep = "") # postscript(file = plotName, width = 8, height = 10, horizontal = FALSE, family = "Helvetica") Then after running plot15, the user needs to give the command dev.off()

See Also

plot1of15

Examples

```
eList <- Choptank_eList

plot15(eList, yearStart = 1980, yearEnd = 2010)
dev.off()
```

plot1of15 *plots 1 of the 15 graphs of streamflow statistics on a single page*

Description

Part of the flowHistory system. It is designed to create each of the component graphs for the function plot15. The 15 graphs include annual and four seasonal graphs for each of 3 flow statistics: 1-day maximum, mean, and 7-day minimum. The computations involved are the same as the ones used in plotFlowSingle or in makeAnnualSeries

Usage

```
plot1of15(eList, yearStart, yearEnd, qf, istat, isBottom = FALSE)
```

Arguments

eList	named list with at least the Daily and INFO dataframes
yearStart	A numeric value for the year in which the graph should start
yearEnd	A numeric value for the year in which the graph should end
qf	a scale factor to convert discharge in cubic feet per second to mm/day, it is 86 / (drainage area in square kilometers)
istat	A numeric value selecting the flow statistic to be plotted, must be an integer from 1 to 8
isBottom	logical, if TRUE the graph is from the bottom row and thus needs x axis labels, if FALSE it does not need labels

Examples

```
eList <- Choptank_eList
plot1of15(eList, 1980, 2010, 0.2938476, 5)
```

plotConcHist *Graph of annual concentration and flow normalized concentration versus year*

Description

Data come from named list (eList), which contains a Daily dataframe with the daily flow data, and an INFO dataframe with metadata.

The annual concentrations are "time-weighted" mean concentrations (as opposed to "flow-weighted"). The annual results reported are for a specified "period of analysis" which can be an entire water year, a calendar, a season or even an individual month. User specifies this period of analysis in the call to setupYears.

User can specify plotting of three possible series. All are in units of mg/L. Annual mean concentration WRTDS_K version of annual mean concentration (requires that WRTDSKalman has been run) Flow normalized mean concentration

Although there are a lot of optional arguments to this function, most are set to a logical default.

Usage

```
plotConcHist(eList, yearStart = NA, yearEnd = NA, concMax = NA,
  printTitle = TRUE, tinyPlot = FALSE, usgsStyle = FALSE,
  plotFlowNorm = TRUE, plotAnnual = TRUE, plotGenConc = FALSE,
  cex = 0.8, cex.axis = 1.1, cex.main = 1.1, lwd = 2, col = "black",
  col.pred = "green", concLab = 1, col.gen = "red", customPar = FALSE,
  ...)
```

Arguments

eList	named list with at least the Daily and INFO dataframes
yearStart	numeric is the calendar year containing the first estimated annual value to be plotted, default is NA (which allows it to be set automatically by the data)
yearEnd	numeric is the calendar year just after the last estimated annual value to be plotted, default is NA (which allows it to be set automatically by the data)
concMax	numeric. Maximum value of concentration to be plotted.
printTitle	logical variable if TRUE title is printed, if FALSE title is not printed (this is best for a multi-plot figure)
tinyPlot	logical variable, if TRUE plot is designed to be plotted small, as a part of a multipart figure, default is FALSE
usgsStyle	logical option to use USGS style guidelines. Setting this option to TRUE does NOT guarantee USGS compliance. It will only change automatically generated labels
plotFlowNorm	logical variable if TRUE flow normalized line is plotted, if FALSE not plotted
plotAnnual	logical variable if TRUE, annual concentration points from WRTDS output are plotted, if FALSE not plotted
plotGenConc	logical variable. If TRUE, annual concentration points from WRTDSKalman output are plotted, if FALSE not plotted
cex	numerical value giving the amount by which plotting symbols should be magnified
cex.axis	magnification to be used for axis annotation relative to the current setting of cex
cex.main	magnification to be used for main titles relative to the current setting of cex
lwd	number magnification of line width.
col	color of points on plot, see ?par 'Color Specification'
col.pred	color of flow normalized line on plot, see ?par 'Color Specification'

concLab	object of concUnit class, or numeric represented the short code, or character representing the descriptive name. By default, this argument sets concentration labels to use either Concentration or Conc (for tiny plots). Units are taken from the eList$INFO$param.units. To use any other words than "Concentration" see `vignette(topic = "units", package = "EGRET")`.
col.gen	color of points for WRTDS_K output on plot, see ?par 'Color Specification'
customPar	logical defaults to FALSE. If TRUE, par() should be set by user before calling this function (for example, adjusting margins with par(mar=c(5,5,5,5))). If customPar FALSE, EGRET chooses the best margins depending on tinyPlot.
...	arbitrary graphical parameters that will be passed to genericEGRETDotPlot function (see ?par for options)

See Also

setupYears, genericEGRETDotPlot

Examples

```
yearStart <- 2001
yearEnd <- 2010
eList <- Choptank_eList

plotConcHist(eList, yearStart, yearEnd)
```

---

plotConcPred *Plot of Observed Concentration versus Estimated Concentration*

Description

Data come from named list, which contains a Sample dataframe with the sample data, and an INFO dataframe with metadata.

Although there are a lot of optional arguments to this function, most are set to a logical default.

Usage

```
plotConcPred(eList, concMax = NA, logScale = FALSE, printTitle = TRUE,
  tinyPlot = FALSE, cex = 0.8, cex.axis = 1.1, cex.main = 1.1,
  customPar = FALSE, col = "black", lwd = 1, randomCensored = FALSE,
  concLab = 1, usgsStyle = FALSE, ...)
```

Arguments

eList	named list with at least the Sample and INFO dataframes
concMax	number specifying the maximum value to be used on the vertical axis, default is NA (which allows it to be set automatically by the data)

logScale	logical, default TRUE, TRUE indicates x and y axes are on a log scale. FALSE indicates both x and y are on an arithmetic scale.
printTitle	logical variable if TRUE title is printed, if FALSE not printed (this is best for a multi-plot figure)
tinyPlot	logical variable, if TRUE plot is designed to be plotted small, as a part of a multipart figure, default is FALSE
cex	numerical value giving the amount by which plotting symbols should be magnified
cex.axis	magnification to be used for axis annotation relative to the current setting of cex
cex.main	magnification to be used for main titles relative to the current setting of cex
customPar	logical defaults to FALSE. If TRUE, par() should be set by user before calling this function (for example, adjusting margins with par(mar=c(5,5,5,5))). If customPar FALSE, EGRET chooses the best margins depending on tinyPlot.
col	color of points on plot, see ?par 'Color Specification'
lwd	number line width
randomCensored	logical. Show censored values as randomized.
concLab	object of concUnit class, or numeric represented the short code, or character representing the descriptive name. By default, this argument sets concentration labels to use either Concentration or Conc (for tiny plots). Units are taken from the eList$INFO$param.units. To use any other words than "Concentration" see `vignette(topic = "units", package = "EGRET")`.
usgsStyle	logical option to use USGS style guidelines. Setting this option to TRUE does NOT guarantee USGS compliance. It will only change automatically generated labels
...	arbitrary graphical parameters that will be passed to genericEGRETDotPlot function (see ?par for options)

See Also

selectDays, genericEGRETDotPlot

Examples

```
eList <- Choptank_eList
# Water year:
plotConcPred(eList)

# Graphs consisting of Jun-Aug
eList <- setPA(eList, paStart=6,paLong=3)
plotConcPred(eList)
```

plotConcQ *Plot of Observed Concentration versus Discharge*

Description

Data come from named list, which contains a Sample dataframe with the sample data, and an INFO dataframe with metadata. Discharge is plotted on a log scale.

Although there are a lot of optional arguments to this function, most are set to a logical default.

Usage

```
plotConcQ(eList, qUnit = 2, tinyPlot = FALSE, logScale = FALSE,
  randomCensored = FALSE, concMax = NA, concMin = NA,
  printTitle = TRUE, cex = 0.8, cex.axis = 1.1, cex.main = 1.1,
  usgsStyle = FALSE, rmSciX = FALSE, rmSciY = FALSE, customPar = FALSE,
  col = "black", lwd = 1, concLab = 1, ...)
```

Arguments

eList	named list with at least the Sample and INFO dataframes
qUnit	object of qUnit class printqUnitCheatSheet, or numeric represented the short code, or character representing the descriptive name.
tinyPlot	logical variable, if TRUE plot is designed to be plotted small as part of a multi-part figure, default is FALSE.
logScale	logical if TRUE x and y plotted in log axis
randomCensored	logical. Show censored values as randomized.
concMax	number specifying the maximum value to be used on the vertical axis, default is NA (which allows it to be set automatically by the data)
concMin	numeric value for lower limit on concentration shown on the vertical log graph, default is NA (which causes the lower limit to be set automatically, based on the data). This value is ignored for linear scales, using 0 as the minimum value for the concentration axis.
printTitle	logical variable if TRUE title is printed, if FALSE title is not printed (this is best for a multi-plot figure)
cex	numerical value giving the amount by which plotting symbols should be magnified
cex.axis	magnification to be used for axis annotation relative to the current setting of cex
cex.main	magnification to be used for main titles relative to the current setting of cex
usgsStyle	logical option to use USGS style guidelines. Setting this option to TRUE does NOT guarantee USGS compliance. It will only change automatically generated labels.
rmSciX	logical defaults to FALSE, changes x label from scientific to fixed
rmSciY	logical defaults to FALSE, changes y label from scientific to fixed

customPar	logical defaults to FALSE. If TRUE, par() should be set by user before calling this function (for example, adjusting margins with par(mar=c(5,5,5,5))). If customPar FALSE, EGRET chooses the best margins depending on tinyPlot.
col	color of points on plot, see ?par 'Color Specification'
lwd	number line width
concLab	object of concUnit class, or numeric represented the short code, or character representing the descriptive name. By default, this argument sets concentration labels to use either Concentration or Conc (for tiny plots). Units are taken from the eList$INFO$param.units. To use any other words than "Concentration" see `vignette(topic = "units", package = "EGRET")`.
...	arbitrary graphical parameters that will be passed to genericEGRETDotPlot function (see ?par for options)

Details

The function has two possible ways to plot censored values (e.g. "less-than values").

The default is to plot them as a vertical line that goes from the reporting limit down to the bottom of the graph.

The alternative is to set randomCensored = TRUE. In this case a random value is used for plotting the individual sample value. This random value lies between the reporting limit and zero and it is distributed as a truncated log normal based on the fitted WRTDS model.

The function makeAugmentedSample must be run first if randomCensored = TRUE. Running makeAugmentedSample requires that modelEstimation has already been run.

These random censored values are used to create more readable plots and are not used in any computations about the data set. The random censored values are shown as open circles and the non-censored data are shown as filled dots.

See Also

selectDays, genericEGRETDotPlot

Examples

```
eList <- Choptank_eList
# Water year:
plotConcQ(eList)
plotConcQ(eList, logScale=TRUE)
# Graphs consisting of Jun-Aug
eList <- setPA(eList, paStart=6,paLong=3)
plotConcQ(eList)
```

plotConcQSmooth *Plot up to three curves representing the concentration versus discharge relationship. Each curve is a different point in time.*

Description

These plots are like a vertical slice of the estimated concentration surface that is seen in the plotContours function. These plots show how the concentration-discharge relationship is changing over time. Typically the time points selected would be in three years at the same time of year spaced out over the period of record. But that is not necessary. Another possibility is to use this to explore seasonal differences. In this case the three dates would be in the same year but different times during the year.

This plot can also help identify situations where the windowQ may be too small. If there are substantial oscillations of some of the curves, then the windowQ should be increased. Alternatively, windowQ may be too large. This can be seen when the windowQ is reduced (say to 1.0). A good choice of windowQ would be a value just great enough to damp out oscillations in the curves.

Although there are a lot of optional arguments to this function, most are set to a logical default.

Data come from named list, which contains a Sample dataframe with the sample data, and an INFO dataframe with metadata.

Usage

```
plotConcQSmooth(eList, date1, date2, date3, qLow, qHigh, qUnit = 2,
  legendLeft = 0, legendTop = 0, concMax = NA, concMin = NA,
  bw = FALSE, printTitle = TRUE, printValues = FALSE, minNumObs = 100,
  minNumUncen = 50, colors = c("black", "red", "green"),
  printLegend = TRUE, windowY = 7, windowQ = 2, windowS = 0.5,
  tinyPlot = FALSE, customPar = FALSE, lwd = 2, cex = 0.8,
  cex.axis = 1.1, cex.main = 1.1, cex.legend = 1.2, lineVal = c(1, 1,
  1), logScale = FALSE, edgeAdjust = TRUE, concLab = 1,
  usgsStyle = FALSE, ...)
```

Arguments

eList	named list with at least the Sample and INFO dataframes
date1	character specifying the date for the first curve on the graph, it is in the form "yyyy-mm-dd" (must be in quotes)
date2	character specifying the date for the second curve on the graph, it is in the form "yyyy-mm-dd" (must be in quotes). If only one curve is wanted this should be NA
date3	character specifying the date for the third curve on the graph, it is in the form "yyyy-mm-dd" (must be in quotes). If a third curve is not wanted this should be NA
qLow	numeric value for the lowest discharge to be considered, expressed in the units of discharge that are being used (as specified in qUnit)

qHigh	numeric value for the highest discharge to be considered, expressed in the units of discharge that are being used (as specified in qUnit)
qUnit	object of qUnit class. printqUnitCheatSheet, or numeric represented the short code, or character representing the descriptive name.
legendLeft	numeric which represents the left edge of the legend in the units of the plot.
legendTop	numeric which represents the top edge of the legend in the units of the plot.
concMax	numeric value for upper limit on concentration shown on the graph, default = NA (which causes the upper limit to be set automatically, based on the data)
concMin	numeric value for lower limit on concentration shown on the vertical log graph, default is NA (which causes the lower limit to be set automatically, based on the data). This value is ignored for linear scales, using 0 as the minimum value for the concentration axis.
bw	logical if TRUE graph is produced in black and white, default is FALSE (which means it will use color)
printTitle	logical variable if TRUE title is printed, if FALSE not printed
printValues	logical variable if TRUE the results shown on the graph are also printed to the console and returned in a dataframe (this can be useful for quantifying the changes seen visually in the graph), default is FALSE (not printed)
minNumObs	numeric specifying the miniumum number of observations required to run the weighted regression, default is 100
minNumUncen	numeric specifying the minimum number of uncensored observations to run the weighted regression, default is 50
colors	color vector of lines on plot, see ?par 'Color Specification'. Defaults to c("black","red","green")
printLegend	logical if TRUE, legend is included
windowY	numeric specifying the half-window width in the time dimension, in units of years, default is 7
windowQ	numeric specifying the half-window width in the discharge dimension, units are natural log units, default is 2
windowS	numeric specifying the half-window with in the seasonal dimension, in units of years, default is 0.5
tinyPlot	logical variable, if TRUE plot is designed to be plotted small as part of a multi-part figure, default is FALSE.
customPar	logical defaults to FALSE. If TRUE, par() should be set by user before calling this function (for example, adjusting margins with par(mar=c(5,5,5,5))). If customPar FALSE, EGRET chooses the best margins depending on tinyPlot.
lwd	number line width, default is 2
cex	numerical value giving the amount by which plotting symbols should be magnified
cex.axis	magnification to be used for axis annotation relative to the current setting of cex
cex.main	magnification to be used for main titles relative to the current setting of cex
cex.legend	magnification to be used for legend annotation relative to the current setting of cex

lineVal	vector of line types. Defaults to c(1,1,1) which is a solid line for each line. Options: 0=blank, 1=solid (default), 2=dashed, 3=dotted, 4=dotdash, 5=longdash, 6=twodash
logScale	logical whether or not to use a log scale in the y axis. Default is FALSE
edgeAdjust	logical specifying whether to use the modified method for calculating the windows at the edge of the record. The modified method tends to reduce curvature near the start and end of record. Default is TRUE.
concLab	object of concUnit class, or numeric represented the short code, or character representing the descriptive name. By default, this argument sets concentration labels to use either Concentration or Conc (for tiny plots). Units are taken from the eList$INFO$param.units. To use any other words than "Concentration" see `vignette(topic = "units", package = "EGRET")`.
usgsStyle	logical option to use USGS style guidelines. Setting this option to TRUE does NOT guarantee USGS compliance. It will only change automatically generated labels
...	arbitrary graphical parameters that will be passed to genericEGRETDotPlot function (see ?par for options)

See Also

genericEGRETDotPlot, runSurvReg

Examples

```
date1 <- "1982-06-01"
date2 <- "1994-06-01"
date3 <- "2010-06-01"
qLow <- 0.5
qHigh <- 50
eList <- Choptank_eList

plotConcQSmooth(eList, date1, date2, date3, qLow, qHigh,
                 legendLeft = 0.6, legendTop = 0.7)
plotConcQSmooth(eList, date1, date2, date3, qLow, qHigh,
                logScale=TRUE, legendLeft = 0.6, legendTop = 0.7)
```

---

plotConcTime *Plot of Observed Concentration versus Time*

Description

This function allows the user to plot all of the data, but also to limit it in two ways. The data can be limited to only those observed concentrations collected in a specified discharge range. The data can also be limited to only those observed in certain months of the year. These two selection criteria can be combined. For example, we may only want to plot data for discharges between 100 and 500 cubic feet per second in the months of March, April and May.

Although there are a lot of optional arguments to this function, most are set to a logical default.

Data come from named list, which contains a Sample dataframe with the sample data, and an INFO dataframe with metadata.

Usage

```
plotConcTime(eList, qUnit = 2, yearStart = NA, yearEnd = NA,
  qLower = NA, qUpper = NA, randomCensored = FALSE, tinyPlot = FALSE,
  concMax = NA, concMin = NA, printTitle = TRUE, logScale = FALSE,
  cex = 0.8, cex.axis = 1.1, cex.main = 1.1, customPar = FALSE,
  col = "black", lwd = 1, concLab = 1, usgsStyle = FALSE, ...)
```

Arguments

eList	named list with at least the Sample and INFO dataframes
qUnit	object of qUnit class printqUnitCheatSheet, or numeric represented the short code, or character representing the descriptive name.
yearStart	numeric is the calendar year containing the first estimated annual value to be plotted, default is NA (which allows it to be set automatically by the data)
yearEnd	numeric is the calendar year just after the last estimated annual value to be plotted, default is NA (which allows it to be set automatically by the data)
qLower	numeric the lower bound on values of discharge used to select the data points to be plotted, units are those specified by qUnit, default = NA which is equivalent to a lower bound of zero but if the desired lower bound is zero use qLower = NA
qUpper	numeric the upper bound on values of discharge for selection of data points to be plotted, units are those specified by qUnit, default = NA which is equivalent to an upper bound of infinity
randomCensored	logical. Show censored values as randomized.
tinyPlot	logical variable, if TRUE plot is designed to be plotted small as part of a multi-part figure, default is FALSE.
concMax	numeric value for the maximum value to be used on the vertical axis, default is NA (which allows it to be set automatically by the data)
concMin	numeric value for lower limit on concentration shown on the vertical log graph, default is NA (which causes the lower limit to be set automatically, based on the data). This value is ignored for linear scales, using 0 as the minimum value for the concentration axis.
printTitle	logical variable if TRUE title is printed, if FALSE title is not printed (this is best for a multi-plot figure).
logScale	logical. If TRUE concentration is plotted on a log axis, default FALSE.
cex	numerical value giving the amount by which plotting symbols should be magnified.
cex.axis	magnification to be used for axis annotation relative to the current setting of cex.
cex.main	magnification to be used for main titles relative to the current setting of cex.

customPar	logical defaults to FALSE. If TRUE, par() should be set by user before calling this function. (for example, adjusting margins with par(mar=c(5,5,5,5))). If customPar FALSE, EGRET chooses the best margins depending on tinyPlot.
col	color of points on plot, see ?par 'Color Specification'
lwd	number line width.
concLab	object of concUnit class, or numeric represented the short code, or character representing the descriptive name. By default, this argument sets concentration labels to use either Concentration or Conc (for tiny plots). Units are taken from the eList$INFO$param.units. To use any other words than "Concentration" see `vignette(topic = "units", package = "EGRET")`.
usgsStyle	logical option to use USGS style guidelines. Setting this option to TRUE does NOT guarantee USGS compliance. It will only change automatically generated labels
...	arbitrary functions sent to the generic plotting function. See ?par for details on possible parameters.

Details

The function has two possible ways to plot censored values (e.g. "less-than values").

The default is to plot them as a vertical line that goes from the reporting limit down to the bottom of the graph.

The alternative is to set randomCensored = TRUE. In this case a random value is used for plotting the individual sample value. This random value lies between the reporting limit and zero and it is distributed as a truncated log normal based on the fitted WRTDS model.

The function makeAugmentedSample must be run first if randomCensored = TRUE. Running makeAugmentedSample requires that modelEstimation has already been run.

These random censored values are used to create more readable plots and are not used in any computations about the data set. The random censored values are shown as open circles and the non-censored data are shown as filled dots.

See Also

selectDays, genericEGRETDotPlot

Examples

```
eList <- Choptank_eList
# Water year:
plotConcTime(eList)
# Graphs consisting of Jun-Aug
eList <- setPA(eList, paStart=6,paLong=3)
plotConcTime(eList,
             qUnit = 1,
             qLower = 100,
             qUpper = 10000)
plotConcTime(eList, logScale=TRUE)
plotConcTime(eList,
```

```
qUnit = 1,
qLower = 100, qUpper = 10000,
 randomCensored = TRUE)
```

`plotConcTimeDaily` *Plot of the time series of daily concentration estimates and the sample values for the days that were sampled*

Description

This plot is useful for visual examination of the ability of the WRTDS, or other model, to fit the data, seen in a time-series perspective. The graph is most useful when it covers a period of just a few years and not the complete record but a complete record can be done by repeated use over a series of segments.

Although there are a lot of optional arguments to this function, most are set to a logical default.

Data come from named list, which contains a Sample dataframe with the sample data, a Daily dataframe with the daily flow data, and an INFO dataframe with metadata.

Usage

```
plotConcTimeDaily(eList, yearStart = NA, yearEnd = NA, tinyPlot = FALSE,
  concMax = NA, printTitle = TRUE, plotGenConc = TRUE, cex = 0.8,
  cex.axis = 1.1, randomCensored = FALSE, cex.main = 1.1,
  customPar = FALSE, col = "black", lwd = 1, prettyDate = TRUE,
  usgsStyle = FALSE, ...)
```

Arguments

`eList`	named list with at least the Daily, Sample, and INFO dataframes
`yearStart`	numeric specifying the starting date (expressed as decimal years, for example 1989.0) for the plot
`yearEnd`	numeric specifying the ending date for the plot
`tinyPlot`	logical variable, if TRUE plot is designed to be short and wide, default is FALSE.
`concMax`	number specifying the maximum value to be used on the vertical axis, default is NA (which allows it to be set automatically by the data)
`printTitle`	logical variable if TRUE title is printed, if FALSE title is not printed (this is best for a multi-plot figure)
`plotGenConc`	logical variable. If `TRUE`, annual concentration points from `WRTDSKalman` output are plotted, if `FALSE` WRTDS concentration is plotted,
`cex`	numerical value giving the amount by which plotting symbols should be magnified
`cex.axis`	magnification to be used for axis annotation relative to the current setting of cex
`randomCensored`	if TRUE plot a random value for censored data. Default is FALSE.
`cex.main`	magnification to be used for main titles relative to the current setting of cex

customPar	logical defaults to FALSE. If TRUE, par() should be set by user before calling this function (for example, adjusting margins with par(mar=c(5,5,5,5))). If customPar FALSE, EGRET chooses the best margins depending on tinyPlot.
col	color of points on plot, see ?par 'Color Specification'
lwd	number line width
prettyDate	logical use 'pretty' limits for date axis if TRUE, or force the yearStart/yearEnd as limits if FALSE
usgsStyle	logical option to use USGS style guidelines. Setting this option to TRUE does NOT guarantee USGS compliance. It will only change automatically generated labels
...	arbitrary functions sent to the generic plotting function. See ?par for details on possible parameters

See Also

selectDays, genericEGRETDotPlot

Examples

```
eList <- Choptank_eList
# Water year:
plotConcTimeDaily(eList)
plotConcTimeDaily(eList,
                  yearStart = 1998,
                  yearEnd = 2001,
                  plotGenConc = FALSE)
```

---

plotConcTimeSmooth *Plot up to three curves representing the concentration versus time relationship, each curve representing a different flow.*

Description

These plots show how the concentration-time relationship is changing over flow.

This plot can also help identify situations where the windowY may be too small. If there are substantial oscillations of some of the curves, then the windowY should be increased. Alternatively, windowY may be too large. This can be seen when the windowY is reduced (say to 4.0). A good choice of windowY would be a value just great enough to damp out oscillations in the curves.

Although there are a lot of optional arguments to this function, most are set to a logical default.

Data come from named list, which contains a Sample dataframe with the sample data and an INFO dataframe with metadata.

Usage

```
plotConcTimeSmooth(eList, q1, q2, q3, centerDate, yearStart, yearEnd,
  qUnit = 2, legendLeft = 0, legendTop = 0, concMax = NA,
  concMin = NA, bw = FALSE, printTitle = TRUE, colors = c("black",
  "red", "green"), printValues = FALSE, tinyPlot = FALSE, concLab = 1,
  monthLab = 1, minNumObs = 100, minNumUncen = 50, windowY = 10,
  windowQ = 2, windowS = 0.5, cex.main = 1.1, lwd = 2,
  printLegend = TRUE, cex.legend = 1.2, cex = 0.8, cex.axis = 1.1,
  customPar = FALSE, lineVal = c(1, 1, 1), logScale = FALSE,
  edgeAdjust = TRUE, usgsStyle = FALSE, ...)
```

Arguments

eList	named list with at least the Sample and INFO dataframes
q1	numeric This is the discharge value for the first curve to be shown on the plot. It is expressed in units specified by qUnit.
q2	numeric This is the discharge value for the second curve to be shown on the plot. It is expressed in units specified by qUnit. If you don't want a second curve then the argument must be q2=NA
q3	numeric This is the discharge value for the third curve to be shown on the plot. It is expressed in units specified by qUnit. If you don't want a third curve then the argument must be q3=NA
centerDate	character This is the time of year to be used as the center date for the smoothing. It is expressed as a month and day and must be in the form "mm-dd"
yearStart	numeric This is the starting year for the graph. The first value plotted for each curve will be at the first instance of centerDate in the year designated by yearStart.
yearEnd	numeric This is the end of the sequence of values plotted on the graph.The last value will be the last instance of centerDate prior to the start of yearEnd. (Note, the number of values plotted on each curve will be yearEnd-yearStart.)
qUnit	object of qUnit class. printqUnitCheatSheet, or numeric represented the short code, or character representing the descriptive name.
legendLeft	numeric which represents the left edge of the legend in the units of the plot.
legendTop	numeric which represents the top edge of the legend in the units of the plot.
concMax	numeric value for upper limit on concentration shown on the graph, default = NA (which causes the upper limit to be set automatically, based on the data)
concMin	numeric value for lower limit on concentration shown on the vertical log graph, default is NA (which causes the lower limit to be set automatically, based on the data). This value is ignored for linear scales, using 0 as the minimum value for the concentration axis.
bw	logical if TRUE graph is produced in black and white, default is FALSE (which means it will use color)
printTitle	logical variable if TRUE title is printed, if FALSE not printed
colors	color vector of lines on plot, see ?par 'Color Specification'. Defaults to c("black","red","green")

printValues	logical variable if TRUE the results shown on the graph are printed to the console and returned in a dataframe (this can be useful for quantifying the changes seen visually in the graph), default is FALSE (not printed)
tinyPlot	logical variable, if TRUE plot is designed to be plotted small, as a part of a multipart figure, default is FALSE
concLab	object of concUnit class, or numeric represented the short code, or character representing the descriptive name. By default, this argument sets concentration labels to use either Concentration or Conc (for tiny plots). Units are taken from the eList$INFO$param.units. To use any other words than "Concentration" see `vignette(topic = "units", package = "EGRET")`.
monthLab	object of monthLabel class, or numeric represented the short code, or character representing the descriptive name.
minNumObs	numeric specifying the miniumum number of observations required to run the weighted regression, default is 100
minNumUncen	numeric specifying the minimum number of uncensored observations to run the weighted regression, default is 50
windowY	numeric specifying the half-window width in the time dimension, in units of years, default is 10
windowQ	numeric specifying the half-window width in the discharge dimension, units are natural log units, default is 2
windowS	numeric specifying the half-window with in the seasonal dimension, in units of years, default is 0.5
cex.main	magnification to be used for main titles relative to the current setting of cex
lwd	line width, a positive number, defaulting to 2
printLegend	logical if TRUE, legend is included
cex.legend	number magnification of legend
cex	numerical value giving the amount by which plotting symbols should be magnified
cex.axis	magnification to be used for axis annotation relative to the current setting of cex
customPar	logical defaults to FALSE. If TRUE, par() should be set by user before calling this function (for example, adjusting margins with par(mar=c(5,5,5,5))). If customPar FALSE, EGRET chooses the best margins depending on tinyPlot.
lineVal	vector of line types. Defaults to c(1,1,1) which is a solid line for each line. Options: 0=blank, 1=solid (default), 2=dashed, 3=dotted, 4=dotdash, 5=longdash, 6=twodash
logScale	logical whether or not to use a log scale in the y axis.
edgeAdjust	logical specifying whether to use the modified method for calculating the windows at the edge of the record. The modified method tends to reduce curvature near the start and end of record. Default is TRUE.
usgsStyle	logical option to use USGS style guidelines. Setting this option to TRUE does NOT guarantee USGS compliance. It will only change automatically generated labels
...	arbitrary functions sent to the generic plotting function. See ?par for details on possible parameters

See Also

genericEGRETDotPlot, runSurvReg

Examples

```
q1 <- 1
q2 <- 10
q3 <- 100
centerDate <- "07-01"
yearStart <- 1990
yearEnd <- 2010
eList <- Choptank_eList
plotConcTimeSmooth(eList, q1, q2,q3, centerDate,
                   yearStart, yearEnd, legendLeft = 1997,
                   legendTop = 0.44, cex.legend = 0.9)
plotConcTimeSmooth(eList, q1, q2,q3, centerDate, yearStart,
                   yearEnd, logScale = TRUE, legendLeft = 1994,
                   legendTop = 0.4, cex.legend = 0.9)
```

---

plotContours *Color contour plot of the estimated surfaces as a function of discharge and time (surfaces include log concentration, standard error, and concentration)*

Description

These plots are normally used for plotting the estimated concentration surface (whatSurface = 3) but can be used to explore the estimated surfaces for the log of concentration or for the standard error (in log space) which is what determines the bias correction. The plots are often more interpretable when the time limits for the plot are less than a decade. To explore changes over a long time period it is best to do this multiple times, for various time slices of 2 years (for example) or to use the function plotDiffContours.

Although there are a lot of optional arguments to this function, most are set to a logical default.

Obtaining a plot that provides good insight it is useful to experiment with several of the arguments such as yearStart, yearEnd, qBottom, qTop, and contourLevels.

Data come from named list, which contains a Sample dataframe with the sample data, a Daily dataframe with the daily flow data, and an INFO dataframe with metadata.

Usage

```
plotContours(eList, yearStart, yearEnd, qBottom = NA, qTop = NA,
  whatSurface = 3, qUnit = 2, contourLevels = NA, span = 60,
  pval = 0.05, printTitle = TRUE, vert1 = NA, vert2 = NA, horiz = NA,
  tcl = 0.03, flowDuration = TRUE, customPar = FALSE, yTicks = NA,
  tick.lwd = 1, usgsStyle = FALSE, lwd = 2, cex.main = 1,
  cex.axis = 1, concLab = 1,
  color.palette = grDevices::colorRampPalette(c("white", "gray", "blue",
  "red")), ...)
```

Arguments

eList	named list with at least the Daily and INFO dataframes, and surfaces matrix
yearStart	numeric value for the starting date for the graph, expressed as decimal year (typically whole number such as 1989.0)
yearEnd	numeric value for the ending date for the graph, expressed as decimal year, (for example 1993.0)
qBottom	numeric value for the bottom edge of the graph, expressed in the units of discharge that are being used (as specified in qUnit). NA will choose a "pretty" lower limit nearest to the 5% of discharge. If yTicks are specified, then the first value of yTicks becomes the lowest discharge shown on the figure.
qTop	numeric value for the top edge of the graph, expressed in the units of discharge that are being used (as specified in qUnit). NA will choose a "pretty" upper limit nearest to the 95% of discharge. If yTicks are specified, then the last value of yTicks becomes the highest discharge shown on the figure.
whatSurface	numeric value, can only accept 1, 2, or 3; whatSurface=1 is yHat (log concentration), whatSurface=2 is SE (standard error of log concentration), and whatSurface=3 is ConcHat (unbiased estimate of concentration), default = 3.
qUnit	object of qUnit class. printqUnitCheatSheet, or numeric represented the short code, or character representing the descriptive name.
contourLevels	numeric vector containing the contour levels for the contour plot, arranged in ascending order, default is NA (which causes the contour levels to be set automatically, based on the data)
span	numeric, it is the half-width (in days) of the smoothing window for computing the flow duration information, default = 60
pval	numeric, the probability value for the lower flow frequency line on the graph
printTitle	logical variable if TRUE title is printed, if FALSE not printed
vert1	numeric, the location in time for a black vertical line on the figure, yearStart<vert1<yearEnd, default is NA (vertical line is not drawn)
vert2	numeric, the location in time for a black vertical line on the figure, yearStart<vert2<yearEnd, default is NA (vertical line is not drawn)
horiz	numeric, the location in discharge for a black horizontal line on the figure, qBottom<vert1<qTop, default is NA (no horizontal line is drawn)
tcl	numeric, length of tick marks in inches, default is 0.03
flowDuration	logical variable if TRUE plot the flow duration lines (5 and 95 flow percentiles), if FALSE do not plot them, default = TRUE
customPar	logical defaults to FALSE. If TRUE, par() should be set by user before calling this function (for example, adjusting margins with par(mar=c(5,5,5,5))). If customPar FALSE, EGRET chooses the best margins.
yTicks	vector of yTick labels and marks that will be plotted in log space. (for example yTicks = c(3, 5, 10, 20, 50, 100, 200, 400). The first and last values determine the range of the y axis. If NA, the tick marks will be automatically generated.
tick.lwd	line width for axis ticks, default is 1

usgsStyle	logical option to use USGS style guidelines. Setting this option to TRUE does NOT guarantee USGS compliance. It will only change automatically generated labels.
lwd	numeric, line width of flowDuration curve, default is 2
cex.main	magnification to be used for main titles relative to the current setting of cex
cex.axis	magnification to be used for axis annotation relative to the current setting of cex
concLab	object of concUnit class, or numeric represented the short code, or character representing the descriptive name. By default, this argument sets concentration labels to use either Concentration or Conc (for tiny plots). Units are taken from the eList$INFO$param.units. To use any other words than "Concentration" see `vignette(topic = "units", package = "EGRET")`.
color.palette	a function that creates a color palette for the contour plot. Default goes from white to gray to blue to red using the function `colorRampPalette(c("white","gray","blue","red"))` A few preset options are heat.colors, topo.colors, and terrain.colors.
...	arbitrary functions sent to the generic plotting function. See ?par for details on possible parameters

Examples

```
yearStart <- 2002
yearEnd <- 2010
qBottom <- 0.5
qTop<- 20
clevel <- seq(0,2,0.25)
eList <- Choptank_eList
plotContours(eList, yearStart, yearEnd, qBottom, qTop,
             contourLevels = clevel)
plotContours(eList, yearStart, yearEnd, qBottom, qTop = 50,
             contourLevels = clevel, flowDuration = FALSE)
colors <- grDevices::colorRampPalette(c("white","black"))
plotContours(eList, yearStart, yearEnd, qBottom, qTop = 50,
             contourLevels = clevel, color.palette = colors,
             flowDuration = FALSE)
```

---

plotDiffContours *Plots the difference between two years from a contour plot created by plotContours*

Description

These plots are normally used for plotting changes in the estimated concentration surface (whatSurface=3) but can be used to explore the changes in estimated surfaces for the log of concentration or for the standard error (in log space) which is what determines the bias correction.

The difference can be shown either as an arithmetic difference or as a percentage difference.

Although there are a lot of optional arguments to this function, most are set to a logical default.

Data come from named list, which contains a Sample dataframe with the sample data, a Daily dataframe with the daily flow data, and an INFO dataframe with metadata.

Usage

```
plotDiffContours(eList, year0, year1, qBottom = NA, qTop = NA,
  maxDiff = NA, whatSurface = 3, tcl = 0.03, qUnit = 2, span = 60,
  pval = 0.05, printTitle = TRUE, plotPercent = FALSE, vert1 = NA,
  vert2 = NA, horiz = NA, flowDuration = TRUE, yTicks = NA,
  tick.lwd = 1, lwd = 2, cex.main = 0.95, cex.axis = 1,
  customPar = FALSE, usgsStyle = FALSE,
  color.palette = grDevices::colorRampPalette(c("blue", "white", "red")),
  concLab = 1, monthLab = 1, ...)
```

Arguments

eList	named list with at least the Daily and INFO dataframes, and surfaces matrix
year0	numeric value for the calendar year that is the first year of the pair of years for the analysis, should be a whole number
year1	numeric value for the calendar year that is the second year of the pair of years for the analysis, should be a whole number
qBottom	numeric value for the bottom edge of the graph, expressed in the units of discharge that are being used (as specified in qUnit). NA will choose a "pretty" lower limit nearest to the 5% of discharge. If yTicks are specified, then the first value of yTicks becomes the lowest discharge shown on the figure.
qTop	numeric value for the top edge of the graph, expressed in the units of discharge that are being used (as specified in qUnit). NA will choose a "pretty" upper limit nearest to the 95% of discharge. If yTicks are specified, then the last value of yTicks becomes the highest discharge shown on the figure.
maxDiff	numeric value which is the absolute value of the largest change in concentration that will be shown on the figure. If NA, the scale will be set from 5% to 95% of the concentration difference. If plotPercent = TRUE then maxDiff will be the maximum percentage difference.
whatSurface	numeric value, can only accept 1, 2, or 3; whatSurface = 1 is yHat (log concentration), whatSurface = 2 is SE (standard error of log concentration), and whatSurface = 3 is ConcHat (unbiased estimate of concentration), default = 3
tcl	numeric, length of tick marks in inches, default is 0.1
qUnit	object of qUnit class. printqUnitCheatSheet, or numeric represented the short code, or character representing the descriptive name.
span	numeric, it is the half-width (in days) of the smoothing window for computing the flow duration information, default = 60
pval	numeric, the probability value for the lower flow frequency line on the graph
printTitle	logical variable if TRUE title is printed, if FALSE not printed
plotPercent	logical. If TRUE, plots percent difference, if FALSE, plots arithmetic differences. Defaults to FALSE.
vert1	numeric, the location in time for a black vertical line on the figure, yearStart < vert1 < yearEnd, default is NA (vertical line is not drawn)

vert2	numeric, the location in time for a black vertical line on the figure, yearStart < vert2 < yearEnd, default is NA (vertical line is not drawn)
horiz	numeric, the location in discharge for a black horizontal line on the figure, qBottom<vert1<qTop, default is NA (no horizontal line is drawn)
flowDuration	logical variable if TRUE plot the flow duration lines (5 and 95 flow percentiles), if FALSE do not plot them, default = TRUE
yTicks	vector of yTick labels and marks that will be plotted in log space. (for example yTicks = c(3, 5, 10, 20, 50, 100, 200, 400). The first and last values determine the range of the y axis. If NA, the tick marks will be automatically generated.
tick.lwd	line width for axis ticks, default is 2
lwd	numeric, line width of flowDuration curve, default is 1
cex.main	magnification to be used for main titles relative to the current setting of cex
cex.axis	magnification to be used for axis annotation relative to the current setting of cex
customPar	logical defaults to FALSE. If TRUE, par() should be set by user before calling this function (for example, adjusting margins with par(mar=c(5,5,5,5))). If customPar FALSE, EGRET chooses the best margins.
usgsStyle	logical option to use USGS style guidelines. Setting this option to TRUE does NOT guarantee USGS compliance. It will only change automatically generated labels.
color.palette	a function that creates a color palette for the contour plot. Default goes from blue to white to red using the function grDevices::colorRampPalette(c("blue","white","red")). A few preset options are heat.colors, topo.colors, and terrain.colors.
concLab	object of concUnit class, or numeric represented the short code, or character representing the descriptive name. By default, this argument sets concentration labels to use either Concentration or Conc (for tiny plots). Units are taken from the eList$INFO$param.units. To use any other words than "Concentration" see vignette(topic = "units", package = "EGRET").
monthLab	object of monthLabel class, or numeric represented the short code, or character representing the descriptive name.
...	arbitrary functions sent to the generic plotting function. See ?par for details on possible parameters

Examples

```
year0 <- 1990
year1 <- 2009
qBottom <- 0.5
qTop <- 20
maxDiff<-0.5
eList <- Choptank_eList
plotDiffContours(eList, year0, year1, qBottom, qTop, maxDiff = 0.5)
plotDiffContours(eList, year0, year1, qBottom, qTop, maxDiff = 50, plotPercent = TRUE)
```

plotFlowSingle *Creates a plot of a time series of a particular flow statistic and a loess smooth of that flow statistic*

Description

A part of the flowHistory system. The index of the flow statistics is istat. These statistics are: (1) 1-day minimum, (2) 7-day minimum, (3) 30-day minimum, (4) median (5) mean, (6) 30-day maximum, (7) 7-day maximum, and (8) 1-day maximum

Although there are a lot of optional arguments to this function, most are set to a logical default.

Data come from named list, which contains a Daily dataframe with the daily flow data, and an INFO dataframe with metadata.

Usage

```
plotFlowSingle(eList, istat, yearStart = NA, yearEnd = NA, qMax = NA,
  printTitle = TRUE, tinyPlot = FALSE, customPar = FALSE,
  runoff = FALSE, qUnit = 1, printStaName = TRUE, printPA = TRUE,
  usgsStyle = FALSE, printIstat = TRUE, cex = 0.8, cex.axis = 1.1,
  cex.main = 1.1, lwd = 2, col = "black", ...)
```

Arguments

eList	named list with at least the Daily and INFO dataframes
istat	A numeric value for the flow statistic to be graphed (possible values are 1 through 8)
yearStart	A numeric value for year in which the graph should start, default is NA, which indicates that the graph should start with first annual value
yearEnd	A numeric value for year in which the graph should end, default is NA, which indicates that the graph should end with last annual value
qMax	A numeric value for the maximum value to be used for y-axis of graph, default is NA means that graph is self-scaling
printTitle	logical variable, if TRUE title is printed, if FALSE title is not printed, default is TRUE
tinyPlot	logical variable, if TRUE plot is designed to be plotted small, as a part of a multipart figure, default is FALSE
customPar	logical defaults to FALSE. If TRUE, par() should be set by user before calling this function (for example, adjusting margins with par(mar=c(5,5,5,5))). If customPar FALSE, EGRET chooses the best margins depending on tinyPlot.
runoff	logical variable, if TRUE the streamflow data are converted to runoff values in mm/day
qUnit	object of qUnit class printqUnitCheatSheet, or numeric represented the short code, or character representing the descriptive name.

printStaName	logical variable, if TRUE station name is printed in title, if FALSE not printed, default is TRUE
printPA	logical variable, if TRUE Period of Analysis information is printed in title, if FALSE not printed, default is TRUE
usgsStyle	logical option to use USGS style guidelines. Setting this option to TRUE does NOT guarantee USGS compliance. It will only change automatically generated labels.
printIstat	logical variable, if TRUE print the statistic name is printed in title, if FALSE not printed, default is TRUE
cex	numerical value giving the amount by which plotting symbols should be magnified
cex.axis	magnification to be used for axis annotation relative to the current setting of cex
cex.main	magnification to be used for main titles relative to the current setting of cex
lwd	number line width
col	color of points on plot, see ?par 'Color Specification'
...	arbitrary graphical parameters that will be passed to genericEGRETDotPlot function (see ?par for options)

Details

The curve plotted on the graph is a loess smooth of the data. This smooth is computed on the logs of the data and then transformed back to plot. The width of the smoothing window is 20 years on either side of the year being plotted However, the window width can be adjusted using setPA function.

See Also

makeAnnualSeries, genericEGRETDotPlot

Examples

```
eList <- Choptank_eList
# Water year:
plotFlowSingle(eList, 1)
# Graphs consisting of Jun-Aug
eList <- setPA(eList, paStart=6,paLong=3)
plotFlowSingle(eList, 1)
```

plotFluxHist *Graph of annual flux and flow normalized flux versus year*

Description

The annual results reported are for a specified "period of analysis" which can be an entire water year, a calendar, a season or even an individual month. The user specifies this period of analysis in the call to setupYears. Values plotted express a flux rate, such as thousand kg per year For a period of analysis that is less than a year, this does not equal the mass transported over the period of analysis

Although there are a lot of optional arguments to this function, most are set to a logical default.

Data come from named list (eList), which contains a Daily dataframe with the daily flow data, and an INFO dataframe with metadata.

Usage

```
plotFluxHist(eList, yearStart = NA, yearEnd = NA, fluxUnit = 9,
  fluxMax = NA, printTitle = TRUE, usgsStyle = FALSE,
  plotFlowNorm = TRUE, plotAnnual = TRUE, plotGenFlux = FALSE,
  tinyPlot = FALSE, col = "black", col.pred = "green", col.gen = "red",
  cex = 0.8, cex.axis = 1.1, cex.main = 1.1, lwd = 2,
  customPar = FALSE, ...)
```

Arguments

eList	named list with at least the Daily and INFO dataframes
yearStart	numeric is the calendar year containing the first estimated annual value to be plotted, default is NA (which allows it to be set automatically by the data)
yearEnd	numeric is the calendar year just after the last estimated annual value to be plotted, default is NA (which allows it to be set automatically by the data)
fluxUnit	number representing entry in pre-defined fluxUnit class array. printFluxUnitCheatSheet
fluxMax	number specifying the maximum value to be used on the vertical axis, default is NA (which allows it to be set automatically by the data)
printTitle	logical variable if TRUE title is printed, if FALSE title is not printed (this is best for a multi-plot figure)
usgsStyle	logical option to use USGS style guidelines. Setting this option to TRUE does NOT guarantee USGS compliance. It will only change automatically generated labels.
plotFlowNorm	logical variable if TRUE the flow normalized line is plotted, if FALSE not plotted
plotAnnual	logical variable if TRUE annual flux points are plotted, if FALSE not plotted
plotGenFlux	logical variable. If TRUE, annual flux points from WRTDSKalman output are plotted, if FALSE the generalized flux is not plotted.

tinyPlot	logical variable, if TRUE plot is designed to be plotted small, as a part of a multipart figure, default is FALSE
col	color of points on plot, see ?par 'Color Specification'
col.pred	color of flow normalized line on plot, see ?par 'Color Specification'
col.gen	color of points for WRTDS_K output on plot, see ?par 'Color Specification'
cex	numerical value giving the amount by which plotting symbols should be magnified
cex.axis	magnification to be used for axis annotation relative to the current setting of cex
cex.main	magnification to be used for main titles relative to the current setting of cex
lwd	number line width
customPar	logical defaults to FALSE. If TRUE, par() should be set by user before calling this function (for example, adjusting margins with par(mar=c(5,5,5,5))). If customPar FALSE, EGRET chooses the best margins depending on tinyPlot.
...	arbitrary graphical parameters that will be passed to genericEGRETDotPlot function (see ?par for options)

See Also

setupYears

Examples

```
yearStart <- 2001
yearEnd <- 2010
eList <- Choptank_eList
# Water year:

plotFluxHist(eList)
plotFluxHist(eList, yearStart, yearEnd, fluxUnit = 1)
plotFluxHist(eList, yearStart, yearEnd, fluxUnit = 'kgDay')
```

---

plotFluxPred *Graph of observed versus estimated flux*

Description

Data come from named list, which contains a Sample dataframe with the sample data, and an INFO dataframe with metadata.

Although there are a lot of optional arguments to this function, most are set to a logical default.

Usage

```
plotFluxPred(eList, fluxUnit = 3, fluxMax = NA, printTitle = TRUE,
  oneToOneLine = TRUE, customPar = FALSE, col = "black", lwd = 1,
  cex = 0.8, cex.axis = 1.1, cex.main = 1.1, tinyPlot = FALSE,
  usgsStyle = FALSE, logScale = FALSE, randomCensored = FALSE, ...)
```

Arguments

eList	named list with at least the Sample and INFO dataframes
fluxUnit	number representing entry in pre-defined fluxUnit class array. printFluxUnitCheatSheet
fluxMax	number specifying the maximum value to be used on the vertical axis, default is NA (which allows it to be set automatically by the data)
printTitle	logical variable if TRUE title is printed, if FALSE not printed (this is best for a multi-plot figure)
oneToOneLine	inserts 1:1 line
customPar	logical defaults to FALSE. If TRUE, par() should be set by user before calling this function (for example, adjusting margins with par(mar=c(5,5,5,5))). If customPar FALSE, EGRET chooses the best margins depending on tinyPlot.
col	color of points on plot, see ?par 'Color Specification'
lwd	number line width
cex	numerical value giving the amount by which plotting symbols should be magnified
cex.axis	magnification to be used for axis annotation relative to the current setting of cex
cex.main	magnification to be used for main titles relative to the current setting of cex
tinyPlot	logical variable if TRUE plot is designed to be small, if FALSE it is designed for page size, default is FALSE (not fully implemented yet)
usgsStyle	logical option to use USGS style guidelines. Setting this option to TRUE does NOT guarantee USGS compliance. It will only change automatically generated labels.
logScale	logical if TRUE x and y plotted in log axis
randomCensored	logical. Show censored values as randomized.
...	arbitrary graphical parameters that will be passed to genericEGRETDotPlot function (see ?par for options)

See Also

selectDays, genericEGRETDotPlot

Examples

```
eList <- Choptank_eList
# Water year:
plotFluxPred(eList)
plotFluxPred(eList, fluxUnit = 'poundsDay')
plotFluxPred(eList, logScale=TRUE)
# Graphs consisting of Jun-Aug
eList <- setPA(eList, paStart=6,paLong=3)
plotFluxPred(eList)
```

plotFluxQ *Sample data plot: observed log flux vs log discharge*

Description

Concentration and discharge data used to compute flux come from a data frame named Sample which contains the sample data. The metadata come from a data frame named INFO.

Although there are a lot of optional arguments to this function, most are set to a logical default.

Data come from named list, which contains a Sample dataframe with the sample data, and an INFO dataframe with metadata.

Usage

```
plotFluxQ(eList, qUnit = 2, logScale = TRUE, fluxUnit = 3,
  tinyPlot = FALSE, fluxMax = NA, fluxMin = NA, col = "black",
  lwd = 1, printTitle = TRUE, usgsStyle = FALSE, cex = 0.8,
  cex.axis = 1.1, cex.main = 1.1, customPar = FALSE, ...)
```

Arguments

eList	named list with at least the Sample and INFO dataframes
qUnit	object of qUnit class. printqUnitCheatSheet, or numeric represented the short code, or character representing the descriptive name.
logScale	logical, default TRUE, TRUE creates a log-log scale, FALSE creates an arithmatic scale.
fluxUnit	object of fluxUnit class. printFluxUnitCheatSheet, or numeric represented the short code, or character representing the descriptive name.
tinyPlot	logical variable if TRUE plot is designed to fit into a multi-plot array, default is FALSE
fluxMax	numeric specifying the maximum value to be used on the vertical axis, default is NA (which allows it to be set automatically by the data)
fluxMin	numeric specifying the minimum value to be used on the vertical axis, default is NA (which allows it to be set automatically by the data)
col	color of points on plot, see ?par 'Color Specification'
lwd	number line width
printTitle	logical variable if TRUE title is printed, if FALSE not printed (this is best for a multi-plot figure)
usgsStyle	logical option to use USGS style guidelines. Setting this option to TRUE does NOT guarantee USGS compliance. It will only change automatically generated labels.
cex	numerical value giving the amount by which plotting symbols should be magnified
cex.axis	magnification to be used for axis annotation relative to the current setting of cex

cex.main	magnification to be used for main titles relative to the current setting of cex
customPar	logical defaults to FALSE. If TRUE, par() should be set by user before calling this function (for example, adjusting margins with par(mar=c(5,5,5,5))). If customPar FALSE, EGRET chooses the best margins depending on tinyPlot.
...	arbitrary graphical parameters that will be passed to genericEGRETDotPlot function (see ?par for options)

See Also

selectDays, genericEGRETDotPlot

Examples

```
eList <- Choptank_eList
# Water year:
plotFluxQ(eList, qUnit = 1, fluxUnit = 1)
plotFluxQ(eList, fluxUnit = 'kgDay')
plotFluxQ(eList)
# Graphs consisting of Jun-Aug
eList <- setPA(eList, paStart=6,paLong=3)
plotFluxQ(eList)
```

---

plotFluxTimeDaily *Plot of the time series of daily flux estimates and the sample values for the days that were sampled*

Description

This plot is useful for visual examination of the ability of the WRTDS, or other model, to fit the data, as seen in a time-series perspective.

Although there are a lot of optional arguments to this function, most are set to a logical default.

Data come from named list, which contains a Sample dataframe with the sample data, a Daily dataframe with the daily flow data, and an INFO dataframe with metadata.

Usage

```
plotFluxTimeDaily(eList, yearStart = NA, yearEnd = NA, tinyPlot = FALSE,
  fluxUnit = 3, fluxMax = NA, randomCensored = FALSE,
  printTitle = TRUE, plotGenFlux = TRUE, usgsStyle = FALSE, cex = 0.8,
  cex.axis = 1.1, cex.main = 1.1, customPar = FALSE, col = "black",
  lwd = 1, prettyDate = TRUE, ...)
```

Arguments

eList	named list with at least the Daily, Sample, and INFO dataframes
yearStart	numeric specifying the starting date (expressed as decimal years, for example 1989.0) for the plot
yearEnd	numeric specifying the ending date for the plot
tinyPlot	logical variable, if TRUE plot is designed to be short and wide, default is FALSE.
fluxUnit	number representing in pre-defined fluxUnit class array. printFluxUnitCheatSheet
fluxMax	number specifying the maximum value to be used on the vertical axis, default is NA (which allows it to be set automatically by the data)
randomCensored	logical, if TRUE plot a random value for censored data. Default is FALSE.
printTitle	logical variable if TRUE title is printed, if FALSE title is not printed (this is best for a multi-plot figure)
plotGenFlux	logical variable. If TRUE, annual flux points from WRTDSKalman output are plotted, if FALSE WRTDS flux is plotted.
usgsStyle	logical option to use USGS style guidelines. Setting this option to TRUE does NOT guarantee USGS compliance. It will only change automatically generated labels.
cex	numerical value giving the amount by which plotting symbols should be magnified
cex.axis	magnification to be used for axis annotation relative to the current setting of cex
cex.main	magnification to be used for main titles relative to the current setting of cex
customPar	logical defaults to FALSE. If TRUE, par() should be set by user before calling this function (for example, adjusting margins with par(mar=c(5,5,5,5))). If customPar FALSE, EGRET chooses the best margins depending on tinyPlot.
col	color of points on plot, see ?par 'Color Specification'
lwd	number line width
prettyDate	logical use 'pretty' limits for date axis if TRUE, or force the yearStart/yearEnd as limits if FALSE
...	arbitrary graphical parameters that will be passed to genericEGRETDotPlot function (see ?par for options)

See Also

selectDays, genericEGRETDotPlot

Examples

```
eList <- Choptank_eList
# Water year:
plotFluxTimeDaily(eList)
plotFluxTimeDaily(eList, 2001,2009)
```

plotFour *Makes four graphs of streamflow statistics on a single page*

Description

Part of the flowHistory system. The four statistics are 1-day maximum, annual mean, annual 7-day minimum, and the running standard deviation of the log daily discharge values.

Although there are a lot of optional arguments to this function, most are set to a logical default.

Data come from named list, which contains a Daily dataframe with the daily flow data, and an INFO dataframe with metadata. Each graph shows a loess smooth of the data that are plotted.

Usage

```
plotFour(eList, yearStart = NA, yearEnd = NA, printTitle = TRUE,
  runoff = FALSE, qUnit = 1, window = 15, cex = 0.8, cex.axis = 1.2,
  cex.main = 1.2, col = "black", lwd = 1, ...)
```

Arguments

eList	named list with at least Daily and INFO dataframes
yearStart	A numeric value for year in which the graph should start, default is NA, which indicates that the graph should start with first annual value
yearEnd	A numeric value for year in which the graph should end, default is NA, which indicates that the graph should end with last annual value
printTitle	logical variable, if TRUE title is printed, if FALSE title is not printed, default is TRUE
runoff	logical variable, if TRUE the streamflow data are converted to runoff values in mm/day
qUnit	object of qUnit class printqUnitCheatSheet, or numeric represented the short code, or character representing the descriptive name.
window	numeric which is the full width, in years, of the time window over which the standard deviation is computed, default = 15
cex	numerical value giving the amount by which plotting symbols should be magnified
cex.axis	magnification to be used for axis annotation relative to the current setting of cex
cex.main	magnification to be used for main titles relative to the current setting of cex
col	color of points on plot, see ?par 'Color Specification'
lwd	number line width. Default is 1.
...	arbitrary graphical parameters that will be passed to genericEGRETDotPlot function (see ?par for options)

See Also

plotFlowSingle

Examples

```
eList <- Choptank_eList

#Water year:
plotFour(eList)
# Graphs consisting of Jun-Aug
eList <- setPA(eList,paStart=6,paLong=3)
plotFour(eList)
```

`plotFourStats` *Makes four graphs of annual streamflow statistics on a single page*

Description

Part of the flowHistory system. The four statistics are 1-day maximum, annual mean, annual median, and annual 7-day minimum. Although there are a lot of optional arguments to this function, most are set to a logical default.

Data come from named list, which contains a Sample dataframe with the sample data, a Daily dataframe with the daily flow data, and an INFO dataframe with metadata. Each graph shows a loess smooth of the data that are plotted.

Usage

```
plotFourStats(eList, yearStart = NA, yearEnd = NA, printTitle = TRUE,
  runoff = FALSE, cex.main = 1.2, qUnit = 1, cex.axis = 1.2,
  cex = 0.8, col = "black", lwd = 1, ...)
```

Arguments

`eList`	named list with at least Daily and INFO dataframes
`yearStart`	A numeric value for year in which the graph should start, default is NA, which indicates that the graph should start with first annual value
`yearEnd`	A numeric value for year in which the graph should end, default is NA, which indicates that the graph should end with last annual value
`printTitle`	logical variable, if TRUE title is printed, if FALSE title is not printed, default is TRUE
`runoff`	logical variable, if TRUE the streamflow data are converted to runoff values in mm/day
`cex.main`	magnification to be used for main titles relative to the current setting of cex
`qUnit`	object of qUnit class `printqUnitCheatSheet`, or numeric represented the short code, or character representing the descriptive name.
`cex.axis`	magnification to be used for axis annotation relative to the current setting of cex
`cex`	numerical value giving the amount by which plotting symbols should be magnified

col	color of points on plot, see ?par 'Color Specification'
lwd	number line width. Default is 1.
...	arbitrary graphical parameters that will be passed to genericEGRETDotPlot function (see ?par for options)

See Also

plotFlowSingle

Examples

```
eList <- Choptank_eList

# Water year:
plotFourStats(eList)
# Graphs consisting of Jun-Aug
eList <- setPA(eList,paStart=6,paLong=3)
plotFourStats(eList)
```

---

plotMonthTrend *Plot monthly trend result from runPairs*

Description

Plot monthly trend result from runPairs. The change in concentration or flux is calculated from the runPairs function. This plotting function shows an arrow for each month. If the trend from year1 to year2 was increasing, the arrow is red and pointing up. If the trend was decreasing, the arrow is black and pointing down.

Usage

```
plotMonthTrend(pairResults, yMax = NA, arrowFactor = 0.75, flux = TRUE,
  printTitle = TRUE, concLab = 1, monthLab = 1)
```

Arguments

pairResults	results from runPairs.
yMax	numeric. Upper limit for plot. Default is NA, which will use the maximum of the data.
arrowFactor	numeric. Scaling factor for the size of the arrows. The arrows are automatically scaled to the overall trend. This scaling factor helps adjust how big/small they are.
flux	logical. TRUE is flux, FALSE is concentration. Default is TRUE.
printTitle	logical variable if TRUE title is printed, if FALSE title is not printed (this is best for a multi-plot figure)

concLab	object of concUnit class, or numeric represented the short code, or character representing the descriptive name. By default, this argument sets concentration labels to use either Concentration or Conc (for tiny plots). Units are taken from the eList$INFO$param.units. To use any other words than "Concentration" see `vignette(topic = "units", package = "EGRET")`.
monthLab	object of monthLabel class, or numeric represented the short code, or character representing the descriptive name.

Details

The flux values for each month are flow normalized monthly watershed yields expressed as kg/month/km^2. The concentrations are the mean flow normalized concentration, expressed in whatever concentration units the raw data are expressed as (typically mg/L).

Value

Base R plot of monthly trends

Examples

```
eList <- Choptank_eList
year1 <- 1985
year2 <- 2010



pairOut_1 <- runPairs(eList, year1, year2, windowSide = 0)

plotMonthTrend(pairOut_1)
plotMonthTrend(pairOut_1, flux = FALSE)

eList <- setPA(eList, paStart = 12, paLong = 3)
pairOut_2 <- runPairs(eList, year1, year2, windowSide = 0)

plotMonthTrend(pairOut_2)

eList <- setPA(eList, paStart = 1, paLong = 12)
pairOut_3 <- runPairs(eList, year1, year2, windowSide = 0)

plotMonthTrend(pairOut_3)
```

---

plotQTimeDaily *Plot of the discharge time series*

Description

Part of flowHistory component. Allows discharge record to only show those discharges above a given threshold

Although there are a lot of optional arguments to this function, most are set to a logical default.

Data come from named list, which contains a Daily dataframe with the daily flow data, and an INFO dataframe with metadata.

Usage

```
plotQTimeDaily(eList, yearStart = NA, yearEnd = NA, qLower = NA,
  qUnit = 1, logScale = FALSE, tinyPlot = FALSE, printTitle = TRUE,
  usgsStyle = FALSE, lwd = 3, col = "red", cex.main = 1.2,
  cex.lab = 1.2, customPar = FALSE, prettyDate = TRUE, ...)
```

Arguments

eList	named list with at least the Daily and INFO dataframes
yearStart	numeric indicating the starting year for the graph
yearEnd	numeric indicating the ending year for the graph (should be a time in decimal years that is after the last observations to be plotted)
qLower	numeric specifying the lower bound on discharges that are to be plotted, must be in the units specified by qUnit, default is NA (lower bound is zero)
qUnit	object of qUnit class. printqUnitCheatSheet, or numeric represented the short code, or character representing the descriptive name. Default is qUnit=1 (cubic feet per second)
logScale	logical whether or not to use a log scale in the y axis. Default is FALSE.
tinyPlot	logical variable, if TRUE plot is designed to be short and wide, default is FALSE.
printTitle	logical variable if TRUE title is printed, if FALSE title is not printed (this is best for a multi-plot figure)
usgsStyle	logical option to use USGS style guidelines. Setting this option to TRUE does NOT guarantee USGS compliance. It will only change automatically generated labels.
lwd	line width, a positive number, defaulting to 3
col	specification for the default plotting color
cex.main	magnification to be used for main titles relative to the current setting of cex
cex.lab	magnification to be used for x and y labels relative to the current setting of cex
customPar	logical defaults to FALSE. If TRUE, par() should be set by user before calling this function (for example, adjusting margins with par(mar=c(5,5,5,5))). If customPar FALSE, EGRET chooses the best margins depending on tinyPlot.
prettyDate	logical use 'pretty' limits for date axis if TRUE, or force the yearStart/yearEnd as limits if FALSE
...	arbitrary graphical parameters that will be passed to genericEGRETDotPlot function (see ?par for options)

See Also

selectDays, genericEGRETDotPlot

Examples

```
eList <- Choptank_eList
# Water year:
plotQTimeDaily(eList)
plotQTimeDaily(eList, yearStart=1990, yearEnd=2000,qLower=1500)
plotQTimeDaily(eList, prettyDate=FALSE)
```

---

plotResidPred *Plot of the residuals from WRTDS versus the estimated values (all in log concentration units)*

Description

This function produces a plot of the residuals from WRTDS, expressed in natural log concentration units versus the estimated values, also in natural log concentration units. These estimates are the log-space estimates prior to bias-correction. The function provides an alternative for viewing the standardized residuals, where the each residual is divided by its estimated standard error.

Although there are a lot of optional arguments to this function, most are set to a logical default.

Data come from named list, which contains a Sample dataframe with the sample data, and an INFO dataframe with metadata.

Usage

```
plotResidPred(eList, stdResid = FALSE, tinyPlot = FALSE,
  printTitle = TRUE, col = "black", lwd = 1, cex = 0.8,
  cex.axis = 1.1, cex.main = 1.1, customPar = FALSE,
  randomCensored = FALSE, concLab = 1, ...)
```

Arguments

eList	named list with at least the Sample and INFO dataframes
stdResid	logical variable, if TRUE it uses the standardized residual, if FALSE it uses the actual, default is FALSE
tinyPlot	logical variable, if TRUE plot is designed to be plotted small as part of a multi-part figure, default is FALSE.
printTitle	logical variable if TRUE title is printed, if FALSE not printed (this is best for a multi-plot figure)
col	color of points on plot, see ?par 'Color Specification'
lwd	number line width
cex	numerical value giving the amount by which plotting symbols should be magnified

cex.axis	magnification to be used for x and y labels relative to the current setting of cex
cex.main	magnification to be used for main titles relative to the current setting of cex
customPar	logical defaults to FALSE. If TRUE, par() should be set by user before calling this function (for example, adjusting margins with par(mar=c(5,5,5,5))). If customPar FALSE, EGRET chooses the best margins depending on tinyPlot.
randomCensored	logical. Show censored residuals as randomized.
concLab	object of concUnit class, or numeric represented the short code, or character representing the descriptive name. By default, this argument sets concentration labels to use either Concentration or Conc (for tiny plots). Units are taken from the eList$INFO$param.units. To use any other words than "Concentration" see `vignette(topic = "units", package = "EGRET")`.
...	arbitrary graphical parameters that will be passed to genericEGRETDotPlot function (see ?par for options)

See Also

selectDays, genericEGRETDotPlot

Examples

```
eList <- Choptank_eList
# Water year:
plotResidPred(eList)
# Graphs consisting of Jun-Aug
eList <- setPA(eList, paStart=6,paLong=3)
plotResidPred(eList)
```

---

plotResidQ *Plot of the residuals from WRTDS (in log concentration units) versus the discharge*

Description

This function produces a plot of the residuals from WRTDS, expressed in natural log concentration units versus the discharge shown on a log scale. The function also provides an alternative for viewing the standardized residuals, where the each residual is divided by its estimated standard error

Although there are a lot of optional arguments to this function, most are set to a logical default.

Data come from named list, which contains a Sample dataframe with the sample data, and an INFO dataframe with metadata.

Usage

```
plotResidQ(eList, qUnit = 2, tinyPlot = FALSE, stdResid = FALSE,
  printTitle = TRUE, col = "black", lwd = 1, cex = 0.8,
  cex.axis = 1.1, cex.main = 1.1, rmSciX = FALSE, customPar = FALSE,
  randomCensored = FALSE, usgsStyle = FALSE, ...)
```

Arguments

eList	named list with at least the Sample and INFO dataframes
qUnit	object of qUnit class printqUnitCheatSheet, or numeric represented the short code, or character representing the descriptive name.
tinyPlot	logical variable, if TRUE plot is designed to be plotted small as part of a multi-part figure, default is FALSE.
stdResid	logical variable, if TRUE it uses the standardized residual, if FALSE it uses the actual, default is FALSE
printTitle	logical variable if TRUE title is printed, if FALSE not printed (this is best for a multi-plot figure)
col	color of points on plot, see ?par 'Color Specification'
lwd	number line width
cex	numerical value giving the amount by which plotting symbols should be magnified
cex.axis	magnification to be used for axis annotation relative to the current setting of cex
cex.main	magnification to be used for main titles relative to the current setting of cex
rmSciX	logical defaults to FALSE, changes x label from scientific to fixed
customPar	logical defaults to FALSE. If TRUE, par() should be set by user before calling this function (for example, adjusting margins with par(mar=c(5,5,5,5))). If customPar FALSE, EGRET chooses the best margins depending on tinyPlot.
randomCensored	logical. Show censored residuals as randomized.
usgsStyle	logical option to use USGS style guidelines. Setting this option to TRUE does NOT guarantee USGS complience. It will only change automatically generated labels.
...	arbitrary graphical parameters that will be passed to genericEGRETDotPlot function (see ?par for options)

See Also

selectDays, genericEGRETDotPlot

Examples

```
eList <- Choptank_eList
# Water year:
plotResidQ(eList)
# Graphs consisting of Jun-Aug
eList <- setPA(eList, paStart=6,paLong=3)
plotResidQ(eList)
```

`plotResidTime` *Plot of the residuals from WRTDS (in log concentration units) versus time*

Description

This function produces a plot of the residuals from WRTDS, expressed in natural log concentration units versus time. It also provides an alternative for viewing the standardized residuals, where the each residual is divided by its estimated standard error.

Although there are a lot of optional arguments to this function, most are set to a logical default.

Data come from named list, which contains a Sample dataframe with the sample data, and an INFO dataframe with metadata.

Usage

```
plotResidTime(eList, stdResid = FALSE, printTitle = TRUE, hLine = TRUE,
  tinyPlot = FALSE, col = "black", lwd = 1, cex = 0.8,
  cex.axis = 1.1, cex.main = 1.1, customPar = FALSE,
  randomCensored = FALSE, ...)
```

Arguments

`eList`	named list with at least the Sample and INFO dataframes
`stdResid`	logical variable, if TRUE it uses the standardized residual, if FALSE it uses the actual, default is FALSE
`printTitle`	logical variable if TRUE title is printed, if FALSE not printed (this is best for a multi-plot figure)
`hLine`	inserts horizontal line at zero
`tinyPlot`	logical variable, if TRUE plot is designed to be plotted small, as a part of a multipart figure, default is FALSE
`col`	color of points on plot, see ?par 'Color Specification'
`lwd`	number line width
`cex`	numerical value giving the amount by which plotting symbols should be magnified
`cex.axis`	magnification to be used for axis annotation relative to the current setting of cex
`cex.main`	magnification to be used for main titles relative to the current setting of cex
`customPar`	logical defaults to FALSE. If TRUE, par() should be set by user before calling this function (for example, adjusting margins with par(mar=c(5,5,5,5))). If customPar FALSE, EGRET chooses the best margins depending on tinyPlot.
`randomCensored`	logical. Show censored residuals as randomized.
`...`	arbitrary graphical parameters that will be passed to genericEGRETDotPlot function (see ?par for options)

See Also

selectDays, genericEGRETDotPlot

Examples

```
eList <- Choptank_eList
# Water year:
plotResidTime(eList)
# Graphs consisting of Jun-Aug
eList <- setPA(eList, paStart=6,paLong=3)
plotResidTime(eList)
```

---

plotSDLogQ *Graph of the standard deviation of the log of daily discharge versus year*

Description

Graph of the standard deviation of the log of daily discharge versus year

Although there are a lot of optional arguments to this function, most are set to a logical default.

Data come from named list, which contains a Daily dataframe with the daily flow data, and an INFO dataframe with metadata.

Usage

```
plotSDLogQ(eList, yearStart = NA, yearEnd = NA, window = 15,
  sdMax = NA, printTitle = TRUE, tinyPlot = FALSE, printStaName = TRUE,
  printPA = TRUE, cex = 0.8, cex.main = 1.1, cex.axis = 1.1, lwd = 2,
  customPar = FALSE, ...)
```

Arguments

eList	named list with at least the Daily and INFO dataframes
yearStart	numeric is the calendar year of the first value to be included in graph, default is NA, which plots from the start of the period of record
yearEnd	numeric is the calendar year of the last value to be included in graph, default is NA, which plots to the end of the period of record
window	numeric which is the full width, in years, of the time window over which the standard deviation is computed, default = 15
sdMax	numeric is the maximum value to be used on the vertical axis of the graph, default is NA (which allows it to be set automatically by the data)
printTitle	logical variable if TRUE title is printed, if FALSE title is not printed (this is best for a multi-plot figure), default is TRUE
tinyPlot	logical variable if TRUE plot is designed to be small, if FALSE it is designed for page size, default is FALSE (not fully implemented yet)

printStaName	logical variable, if TRUE print the station name, if FALSE do not, default is TRUE
printPA	logical variable, if TRUE print the period of analysis information in the plot title, if FALSE leave it out, default is TRUE
cex	numerical value giving the amount by which plotting symbols should be magnified
cex.main	magnification to be used for main titles relative to the current setting of cex
cex.axis	magnification to be used for axis annotation relative to the current setting of cex
lwd	line width, a positive number, defaulting to 2
customPar	logical defaults to FALSE. If TRUE, par() should be set by user before calling this function (for example, adjusting margins with par(mar=c(5,5,5,5))). If customPar FALSE, EGRET chooses the best margins depending on tinyPlot.
...	arbitrary graphical parameters that will be passed to genericEGRETDotPlot function (see ?par for options)

See Also

selectDays, genericEGRETDotPlot

Examples

```
eList <- Choptank_eList

# Water year:
plotSDLogQ(eList)
plotSDLogQ(eList, 1998, 2000)
```

---

plotTimeSlice *plotTimeSlice*

Description

Plot of either concentration or flux over time showing both the WRTDS and WRTDSKalman estimates.

Usage

```
plotTimeSlice(eList, start = NA, end = NA, conc = TRUE, fluxUnit = 3,
  usgsStyle = FALSE)
```

Arguments

eList	named list with at least the Daily, Sample, and INFO dataframes. This eList must be run through WRTDSKalman.
start	numeric start of DecYear for plot. If NA, plot will start at the earliest date in the record.
end	numeric end of DecYear for plot. If NA, plot will end at the latest date in the record.
conc	logical. If TRUE, plot concentration, otherwise plot flux.
fluxUnit	number representing entry in pre-defined fluxUnit class array. printFluxUnitCheatSheet
usgsStyle	logical option to use USGS style guidelines. Setting this option to TRUE does NOT guarantee USGS compliance. It will only change automatically generated labels

Details

In the plot title, Ratio of means is mean of WRTDSKalman estimates to the WRTDS Classic estimates. Ratio only calculated on the data shown in the figure, not the whole series. In the plot, red dots are measured values, blue dots are plotted at the reporting limit for those values that are censored.

Examples

```
eList <- Choptank_eList
eList <- WRTDSKalman(eList, niter = 10)

plotTimeSlice(eList, start = 1990, end = 1991, conc = TRUE)

plotTimeSlice(eList, start = 1990, end = 1991, conc = FALSE)

plotTimeSlice(eList, start = NA, end = 1991, conc = FALSE)
```

---

plotWRTDSKalman *plotWRTDSKalman*

Description

Two plots to check the flux estimates using the WRTDS_K vs classic WRTDS. The first is annual flux over time, where the two fluxes are shown in different colors. The second is WRTDS vs WRTDSKalman flux estimates. The graphs can be output either on top of each other, or side by side using the sideBySide argument.

Usage

```
plotWRTDSKalman(eList, sideBySide = FALSE, fluxUnit = 9,
  usgsStyle = FALSE)
```

Arguments

eList	named list with at least the Daily, Sample, and INFO dataframes. This eList must be run through WRTDSKalman.
sideBySide	logical. If TRUE, the two plots will be plotted side by side, otherwise, one by one vertically.
fluxUnit	number representing entry in pre-defined fluxUnit class array. printFluxUnitCheatSheet
usgsStyle	logical option to use USGS style guidelines. Setting this option to TRUE does NOT guarantee USGS compliance. It will only change automatically generated labels

Examples

```
eList <- Choptank_eList
eList <- WRTDSKalman(eList, niter = 10)
plotWRTDSKalman(eList)

plotWRTDSKalman(eList, sideBySide = TRUE)
```

---

populateConcentrations

Populate Concentration Columns

Description

Creates ConcLow, ConcHigh, Uncen (0 if censored, 1 if uncensored) columns for Sample data frame for WRTDS analysis.

Usage

```
populateConcentrations(rawData)
```

Arguments

rawData	vector with value and code columns

Value

concentrationColumns dataframe

Examples

```
code <- c("","<","")
value <- c(1,2,3)
dataInput <- data.frame(value, code, stringsAsFactors=FALSE)
concentrationDF <- populateConcentrations(dataInput)
```

`populateDaily` *Populate Daily data frame*

Description

Using a data frame that has at least Date, Q, Qualifier, populates the rest of the basic Daily data frame used in EGRET analysis.

Usage

```
populateDaily(rawData, qConvert, verbose = TRUE, adjust = TRUE,
  fill = FALSE, maxgap = 21, fill_type = "interpolation")
```

Arguments

`rawData`	dataframe contains at least Date, Q, Qualifier columns.
`qConvert`	numeric conversion to cubic meters per second.
`verbose`	logical specifying whether or not to display messages.
`adjust`	logical specifying whether or not to add a constant to zero values to allow log transformation. Defaults to TRUE.
`fill`	logical specifying whether to fill NA values by linear interpolation. Defaults to FALSE.
`maxgap`	Maximum number of NA days allowed for interpolating gaps. Default is 21. Only used if fill is set to TRUE.
`fill_type`	character to define what process to fill missing data. Options are "interpolation" - linear interpolation from the ‘zoo::na.approx‘, or "log_interp" - linear interpolation in the log space. Only used if fill is set to TRUE.

Value

A data frame 'Daily' with the following columns:

Name	Type	Description
Q	numeric	Discharge in m^3/s
Julian	integer	Number of days since Jan. 1, 1850
Month	integer	Month of the year [1-12]
Day	integer	Day of the year [1-366]
DecYear	numeric	Decimal year
MonthSeq	integer	Number of months since January 1, 1850
Qualifier	character	Qualifying code
i	integer	Index of days, starting with 1
LogQ	numeric	Natural logarithm of Q
Q7	numeric	7 day running average of Q
Q30	numeric	30 day running average of Q

Author(s)

Robert M. Hirsch <rhirsch@usgs.gov>

See Also

readNWISDaily, readUserDaily

Examples

```
Date <- as.character(seq(from = as.Date("2001/1/1"),
                         to = as.Date("2002/1/2"),
                         by = "day"))
Q <- c(-1:365)
Qualifier <- rep("",367)
dataInput_complete <- data.frame(Date, Q, Qualifier)
dataInput <- dataInput_complete[-4:-5,]

# No fill, but with 0 and negative:
Daily <- populateDaily(dataInput, qConvert = 1)

# No negatives/zeros:
Q <- 2+sin(seq(from = 0, to = 2*pi, length.out = 367))
Q <- jitter(Q, factor = 500)
plot(Q, ylim = c(0, 3.2))
dataInput_complete <- data.frame(Date, Q, Qualifier)
# Remove some rows to test missing:
dataInput <- dataInput_complete[-4:-5,]
dataInput <- dataInput[-10:-20,]

# No fill:
Daily <- populateDaily(dataInput, qConvert = 1)
plot(Daily$Date[1:30], Daily$Q[1:30], type = "b", ylim = c(0, 3.2))

# Linear interpolation:
Daily_fill <- populateDaily(dataInput,
                            qConvert = 1,
                            fill = TRUE,
                            fill_type = "interpolation")
plot(Daily_fill$Date[1:30],
     Daily_fill$Q[1:30],
     col = as.factor(Daily_fill$Qualifier[1:30]),
     type = "b", pch = 16, ylim = c(0, 3.2),
     main = "Linear Interpolation")


# Add a gap that is too big do deal with:
dataInput <- dataInput_complete[-4:-20,]
dataInput <- dataInput[-200:-255,]

Daily_interp <- populateDaily(dataInput,
                              qConvert = 1,
                              fill = TRUE,
```

```
                              fill_type = "interpolation")
plot(Daily_interp$Date, Daily_interp$Q,
     col = as.factor(Daily_interp$Qualifier),
     main = "Linear Interpolation",
     type = "b", pch = 16, ylim = c(0, 3.2))
plot(Daily_interp$Date[1:50], Daily_interp$Q[1:50],
     col = as.factor(Daily_interp$Qualifier[1:50]),
     main = "Linear Interpolation",
     type = "b", pch = 16, ylim = c(0, 3.2))

Daily_log_interp <- populateDaily(dataInput,
                              qConvert = 1,
                              fill = TRUE,
                              fill_type = "log_interp")
plot(Daily_log_interp$Date, Daily_log_interp$Q,
     col = as.factor(Daily_log_interp$Qualifier),
     main = "Linear Interpolation in Log Scale",
     type = "b", pch = 16, ylim = c(0, 3.2))
plot(Daily_log_interp$Date[1:50], Daily_log_interp$Q[1:50],
     col = as.factor(Daily_log_interp$Qualifier[1:50]),
     main = "Linear Interpolation in Log Scale",
     type = "b", pch = 16, ylim = c(0, 3.2))


# Real data:
eList <- Choptank_eList
Daily_chop <- eList$Daily
df <- Daily_chop[,c("Date", "Q")]
df <- df[-2:-5, ]
df <- df[-100:-200,]
D2 <- populateDaily(df, 1, fill = TRUE)
plot(D2$Date[1:20], D2$Q[1:20],
     col = as.factor(D2$Qualifier[1:20]),
     main = "Linear Interpolation",
     type = "b", pch = 16)
plot(D2$Date[1:110], D2$Q[1:110],
     col = as.factor(D2$Qualifier[1:110]),
     main = "Linear Interpolation",
     type = "b", pch = 16)
```

`populateDailySamp` *Merge concentration to Daily*

Description

Used for the WRTDS Kalman set of functions, this function merges the ConcAve into the Daily data frame, renaming it "trueConc", then calculates the "trueFlux", and "stdResid".

Usage

```
populateDailySamp(eList)
```

Arguments

eList	named list with the INFO, Daily, and Sample dataframes and surfaces matrix

Examples

```
eList <- Choptank_eList
Daily2 <- populateDailySamp(eList)
```

---

populateDateColumns *Populate Date Columns*

Description

Creates various date columns for WRTDS study.

Usage

```
populateDateColumns(rawData)
```

Arguments

rawData	vector with dateTime

Value

DateFrame dataframe

Examples

```
dateTime <- c('1984-02-28 13:56',
              '1984-03-01 00:00',
              '1986-03-01 00:00',
              '1986-10-15 00:00')

expandedDateDF <- populateDateColumns(dateTime)
expandedDateDF

dateTime <- c('1984-02-28',
              '1984-03-01',
              '1986-03-01',
              '1986-10-15')
expandedDateDF <- populateDateColumns(dateTime)
expandedDateDF
```

populateParameterINFO *Populate Parameter Information Columns*

Description

Populates INFO data frame with additional user-supplied information concerning the measured parameter.

Usage

```
populateParameterINFO(parameterCd, INFO, interactive = TRUE)
```

Arguments

parameterCd	character USGS parameter code
INFO	dataframe with value and code columns. Default is INFO
interactive	logical Option for interactive mode. If TRUE, there is user interaction for error handling and data checks. Default is TRUE. If running in batch, should be set to FALSE.

Value

INFO dataframe

Examples

```
INFO <- dataRetrieval::read_waterdata_monitoring_location('USGS-01594440')
parameterCd <- "01075"
parameterData <- dataRetrieval::read_waterdata_parameter_codes(parameterCd)
INFO$param.nm <- parameterData$parameter_name
INFO$param.units <- parameterData$unit_of_measure
INFO$paramShortName <- parameterData$parameter_name
INFO$paramNumber <- parameterData$parameter_code

INFO <- populateParameterINFO(parameterCd, INFO, interactive = FALSE)
```

populateSampleColumns *Populate Sample Columns*

Description

Creates ConcAve and ConcLow based on Uncen. Removes any samples with NA values in ConcHigh.

Usage

```
populateSampleColumns(rawData)
```

Arguments

rawData	dataframe with dateTime, ConcLow, ConcHigh, Uncen

Value

Sample2 dataframe with columns: Date, ConcLow, ConcHigh, Uncen, ConcAve, Julian, Month, Day, DecYear, MonthSeq, waterYear, SinDY, and CosDY (DY = decimal year)

Examples

```
dateTime <- c('1985-01-01', '1985-01-02', '1985-01-03')
ConcLow <- c(1,2,0)
ConcHigh <- c(1,2,3)
Uncen <- c(1,1,0)
dataInput <- data.frame(dateTime, ConcLow, ConcHigh, Uncen, stringsAsFactors=FALSE)
Sample <- populateSampleColumns(dataInput)
```

---

populateSiteINFO *Populate Site Information Columns*

Description

Populates INFO data frame with additional user-supplied information. Also removes fields not related to WRTDS study.

Usage

```
populateSiteINFO(INFO, siteNumber, interactive = TRUE)
```

Arguments

INFO	dataframe with value and code columns
siteNumber	character USGS site number
interactive	logical Option for interactive mode. If TRUE, there is user interaction for error handling and data checks. Default is TRUE. If running in batch, should be set to FALSE.

Value

INFO dataframe

Examples

```
INFO <- dataRetrieval::read_waterdata_monitoring_location('USGS-01594440')
siteNumber <- "01594440"
siteINFO <- populateSiteINFO(INFO, siteNumber, interactive = FALSE)
```

`precompute_L` *precompute_L*

Description

Builds the bidiagonal Cholesky factor of the tridiagonal PRECISION matrix shared by every call of genmissing for a given value of rho.

Usage

```
precompute_L(rho, N_max)
```

Arguments

`rho`	the lag one autocorrelation
`N_max`	the maximum number of days between samples

Value

named list with elements Ldiag a numeric vector of the diagonal of the Cholesky factor and Lsub a numeric vector of the subdiagonal of the Cholesky factor

---

`print.egret` *EGRET helper functions*

Description

A small collection of helper functions

Usage

```
## S3 method for class 'egret'
print(x, ...)

## S3 method for class 'egret'
plot(x, ...)

nDischarge(x)
```

```
nObservations(x)

nCensoredVals(x)
```

Arguments

x	EGRET object
...	additional parameters

See Also

multiPlotDataOverview

Examples

```
Choptank_eList
print(Arkansas_eList)
plot(Choptank_eList)
plot(Choptank_eList, cex.main=0.7)
nDischarge(Arkansas_eList)
nObservations(Arkansas_eList)
nCensoredVals(Arkansas_eList)
```

---

printFluxUnitCheatSheet

Reminder to user of flux unit properties (such as kg/day, tons/year, etc).

Description

Cheat sheet to print out pre-defined flux unit properties from fluxUnit class Flux units included:

Number	ObjectName	shortName	unitFactor	unitName
1	POUNDS_DAY	lbs/day	2.204623	pounds/day
2	TONS_DAY	tons/day	0.001102	tons/day
3	KG_DAY	kg/day	1	kg/day
4	THOUSAND_KG_DAY	10^3 kg/day	0.001	thousands of kg/day
5	TONS_YEAR	tons/yr	0.402619	tons/year
6	THOUSAND_TONS_YEAR	10^3 tons/yr	0.000402619	thousands of tons/year
7	MILLION_TONS_YEAR	10^6 tons/yr	4.02619e-07	millions of tons/year
8	THOUSAND_KG_YEAR	10^3 kg/yr	0.36525	thousands of kg/year
9	MILLION_KG_YEAR	10^6 kg/yr	0.00036525	millions of kg/year
10	BILLION_KG_YEAR	10^9 kg/yr	3.6525e-07	billions of kg/year
11	thousandTonsDay	10^3 tons/day	1.102e-06	thousands of tons/day
12	millionKgDay	10^6 kg/day	1e-06	millions of kg/day
13	kgYear	kg/year	365.25	kg/year

Usage

```
printFluxUnitCheatSheet()
```

Examples

```
printFluxUnitCheatSheet()
```

`printGroups` *Print information about group analysis*

Description

Prints the information from the `runGroups` function. This could be used to save the output to a text file.

Usage

```
printGroups(eList, groupResults)
```

Arguments

`eList`	named list with at least the Daily, Sample, and INFO dataframes
`groupResults`	output of `runGroups`.

Value

text to console

Examples

```
eList <- Choptank_eList


groupOut_1 <- runGroups(eList,  windowSide = 0,
                        group1firstYear = 1980, group1lastYear = 1990,
                        group2firstYear = 1995, group2lastYear = 2005)

printGroups(eList, groupOut_1)
```

printPairs *Print information about pairs analysis*

Description

Prints the information from the runPairs function. This could be used to save the output to a text file.

Usage

```
printPairs(eList, pairResults)
```

Arguments

eList	named list with at least the Daily, Sample, and INFO dataframes
pairResults	output of runGroups.

Value

text to console

Examples

```
eList <- Choptank_eList
year1 <- 1985
year2 <- 2010


pairOut_1 <- runPairs(eList,
                      year1, year2,
                      windowSide = 0)

printPairs(eList, pairOut_1)
```

printqUnitCheatSheet *Reminder to user of flow Unit properties such as cubic meters per second or thousands of cubic feet per second.*

Description

Cheat sheet to print out pre-defined qUnit properties from qUnit class. Flow units included:

Number	ObjectName	shortName	unitFactor
1	cfs	Cubic Feet per Second	35.31467
2	cms	Cubic Meters per Second	1
3	thousandCfs	Thousand Cubic Feet per Second	0.03531467
4	thousandCms	Thousand Cubic Meters per Second	0.001
5	mmDay	mm per day	
6	mmYear	mm per year	

Usage

```
printqUnitCheatSheet()
```

Examples

```
printqUnitCheatSheet()
```

`printSeries` *Print annual results for a given streamflow statistic*

Description

Part of the flowHistory system. The index of the flow statistics is istat. These statistics are: (1) 1-day minimum, (2) 7-day minimum, (3) 30-day minimum, (4) median (5) mean, (6) 30-day maximum, (7) 7-day maximum, and (8) 1-day maximum.

Usage

```
printSeries(eList, istat, qUnit = 1, runoff = FALSE)
```

Arguments

`eList`	named list with at least the Daily and INFO dataframes
`istat`	A numeric value for the flow statistic to be graphed (possible values are 1 through 8)
`qUnit`	object of qUnit class `printqUnitCheatSheet`, or numeric represented the short code, or character representing the descriptive name. Default is 1, which is cubic feet per second.
`runoff`	logical variable, if TRUE the streamflow data are converted to runoff values in mm/day

Details

The smoothing algorithm is as defined in `makeAnnualSeries`. smoothing window is defined by the eList$INFO$window value (default = 20)

Value

data frame with:

years	integer	year
qActual	numeric	Actual flow statistic (based on istat)
qSmooth	numeric	Smoothed flow statistic

Examples

```
eList <- Choptank_eList
printReturn <- printSeries(eList, 5)
```

---

processQWData *Processing of Water Quality Data*

Description

Processes water quality data. This function looks at detection limit and detection conditions to determine if a value is left censored or not. Censored values are given the qualifier "<". The dataframe is also converted from a long to wide format.

Usage

```
processQWData(data)
```

Arguments

data	dataframe from Water Quality Portal

Value

data dataframe with first column dateTime, and at least one qualifier and value columns (subsequent qualifier/value columns could follow depending on the number of parameter codes)

See Also

readWQPqw

Examples

```
rawWQP <- dataRetrieval::readWQPdata(siteid = '21FLEECO_WQX-IMPRGR80',
                                     characteristicName = 'Phosphorus',
                                     service = "Result",
                                     dataProfile = "resultPhysChem")
Sample2 <- processQWData(rawWQP)
```

qUnit-class *qUnit class*

Description

Some details about the qUnit class

Details

qshortName A character specifying the short name.

qUnitFactor A numeric representing the conversion factor

qUnitName A character specifying the full name.

qUnitExpress An expression specifying the full name.

unitUSGS A character specifying flux with full text.

qUnitTiny An expression specifying the abbreviated name.

shortCode A number for quick lookup

prefix A character specifying the general type of measurement.

randomSubset *randomSubset*

Description

Calculates a random subset of the data based on repeated values from a specified column.

Usage

```
randomSubset(df, colName, seed = NA)
```

Arguments

df	data frame. Must include a column named by the argument colName.
colName	column name to check for duplicates
seed	integer value. Defaults to NA, which will not change the current seed. Setting the seed to any given value can be used to create repeatable output.

Examples

```
df <- data.frame(Julian = c(1,2,2,3,4,4,4,6),
                 y = 1:8)
df
df_random <- randomSubset(df, "Julian")
df_random
```

`readDataFromFile` *Basic Data Import for Water Flow Data*

Description

Imports data from user-supplied data file. Specifically used to import water flow data for use in the EGRET package. For EGRET usage, the first column is expected to be dates. If the data is daily data, then next column is expected to be the measured values. If the data is sampled data, the next column is remark codes, and the third column is values. If the date format is not automatically detected, the format can be specified using the "format" argument.

Usage

```
readDataFromFile(filePath, fileName, hasHeader = TRUE, separator = ",",
  format = "%m/%d/%Y")
```

Arguments

`filePath`	character specifying the path to the file. If it is in the working directory, use ".".
`fileName`	character name of file to open.
`hasHeader`	logical true if the first row of data is the column headers
`separator`	character character that separates data cells. The default is "," for a csv file. Tab delimited would be "\t".
`format`	character indicating the format of the date (which should be in the first column). Default is "%m\/%d/%Y". See `?strptime` for options. The code will initially look for R's standard YYYY-MM-DD, and check this format as a backup.

Value

retval dataframe with dateTime, value, and code columns

Examples

```
filePath <- system.file("extdata", package="EGRET")
fileName <- 'ChoptankRiverFlow.txt'
ChopData <- readDataFromFile(filePath,fileName, separator="\t")
```

readNWISDaily *Import NWIS Daily Data for EGRET analysis*

Description

Imports daily data from NWIS web service. This function gets the data from here: https://waterservices.usgs.gov/

Usage

```
readNWISDaily(siteNumber, parameterCd = "00060", startDate = "",
  endDate = "", verbose = TRUE, convert = TRUE, adjust = TRUE,
  fill = FALSE, maxgap = 21, fill_type = "interpolation")
```

Arguments

siteNumber	character USGS site number. This is usually an 8 digit number.
parameterCd	character USGS parameter code. This is usually an 5 digit number.
startDate	character starting date for data retrieval in the form YYYY-MM-DD.
endDate	character ending date for data retrieval in the form YYYY-MM-DD.
verbose	logical specifying whether or not to display messages.
convert	logical Option to include a conversion from cfs to cms (35.314667). The default is TRUE which is appropriate for using NWIS data in the EGRET package. Set this to FALSE to not include the conversion. If the parameter code is not 00060 (NWIS discharge), there is no conversion applied.
adjust	logical specifying whether or not to add a constant to zero values to allow log transformation. Defaults to TRUE.
fill	logical specifying whether to fill NA values by linear interpolation. Defaults to FALSE.
maxgap	Maximum number of NA days allowed for interpolating gaps. Default is 21. Only used if fill is set to TRUE.
fill_type	character to define what process to fill missing data. Options are "interpolation" - linear interpolation from the ‘zoo::na.approx‘, or "log_interp" - linear interpolation in the log space. Only used if fill is set to TRUE.

Value

A data frame 'Daily' with the following columns:

Name	Type	Description
Date	Date	Date
Q	numeric	Discharge in m^3/s
Julian	integer	Number of days since Jan. 1, 1850
Month	integer	Month of the year [1-12]

Day	integer	Day of the year [1-366]
DecYear	numeric	Decimal year
MonthSeq	integer	Number of months since January 1, 1850
Qualifier	character	Qualifying code
i	integer	Index of days, starting with 1
LogQ	numeric	Natural logarithm of Q
Q7	numeric	7 day running average of Q
Q30	numeric	30 day running average of Q

See Also

read_waterdata_daily, populateDaily

Examples

```
Daily <- readNWISDaily('01594440','00060',
                       '2004-01-01', '2024-03-31')
DailySuspSediment <- readNWISDaily('01594440','80154',
                                   '1985-01-01', '1985-03-31',
                                   convert = FALSE)
```

---

readNWISSample *Import USGS Sample Data for EGRET analysis*

Description

Imports data from USGS web service. For raw data, use read_waterdata_samples from the dataRetrieval package. This function will retrieve the raw data, and compress it (summing constituents) if more than 1 parameter code is supplied. See section 3.2.4 of the vignette for more details.

Usage

```
readNWISSample(siteNumber, parameterCd, startDate = "", endDate = "",
  verbose = TRUE)
```

Arguments

siteNumber	character USGS site number. This is usually an 8 digit number
parameterCd	character USGS parameter code. This is usually an 5 digit number.
startDate	character starting date for data retrieval in the form YYYY-MM-DD. Default is empty quotes "" which will retrieve the full period of record.

endDate	character ending date for data retrieval in the form YYYY-MM-DD. Default is empty quotes "" which will retrieve the full period of record.
verbose	logical specifying whether or not to display progress message

Value

A data frame 'Sample' with the following columns:

Name	Type	Description
Date	Date	Date
ConcLow	numeric	Lower limit of concentration
ConcHigh	numeric	Upper limit of concentration
Uncen	integer	Uncensored data (1=TRUE, 0=FALSE)
ConcAve	numeric	Average concentration
Julian	integer	Number of days since Jan. 1, 1850
Month	integer	Month of the year [1-12]
Day	integer	Day of the year [1-366]
DecYear	numeric	Decimal year
MonthSeq	integer	Number of months since January 1, 1850
SinDY	numeric	Sine of the DecYear
CosDY	numeric	Cosine of the DecYear

See Also

compressData, populateSampleColumns, read_waterdata_samples

Examples

```
# These examples require an internet connection to run

Sample_01075 <- readNWISSample(siteNumber = '01594440',
                               parameterCd = '01075',
                               startDate = '1985-01-01',
                               endDate = '1995-03-31')
Sample_char <- readNWISSample(siteNumber = 'USGS-01594440',
                              parameterCd = 'Silver',
                              startDate = '1985-01-01',
                              endDate = '1995-03-31')
```

---

readUserDaily *Import user daily data for EGRET analysis*

Description

Imports data from a user-supplied file, and converts it to a Daily data frame, appropriate for WRTDS calculations. The file can use most any separators as a delimiter. The default is comma separated.

Usage

```
readUserDaily(filePath, fileName, hasHeader = TRUE, separator = ",",
  qUnit = 1, format = "%m/%d/%Y", verbose = TRUE, adjust = TRUE,
  fill = FALSE, maxgap = 21, fill_type = "interpolation")
```

Arguments

filePath	character specifying the path to the file. If it is in the working directory, use ".".
fileName	character name of file to open
hasHeader	logical true if the first row of data is the column headers
separator	character character that separates data cells. The default is "," for a csv file. Tab delimited would be "\t".
qUnit	number 1 is cubic feet per second, 2 is cubic meters per second, 3 is 10^3 cubic feet per second, and 4 is 10^3 cubic meters per second. The default is 1.
format	character indicating the format of the date (which should be in the first column). Default is "%m\/%d/%Y". See ?strptime for options. The code will initially look for R's standard YYYY-MM-DD, and check this format as a backup.
verbose	logical specifying whether or not to display progress message
adjust	logical specifying whether or not to add a constant to zero values to allow log transformation. Defaults to TRUE.
fill	logical specifying whether to fill NA values by linear interpolation. Defaults to FALSE.
maxgap	Maximum number of NA days allowed for interpolating gaps. Default is 21. Only used if fill is set to TRUE.
fill_type	character to define what process to fill missing data. Options are "interpolation" - linear interpolation from the 'zoo::na.approx', or "log_interp" - linear interpolation in the log space. Only used if fill is set to TRUE.

Details

The first column is expected to be dates, the second column is expected to be discharge values. If the date format is not automatically detected, the format can be specified using the "format" argument.

Value

A data frame 'Daily' with the following columns:

Name	Type	Description
Date	Date	Date
Q	numeric	Discharge in m^3/s
Julian	integer	Number of days since Jan. 1, 1850
Month	integer	Month of the year [1-12]
Day	integer	Day of the year [1-366]
DecYear	numeric	Decimal year
MonthSeq	integer	Number of months since January 1, 1850

Qualifier	character	Qualifying code
i	integer	Index of days, starting with 1
LogQ	numeric	Natural logarithm of Q
Q7	numeric	7 day running average of Q
Q30	numeric	30 day running average of Q

Examples

```
filePath <- system.file("extdata", package="EGRET")
fileName <- "ChoptankRiverFlow.txt"
Daily <- readUserDaily(filePath,fileName,separator="\t")
```

`readUserSample` *Import user-supplied sample data for EGRET analysis*

Description

Imports data from a user-supplied file, and converts it to a Sample data frame (including summing multiple constituents), appropriate for EGRET analysis. First column is date, second is remark code, and third is value. If multiple constituents are to be combined with interval censoring, additional pairs of columns can be inserted, each pair starting with remark code (specifically looking for <), followed by the values. If the date format is not automatically detected, the format can be specified using the "format" argument.

Usage

```
readUserSample(filePath, fileName, hasHeader = TRUE, separator = ",",
  verbose = TRUE, format = "%m/%d/%Y")
```

Arguments

`filePath`	character specifying the path to the file. If it is in the working directory, use ".".
`fileName`	character name of file to open
`hasHeader`	logical true if the first row of data is the column headers
`separator`	character character that separates data cells. , default is "," which is separator used in a .csv file.
`verbose`	logical specifying whether or not to display progress message
`format`	character indicating the format of the date (which should be in the first column). Default is "%m\/%d/%Y". See `?strptime` for options. The code will initially look for R's standard YYYY-MM-DD, and check this format as a backup.

Value

A data frame 'Sample' with the following columns:

Name	Type	Description
Date	Date	Date
ConcLow	numeric	Lower limit of concentration
ConcHigh	numeric	Upper limit of concentration
Uncen	integer	Uncensored data (1=TRUE, 0=FALSE)
ConcAve	numeric	Average concentration
Julian	integer	Number of days since Jan. 1, 1850
Month	integer	Month of the year [1-12]
Day	integer	Day of the year [1-366]
DecYear	numeric	Decimal year
MonthSeq	integer	Number of months since January 1, 1850
SinDY	numeric	Sine of the DecYear
CosDY	numeric	Cosine of the DecYear

See Also

compressData, populateSampleColumns

Examples

```
filePath <- system.file("extdata", package="EGRET")
fileName <- 'ChoptankRiverNitrate.csv'
Sample <- readUserSample(filePath,fileName, separator=";",verbose=FALSE)
```

---

readWQPSample *Import Sample Data from the Water Quality Portal for WRTDS*

Description

Imports data from the Water Quality Portal, so it could be STORET, USGS, or USDA data. This function gets the data from: https://www.waterqualitydata.us For raw data, use readWQPdata. This function will retrieve the raw data, compress it (summing constituents), then converts it to the Sample dataframe structure. See chapter 7 of the EGRET user guide for more details.

Usage

```
readWQPSample(siteNumber, characteristicName, startDate = "", endDate = "",
  verbose = TRUE, legacy = FALSE)
```

Arguments

siteNumber	character site number. If USGS, it should be in the form :'USGS-XXXXXXXXX...'
characteristicName	character. Either a valid characteristic name, or a 5 digit USGS parameter code.
startDate	character starting date for data retrieval in the form YYYY-MM-DD. Default is empty quotes "" which will retrieve the full period of record.
endDate	character ending date for data retrieval in the form YYYY-MM-DD. Default is empty quotes "" which will retrieve the full period of record.
verbose	logical specifying whether or not to display progress message
legacy	logical specifying whether to use the "legacy" WQP services, or the new ones. Legacy works well for non-USGS sites. For USGS sites it is recommended to use the readNWISSample function. Default is FALSE to get the newest data.

Value

A data frame 'Sample' with the following columns:

Name	Type	Description
Date	Date	Date
ConcLow	numeric	Lower limit of concentration
ConcHigh	numeric	Upper limit of concentration
Uncen	integer	Uncensored data (1=TRUE, 0=FALSE)
ConcAve	numeric	Average concentration
Julian	integer	Number of days since Jan. 1, 1850
Month	integer	Month of the year [1-12]
Day	integer	Day of the year [1-366]
DecYear	numeric	Decimal year
MonthSeq	integer	Number of months since January 1, 1850
SinDY	numeric	Sine of the DecYear
CosDY	numeric	Cosine of the DecYear

See Also

readWQPdata, dataRetrieval::whatWQPsites, readWQPqw, compressData, populateSampleColumns

Examples

```
# These examples require an internet connection to run

Sample_All <- readWQPSample('WIDNR_WQX-10032762','Specific conductance', '', '')
```

`removeDuplicates` *Remove duplicates values from Sample data frame.*

Description

Removes observations from the data frame Sample when the observation has the identical date and value as another observation

Usage

```
removeDuplicates(Sample)
```

Arguments

`Sample`	dataframe with at least DecYear and ConcHigh, default name is Sample

Value

A data frame 'Sample' with the following columns:

Name	Type	Description
Date	Date	Date
ConcLow	numeric	Lower limit of concentration
ConcHigh	numeric	Upper limit of concentration
Uncen	integer	Uncensored data (1=TRUE, 0=FALSE)
ConcAve	numeric	Average concentration
Julian	integer	Number of days since Jan. 1, 1850
Month	integer	Month of the year [1-12]
Day	integer	Day of the year [1-366]
DecYear	numeric	Decimal year
MonthSeq	integer	Number of months since January 1, 1850
SinDY	numeric	Sine of the DecYear
CosDY	numeric	Cosine of the DecYear

Examples

```
DecYear <- c('1985.01', '1985.01', '1985.02', '1985.02', '1985.03')
ConcHigh <- c(1,2,3,3,5)
dataInput <- data.frame(DecYear, ConcHigh, stringsAsFactors=FALSE)
Sample <- removeDuplicates(dataInput)
```

runGroups *Runs a comparison of any group of years in the record.*

Description

runGroups provides comparisons of results, in terms of flow-normalized concentration and flow-normalized flux for any groups of years of years in the water quality record. Comparison could involve the use of the "wall" and/or use of "generalized flow-normalization". These two concepts are described in detail in the vignette: vignette("Enhancements", package = "EGRET").

Usage

```
runGroups(eList, windowSide, group1firstYear, group1lastYear, group2firstYear,
  group2lastYear, surfaceStart = NA, surfaceEnd = NA, flowBreak = FALSE,
  Q1EndDate = NA, QStartDate = NA, QEndDate = NA, wall = FALSE,
  oldSurface = FALSE, fractMin = 0.75, sample1EndDate = NA,
  sampleStartDate = NA, sampleEndDate = NA, paStart = NA, paLong = NA,
  minNumObs = 100, minNumUncen = 50, windowY = 7, windowQ = 2,
  windowS = 0.5, edgeAdjust = TRUE, verbose = TRUE, saveOutput = FALSE,
  fileName = "temp.txt")
```

Arguments

eList	named list with at least the Daily, Sample, and INFO dataframes
windowSide	integer. The width of the flow normalization window on each side of the year being estimated. A common value is 11, but no default is specified. If stationary flow normalization is to be used, then windowSide = 0 (this means that flow-normalization period for all years is the same).
group1firstYear	decimal year. Starting year of first group.
group1lastYear	decimal year. Ending year of first group.
group2firstYear	decimal year. Starting year of second group.
group2lastYear	decimal year. Ending year of second group.
surfaceStart	The Date (or character in YYYY-MM-DD) that is the start of the WRTDS model to be estimated and the first of the daily outputs to be generated. Default is NA, which means that the surfaceStart is based on the date of the first sample.
surfaceEnd	The Date (or character in YYYY-MM-DD) that is the end of the WRTDS model to be estimated and the last of the daily outputs to be generated. Default is NA, which means that the surfaceEnd is based on the date of the last sample.
flowBreak	logical. Is there an abrupt break in the discharge record, default is FALSE.
Q1EndDate	The Date (as character in YYYY-MM-DD) which is the last day, just before the flowBreak.

QStartDate	The first Date (as character in YYYY-MM-DD) used in the flow normalization method. Default is NA, which makes the QStartDate become the first Date in eList$Daily.
QEndDate	The last Date (as character in YYYY-MM-DD) used in the flow normalization method. Default is NA, which makes the QEndDate become the last Date in eList$Daily.
wall	logical. Whether there is an abrupt break in the concentration versus discharge relationship due to some major change in pollution control or water management. Default is FALSE.
oldSurface	logical specifying whether to use the original surface, or create a new one. Default is FALSE.
fractMin	numeric specifying the minimum fraction of the observations required to run the weighted regression, default is 0.75. The minimum number will be the maximum of minNumObs and fractMin multiplied by total number of observations.
sample1EndDate	The Date (as character in YYYY-MM-DD) of the last date just before the wall. Default = NA. A date must be specified if wall = TRUE.
sampleStartDate	The Date (as character in YYYY-MM-DD) of the first sample to be used. Default is NA which sets it to the first Date in eList$Sample.
sampleEndDate	The Date (as character in YYYY-MM-DD) of the last sample to be used. Default is NA which sets it to the last Date in eList$Sample.
paStart	numeric integer specifying the starting month for the period of analysis, 1<=paStart<=12. Default is NA, which will use the paStart in the eList$INFO data frame. See also setPA.
paLong	numeric integer specifying the length of the period of analysis, in months, 1<=paLong<=12. Default is NA, which will use the paLong in the eList$INFO data frame. See also setPA.
minNumObs	numeric specifying the miniumum number of observations required to run the weighted regression, default is 100
minNumUncen	numeric specifying the minimum number of uncensored observations to run the weighted regression, default is 50
windowY	numeric specifying the half-window width in the time dimension, in units of years, default is 7
windowQ	numeric specifying the half-window width in the discharge dimension, units are natural log units, default is 2
windowS	numeric specifying the half-window with in the seasonal dimension, in units of years, default is 0.5
edgeAdjust	logical specifying whether to use the modified method for calculating the windows at the edge of the record. The edgeAdjust method tends to reduce curvature near the start and end of record. Default is TRUE.
verbose	logical specifying whether or not to display progress message
saveOutput	logical. If TRUE, a text file will be saved in the working directory of the printout of what is in the console output. Default is FALSE.
fileName	character. Name to save the output file if saveOutput=TRUE.

Details

When using generalized flow-normalization, it is best to have the Daily data frame extend well beyond the years that are in the Sample data frame. Ideally, the Daily data frame would start windowSide years before the start of the Sample data set, if the data exist to provide for that. Generally that isn't possible for the end of the record because the Sample data may end very close to the present. To the extent that is possible therefore, it is better to include more discharge data after the end of the Sample record. Also note that in the case run in the examples don't do that, because the data set needs to be appropriate for stationary flow normalization as well (and package size considerations make it difficult to include specialized examples).

Value

Dataframe with 7 columns and 2 rows. The first row is about trends in concentration (mg/L), the second column is about trends in flux (million kg/year). The data frame has a number of attributes.

Column Name	Description
Total Change	The difference between the results for group2 - group1 (x22 - x11).
CQTC	CQTC is the "Concentration v. Q Trend Component." It is the component of total change due to the change in
QTC	QTC is the "Q Trend Component." It is the component of total change due to the trend in the QD (Discharge
x10	The estimated value based on the CQR computed for the years in group1, integrated over the QD for the entir
x11	The estimated value based on the CQR for the years in group1, integrated over the QD specified by the user f
x20	The estimated value based on the CQR computed for the years in group2, integrated over the QD for the entir
x22	The estimated value based on the CQR for the years in group2, integrated over the QD specified by the user f

Examples

```
eList <- Choptank_eList


#Option 1:  Use all years for group flow normalization.
groupOut_1 <- runGroups(eList,  windowSide = 0,
                        group1firstYear = 1980, group1lastYear = 1990,
                        group2firstYear = 1995, group2lastYear = 2005)

# Option 2: Use sliding window.
#                In each case it is a 23 year window (23 = 1 + 2 * 11)

groupOut_2 <- runGroups(eList,  windowSide = 11,
                        group1firstYear = 1980, group1lastYear = 1990,
                        group2firstYear = 1995, group2lastYear = 2005)

# Option 3: Flow normalization is based on splitting the flow record at 1990-09-30
#              But in years before the break it uses all flow data from before the break,
#                and years after the break uses all flow data after the break

groupOut_3 <- runGroups(eList,  windowSide = 0,
                        group1firstYear = 1980, group1lastYear = 1990,
                        group2firstYear = 1995, group2lastYear = 2005,
                        flowBreak = TRUE,
```

```
                      Q1EndDate = "1990-09-30")

# Option 4: Flow normalization is based on splitting the flow record at 1990-09-30
#                but before the break uses a 23 year window of years before the break
#                after the break uses a 23 year window of years after the break
groupOut_4 <- runGroups(eList,  windowSide = 11,
                      group1firstYear = 1980, group1lastYear = 1990,
                      group2firstYear = 1995, group2lastYear = 2005,
                      flowBreak = TRUE,
                      Q1EndDate = "1990-09-30")
```

---

runPairs *Runs a comparison of any two years in the record.*

Description

runPairs provides comparisons of results, in terms of flow-normalized concentration and flow-normalized flux for any pair of years in the water quality record. Comparison could involve the use of the "wall" and/or use of "generalized flow normalization". These two concepts are described in detail in the vignette: vignette("Enhancements", package = "EGRET").

Usage

```
runPairs(eList, year1, year2, windowSide, flowBreak = FALSE,
  Q1EndDate = NA, QStartDate = NA, QEndDate = NA, wall = FALSE,
  oldSurface = FALSE, sample1EndDate = NA, sampleStartDate = NA,
  sampleEndDate = NA, paStart = NA, paLong = NA, minNumObs = 100,
  minNumUncen = 50, fractMin = 0.75, windowY = 7, windowQ = 2,
  windowS = 0.5, edgeAdjust = TRUE, saveOutput = FALSE,
  fileName = "temp.txt", verbose = TRUE)
```

Arguments

eList	named list with at least the Daily, Sample, and INFO dataframes
year1	integer the ending year of the first year in the pair
year2	integer the ending year of the second year in the pair
windowSide	integer. The width of the flow normalization window on each side of the year being estimated. A common value is 11, but no default is specified. If stationary flow normalization is to be used, then windowSide = 0 (this means that flow-normalization period for all years is the same).
flowBreak	logical. Is there an abrupt break in the discharge record, default is FALSE.
Q1EndDate	The Date (as character in YYYY-MM-DD) which is the last day, just before the flowBreak.

QStartDate	The first Date (as character in YYYY-MM-DD) used in the flow normalization method. Default is NA, which makes the QStartDate become the first Date in eList$Daily.
QEndDate	The last Date (as character in YYYY-MM-DD) used in the flow normalization method. Default is NA, which makes the QEndDate become the last Date in eList$Daily.
wall	logical. Whether there is an abrupt break in the concentration versus discharge relationship due to some major change in pollution control or water management. Default is FALSE.
oldSurface	logical specifying whether to use the original surface, or create a new one. Default is FALSE.
sample1EndDate	The Date (as character in YYYY-MM-DD) of the last date just before the wall. Default = NA. A date must be specified if wall = TRUE.
sampleStartDate	The Date (as character in YYYY-MM-DD) of the first sample to be used. Default is NA which sets it to the first Date in eList$Sample.
sampleEndDate	The Date (as character in YYYY-MM-DD) of the last sample to be used. Default is NA which sets it to the last Date in eList$Sample.
paStart	numeric integer specifying the starting month for the period of analysis, 1<=paStart<=12. Default is NA, which will use the paStart in the eList$INFO data frame. See also setPA.
paLong	numeric integer specifying the length of the period of analysis, in months, 1<=paLong<=12. Default is NA, which will use the paLong in the eList$INFO data frame. See also setPA.
minNumObs	numeric specifying the miniumum number of observations required to run the weighted regression, default is 100
minNumUncen	numeric specifying the minimum number of uncensored observations to run the weighted regression, default is 50
fractMin	numeric specifying the minimum fraction of the observations required to run the weighted regression, default is 0.75. The minimum number will be the maximum of minNumObs and fractMin multiplied by total number of observations.
windowY	numeric specifying the half-window width in the time dimension, in units of years, default is 7
windowQ	numeric specifying the half-window width in the discharge dimension, units are natural log units, default is 2
windowS	numeric specifying the half-window with in the seasonal dimension, in units of years, default is 0.5
edgeAdjust	logical specifying whether to use the modified method for calculating the windows at the edge of the record. The edgeAdjust method tends to reduce curvature near the start and end of record. Default is TRUE.
saveOutput	logical. If TRUE, a text file will be saved in the working directory of the printout of what is in the console output. Default is FALSE.
fileName	character. Name to save the output file if saveOutput=TRUE.
verbose	logical specifying whether or not to display progress message

Details

When using generalized flow-normalization, it is best to have the Daily data frame extend well beyond the years that are in the Sample data frame. Ideally, the Daily data frame would start windowSide years before the start of the Sample data set, if the data exist to provide for that. Generally that isn't possible for the end of the record because the Sample data may end very close to the present. To the extent that is possible therefore, it is better to include more discharge data after the end of the Sample record. Also note that in the case run in the examples don't do that, because the data set needs to be appropriate for stationary flow normalization as well (and package size considerations make it difficult to include specialized examples).

Value

Data frame with 7 columns and 2 rows. The first row is about trends in concentration (mg/L), the second column is about trends in flux (million kg/year). The data frame has a number of attributes.

Column Name	Description
Total Change	The difference between the results for year2 - year1 (x22 - x11)
CQTC	CQTC is the "Concentration v. Q Trend Component." It is the component of total change due to the change in
QTC	QTC is the "Q Trend Component." It is the component of total change due to the trend in the QD (Discharge
x10	The estimated value based on the CQR computed for year1, integrated over the QD for the entire timespan of
x11	The estimated value based on the CQR for year1, integrated over the QD specified by the user for year1.
x20	The estimated value based on the CQR computed for year2, integrated over the QD for the entire period of re
x22	The estimated value based on the CQR for year2, integrated over the QD specified by the user for year2.

Additionally, there is an attribute on the data frame "Other", containing a list that includes minNumObs=minNumObs, minNumUncen, windowY, windowQ, siteName, windowS, wall, edgeAdjust, QStartDate, QEndDate, PercentChangeConc, and PercentChangeFlux.

PercentChangeConc, and PercentChangeFlux are vectors where: Total Percent Change is the Total Change divided by x11 CQTC Percent is the CQTC divided by x11 QTC Percent is the QTC divided by x11

Another attribute is "byMonth". This is a data frame of 4 columns and 14 rows. The columns represent the concentrations and fluxes for the starting and ending year. The flux values for each month are flow normalized monthly watershed yields expressed as kg/month/km^2. The concentrations are the mean flow normalized concentration, expressed in whatever concentration units the raw data are expressed as (typically mg/L). This data frame is used as the input to the `plotMonthTrend` function.

Examples

```
eList <- Choptank_eList
year1 <- 1985
year2 <- 2010


# Automatic calculations based on windowSide = 11
# four possible ways to do generalized flow normalization:
```

```
#Option 1: Use all years for flow normalization.

pairOut_1 <- runPairs(eList, year1, year2, windowSide = 0)

# Option 2:  Use different windows for flow normalization for year1 versus year2
#            In each case it is a 23 year window (23 = 1 + 2*11)

pairOut_2 <- runPairs(eList, year1, year2, windowSide = 11)

# Option 3: Flow normalization is based on splitting the flow record at 1990-09-30
#          But year1 uses all flow data from before the break,
#          year2 uses all flow data after the break

pairOut_3 <- runPairs(eList, year1, year2,
                      windowSide = 0, flowBreak = TRUE,
                      Q1EndDate = "1990-09-30")

# Option 4: Flow normalization is based on splitting the flow record at 1990-09-30
#           but year1 uses a 23 year window before the break
#           year2 uses a 23 year window after the break

pairOut_4 <- runPairs(eList, year1, year2,
                      windowSide = 11, flowBreak = TRUE,
                      Q1EndDate = "1990-09-30")

monthly_trends <- attr(pairOut_4, "byMonth")
plotMonthTrend(pairOut_4)

eList <- setPA(eList, paLong = 3, paStart = 12)
pairOut_5 <- runPairs(eList, year1, year2,
                      windowSide = 11)
monthly_trends <- attr(pairOut_5, "byMonth")
plotMonthTrend(pairOut_5)
```

`runSeries` *Annual series of flow-normalized concentration and flow-normalized flux*

Description

`runSeries` provides annual series of flow-normalized concentration and flow-normalized flux for the water quality record. Computations could involve the use of the "wall" and/or use of "generalized flow normalization". These two concepts are described in detail in the vignette: `vignette("Enhancements", package = "EGRET")`.

Usage

```
runSeries(eList, windowSide, surfaceStart = NA, surfaceEnd = NA,
```

```
flowBreak = FALSE, Q1EndDate = NA, QStartDate = NA, QEndDate = NA,
wall = FALSE, oldSurface = FALSE, sample1EndDate = NA,
sampleStartDate = NA, sampleEndDate = NA, paStart = NA, paLong = NA,
fractMin = 0.75, minNumObs = 100, minNumUncen = 50, windowY = 7,
windowQ = 2, windowS = 0.5, edgeAdjust = TRUE, verbose = TRUE)
```

Arguments

eList	named list with at least the Daily, Sample, and INFO dataframes
windowSide	integer. The width of the flow normalization window on each side of the year being estimated. A common value is 11, but no default is specified. If stationary flow normalization is to be used, then windowSide = 0 (this means that flow-normalization period for all years is the same).
surfaceStart	The Date (or character in YYYY-MM-DD) that is the start of the WRTDS model to be estimated and the first of the daily outputs to be generated. Default is NA, which means that the surfaceStart is based on the date of the first sample.
surfaceEnd	The Date (or character in YYYY-MM-DD) that is the end of the WRTDS model to be estimated and the last of the daily outputs to be generated. Default is NA, which means that the surfaceEnd is based on the date of the last sample.
flowBreak	logical, is there an abrupt break in the discharge record, default is FALSE.
Q1EndDate	The Date (as character in YYYY-MM-DD format) which is the last day, just before the flowBreak. Required if flowBreak = TRUE.
QStartDate	The first Date (as character in YYYY-MM-DD format) used in the flow normalization. Default is NA, which makes the QStartDate become the first Date in eList$Daily.
QEndDate	The last Date (as character in YYYY-MM-DD format) used in the flow normalization. Default is NA, which makes the QEndDate become the last Date in eList$Daily.
wall	logical. Whether there is an abrupt break in the concentration versus discharge relationship due to some major change in pollution control or water management. Default is FALSE.
oldSurface	logical, if TRUE, use surface previously estimated using modelEstimation. Default is FALSE.
sample1EndDate	The Date (as character in YYYY-MM-DD format) of the last day just before the wall. Default = NA. A date must be specified if wall = TRUE.
sampleStartDate	The Date (as character in YYYY-MM-DD format) of the first sample to be used. Default is NA which sets it to the first Date in eList$Sample.
sampleEndDate	The Date (as character in YYYY-MM-DD format) of the last sample to be used. Default is NA which sets it to the last Date in eList$Sample.
paStart	numeric integer specifying the starting month for the period of analysis, 1<=paStart<=12. Default is NA, which will use the paStart in the eList$INFO data frame. See also setPA.
paLong	numeric integer specifying the length of the period of analysis, in months, 1<=paLong<=12. Default is NA, which will use the paLong in the eList$INFO data frame. See also setPA.

fractMin	numeric specifying the minimum fraction of the observations required to run the weighted regression, default is 0.75. The minimum number will be the maximum of minNumObs and fractMin multiplied by total number of observations.
minNumObs	numeric specifying the miniumum number of observations required to run the weighted regression, default is 100
minNumUncen	numeric specifying the minimum number of uncensored observations to run the weighted regression, default is 50
windowY	numeric specifying the half-window width in the time dimension, in units of years, default is 7
windowQ	numeric specifying the half-window width in the discharge dimension, units are natural log units, default is 2
windowS	numeric specifying the half-window with in the seasonal dimension, in units of years, default is 0.5
edgeAdjust	logical specifying whether to use the modified method for calculating the windows at the edge of the record. The edgeAdjust method tends to reduce curvature near the start and end of record. Default is TRUE.
verbose	logical specifying whether to output status messages.

Details

When using generalized flow-normalization, it is best to have the Daily data frame extend well beyond the years that are in the Sample data frame. Ideally, the Daily data frame would start windowSide years before the start of the Sample data set, if the data exist to provide for that. Generally that isn't possible for the end of the record because the Sample data may end very close to the present. To the extent that is possible therefore, it is better to include more discharge data after the end of the Sample record. Also note that in the case run in the examples don't do that, because the data set needs to be appropriate for stationary flow normalization as well (and package size considerations make it difficult to include specialized examples).

Value

eList named list with INFO, Daily, and Sample dataframes, along with the surfaces matrix.

Examples

```
eList <- Choptank_eList


# Automatic calculations based on windowSide = 11
# four possible ways to do generalized flow normalization

#Option 1:  Use all years for flow normalization.
seriesOut_1 <- runSeries(eList,  windowSide = 0)
plotConcHist(seriesOut_1)
plotFluxHist(seriesOut_1)

# Option 2: Use sliding window throughout the whole flow normalization process.
#                In each case it is a 15 year window (23 = 1 + 2*11)
```

```
seriesOut_2 <- runSeries(eList, windowSide = 11)

plotConcHist(seriesOut_2)
plotFluxHist(seriesOut_2)

# Option 3: Flow normalization is based on splitting the flow record at 1990-09-30
#              But in years before the break it uses all flow data from before the break,
#                and years after the break uses all flow data after the break
seriesOut_3 <- runSeries(eList,
                         windowSide = 0,
                         flowBreak = TRUE,
                         Q1EndDate = "1990-09-30")

plotConcHist(seriesOut_3)
plotFluxHist(seriesOut_3)

# Option 4: Flow normalization is based on splitting the flow record at 1990-09-30
#                but before the break uses a 23 year window of years before the break
#                after the break uses a 23 year window of years after the break
seriesOut_4 <- runSeries(eList,
                        windowSide = 11, flowBreak = TRUE,
                        Q1EndDate = "1990-09-30")

plotConcHist(seriesOut_4)
plotFluxHist(seriesOut_4)
```

---

runSurvReg *Run the weighted survival regression for a set of estimation points (defined by DecYear and Log(Q))*

Description

This function runs the survival regression which is the concentration estimation method of WRTDS. It uses sample data from the data frame Sample. It does the estimation for a set of data points defined by two vectors: estPtYear and estPtLQ. It returns an array of results for the estimation points. The array returned contains yHat, SE and ConcHat (in that order). yHat is the expected value of log(concentration), SE is the standard error of log(concentration) and ConcHat is the expected value of concentration.

Usage

```
runSurvReg(estPtYear, estPtLQ, DecLow, DecHigh, Sample, windowY = 7,
  windowQ = 2, windowS = 0.5, minNumObs = 100, minNumUncen = 50,
  verbose = TRUE, interactive = NULL, edgeAdjust = TRUE,
  run.parallel = FALSE)

run_WRTDS(estY, estLQ, localSample, DecLow, DecHigh, minNumObs, minNumUncen,
  windowY, windowQ, windowS, edgeAdjust)
```

Arguments

estPtYear	numeric vector of Decimal Year values at the estimation points
estPtLQ	numeric vector of ln(Q) values at the estimation points, must be the same length as estPtYear
DecLow	number specifying minimum decimal year (left edge of the estimated surfaces).
DecHigh	number specifying maximum decimal year (right edge of the estimated surfaces).
Sample	dataframe created for EGRET analysis
windowY	numeric specifying the half-window width in the time dimension, in units of years, default is 7
windowQ	numeric specifying the half-window width in the discharge dimension, units are natural log units, default is 2
windowS	numeric specifying the half-window with in the seasonal dimension, in units of years, default is 0.5
minNumObs	numeric specifying the miniumum number of observations required to run the weighted regression, default is 100
minNumUncen	numeric specifying the minimum number of uncensored observations to run the weighted regression, default is 50
verbose	logical specifying whether or not to display progress message
interactive	logical deprecated. Use 'verbose' instead
edgeAdjust	logical specifying whether to use the modified method for calculating the windows at the edge of the record. The modified method tends to reduce curvature near the start and end of record. Default is TRUE.
run.parallel	logical to run bootstrapping in parallel or not
estY	numeric decimal year values at the estimation point
estLQ	numeric ln(Q) values at the estimation point
localSample	"Sample" data frame from the eList.

Value

resultSurvReg numeric array containing the yHat, SE, and ConcHat values array dimensions are (numEstPts,3)

Examples

```
eList <- Choptank_eList
estPtYear<-c(2001.0,2005.0,2009.0)
estPtLQ<-c(1,1,1)
Sample <- getSample(eList)
DecLow <- Sample$DecYear[1]
DecHigh <- Sample$DecYear[nrow(Sample)]
resultSurvReg <- runSurvReg(estPtYear,estPtLQ,
                            DecLow,DecHigh,Sample,
                            run.parallel = FALSE)
```

`saveResults` *A utility program for saving the contents of the workspace*

Description

This function saves the workspace. Future versions of EGRET will not include this function, use saveRDS to save individual eList objects. It assigns the file a name using the abbreviations for station and constituent.

Usage

```
saveResults(savePath, eList)
```

Arguments

`savePath`	character specifying the full pathname of the folder where the file is to be saved ending with the final slash
`eList`	named list with at least the INFO dataframe

`selectDays` *Creates a subset Daily data frame that only contains daily estimates for the specified period of analysis*

Description

This function uses the user-defined 'period of analysis', and subsets the Daily data frame, it doesn't have any effect on the Sample data frame. If you want to examine your data set as a time series of water years, then the period of analysis is October through September. If you want to examine the data set as calendar years then the period of analysis is January through December. You might want to examine the winter season, which you could define as December through February, then those 3 months become the period of analysis. The only constraints on the definition of a period of analysis are these: it must be defined in terms of whole months; it must be a set of contiguous months (like March-April-May), and have a length that is no less than 1 month and no more than 12 months. Define the PA by using two arguments: paLong and paStart. paLong is the length of the period of analysis, and paStart is the starting month.

Usage

```
selectDays(df, paLong, paStart)
```

Arguments

df	dataframe which must contain a column named Month (for month of the calendar year, typically this is a Daily data frame.
paLong	a numeric value for the length of the period of analysis, must be an integer from 1 to 12
paStart	a numeric value for the starting month of the period of analysis, must be an integer from 1 to 12

Value

localDaily a data frame containing the daily data but only for the period of analysis (not all months)

Examples

```
eList <- Choptank_eList
Daily <- getDaily(eList)
DailySubset <- selectDays(Daily, 4, 11)
```

---

setPA *Sets up the period of analysis (the portion of the year being evaluated).*

Description

Period of analysis is defined by the starting month (paStart) and length in months (paLong). paStart and paLong are constrained to be integers from 1 to 12. For example, a water year would be paStart = 10 and paLong = 12. For example, the winter season, defined by Dec,Jan,Feb would be paStart = 12 and paLong = 3.

Usage

```
setPA(eList, paStart = 10, paLong = 12, window = 20)
```

Arguments

eList	named list with at least the INFO dataframe
paStart	A numeric value for the starting month of the Period of Analysis, default is 10
paLong	A numeric value for the length of the Period of Analysis in months, default is 12
window	A numeric value for the half-width of a smoothing window for annual streamflow values, default is 20

Value

eList named list with at least the INFO dataframe. Any of these values can be NA, but not all EGRET functions will work with missing parts of the named list eList.

Examples

```
eList <- Choptank_eList
eList <- setPA(eList, paStart = 12, paLong = 3)
```

`setSeasonLabel` *Create a character variable that describes the period of analysis, when period of analysis has already been set in AnnualResults*

Description

The period of analysis can be of any length from 1 month to 12 months. The period of analysis can have any starting month from 1 (January) through 12 (December). This function produces a character character that describes this period of analysis. For example "water year", "calendar year", "year starting with April", or "Season consisting of April, May, June". There is an alternative version of this function for the case where AnnualResults does not exist. This might arise in a call from plotConcTime or plotLogConcTime. That function is called setSeasonLabelByUser.

Usage

```
setSeasonLabel(localAnnualResults, monthLab = 1)
```

Arguments

`localAnnualResults`	data frame that contains the annual results, default is AnnualResults
`monthLab`	object of monthLabel class, or numeric represented the short code, or character representing the descriptive name.

Value

periodName character which describes the period of analysis

Examples

```
eList <- Choptank_eList
Daily <- getDaily(eList)
AnnualResults <- setupYears(Daily)
setSeasonLabel(AnnualResults)

AnnualResultsWinter <- setupYears(Daily,
                                  paLong = 3,
                                  paStart = 12)
setSeasonLabel(AnnualResultsWinter)
```

setSeasonLabelByUser *Creates a character variable that describes the period of analysis, when the period of analysis is being set by the user and not from AnnualResults*

Description

The period of analysis can be of any length from 1 month to 12 months. The period of analysis can have any starting month from 1 (January) through 12 (December). This function produces a character that describes this period of analysis. For example "water year", "calendar year", "year starting with April", or "Season consisting of April, May, June". There is an alternative version of this function for the case where AnnualResults exists. And we want to use the period of analysis defined there. That function is called setSeasonLabel.

Usage

```
setSeasonLabelByUser(paStartInput = 10, paLongInput = 12, monthLab = 1)
```

Arguments

paStartInput	numeric the month which is the start of the period of analysis, default is 10 which would be the case if the period of analysis is the water year
paLongInput	numeric the length of the the period of analysis, in months, default is 12 which would be the case if the period of analysis is the water year
monthLab	object of monthLabel class, or numeric represented the short code, or character representing the descriptive name.

Value

periodName character which describes the period of analysis

Examples

```
setSeasonLabelByUser(paStartInput=1,paLongInput=12)
setSeasonLabelByUser(paStartInput=4,paLongInput=3)
```

setUpEstimation *setUpEstimation*

Description

Set up the INFO data frame for a modelEstimation

Usage

```
setUpEstimation(eList, windowY = 7, windowQ = 2, windowS = 0.5,
  minNumObs = 100, minNumUncen = 50, edgeAdjust = TRUE, verbose = TRUE,
  interactive = NULL)
```

Arguments

eList	named list with at least the Daily, Sample, and INFO dataframes
windowY	numeric specifying the half-window width in the time dimension, in units of years, default is 7
windowQ	numeric specifying the half-window width in the discharge dimension, units are natural log units, default is 2
windowS	numeric specifying the half-window with in the seasonal dimension, in units of years, default is 0.5
minNumObs	numeric specifying the miniumum number of observations required to run the weighted regression, default is 100
minNumUncen	numeric specifying the minimum number of uncensored observations to run the weighted regression, default is 50
edgeAdjust	logical specifying whether to use the modified method for calculating the windows at the edge of the record. The modified method tends to reduce curvature near the start and end of record. Default is TRUE.
verbose	logical specifying whether or not to display progress message
interactive	logical deprecated. Use 'verbose' instead

Value

eList named list with Daily, Sample, and INFO dataframes.

Examples

```
eList <- Choptank_eList
eList <- setUpEstimation(eList)
```

---

setupYears *Creates the AnnualResults data frame from the Daily data frame*

Description

This function aggregates the results stored on a daily basis in the Daily data frame and stores the average values of these in the new data frame called AnnualResults. Note that the flux values are rates (kg/day) and not a mass (kg). The "annual values" can be a full 12 months, or they can be shorter. See manual to understand paLong and paStart arguments. The simplest case, a Water Year (October through September), would have paLong=12, and paStart=10. A calendar year would be paLong=12 and paStart=1. A winter season of Dec, Jan, Feb would be paLong=3 and paStart=12

Usage

```
setupYears(localDaily, paLong = 12, paStart = 10)
```

Arguments

localDaily	data frame containing the daily values, default is Daily
paLong	numeric integer specifying the length of the period of analysis, in months, 1<=paLong<=12, default is 12
paStart	numeric integer specifying the starting month for the period of analysis, 1<=paStart<=12, default is 10

Value

A data frame 'AnnualResults' of numeric values with the following columns

Name	Description
DecYear	Middle of the period in decimal years
Q	Mean discharge, in m^3/s
Conc	Estimated mean concentration, in mg/L
Flux	Estimated mean flux, in kg/day
FNConc	Flow-normalized concentration, in mg/L
FNFlux	Flow-normalized flux, in kg/day
GenConc	Generalized mean concentration, in mg/L. This column is only returned if the WRTDSKalman function was run
GenFlux	Generalized mean flux, in kg/day. This column is only returned if the WRTDSKalman function was run, which
PeriodLong	Length of period of analysis (paLong), in months
PeriodStart	Starting month of period of analysis (paStart), in months (1 = January)

Examples

```
eList <- Choptank_eList
Daily <- getDaily(eList)
AnnualResults <- setupYears(Daily, 4, 10)
```

---

startEnd *startEnd*

Description

Returns two date variables representing the starting date and ending date for a combination of paStart, paLong, and year

Usage

```
startEnd(paStart, paLong, year)
```

Arguments

paStart	numeric integer specifying the starting month for the period of analysis, 1<=paStart<=12, default is 10
paLong	numeric integer specifying the length of the period of analysis, in months, 1<=paLong<=12, default is 12
year	integer year, which is the calendar year in which the period ends

Value

Date list

Examples

```
paStart <- 10
paLong <- 12
year <- 1999
startEnd(paStart, paLong, year)
```

---

stitch *stitch surfaces*

Description

This function creates a continuous surfaces object that starts just before surfaceStart and ends just after surfaceEnd. It is made up from two surfaces objects created when there is a wall specified for the analysis. The first surfaces object is based on data prior to the wall and the second surfaces object is based on data after the wall. The wall is located just after sample1EndDate. The Daily data frame is used only to set the minimum and maximum discharges used to construct the indices for discharges in the surfaces.

Usage

```
stitch(eList, sample1StartDate, sample1EndDate, sample2StartDate,
  sample2EndDate, surfaceStart = NA, surfaceEnd = NA, minNumObs = 100,
  minNumUncen = 50, fractMin = 0.75, windowY = 7, windowQ = 2,
  windowS = 0.5, edgeAdjust = TRUE, verbose = FALSE,
  run.parallel = FALSE)
```

Arguments

eList	named list with at least the Daily, Sample, and INFO dataframes
sample1StartDate	The Date (or character in YYYY-MM-DD) of the first sample to be used in estimating the first segment of the surfaces object.
sample1EndDate	The Date (or character in YYYY-MM-DD) of the last sample to be used in the first segment of the surfaces object.

sample2StartDate
: The Date (or character in YYYY-MM-DD) of the first sample to be used in the second segment of the surfaces object.

sample2EndDate
: The Date (or character in YYYY-MM-DD) of the last sample to be used in the second segment of the surfaces object.

surfaceStart
: The Date (or character in YYYY-MM-DD) that is the start of the WRTDS model to be estimated and the first of the daily outputs to be generated. Default is NA, which means that the surfaceStart is based on the date of the first sample.

surfaceEnd
: The Date (or character in YYYY-MM-DD) that is the end of the WRTDS model to be estimated and the last of the daily outputs to be generated. Default is NA, which means that the surfaceEnd is based on the date of the last sample.

minNumObs
: numeric specifying the miniumum number of observations required to run the weighted regression, default is 100

minNumUncen
: numeric specifying the minimum number of uncensored observations to run the weighted regression, default is 50

fractMin
: numeric specifying the minimum fraction of the observations required to run the weighted regression, default is 0.75. The minimum number will be the maximum of minNumObs and fractMin multiplied by total number of observations.

windowY
: numeric specifying the half-window width in the time dimension, in units of years, default is 7

windowQ
: numeric specifying the half-window width in the discharge dimension, units are natural log units, default is 2

windowS
: numeric specifying the half-window with in the seasonal dimension, in units of years, default is 0.5

edgeAdjust
: logical specifying whether to use the modified method for calculating the windows at the edge of the record. The edgeAdjust method tends to reduce curvature near the start and end of record. Default is TRUE.

verbose
: logical specifying whether or not to display progress message

run.parallel
: logical to run bootstrapping in parallel or not

Examples

```
eList <- Choptank_eList

surfaceStart <- "1986-10-01"
surfaceEnd <- "2010-09-30"

# Surface skips a few years:
sample1StartDate <- "1986-10-01"
sample1EndDate <- "1992-09-30"
sample2StartDate <- "1996-10-01"
sample2EndDate <- "2011-09-30"


surface_skip <- stitch(eList,
                       sample1StartDate, sample1EndDate,
```

```
                        sample2StartDate, sample2EndDate,
                        surfaceStart, surfaceEnd)

# Surface overlaps a few years:
sample1StartDate <- "1986-10-01"
sample1EndDate <- "1996-09-30"
sample2StartDate <- "1992-10-01"
sample2EndDate <- "2011-09-30"

surface_overlap <- stitch(eList,
                        sample1StartDate, sample1EndDate,
                        sample2StartDate, sample2EndDate)
```

`surfaceIndex` *Compute the 6 parameters needed to lay out the grid for the surfaces computed in estSurfaces*

Description

The code here is a repetition of the first part of the code for estSurfaces

Usage

```
surfaceIndex(Daily)
```

Arguments

`Daily`	data frame containing the daily values, default is Daily

Value

surfaceIndexParameters a numeric vector of length 6, defining the grid for the surfaces

Examples

```
eList <- Choptank_eList
Daily <- getDaily(eList)
surfaceIndex(Daily)
```

surfaceStartEnd *Surface date limits*

Description

Sets the Date limits to the surfaces being estimated from the Sample data set. The start is less than a year prior to the first date (typically the date of the first sample) and the end is less than a year after the last date (typically the date of the last sample). The start is constrained to be on the first day of the period of analysis and the end is constrained to be on the last day of the the period of analysis

Usage

```
surfaceStartEnd(paStart, paLong, Date1, Date2)
```

Arguments

paStart	numeric integer specifying the starting month for the period of analysis, 1<=paStart<=12, default is 10
paLong	numeric integer specifying the length of the period of analysis, in months, 1<=paLong<=12, default is 12
Date1	Date set to Date of earliest data in Sample.
Date2	Date set to Date of latest data in Sample.

Examples

```
eList <- Choptank_eList
Date1 <- eList$Sample$Date[1]
Date2 <- range(eList$Sample$Date)[2]
surfaceStartEnd(10, 12, Date1, Date2)
```

tableChange *Create a table of the changes in flow-normalized values between various points in time in the record*

Description

These tables describe trends in flow-normalized concentration and in flow-normalized flux. They are described as changes in real units or in percent and as slopes in real units per year or in percent per year. They are computed over pairs of time points. These time points can be user-defined or they can be set by the program to be the final year of the record and a set of years that are multiples of 5 years prior to that. tableChangeSingle is a version of the same code that will produce output only for flow-normalized concentration or flow-normalized flux, but not both

Usage

```
tableChange(eList, fluxUnit = 9, yearPoints = NA)

tableChangeSingle(eList, fluxUnit = 9, yearPoints = NA, flux = FALSE)
```

Arguments

eList	named list with at least the Daily and INFO dataframes
fluxUnit	object of fluxUnit class. printFluxUnitCheatSheet, or numeric represented the short code, or character representing the descriptive name.
yearPoints	numeric vector listing the years for which the change or slope computations are made, they need to be in chronological order. For example yearPoints=c(1975,1985,1995,2005), default is NA (which allows the program to set yearPoints automatically)
flux	logical if TRUE results are returned in flux, if FALSE concentration. Default is set to FALSE.

Value

dataframe with Year1, Year2, change[mg/L], slope[mg/L], change[percent], slope[percent] columns. The data in each row is the change or slope calculated from Year1 to Year2

Examples

```
eList <- Choptank_eList
# Water Year:

tableChange(eList, fluxUnit = 8, yearPoints = c(1980, 1995, 2011))
tableChange(eList, fluxUnit = 5)
# Winter:
eList <- setPA(eList, paStart = 12, paLong = 3)
tableChange(eList, fluxUnit = 8, yearPoints = c(1980, 1995, 2011))

# Water Year:
eList <- setPA(eList, paStart = 10, paLong = 12)
#This returns concentration ASCII table in the console:
tableChangeSingle(eList, fluxUnit = 8, yearPoints = c(1980, 1995, 2011), flux = FALSE)
#Returns a data frame:
change <- tableChangeSingle(eList, fluxUnit = 8, yearPoints=c(1980, 1995, 2011), flux = FALSE)
#This returns flux values as a data frame:
df <- tableChangeSingle(eList, fluxUnit = 8, yearPoints=c(1980, 1995, 2011), flux = TRUE)
# Winter Concentration only:
eList <- setPA(eList, paStart = 12, paLong = 3)
df.winter <- tableChangeSingle(eList, fluxUnit = 8, yearPoints=c(1980, 1995, 2011), flux = FALSE)
```

tableFlowChange *Prints table of change metrics for a given streamflow statistic*

Description

Part of the flowHistory system. Provides a measure of change (in real units and as percent per year) based on the smoothed values for various streamflow statistics. Smoothing algorithm is the same as is used in plotFlowSingle.

Usage

```
tableFlowChange(eList, istat, qUnit = 1, runoff = FALSE, yearPoints = NA)
```

Arguments

eList	named list with at least Daily and INFO dataframes
istat	A numeric value for the flow statistic to be graphed (possible values are 1 through 8)
qUnit	object of qUnit class printqUnitCheatSheet, or numeric represented the short code, or character representing the descriptive name.
runoff	logical variable, if TRUE the streamflow data are converted to runoff values in mm/day
yearPoints	A vector of numeric values, specifying the years at which change metrics are to be calculated, default is NA (which allows the function to set these automatically), yearPoints must be in ascending order

Details

The index of the flow statistics is istat. These statistics are: (1) 1-day minimum, (2) 7-day minimum, (3) 30-day minimum, (4) median (5) mean, (6) 30-day maximum, (7) 7-day maximum, and (8) 1-day maximum.

Can also run the statistics on any Period of Analysis (individual months or sequence of months) using setPA.

A dataframe is returned, as well as a printout in the R console.

Examples

```
eList <- Choptank_eList
tableFlowChange(eList, istat = 5, yearPoints = c(1981, 1995, 2010))
eList <- setPA(eList, paStart = 4, paLong = 12)
tableFlowChange(eList, istat = 2, qUnit = 2, yearPoints = c(1981, 1995, 2010))
eList <- setPA(eList, paStart = 9, paLong = 1)
df <- tableFlowChange(eList, istat = 5, qUnit = 2, yearPoints = c(1981, 1995, 2010))
df
```

`tableResults` *Table of annual results for discharge, concentration and flux*

Description

Produce an ASCII table showing: year, mean discharge, mean concentration, flow-normalized concentration, mean flux, and flow-normalized flux. Note that the flux and flow-normalized flux are rates and not a mass. As such a value for some period shorter than a full year could be larger than the value for a full year.

Usage

```
tableResults(eList, qUnit = 2, fluxUnit = 9, localDaily = NA)
```

Arguments

`eList`	named list with at least Daily and INFO dataframes
`qUnit`	object of qUnit class. `printqUnitCheatSheet`, or numeric represented the short code, or character representing the descriptive name.
`fluxUnit`	object of fluxUnit class. `printFluxUnitCheatSheet`, or numeric represented the short code, or character representing the descriptive name.
`localDaily`	data frame to override eList$Daily

Details

Can also procude a table for any Period of Analysis (individual months or sequence of months) using `setPA`.

Value

dataframe with columns year, mean discharge, mean concentration, mean flow-normalized concentration, mean Kalman concentration (if it has been calculated), mean flux, mean flow-normalized flux, and mean Kalman flux (if it has been calculated).

Examples

```
eList <- Choptank_eList
# Water Year:

tableResults(eList, fluxUnit = 8)
df <- tableResults(eList, fluxUnit = 1)
df
# Spring:
eList <- setPA(eList, paStart = 3, paLong = 3)
tableResults(eList, fluxUnit = 1, qUnit = "cfs")
```

triCube *Tricube weight function*

Description

Computes the tricube weight function on a vector of distance values (d), based on a half-window width of h, and returns a vector of weights that range from zero to 1.

Usage

```
triCube(d, h)
```

Arguments

d	numeric vector of distances from the point of estimation to the given sample value
h	numeric value, the half-window width, measured in the same units as d

Details

See Cleveland, W. S. (1979). Robust locally weighted regression and smoothing scatterplots, JASA, 74, 829-836

Value

w numeric vector of weights, all 0<=w<=1

Examples

```
h<-10
d<-c(-11,-10,-5,-1,-0.01,0,5,9.9,10,20)
triCube(d,h)
```

WRTDSKalman *WRTDS-Kalman*

Description

This function uses an autoregressive model to produce more accurate estimates of concentration and flux

Usage

```
WRTDSKalman(eList, rho = 0.9, niter = 200, seed = NA, verbose = TRUE)
```

Arguments

eList	named list with the INFO, Daily, and Sample dataframes and surfaces matrix
rho	numeric the lag one autocorrelation. Default is 0.9.
niter	number of iterations. Default is 200.
seed	integer value. Defaults to NA, which will not change the current seed. Setting the seed to any given value can be used to create repeatable output.
verbose	logical specifying whether or not to display progress message

Details

This function takes an existing eList Including the estimated model (the surfaces object in the eList) And produces the daily WRTDSKalman estimates of concentration and flux These generated estimates are called GenConc and GenFlux

Examples

```
eList <- Choptank_eList
eList <- WRTDSKalman(eList, niter = 10)
summary(eList$Daily)

#All flux values in AnnualResults are expressed as a rate in kg/day
AnnualResults <- setupYears(eList$Daily)
head(AnnualResults)
```

---

yPretty *Sets up tick marks for an axis for a graph with an arithmetic scale which starts at zero*

Description

Axis tick marks that run from zero to some specified maximum, creates about 4 to 8 ticks marks.

Usage

```
yPretty(yMax)
```

Arguments

yMax	A numeric value for the maximum value to be plotted, it must be >0

Value

yTicks A numeric vector representing the values for each of the tick marks

Examples

```
yTicks <- yPretty(7.8)
yTicks <- yPretty(125)
```

Index

∗ **USGS**
as.egret, 4
dataOverview, 19
getDaily, 38
getInfo, 39
getSample, 40
getSurfaces, 41
INFOdataframe, 41
is.egret, 43
mergeReport, 48
print.egret, 101
processQWData, 106
readNWISDaily, 109
readNWISSample, 110
readUserDaily, 111
readWQPSample, 114
∗ **WRTDS**
as.egret, 4
checkStartEndDate, 14
compressData, 16
dataOverview, 19
dateFormatCheck, 19
fixSampleFrame, 28
getDaily, 38
getInfo, 39
getSample, 40
getSurfaces, 41
is.egret, 43
mergeReport, 48
populateDaily, 95
print.egret, 101
readNWISDaily, 109
readNWISSample, 110
readUserDaily, 111
readWQPSample, 114
∗ **bias**
fluxBiasStat, 33
∗ **datasets**
Constants, 17
∗ **data**
as.egret, 4
Choptank_eList, 15
dataOverview, 19
getDaily, 38
getInfo, 39
getSample, 40
getSurfaces, 41
INFOdataframe, 41
is.egret, 43
mergeReport, 48
print.egret, 101
processQWData, 106
readDataFromFile, 108
readNWISDaily, 109
readNWISSample, 110
readUserDaily, 111
readUserSample, 113
readWQPSample, 114
∗ **file**
readDataFromFile, 108
readUserSample, 113
∗ **flow**
checkStartEndDate, 14
compressData, 16
dateFormatCheck, 19
fixSampleFrame, 28
populateDaily, 95
∗ **graphics**
boxConcMonth, 6
boxConcThree, 8
boxQTwice, 9
boxResidMonth, 11
fluxBiasMulti, 32
generalAxis, 34
genericEGRETDotPlot, 35
logPretty1, 44
logPretty3, 45
multiPlotDataOverview, 51

plot15, 53
plot1of15, 54
plotConcHist, 54
plotConcPred, 56
plotConcQ, 58
plotConcQSmooth, 60
plotConcTime, 62
plotConcTimeDaily, 65
plotConcTimeSmooth, 66
plotContours, 69
plotDiffContours, 71
plotFlowSingle, 74
plotFluxHist, 76
plotFluxPred, 77
plotFluxQ, 79
plotFluxTimeDaily, 80
plotFour, 82
plotFourStats, 83
plotQTimeDaily, 85
plotResidPred, 87
plotResidQ, 88
plotResidTime, 90
plotSDLogQ, 91
printFluxUnitCheatSheet, 102
printqUnitCheatSheet, 104
setSeasonLabel, 130
setSeasonLabelByUser, 131
yPretty, 142

∗ **import**
as.egret, 4
dataOverview, 19
getDaily, 38
getInfo, 39
getSample, 40
getSurfaces, 41
INFOdataframe, 41
is.egret, 43
mergeReport, 48
print.egret, 101
processQWData, 106
readDataFromFile, 108
readNWISDaily, 109
readNWISSample, 110
readUserDaily, 111
readUserSample, 113
readWQPSample, 114

∗ **quality**
Choptank_eList, 15
setSeasonLabel, 130
setSeasonLabelByUser, 131

∗ **service**
INFOdataframe, 41
processQWData, 106

∗ **statistics**
blankTime, 5
boxConcMonth, 6
boxConcThree, 8
boxQTwice, 9
boxResidMonth, 11
calculateMonthlyResults, 12
estCrossVal, 22
estDailyFromSurfaces, 23
estSurfaces, 24
flowDuration, 31
fluxBiasMulti, 32
fluxBiasStat, 33
generalAxis, 34
genericEGRETDotPlot, 35
logPretty1, 44
logPretty3, 45
makeAnnualSeries, 46
makeAugmentedSample, 47
modelEstimation, 50
multiPlotDataOverview, 51
plot15, 53
plotConcHist, 54
plotConcPred, 56
plotConcQ, 58
plotConcQSmooth, 60
plotConcTime, 62
plotConcTimeDaily, 65
plotConcTimeSmooth, 66
plotContours, 69
plotDiffContours, 71
plotFlowSingle, 74
plotFluxHist, 76
plotFluxPred, 77
plotFluxQ, 79
plotFluxTimeDaily, 80
plotFour, 82
plotFourStats, 83
plotResidPred, 87
plotResidQ, 88
plotResidTime, 90
plotSDLogQ, 91
printFluxUnitCheatSheet, 102

printqUnitCheatSheet, 104
printSeries, 105
runSurvReg, 126
saveResults, 128
selectDays, 128
setPA, 129
setUpEstimation, 131
setupYears, 132
surfaceIndex, 136
tableChange, 137
tableFlowChange, 139
tableResults, 140
triCube, 141
yPretty, 142
∗ **streamflow**
flowDuration, 31
makeAnnualSeries, 46
plot15, 53
plot1of15, 54
plotFlowSingle, 74
plotFour, 82
plotFourStats, 83
plotQTimeDaily, 85
plotSDLogQ, 91
printSeries, 105
selectDays, 128
setPA, 129
tableFlowChange, 139
∗ **trends**
makeAnnualSeries, 46
∗ **water-quality**
blankTime, 5
boxConcMonth, 6
boxConcThree, 8
boxQTwice, 9
boxResidMonth, 11
calculateMonthlyResults, 12
estCrossVal, 22
estDailyFromSurfaces, 23
estSurfaces, 24
fluxBiasMulti, 32
fluxBiasStat, 33
generalAxis, 34
genericEGRETDotPlot, 35
makeAugmentedSample, 47
modelEstimation, 50
multiPlotDataOverview, 51
plotConcHist, 54
plotConcPred, 56
plotConcQ, 58
plotConcQSmooth, 60
plotConcTime, 62
plotConcTimeDaily, 65
plotConcTimeSmooth, 66
plotContours, 69
plotDiffContours, 71
plotFluxHist, 76
plotFluxPred, 77
plotFluxQ, 79
plotFluxTimeDaily, 80
plotResidPred, 87
plotResidQ, 88
plotResidTime, 90
printFluxUnitCheatSheet, 102
printqUnitCheatSheet, 104
runSurvReg, 126
saveResults, 128
setUpEstimation, 131
setupYears, 132
surfaceIndex, 136
tableChange, 137
tableResults, 140
∗ **water**
Choptank_eList, 15
setSeasonLabel, 130
setSeasonLabelByUser, 131
∗ **web**
INFOdataframe, 41
processQWData, 106
∗ **weighting**
triCube, 141

Arkansas_eList *(*Choptank_eList*)*, 15
as.egret, 4

bin_Qs *(*estDailyFromSurfaces*)*, 23
blankTime, 5
boxConcMonth, 6, *52*
boxConcThree, 8
boxplot, *7*, *9*, *10*, *12*
boxQTwice, 9, *52*
boxResidMonth, 11

calculateMonthlyResults, 12
censoredSegments, 13
checkStartEndDate, 14
checkSurfaceSpan, 14

Choptank_eList, 15
cleanUp, 15
compressData, 16, *111*, *114, 115*
concConst *(*Constants*)*, 17
concUnit-class, 17
Constants, 17
cumQdate, 18

dataOverview, 19
dateFormatCheck, 19
decimalDate, 20
decimalHighLow, 20

errorStats, 21
estCrossVal, 22
estDailyFromSurfaces, 23
estSurfaces, 24

fill_missing_daily, 26
fillUpElists, 27
fixSampleFrame, 28
flexFN, 29
flexPlotAddOn, 30
flowDuration, 31
fluxBiasMulti, 32
fluxBiasStat, 33
fluxConst *(*Constants*)*, 17
fluxUnit-class, 34

generalAxis, 34
genericEGRETDotPlot, 35, *56, 57*, *59*, *62*, *64*, *66*, *69*, *75*, *78*, *80, 81*, *87–89*, *91, 92*
genmissing, 38
getConcFluxFromSurface *(*estDailyFromSurfaces*)*, 23
getDaily, 38
getInfo, 39
getSample, 40
getSurfaceEstimates *(*estDailyFromSurfaces*)*, 23
getSurfaces, 41

INFOdataframe, 41
is.egret, 43

jitterSam, 44

logPretty1, 44
logPretty3, 45

makeAnnualSeries, 46, *54*, *75*, *105*
makeAugmentedSample, 47
makeDateInfo, 48
mergeReport, *19*, 48
modelEstimation, *49*, 50
monthInfo *(*Constants*)*, 17
monthLabel-class, 51
multiPlotDataOverview, 51, *102*

nCensoredVals *(*print.egret*)*, 101
nDischarge *(*print.egret*)*, 101
nObservations *(*print.egret*)*, 101

plot.egret *(*print.egret*)*, 101
plot15, 53, *54*
plot1of15, *53*, 54
plotConcHist, 54
plotConcPred, 56
plotConcQ, *52*, 58
plotConcQSmooth, 60
plotConcTime, *52*, 62
plotConcTimeDaily, 65
plotConcTimeSmooth, 66
plotContours, 69
plotDiffContours, 71
plotFlowSingle, *53, 54*, 74, *82*, *84*, *139*
plotFluxHist, 76
plotFluxPred, 77
plotFluxQ, 79
plotFluxTimeDaily, 80
plotFour, 82
plotFourStats, 83
plotMonthTrend, 84
plotQTimeDaily, 85
plotResidPred, 87
plotResidQ, 88
plotResidTime, 90
plotSDLogQ, 91
plotTimeSlice, 92
plotWRTDSKalman, 93
populateConcentrations, 94
populateDaily, 95, *110*
populateDailySamp, 97
populateDateColumns, 98
populateParameterINFO, 99
populateSampleColumns, 99, *111*, *114, 115*
populateSiteINFO, 100
precompute_L, 101
print.egret, 101

printFluxUnitCheatSheet, *32*, *76*, *78, 79*, *81*, *93, 94*, 102, *138*, *140*
printGroups, 103
printPairs, 104
printqUnitCheatSheet, *10*, *31, 32*, *52*, *58*, *61*, *63*, *67*, *70*, *72*, *74*, *79*, *82, 83*, *86*, *89*, 104, *105*, *139, 140*
printSeries, 105
processQWData, 106

qConst *(*Constants*)*, 17
qUnit-class, 107

randomSubset, 107
read_waterdata_daily, *110*
read_waterdata_samples, *110, 111*
readDataFromFile, 108
readNWISDaily, *5*, *39–41*, *49*, *96*, 109
readNWISInfo *(*INFOdataframe*)*, 41
readNWISpCode, *43*
readNWISSample, *5*, *39–41*, *49*, 110
readNWISsite, *43*
readUserDaily, *96*, 111
readUserInfo *(*INFOdataframe*)*, 41
readUserSample, 113
readWQPdata, *115*
readWQPInfo *(*INFOdataframe*)*, 41
readWQPqw, *106*, *115*
readWQPSample, 114
removeDuplicates, 116
run_WRTDS *(*runSurvReg*)*, 126
runGroups, 117
runPairs, 120
runSeries, 123
runSurvReg, *62*, *69*, 126

saveResults, 128
segments, *13*
selectDays, *57*, *59*, *64*, *66*, *78*, *80, 81*, *87–89*, *91, 92*, 128
setPA, *118*, *121*, *124*, 129, *139, 140*
setSeasonLabel, 130
setSeasonLabelByUser, 131
setUpEstimation, 131
setupYears, *56*, *77*, 132
startEnd, 133
stitch, 134
surfaceIndex, 136
surfaceStartEnd, 137

tableChange, 137
tableChangeSingle *(*tableChange*)*, 137
tableFlowChange, 139
tableResults, 140
triCube, 141

whatWQPsites, *43*
WRTDSKalman, *15*, 141

yPretty, 142