Package: rloadest (via r-universe)

October 14, 2026

Type Package

Title River Load Estimation

Version 0.4.5

Date 2017-07-26

Description Collection of functions to make constituent load estimations based on the LOADEST program.

License CC0

Depends R (>= 3.0.0), smwrBase, smwrGraphs, smwrStats, smwrQW

Imports lubridate, stats, segmented

Suggests dataRetrieval, EGRET, survival, knitr, testthat

BugReports https://github.com/USGS-R/rloadest/issues

LazyLoad yes

LazyData yes

RoxygenNote 6.0.1

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

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

Date/Publication 2021-06-15 21:38:58 UTC

RemoteUrl https://github.com/ldecicco-USGS/rloadest

RemoteRef no_vignettes

RemoteSha 5d9f583b8e33cb79d080cedf9baf9795711c5d2e

Contents

rloadest-package 3
AICc 3
app1.calib 4
app2.calib 5
app2.est 6

app4.calib . . . 7
app4.est . . . 8
app5.1999 . . . 9
app5.calib . . . 9
app5.est . . . 10
c2load . . . 11
censoring.Surv . . . 12
center . . . 13
coef.loadReg . . . 14
dailyAg . . . 15
fitted.loadReg . . . 15
jackStats . . . 16
loadConvFactor . . . 18
loadestQadj . . . 18
loadestTadj . . . 19
loadReg . . . 19
loadReport . . . 21
loadStats . . . 21
loadUnitConv . . . 22
logLik.loadReg . . . 23
makepredictcall.center . . . 23
mean.factor . . . 24
model . . . 25
Models . . . 25
nashSutcliffe . . . 26
Orthophosphate . . . 26
plot.loadReg . . . 27
predConc . . . 28
predLoad . . . 29
print.jackStats . . . 31
print.loadReg . . . 31
resampleUVdata . . . 32
residuals.loadReg . . . 33
rmse.loadReg . . . 34
seg . . . 35
segLoadReg . . . 36
segment . . . 37
selBestModel . . . 38
selBestSubset . . . 39
setXLDat . . . 40
vif.loadReg . . . 41

Index **42**

rloadest-package *LOADEST functions for R*

Description

Package:	rloadest
Type:	Package
License:	CC0
LazyLoad:	yes

Details

This package is intended to replicate and extend the LOADEST program for estimating constituent loads in streams and rivers. Some subtle differences between the output for LOADEST and rloadest include:

The least absolute deviation (LAD) method is not supported in rloadest.

LOADEST uses centered time when computing the sine and cosine terms in model numbers 4, 6, 7, 8, and 9, but the functions in rloadest use the actual decimal time so that the seasonality can more easily be assessed by the user.

The order of the terms in the predefined models is different between LOADEST and the rloadest functions.

The printed output of the model descriptions from rloadest matches the format the most users of R would recognize from other linear model output rather then the printed output from LOADEST.

Furthermore, the model building capability in the rloadest functions make easier to explore other forms of rating-curve models than LOADEST.

AICc *Akaike's An Information Criterion with Correction*

Description

Compute Akaike's An Information Criterion with Correction (AICc) for for finite sample sizes.

Usage

```
AICc(object)
```

Arguments

object	the output from loadReg, or any object that has a logLik method.

Value

A numeric value corresponding to the AICc of `object`.

Note

The penalty that `AIC` applies for adding explanatory variables is biased low when the number of samples is small. As a result, models with small smaple sizes can be overfitted. `AICc` can be used to identify more parsimonious models.

References

Hurvitch, C.M. and Tsai, C.L., 1989, Regression and time series model selection in small samples: Biometrika, v. 76, no. 2, p. 297–307.

See Also

`loadReg`,

Examples

```
# From application 1 in the vignettes
data(app1.calib)
app1.lr <- loadReg(Phosphorus ~ model(1), data = app1.calib,
 flow = "FLOW", dates = "DATES", conc.units="mg/L",
 station="Illinois River at Marseilles, Ill.")
AICc(app1.lr)
```

---

app1.calib *app1.calib Data*

Description

Illinois River at Marseilles, Illinois (Helsel & Hirsch, 2002)

Usage

```
app1.calib
```

Format

Data frame with 96 rows and 4 columns

Name	Type	Description
DATES	Date	Date of daily value
TIMES	character	Time corresponding to noon of daily value
FLOW	numeric	Daily mean streamflow

Phosphorus	numeric	Daily mean phosphorus concentration (assumed)

Source

Example calibration dataset from LOADEST

References

Runkel, R.G., Crawford, C.G., and Cohn, T.A., 2004, Load Estimator (LOADEST): a FORTRAN program for estimating constituent loads in streams and rivers: U.S. Geological Survey Techniques and Methods Book 4, Chapter A5, 69 p.

Examples

```
## Not run:
data(app1.calib)
# Plot concentration vs. flow
with(app1.calib, plot(FLOW, Phosphorus, log="xy"))

## End(Not run)
```

`app2.calib` *app2.calib Data*

Description

St.Joseph River near Newville, IN (Station # 04178000)

Usage

```
app2.calib
```

Format

Data frame with 32 rows and 4 columns

Name	Type	Description
DATES	Date	Date of daily value
TIMES	integer	Time that sample was actually taken
FLOW	numeric	Daily mean streamflow
Atrazine	numeric	Daily mean atrazine concentration (assumed)

Source

Example calibration dataset from LOADEST

References

Runkel, R.G., Crawford, C.G., and Cohn, T.A., 2004, Load Estimator (LOADEST): a FORTRAN program for estimating constituent loads in streams and rivers: U.S. Geological Survey Techniques and Methods Book 4, Chapter A5, 69 p.

Examples

```
## Not run:
data(app2.calib)
# Plot concentration vs. flow
with(app2.calib, plot(FLOW, Atrazine, log="xy"))

## End(Not run)
```

`app2.est` *app2.est Data*

Description

St.Joseph River near Newville, IN (Station # 04178000)

Usage

```
app2.est
```

Format

Data frame with 730 rows and 3 columns

Name	Type	Description
DATES	Date	Date of daily value
TIMES	integer	Time corresponding to noon of daily value
FLOW	numeric	Daily mean streamflow

Source

Example estimation dataset from LOADEST

References

Runkel, R.G., Crawford, C.G., and Cohn, T.A., 2004, Load Estimator (LOADEST): a FORTRAN program for estimating constituent loads in streams and rivers: U.S. Geological Survey Techniques and Methods Book 4, Chapter A5, 69 p.

Examples

```
## Not run:
data(app2.est)
summary(app2.est)

## End(Not run)
```

`app4.calib` *app4.calib Data*

Description

White River at Hazleton, Ind. (Station Number 03374100)

Usage

```
app4.calib
```

Format

Data frame with 45 rows and 9 columns

Name	Type	Description
DATES	Date	Date of daily value
TIMES	character	Time that sample was actually taken
FLOW	numeric	Daily mean streamflow
Buty.rmk	character	Remark code for butylate concentration
Buty	numeric	Daily mean butylate concentration (assumed)
Atra	numeric	Daily mean atrazine concentration (assumed)
Alach.rmk	character	Remark code for alachlor concentration
Alach	numeric	Daily mean alachlor concentration (assumed)
SuspSed	numeric	Daily mean suspended sediment concentration (assumed)

Source

Obtained from Charlie Crawford, 5 July 2001

References

Runkel, R.G., Crawford, C.G., and Cohn, T.A., 2004, Load Estimator (LOADEST): a FORTRAN program for estimating constituent loads in streams and rivers: U.S. Geological Survey Techniques and Methods Book 4, Chapter A5, 69 p.

Examples

```
## Not run:
data(app4.calib)
# Plot atrazine concentration vs. flow
with(app4.calib, plot(FLOW, Atra, log="xy"))

## End(Not run)
```

`app4.est` *app4.est Data*

Description

White River at Hazleton, Ind. (Station Number 03374100)

Usage

```
app4.est
```

Format

Data frame with 730 rows and 3 columns

Name	Type	Description
DATES	Date	Date of daily value
TIMES	character	Time corresponding to noon of daily value
FLOW	numeric	Daily mean streamflow

Source

Obtained from Charlie Crawford, 5 July 2001

References

Runkel, R.G., Crawford, C.G., and Cohn, T.A., 2004, Load Estimator (LOADEST): a FORTRAN program for estimating constituent loads in streams and rivers: U.S. Geological Survey Techniques and Methods Book 4, Chapter A5, 69 p.

Examples

```
## Not run:
data(app4.est)
summary(app4.est)

## End(Not run)
```

app5.1999 *app5.1999 Data*

Description

Little Arkansas River near Halstead, Kansas (Station Number 07143672)

Usage

```
app5.1999
```

Format

Data frame with 13 rows and 3 columns

Name	Type	Description
DATES	Date	Date of daily value
TIMES	character	Time that sample was actually taken
Alkalinity	numeric	Daily mean alkalinity (assumed)

Source

Retrieved from NWISweb on July 26, 2013 from URL: https://nwis.waterdata.usgs.gov/ks/nwis/qwdata

Examples

```
data(app5.1999)
head(app5.1999)
```

app5.calib *app5.calib Data*

Description

Little Arkansas River near Halstead, Kansas (Station Number 07143672)

Usage

```
app5.calib
```

Format

Data frame with 103 rows and 5 columns

Name	Type	Description
DATES	Date	Date of daily value
TIMES	character	Time that sample was actually taken
FLOW	numeric	Daily mean streamflow
SC	numeric	Daily mean specific conductance
Alkalinity	numeric	Daily mean alkalinity (assumed)

Source

Example calibration dataset from LOADEST

References

Runkel, R.G., Crawford, C.G., and Cohn, T.A., 2004, Load Estimator (LOADEST): a FORTRAN program for estimating constituent loads in streams and rivers: U.S. Geological Survey Techniques and Methods Book 4, Chapter A5, 69 p.

Examples

```
## Not run:
data(app5.calib)
# Plot concentration vs. specific conductance
with(app5.calib, plot(SC, Alkalinity, log="xy"))

## End(Not run)
```

`app5.est` *app5.est Data*

Description

Little Arkansas River near Halstead, Kansas (Station Number 07143672)

Usage

```
app5.est
```

Format

Data frame with 358 rows and 4 columns

Name	Type	Description
DATES	Date	Date of daily value
TIMES	character	ime corresponding to noon of daily value
FLOW	numeric	Daily mean streamflow
SC	numeric	Daily mean specific conductance

Source

Example estimation dataset from LOADEST

References

Runkel, R.G., Crawford, C.G., and Cohn, T.A., 2004, Load Estimator (LOADEST): a FORTRAN program for estimating constituent loads in streams and rivers: U.S. Geological Survey Techniques and Methods Book 4, Chapter A5, 69 p.

Examples

```
## Not run:
data(app5.est)
summary(app5.est)

## End(Not run)
```

---

c2load *Loads*

Description

Convert concentration and flow to load (flux).

Usage

```
c2load(conc, flow, flow.units = "cfs", conc.units = "", load.units = "kg",
  ignore.censoring = TRUE)
```

Arguments

conc	the concentration data missing values are permitted and result in missing values in the output.
flow	the flow data missing values are permitted and result in missing values in the output.
flow.units	character string describing the flow unit.
conc.units	character string describing the concentration unit.
load.units	character string describing the load unit.
ignore.censoring	logical, see **Value**.

Value

If ignore.censoring is TRUE, the default, then return a vector of numeric values with censored values replaced by 1/2 the detection limit. Otherwise, return a vector that retains the censoring—if conc is numeric, then uncensored; if conc is of class "qw," then the returned data would be of class "lcens" or "mcens."

References

will need some.

See Also

loadReg

Examples

```
# These calls return the conversion factors
c2load(1, 1, conc.units="mg/L")
c2load(1, 1, conc.units="mg/L", load.units="tons")
```

---

censoring.Surv *Describe Censoring*

Description

Gets the type of censoring ("none," "left," "multiple") for an object.

Usage

```
## S3 method for class 'Surv'
censoring(x)
```

Arguments

x	the object to get the type of censoring. For an object of class "Surv," the type must be "interval."

Value

A character string "none," "left," or "multiple" describing the type of censoring present in the object.

Note

This function is mostly used within other functions to determine the 'best' technique to use for analysis.

Examples

```
## Not run:
library(survival)
censoring(Surv(2.3, 2.3, type="interval2"))

## End(Not run)
```

---

center *Centered Linear Terms*

Description

Computes centered values. Used primarily in a linear regression formula.

Usage

```
center(x, center = NULL)
```

Arguments

x	a numeric vector for which to compute the centered values. Missing values are permitted and result in corresponding missing values in the output.
center	an optional value to use for the center of x.

Value

The centered value of x.

Note

The centering value by default is computed by the method described in Cohn and others (1992).

References

Cohn, T.A., Caulder, D.L., Gilroy, E.J., Zynjuk, L.D., and Summers, R.M., 1992, The validity of a simple statistical model for estimating fluvial constituent loads—An empirical study involving nutrient loads entering Chesapeake Bay: Water Resources research, v. 28, no. 5, p. 937–942.

See Also

quadratic, scale

Examples

```
# trivial ceneterd values from 1 to 10
center(seq(10))
```

`coef.loadReg` *Extract Model Coefficients*

Description

Extract the model coefficients from a load regression.

Usage

```
## S3 method for class 'loadReg'
coef(object, summary = FALSE, which = "load", ...)
```

Arguments

`object`	the output from `loadReg`.
`summary`	include standard errors and other information?
`which`	string indicating which coefficients to return; "load" returns the load model and "concentration" returns the concentration model coefficients.
`...`	further arguments passed to or from other methods.

Value

Either a names vector of the coefficients, if `summary` is `FALSE` or a matrix of the coefficients, their standard errors, z-scores, and attained p-values, if `summary` is `TRUE`.

Note

The attained p-values are computed from the log-likelihood test for AMLE regression and from a Wald chi-square test for MLE regression.

See Also

loadReg,

Examples

```
# From application 1 in the vignettes
data(app1.calib)
app1.lr <- loadReg(Phosphorus ~ model(1), data = app1.calib,
 flow = "FLOW", dates = "DATES", conc.units="mg/L",
 station="Illinois River at Marseilles, Ill.")
# Extract the coefficients
coef(app1.lr)
```

`dailyAg` *Daily Mean*

Description

Compute daily mean water-quality values for data frames containing water-quality data.

Usage

```
dailyAg(x, dates = "sample_dt", times = "sample_tm")
```

Arguments

`x`	a data frame containing water-quality data.
`dates`	the name of the sample date column.
`times`	the name of the sample time column.

Value

A data frame like `x` but with the means for each column by day and the sample time set to "1200."

`fitted.loadReg` *Extract Model Fitted Values*

Description

Extract the fitted values of a load regression.

Usage

```
## S3 method for class 'loadReg'
fitted(object, suppress.na.action = FALSE, which = "load",
  ...)
```

Arguments

object	an object of class "loadReg"—output from loadReg
suppress.na.action	logical, suppress the effects of the na.action in the call to loadReg and return only the fitted values corresponding to the fitted data.
which	a character string indicating the type of fitted values. Must be either "load" or "concentration."
...	further arguments passed to or from other methods.

Value

The fitted values from the regression. Note that these are not back- transformed but are in natrual log units.

See Also

loadReg

Examples

```
# From application 1 in the vignettes
data(app1.calib)
app1.lr <- loadReg(Phosphorus ~ model(1), data = app1.calib,
 flow = "FLOW", dates = "DATES", conc.units="mg/L",
 station="Illinois River at Marseilles, Ill.")
# Extract the fitted values
fitted(app1.lr)
```

---

jackStats *Jackknife Statistics*

Description

Compute selected jackknife statistics for a rating-curve load-estimation model.

Usage

```
jackStats(fit, which = "load")
```

Arguments

fit	an object of class "loadReg"—output from loadReg. Can also be an object of class "censReg."
which	a character string indicating the "load" or "concentration" model for an object of class "loadReg" or "censReg" for an object of class "censReg."

Value

An object of class "jackStats" containing these components: coef, the table of coefficient estimates, the jackknife bias and standard errors
coefficients, the jackknifed coefficients
pctcens, the percentage of left-censored values.
The PRESS statistic and individual jackknife differences are also returned when the percentage of censoring is 0.

Note

The `jackStats` function can only be used when the analysis is AMLE.

Abdi and Williams (2010) describe the jackknife as refering to two related techniques: the first estimates the parameters, their bias and standard errors and the second evaluates the predictive performance of the model. The second technique is the PRESS statistic (Helsel and Hirsch, 2002), but can only be used on uncensored data; it is computed by `jackStats` when no data are censored. The first technique can be used to assess the coefficients of the regression—the bias should be small and the jackknife standard errors should not be much different from the standard errors reported for the regression. Efron and Tibshirani (1993) suggest that the bias is small if the relative bias (biuas divided by the jackknife standard error) is less than 0.25.

References

Abdi, H. and Williams, L.J., 2010, Jackknife, in encyclopedia of research design, Salkind, N.J., editor: Thousand Oaks, Calif., SAGE Publications, 1719 p.

Efron, B. and Tibshirani, R.J., 1993, An introduction to the bootstrap: Boca Raton, Fla., Chapman and Hall/CRC, 436 p.

Helsel, D.R. and Hirsch, R.M., 2002, Statistical methods in water resources: U.S. Geological Survey Techniques of Water-Resources Investigations, book 4, chap. A3, 522 p. Salkind,

See Also

`loadReg`

Examples

```
# From application 1 in the vignettes
data(app1.calib)
app1.lr <- loadReg(Phosphorus ~ model(1), data = app1.calib,
 flow = "FLOW", dates = "DATES", conc.units="mg/L",
 station="Illinois River at Marseilles, Ill.")
jackStats(app1.lr)
```

loadConvFactor *Unit Conversion*

Description

Computes the conversion factor to compute the flux units (load units) from the concentration and flow units

Usage

```
loadConvFactor(flow.units, conc.units, load.units)
```

Arguments

flow.units	character string describing the flow unit. The only valid values are "cubic meter per second," "cms," "cubic feet per second," or "cfs."
conc.units	character string describing the concentration unit. The valid units are "mg/l," "mg/L," "ug/l," "ug/L," "ng/l," "ng/L," "milligrams per liter," "micrograms per liter," "nanograms per liter," "col/100mL," "col/dL," or "colonies per 100mL."
load.units	character string describing the load unit. The valid values are "pounds," "tons," "mg," "milligrams," "grams," "g," "kilograms," "kg," "metric tons," "Mg," or "million colonies."

Value

The conversion factor.

Examples

```
loadConvFactor("cubic meter per second","milligrams per liter","pounds")
```

loadestQadj *Center Flow*

Description

Internal support function for rloadest that computes the adjustment to flow.

Usage

```
loadestQadj(x, round = options("round.flow"))
```

Arguments

x	the calibration flow data.
round	either a numeric value indicating the number of decimal places, or a list containing a value indicating the number of decimal places. If NULL, then do no round.

Value

The centering value for flow.

---

loadestTadj *Center Time*

Description

Internal support function for rloadest that computes the adjustment to time.

Usage

```
loadestTadj(x, round = options("round.time"))
```

Arguments

x	the calibration date data.
round	either a numeric value indicating the number of decimal places, or a list containing a value indicating the number of decimal places. If NULL, then do no round.

Value

A vector of length 2 containing the base (reference) year and the centering time correction value. The user value for centered time would be sum.

---

loadReg *Load Estimation*

Description

Build a rating-curve (regression) model for river load estimation.

Usage

```
loadReg(formula, data, subset, na.action, flow, dates, flow.units = "cfs",
  conc.units = "", load.units = "kg", time.step = "day", station = "")
```

Arguments

formula	a formula describing the regression model. See **Details**.
data	the data to search for the variables in formula.
subset	an expression to select a subset of the data.
na.action	what to do with missing values.
flow	character string indicating the name of the flow column.
dates	character string indicating the name of the date column.
flow.units	character string describing the flow units. See **Details**.
conc.units	character string describing the concentration unit. See **Details**.
load.units	character string describing the load unit. See **Details**.
time.step	character string describing the time step of the calibration data. Must be one of "instantaneous," "2 hours," "3 hours," "4 hours," "6 hours," "12 hours," or "day." The default is "day."
station	character string description of the station.

Details

The left-hand side of the formula may be any numeric variable, just as with lm or a variable of class "lcens," "mcens," or "qw." Also permitted are variables constructed using Surv of type "right,", "interval," or "interval2" (for left-censored data, use as.lcens.
For un- or left-censored data, AMLE is used unless weights are specified in the model, then MLE is used, through a call to survreg. For any other censored data, MLE is used.

Typically, loadReg expects the response variable to have units of concentration, mass per volume. For these models, See loadConvFactor for details about valid values for flow.units, conc.units and load.units. For some applications, like bed load estimation, the response variable can have units of flux, mass per time. For these models, conc.units can be expressed as any valid load.units per day. The rate must be expressed in terms of "/d," "/dy," or "/day."

Value

An object of class "loadReg."

References

Runkel, R.L., Crawford, C.G., and Cohn, T.A., 2004, Load estimator (LOADEST): a FORTRAN program for estimating constituent loads in streams and rivers: U.S. Geological Survey Techniques and Methods book 4, chap. A5, 69 p.

See Also

censReg, as.lcens, as.mcens, Surv

Examples

```
# From application 1 in the vignettes
data(app1.calib)
app1.lr <- loadReg(Phosphorus ~ model(1), data = app1.calib,
 flow = "FLOW", dates = "DATES", conc.units="mg/L",
 station="Illinois River at Marseilles, Ill.")
print(app1.lr)
```

---

loadReport *Create Load Report*

Description

Create a 2-page pdf file report of a rating-curve load model. The report contains the text output and 6 diagnostic plots.

Usage

```
loadReport(x, file)
```

Arguments

x	the load model.
file	the output file base name; the .pdf suffix is appended to make the actual file name. if missing, then the name of x is used as the base name.

Value

The actual file name is returned invisibly.

---

loadStats *Summary Statistics*

Description

Compute some summary statistics for a rating-curve load-estimation model.

Usage

```
loadStats(fit, which = "load")
```

Arguments

fit	an object of class "loadReg"—output from loadReg.
which	a character string indicating the "load" or "concentration" model.

Value

A list containing `outSum`, selected summary statistics; of the observed and estimated values and `outBias`, the bias statistics.

See Also

`loadReg`

Examples

```
# From application 1 in the vignettes
data(app1.calib)
app1.lr <- loadReg(Phosphorus ~ model(1), data = app1.calib,
 flow = "FLOW", dates = "DATES", conc.units="mg/L",
 station="Illinois River at Marseilles, Ill.")
loadStats(app1.lr)
```

---

loadUnitConv *Unit Conversion*

Description

Computes the factor to convert from between load units.

Usage

```
loadUnitConv(from, to)
```

Arguments

`from`	character string describing the load units to convert from.
`to`	character string describing the load units to convert to.

Value

The conversion factor.

Examples

```
loadUnitConv("kilograms", "tons")
```

logLik.loadReg *Extract Log-Likelihood*

Description

Compute the log-likelihood statistics for a load regression.

Usage

```
## S3 method for class 'loadReg'
logLik(object, ...)
```

Arguments

object	the output from loadReg.
...	further arguments passed to or from other methods.

Value

An object of class "logLik" containing the log-likelihood and the attributes "df" (degrees of freedom) and "nobs" (number of observations).

See Also

loadReg,

Examples

```
# From application 1 in the vignettes
data(app1.calib)
app1.lr <- loadReg(Phosphorus ~ model(1), data = app1.calib,
 flow = "FLOW", dates = "DATES", conc.units="mg/L",
 station="Illinois River at Marseilles, Ill.")
logLik(app1.lr)
```

makepredictcall.center *Utility Function for Safe Prediction*

Description

A utility to help model.frame.default create the right matrices when predicting from models with center term. Used only internally.

Usage

```
## S3 method for class 'center'
makepredictcall(var, call)
```

Arguments

var	a variable.
call	the term in the formula, as a call.

Value

A replacement for call for the prediction variable.

---

mean.factor *Arithmetic Mean*

Description

Method functions to compute the "mean" for factors and characters. These funcitons are intended primarily as support functions when aggregating unit values data in predLoad.

Usage

```
## S3 method for class 'factor'
mean(x, ...)

## S3 method for class 'character'
mean(x, ...)
```

Arguments

x	an object of class "factor" or "character."
...	further arguments passed to or from other methods.

Value

If all values are identical, then the unique value, otherwise the missing value (NA).

model *Load Model*

Description

Support function for building a predefined rating curve load model.

Usage

```
model(model.no, data, flow, time)
```

Arguments

model.no	the model number.
data	the dataset.
flow	character string indicating the name of the flow column.
time	character string indicating the name of the date/time column.

Value

Row number to select from data to build a predefined model.

Models *Models A dataset that describes the equivalent formula for each of the predefined model number.*

Description

Models A dataset that describes the equivalent formula for each of the predefined model number.

Usage

```
Models
```

Format

Data frame with 9 rows and 2 columns

Name	Type	Description
Number	integer	The predefined model number
Formula	factor	The equivalent formula for the predefined model number

See Also

model

Examples

```
data(Models)
print(Models)
```

---

nashSutcliffe *Nash Sutcliffe*

Description

Compute the Nash-Sutcliffe efficiency rating for model estiamtes.

Usage

```
nashSutcliffe(obs, est, na.rm = TRUE)
```

Arguments

obs	a vector of the observed values.
est	a vector of the model estiamted values. Each value must pair with each value in obs.
na.rm	remove missing values from obs and est before computing the Nash-Sutcliffe efficiency rating.

Value

A single numeric value representing the Nash-Sutcliffe efficiency rating for the observed and estiamted data.

Orthophosphate *Example Orthophosphate data included in LOADEST package*

Description

Example data representing atrazine

plot.loadReg *Diagnostic Plot*

Description

Plot rating-curve load model diagnostics.

Usage

```
## S3 method for class 'loadReg'
plot(x, which = "All", set.up = TRUE, span = 1, ...)
```

Arguments

x	an object of class "loadReg"—output from loadReg
which	either "All" or any of a sequence from 1 to 7 indicating which plot, see **Details**.
set.up	set up the graphics page?
span	the span to use for the loess smooth. Set to 0 to suppress.
...	further arguments passed to or from other methods.

Details

Seven graphs can be produced by this function. If which is "All," then all plots are produced. The argument which can also be the name of an explanatory variable so that a partial residual plot is created for a single variable. Or which can be any of a sequence of numbers from 1 thorugh 7. Numeric values for which:

1. Observed vs. fitted.
2. Fitted vs. Residual
3. S-L plot
4. A correlogram if dates are available in the model or in the data set
5. Q-normal
6. Tukey boxplots for oberved and estimated
7. Partial residual plots for each explanatory variable

Value

The object x is returned invisibly.

Note

This plotting function uses the core routines in the smwrGraphs package. It requires a graphics page that is set up from the functions in that package (setpage or setPDF) if set.up is FALSE. The graphs that are produced by this function are based on the publication guidelines of the USGS.

See Also

censReg, setPage, setPDF

Examples

```
# From application 1 in the vignettes
## Not run:
data(app1.calib)
app1.lr <- loadReg(Phosphorus ~ model(1), data = app1.calib,
 flow = "FLOW", dates = "DATES", conc.units="mg/L",
 station="Illinois River at Marseilles, Ill.")
# Produce the full suite of diagnostic plots
plot(app1.lr)

## End(Not run)
```

---

predConc *Predict Concentrations*

Description

Estimate concentrations from a rating-curve model from loadReg for a new data frame.

Usage

```
predConc(fit, newdata, by = "day", allow.incomplete = FALSE,
  conf.int = 0.95)
```

Arguments

fit	the output from loadReg.
newdata	a data frame of the prediction variables. MIssing values are not permitted in any column in newdata. Observations with missing values NAs must be removed before prediction. Columns that are not needed for prediction that contain missing values can be removed before removing all rows with missing values. The maximum number of rows permitted in newdata is 176000.
by	the time frame for estimates. See Details.
allow.incomplete	compute loads for periods withing missing values or incomplete record? See Details.
conf.int	the confidence interval to compute for concentrations. See Details.

Details

The time frame specified by by must be either "unit" or "day."

If allow.incomplete is TRUE, then concentrations will be computed based on all nonmissing values, otherwise missing values NAs will be returned. For this application, missing values includes NAs and incomplete days. For prediction by "day" when there are variable number of unit values per day, allow.incomplete must be set to TRUE.

The term confidence interval is used here as in the original documentation for LOADEST, but the values that are reported are the prediction intervals, computed from the SEP.

Value

A data frame containing the concnetration estimates.

See Also

loadReg,

Examples

```
# From application 1 in the vignettes
data(app1.calib)
app1.lr <- loadReg(Phosphorus ~ model(1), data = app1.calib,
 flow = "FLOW", dates = "DATES", conc.units="mg/L",
 station="Illinois River at Marseilles, Ill.")
predConc(app1.lr, app1.calib)
```

---

predLoad *Predict Loads*

Description

Estimate loads from a rating-curve model from loadReg for a new data frame, aggregating the loads by specified time periods.

Usage

```
predLoad(fit, newdata, load.units = fit$load.units, by = "total",
  seopt = "exact", allow.incomplete = FALSE, conf.int = 0.95,
  print = FALSE)
```

Arguments

fit	the output from loadReg.
newdata	a data frame of the prediction variables. Missing values are not permitted in any column in newdata. Observations with missing values NAs must be removed before prediction. Columns that are not needed for prediction that contain missing values can be removed before removing all rows with missing values. The maximum number of rows permitted in newdata is 176000.

load.units — a character string indicating the units of the predicted loads/fluxes. By default, uses the value specified in loadReg. See loadReg for a complete list of options.

by — the time frame for estimates. See Details.

seopt — a character string indicating how to comute the standard error of the aggregated load estimates, must be either "exact" or "approximate." Only the first letter is necessary.

allow.incomplete — compute loads for periods withing missing values or incomplete record? See Details.

conf.int — the confidence interval to compute for loads computed by "day" or "unit." The confidence interval is fixed at 0.95 for any other value for by. See Details.

print — print a report summary of the load estimate?

Details

The time frame specified by by can be "unit," "day," "month," "water year," "calendar year," "total," or the name of a column in newdata that can be used to group the data.

If allow.incomplete is TRUE, then loads will be computed based on all nonmissing values, otherwise missing values NAs will be returned. For this application, missing values includes NAs and gaps in the record, except for by set to "total" or user defined groups where missing values only includes NAs. For prediction by "day" when there are variable number of unit values per day, allow.incomplete must be set to TRUE.

The term confidence interval is used here as in the original documentation for LOADEST, but the values that are reported are the prediction intervals, computed from the SEP.

Value

A data frame containing the load estimates.

See Also

loadReg,

Examples

```
# From application 1 in the vignettes
data(app1.calib)
app1.lr <- loadReg(Phosphorus ~ model(1), data = app1.calib,
 flow = "FLOW", dates = "DATES", conc.units="mg/L",
 station="Illinois River at Marseilles, Ill.")
predLoad(app1.lr, app1.calib)
```

print.jackStats *Print Results*

Description

Print the results of a jackknife analysis of a censored regression.

Usage

```
## S3 method for class 'jackStats'
print(x, digits = 4, ...)
```

Arguments

x	an object of class "jackStats"—output from jackStats.
digits	the number of significant digits to print.
...	further arguments passed to or from other methods.

Value

The object x is returned invisibly.

Note

The printed output includes the original original estimate, the jackknife bias and standard error and the relative bias for each parameter in the regression model.

See Also

loadReg

print.loadReg *Print Results*

Description

Print the results of an load rating-curve regression.

Usage

```
## S3 method for class 'loadReg'
print(x, digits = 4, brief = TRUE, load.only = brief,
  ...)
```

Arguments

x	an object of class "loadReg"—output from loadReg.
digits	the number of significant digits to print.
brief	print the brief output? See **Note**.
load.only	print only the load model and not concentration model results.
...	further arguments passed to or from other methods.

Value

The object x is returned invisibly.

Note

The printed output replicates the output described in Runkel (2004) and includes a short section summarizing the data, the load model and coefficients, regression statistics, and comparison of observed and estimated loads. If load.only is set to FALSE, then similar output is generated for the concetration model. If brief is FALSE, then additional descriptions of selected sections of the output are produced.

If the estimation method is "MLE," then the estimated loads used in the comparison to observed loads are approximate becuase they are estimated using MLE, rather than AMLE, wihch is used for predLoad and predConc. The bias is very small when the residual varaince is less than 0.5, but can be large when the residual variance is greater than 1.

References

Runkel, R.L., Crawford, C.G., and Cohn, T.A., 2004, Load estimator (LOADEST): A FORTRAN program for estimating constituent loads in streams and rivers: U.S. Geological Survey Techniques and Methods book 4, chap. A5, 69 p.

See Also

loadReg

---

resampleUVdata *Resample Unit-Value Data*

Description

Unit-value data can be recorded at any arbitrary intervals. For some applications, such as load estimates, a uniform series is required. The resampleUVdata function resamples the orginal unit-value data to a consistent time interval.

Usage

```
resampleUVdata(UVdata, time.step = 15, start.date = "", end.date = "",
  max.diff = "2 hours", missing.days = "exclude")
```

Arguments

UVdata	the dataset containing the unit-values data. Must have one column that represents the time of the observation that is class "POSIXt." Missing values are not permitted in that column.
time.step	the time step of the new data in minutes; must divide an hour exactly evenly. The default value is 15 minutes.
start.date	a character string indicating the first day of the output dataset. The default value ("") indicates use the first day in UVdata.
end.date	a character string indicating the last day of the output dataset. The default value ("") indicates use the last day in UVdata.
max.diff	a character string indicating the maximum difference in time to sucessfully resample the unit-value data. The default is "2 hours" see mergeNearest for details.
missing.days	a characer string indicating what action should be taken for days not present in UVdata. Must be either "exclude" to remove those days from the output, or "include" to retain them. Can be abbreviated. If missing.days is "include," then partial days within max.diff will be included in the output data frame.

Value

A data frame like UVdata but having a uniform time step.

---

residuals.loadReg *Extract Model Residuals*

Description

Extract the residuals from the load or concetration regression model. The residuals will be the same unless the log of flow is not an explanatory variable.

Usage

```
## S3 method for class 'loadReg'
residuals(object, type = "working",
  suppress.na.action = FALSE, model = c("load", "concentration"), ...)
```

Arguments

object	an object of class "loadReg"—output from loadReg
type	The type of residuals, see **Details**.
suppress.na.action	logical, suppress the effects of the na.action in the call to loadReg and return only the fitted values corresponding to the fitted data.
model	the type of model, must be either "load" or "concentration."
...	not used, required for other methods.

Details

The value for type can be any one of the following:

Value	Description
"working"	Residuals with censored residuals replaced by their expected values
"response"	Residuals from the linear predictor
"influence"	An estimate of Cook's D values based on "working" residuals
"leverage"	The hat diagonals
"S-L"	The square-root of the absolute value of the residuals with censored residuals replaced by their expected value

Also, any other value of type for residuals.survreg can be used to obtain those residuals. Note that "working" and "response" are defined in the table above, in keeping with older versions of loadReg.

Value

The residuals from the regression as specified by type.

Note

The residuals from the load regression are the same as those from the concentration regression, so there is no option to distinguish among those models.

See Also

loadReg

---

rmse.loadReg *Root-Mean-Squared Error*

Description

Compute the root-mean-squared error (RMSE) of the difference between observed values and the fitted values for the load or concentration model. The RMSEs will be the same unless the log of flow is not an explanatory variable.

Usage

```
## S3 method for class 'loadReg'
rmse(x, model = c("load", "concentration"), ...)
```

Arguments

x	the output from loadReg.
model	the type of model, must be either "load" or "concentration."
...	not used, required for other methods.

Value

The estimated root-mean-squared error, also know as the residual standard error.

See Also

loadReg,

Examples

```
# From application 1 in the vignettes
data(app1.calib)
app1.lr <- loadReg(Phosphorus ~ model(1), data = app1.calib,
 flow = "FLOW", dates = "DATES", conc.units="mg/L",
 station="Illinois River at Marseilles, Ill.")
rmse(app1.lr)
```

---

seg *Load Model*

Description

Support function for building a segmented rating curve load model. Required in the formula in segLoadReg to define the segmented model.

Usage

```
seg(x, N)
```

Arguments

x	the data to segment. Missing values are permitted and result corresponing in missing values in output.
N	the number of breaks in the segmented model.

Value

The data in x with attributes to build the segmented model.

segLoadReg *Load Estimation*

Description

Build a segmented rating-curve (regression) model for river load estimation.

Usage

```
segLoadReg(formula, data, subset, na.action, flow, dates, flow.units = "cfs",
  conc.units = "", load.units = "kg", time.step = "day", station = "",
  trace = TRUE)
```

Arguments

formula	a formula describing the regression model. See **Details**.
data	the data to search for the variables in formula.
subset	an expression to select a subset of the data.
na.action	what to do with missing values.
flow	character string indicating the name of the flow column.
dates	character string indicating the name of the date column.
flow.units	character string describing the flow unit.
conc.units	character string describing the concentration unit.
load.units	character string describing the load unit.
time.step	character string describing the time step of the calibration data.
station	character string description of the station.
trace	if logical, then if TRUE print a short summary of the segmented fit. Otherwise a character string and the segmented model is saved as that object name.

Details

The left-hand side of the formula can be any numeric variable, just as with lm or a variable of class "lcens," "mcens," "qw," or "Surv." The response variable must be a column in data; it cannot be constructed using as.lcens, as.mcens, or Surv. The initial segmented model is based on uncensored data—simple substitution of 1/2 the reporting limit is used for left-censored values and multiply censored values cause the analysis to fail.

The first term of right-hand side must be defined by the seg function with the number of segments. The first term may be followed by any number of additional terms. The final model will place the segmeted term in the last postition and seg will be replaced by the proper call to segment.

Value

An object of class "loadReg."

References

will need some.

See Also

censReg, link{seg}, segment

Examples

```
# From application 1 in the vignettes
data(app1.calib)
app1.lr <- loadReg(Phosphorus ~ model(1), data = app1.calib,
 flow = "FLOW", dates = "DATES", conc.units="mg/L",
 station="Illinois River at Marseilles, Ill.")
print(app1.lr)
```

---

segment *Segmented Model*

Description

Computes the basis for a segmented model. Used primarily in a linear regression or load model.

Usage

```
segment(x, psi)
```

Arguments

x	the data to segment. Missing values are permitted and result corresponding in missing values in output.
psi	a numeric vector of the breakpoints.

Value

A matrix contining a column named X of the data in x and paired U and P columns for each of the breaks in psi that form the basis for that segment.

`selBestModel` *Load Estimation*

Description

Select the "best" predefined rating-curve (regression) model for river load estimation.

Usage

```
selBestModel(constituent, data, subset, na.action, flow, dates,
  flow.units = "cfs", conc.units = "", load.units = "kg",
  time.step = "day", station = "", criterion = c("AIC", "SPCC", "AICc"))
```

Arguments

`constituent`	a character string giving the name of the response variable for which loads are to be computed.
`data`	the data to search for the variables in `formula`.
`subset`	an expression to select a subset of the data.
`na.action`	what to do with missing values.
`flow`	character string indicating the name of the flow column.
`dates`	character string indicating the name of the date column.
`flow.units`	character string describing the flow unit.
`conc.units`	character string describing the concentration unit.
`load.units`	character string describing the load unit.
`time.step`	character string describing the time step of the calibration data.
`station`	character string description of the station.
`criterion`	the criterion to use for subset selection, must be one of "AIC," "SPCC," or "AICc."

Value

An object of class "loadReg."

References

will need some.

See Also

censReg

Examples

```
# Use the data from application 1 in the vignettes
data(app1.calib)
app1.lr <- selBestModel("Phosphorus", data = app1.calib,
 flow = "FLOW", dates = "DATES", conc.units="mg/L",
 station="Illinois River at Marseilles, Ill.")
# Extract the fitted values
print(app1.lr)
```

`selBestSubset` *Load Estimation*

Description

Select the "best" subset of a user-defined rating-curve (regression) model for rver load estimation.

Usage

```
selBestSubset(formula, min.formula = ~1, data, subset, na.action, flow, dates,
  flow.units = "cfs", conc.units = "", load.units = "kg",
  time.step = "day", station = "", criterion = c("AIC", "SPCC", "AICc"))
```

Arguments

`formula`	a formula describing the regression model. See `loadReg` for details.
`min.formula`	a formula containing the minimum variables to use in the final model. The default is to only force the intercept term, whihc will normally be acceptable. In some rare cases, the log of flow may be needed.
`data`	the data to search for the variables in `formula`.
`subset`	an expression to select a subset of the data.
`na.action`	what to do with missing values.
`flow`	character string indicating the name of the flow column.
`dates`	character string indicating the name of the date column.
`flow.units`	character string describing the flow unit.
`conc.units`	character string describing the concentration unit.
`load.units`	character string describing the load unit.
`time.step`	character string describing the time step of the calibration data.
`station`	character string description of the station.
`criterion`	the criterion to use for subset selection, must be one of "AIC," "SPCC," or "AICc."

Value

An object of class "loadReg."

Note

The printed output of the model inlcudes the anova component from step. That table summarizes the step wise selection and the criterion used for each step. The statistics possibly represent a smaller sample size than used for the final model because the step function requires a data set with no missing values. If missing values are found a warning is printed.

See Also

censReg, step

---

setXLDat *Model Matrix*

Description

Internal support function to extract the model matrix for one of the 9 predefined models in LOADEST.

Usage

```
setXLDat(data, flow, dates, Qadj, Tadj, model.no)
```

Arguments

data	the dataset containing dates and flow columns.
flow	character string indicating the name of the flow column.
dates	character string indicating the name of the date column.
Qadj	the centering value for flow.
Tadj	the centering value for decimal time.
model.no	the model number, must be in the range 1-9.

Value

The model matrix corresponding to the selected model number.

vif.loadReg *Variance Inflation Factors*

Description

Computes the variance inflation factor (Helsel and Hirsch, 2002) for each variable in a load regression.

Usage

```
## S3 method for class 'loadReg'
vif(model, ...)
```

Arguments

model	an object of class "loadReg"—output from loadReg.
...	further arguments passed to or from other methods.

Value

further arguments passed to or from other methods.

See Also

loadReg

Examples

```
# From application 1 in the vignettes
data(app1.calib)
app1.lr <- loadReg(Phosphorus ~ model(1), data = app1.calib,
 flow = "FLOW", dates = "DATES", conc.units="mg/L",
 station="Illinois River at Marseilles, Ill.")
vif(app1.lr, app1.calib)
```

Index

∗ **attribute**
censoring.Surv, 12
∗ **censored**
c2load, 11
censoring.Surv, 12
loadReg, 19
segLoadReg, 36
selBestModel, 38
selBestSubset, 39
∗ **conversions**
loadConvFactor, 18
loadUnitConv, 22
∗ **datasets**
app1.calib, 4
app2.calib, 5
app2.est, 6
app4.calib, 7
app4.est, 8
app5.1999, 9
app5.calib, 9
app5.est, 10
Models, 25
∗ **data**
Orthophosphate, 26
∗ **estimation**
rloadest-package, 3
∗ **hplot**
plot.loadReg, 27
∗ **loads**
c2load, 11
loadReg, 19
segLoadReg, 36
selBestModel, 38
selBestSubset, 39
∗ **load**
rloadest-package, 3
∗ **manip**
center, 13
∗ **quality**
Orthophosphate, 26
∗ **regression**
c2load, 11
coef.loadReg, 14
fitted.loadReg, 15
loadReg, 19
plot.loadReg, 27
residuals.loadReg, 33
segLoadReg, 36
selBestModel, 38
selBestSubset, 39
vif.loadReg, 41
∗ **unit**
loadConvFactor, 18
loadUnitConv, 22
∗ **utilities**
jackStats, 16
loadStats, 21
nashSutcliffe, 26
print.jackStats, 31
print.loadReg, 31
∗ **water**
Orthophosphate, 26

AICc, 3
app1.calib, 4
app2.calib, 5
app2.est, 6
app4.calib, 7
app4.est, 8
app5.1999, 9
app5.calib, 9
app5.est, 10
as.lcens, *20*
as.mcens, *20*

c2load, 11
censoring.Surv, 12
censReg, *20*, *28*, *37*, *38*, *40*
center, 13

coef.loadReg, 14

dailyAg, 15

fitted.loadReg, 15

jackStats, 16

loadConvFactor, 18, *20*
loadestQadj, 18
loadestTadj, 19
loadReg, *4*, *12*, *14*, *16, 17*, 19, *22, 23*, *29–32*, *34, 35*, *39*, *41*
loadReport, 21
loadStats, 21
loadUnitConv, 22
logLik.loadReg, 23

makepredictcall.center, 23
mean.character *(*mean.factor*)*, 24
mean.factor, 24
mergeNearest, *33*
model, 25, *26*
model.frame.default, *23*
Models, 25

nashSutcliffe, 26

Orthophosphate, 26

plot.loadReg, 27
predConc, 28
predLoad, 29
print.jackStats, 31
print.loadReg, 31

quadratic, *14*

resampleUVdata, 32
residuals.loadReg, 33
residuals.survreg, *34*
rloadest-package, 3
rmse.loadReg, 34

scale, *14*
seg, 35
segLoadReg, 36
segment, 37, *37*
selBestModel, 38
selBestSubset, 39
setPage, *28*
setPDF, *28*
setXLDat, 40
step, *40*
Surv, *20*

vif.loadReg, 41