Package 'LakeMetabolizer'

January 20, 2025

Title Tools for the Analysis of Ecosystem Metabolism

Maintainer Jake Zwart <jayzlimno@gmail.com>

Version 1.5.5

Description A collection of tools for the calculation of freewater metabolism from in situ time series of dissolved oxygen, water temperature, and, optionally, additional environmental variables. LakeMetabolizer implements 5 different metabolism models with diverse statistical underpinnings: bookkeeping, ordinary least squares, maximum likelihood, Kalman filter, and Bayesian. Each of these 5 metabolism models can be combined with 1 of 7 models for computing the coefficient of gas exchange across the air–water interface (k). LakeMetabolizer also features a variety of supporting functions that compute conversions and implement calculations commonly applied to raw data prior to estimating metabolism (e.g., oxygen saturation and optical conversion models).

License GPL (>= 2)

Imports plyr, methods

Suggests R2jags, testthat

Depends R (>= 2.15.0), rLakeAnalyzer (>= 1.4)

Repository CRAN

BugReports https://github.com/GLEON/LakeMetabolizer/issues

URL https://www.tandfonline.com/doi/abs/10.1080/IW-6.4.883

RoxygenNote 7.2.1

Encoding UTF-8

NeedsCompilation yes

Author Luke Winslow [aut],
Jake Zwart [cre, aut] (<https://orcid.org/0000-0002-3870-405X>),
Ryan Batt [aut],
Jessica Corman [aut],
Hilary Dugan [aut],
Paul Hanson [aut],
Aline Jaimes [aut],

Jordan Read [aut],
Richard Woolway [aut]

Date/Publication 2022-11-15 23:30:16 UTC

Contents

`calc.lw.net` *Estimate net long wave heat radiation*

Description

Returns the net long wave radiation based on Crawford and Duchon, 1999.

Usage

```
calc.lw.net(ts.data, lat, atm.press)

calc.lw.net.base(dateTime, sw, Ts, lat, atm.press, airT, RH)
```

Arguments

ts.data	Object of class data.frame including the required variables(see details for list of variables and their units)
lat	latitude in degrees north
atm.press	atmospheric pressure in mb
dateTime	vector of datetime in POSIXct format
sw	numeric value of short wave radiation, W/m2
Ts	numeric value of surface water temperature, degC
airT	numeric value of air temperature, degC
RH	numeric value of relative humidity, %

Value

for calc.lw.net.base

A numeric value of net long wave heat flux in W/m^2

for calc.lw.net

A data.frame with columns datetime and lwnet in W/m^2

Author(s)

R Iestyn Woolway Jordan S. Read Hilary Dugan Luke Winslow

References

Crawford, T.M., and Duchon, C.E. 1999. *An improved parameterization for estimating effective atmospheric emissivity for use in calculating daytime downwelling longwave radiation*. Journal of Applied Meteorology 38: 474-480.

See Also

k.read and k.macIntyre

Examples

```
## Base example
dateTime <- as.POSIXct("2013-12-30 23:00")
Uz <- 3
airT <- 20
RH <- 90
sw <- 800
wndZ <- 2
```

```
Kd <- 2
lat <- 54
lake.area <- 5000
atm.press <- 1013
Ts <- 22
calc.lw.net.base(dateTime,sw,Ts,lat,atm.press,airT,RH)

## Example using timeseries in a data frame
data.path = system.file('extdata', package="LakeMetabolizer")

sp.data = load.all.data('sparkling', data.path)

# Prep the input data
ts.data = sp.data$data #pull out just the timeseries data
atm.press = 1018
lat = sp.data$metadata$latitude

lwnet = calc.lw.net(ts.data, lat, atm.press)
plot(lwnet$datetime, lwnet$lwnet)
```

`calc.zeng` *Estimate sensible and latent heat fluxes*

Description

Returns the sensible and latent heat fluxed based on Zeng et al, 1998'

Usage

```
calc.zeng(dateTime,Ts,airT,Uz,RH,atm.press,wnd.z,airT.z,RH.z)
```

Arguments

`dateTime`	vector of datetime in POSIXct format
`Ts`	numeric value of surface water temperature, degC
`airT`	numeric value of air temperature, degC
`Uz`	numeric value of wind speed, m/s
`RH`	numeric value of relative humidity, %
`atm.press`	atmospheric pressure in mb
`wnd.z`	height of wind measurement, m
`airT.z`	height of air temperature measurement, m (optional)
`RH.z`	height of relative humidity measurement, m (optional)

Value

A data.frame including sensible and latent heat flux estimates, and other variables used in calculating these fluxes.

Author(s)

R. Iestyn. Woolway

References

Zeng, X., M. Zhao., and Dickinson, R.E. 1998. *Intercomparison of bulk aerodynamic algorithms for the computation of sea surface fluxes using TOGA COARE and TAO data*. Journal of Climate 11: 2628-2644.

See Also

k.read

Examples

```
dateTime <- as.POSIXct("2013-12-30 23:00")
Ts <- 22.51
airT <- 20
Uz <- 3
RH <- 90
atm.press <- 1013
wnd.z <- 2
calc.zeng(dateTime,Ts,airT,Uz,RH,atm.press,wnd.z)
```

---

get.Ts *gets surface water temperatures*

Description

grabs best available data for surface water temperature

Usage

```
get.Ts(data, s.range = c(0, 1))
```

Arguments

data	Object of class data.frame
s.range	a numeric vector of length=2 with the range for depth measurements to still be considered 'surface'

Value

An object of class data.frame

Author(s)

Jordan S. Read

See Also

has.vars get.vars rmv.vars

`get.vars` *subsets data.frame according to header names*

Description

subsets `data` according to header names

Usage

```
get.vars(data, var.names)
```

Arguments

`data`	Object of class data.frame
`var.names`	A character vector of names to get from `data`

Value

An object of class data.frame

Author(s)

Luke A. Winslow

See Also

has.vars rmv.vars

`getSchmidt` *Returns Schmidt number for a specific gas at a given temperature*

Description

Schmidt number is temperature dependant, and is the ratio of the kinematic viscosity of water to a diffusion coefficient. Coefficients are included for He, O2, CO2, CH4, SF6, N2O, Ar, and N2.

Usage

```
getSchmidt(temperature, gas)
```

Arguments

temperature	Numeric vector of water temperatures in deg. Celsius
gas	String for gas code. Valid inputs include: He, O2, CO2, CH4, SF6, N2O, Ar, and N2

Value

Schmidt number (unitless)

Note

Temperature range is only valid from 4-35 deg Celsius

Author(s)

Jordan S. Read

References

Raymond, Peter A., Christopher J. Zappa, David Butman, Thomas L. Bott, Jody Potter, Patrick Mulholland, Andrew E. Laursen, William H. McDowell, and Denis Newbold. *Scaling the gas transfer velocity and hydraulic geometry in streams and small rivers*. Limnology & Oceanography: Fluids & Environments 2 (2012): 41-53.

Examples

```
getSchmidt(temperature=12, gas="O2")
```

---

has.vars *tests data.frame for column names*

Description

tests data for data column names

Usage

```
has.vars(data, var.names)
```

Arguments

data	Object of class data.frame
var.names	A character vector of names to test against data

Value

a boolean vector of same length as var.names

Author(s)

Luke A. Winslow

See Also

get.vars rmv.vars

`is.day` *determines if measurement was taken during the daytime*

Description

determines if measurement was taken during the daytime

Usage

```
is.day(datetimes, lat)
```

Arguments

`datetimes`	Vector of dates as `POSIXct` or `POSIXlt` (see `DateTimeClasses`) format
`lat`	Single latitude value of site. South should be negative, north positive

Value

a boolean vector of same length as `datetimes`

Author(s)

Luke A. Winslow

See Also

is.night sun.rise.set

is.night *determines if measurement was taken during the night*

Description

determines if measurement was taken during the nighttime

Usage

```
is.night(datetimes, lat)
```

Arguments

datetimes	Vector of dates as POSIXct or POSIXlt (see DateTimeClasses) format
lat	Single latitude value of site. South should be negative, north positive

Value

a boolean vector of same length as datetimes

Author(s)

Luke A. Winslow

See Also

is.day sun.rise.set

k.read *Returns a timeseries of gas exchange velocity*

Description

Returns the gas exchange velocity based on the chosen model in units of m/day

Usage

```
k.cole(ts.data)

k.crusius(ts.data, method='power')

k.read(ts.data, wnd.z, Kd, atm.press, lat, lake.area)

k.read.soloviev(ts.data, wnd.z, Kd, atm.press, lat, lake.area)

k.macIntyre(ts.data, wnd.z, Kd, atm.press,params=c(1.2,0.4872,1.4784))
```

```
k.vachon(ts.data, lake.area, params=c(2.51,1.48,0.39))

k.heiskanen(ts.data, wnd.z, Kd, atm.press)
```

Arguments

ts.data	vector of datetime in POSIXct format
wnd.z	height of wind measurement, m
Kd	Light attenuation coefficient (Units:m^-1)
atm.press	atmospheric pressure in mb
lat	Latitude, degrees north
lake.area	Lake area, m^2
method	Only for k.crusius. String of valid method . Either "linear", "bilinear", or "power"
params	Only for k.vachon.base and k.macIntyre. See details.

Details

Can change default parameters of MacIntyre and Vachon models. Default for Vachon is c(2.51,1.48,0.39). Default for MacIntyre is c(1.2,0.4872,1.4784). Heiskanen 2014 uses MacIntyre model with c(0.5,0.77,0.3) and z.aml constant at 0.15.

Value

Returns a data.frame with a datetime column and a k600 column. k600 is in units of meters per day (m/d).

Author(s)

Hilary Dugan, Jake Zwart, Luke Winslow, R. Iestyn. Woolway, Jordan S. Read

References

Cole, J., J. Nina, and F. Caraco. *Atmospheric exchange of carbon dioxide in a low-wind oligotrophic lake measured by the addition of SF~ 6*. Limnology and Oceanography 43 (1998): 647-656.

MacIntyre, Sally, Anders Jonsson, Mats Jansson, Jan Aberg, Damon E. Turney, and Scott D. Miller. *Buoyancy flux, turbulence, and the gas transfer coefficient in a stratified lake*. Geophysical Research Letters 37, no. 24 (2010).

Read, Jordan S., David P. Hamilton, Ankur R. Desai, Kevin C. Rose, Sally MacIntyre, John D. Lenters, Robyn L. Smyth et al. *Lake-size dependency of wind shear and convection as controls on gas exchange*. Geophysical Research Letters 39, no. 9 (2012).

Crusius, John, and Rik Wanninkhof. *Gas transfer velocities measured at low wind speed over a lake*. Limnology and Oceanography 48, no. 3 (2003): 1010-1017.

Dominic Vachon and Yves T. Prairie. *The ecosystem size and shape dependence of gas transfer velocity versus wind speed relationships in lakes*. Can. J. Fish. Aquat. Sci. 70 (2013): 1757-1764.

Jouni J. Heiskanen, Ivan Mammarella, Sami Haapanala, Jukka Pumpanen, Timo Vesala, Sally MacIntyre Anne Ojala. *Effects of cooling and internal wave motions on gas transfer coefficients in a boreal lake*. Tellus B 66, no.22827 (2014)

Alexander Soloviev, Mark Donelan, Hans Graber, Brian Haus, Peter Schlussel. *An approach to estimation of near-surface turbulence and CO2 transfer velocity from remote sensing data*. Journal of Marine Systems 66, (2007): 182-194.

See Also

k.cole k.crusius k.macIntyre k.vachon k.heiskanen

Examples

```
data.path = system.file('extdata', package="LakeMetabolizer")

tb.data = load.all.data('sparkling', data.path)

ts.data = tb.data$data #pull out just the timeseries data

#calculate U10 and add it back onto the original

u10 = wind.scale(ts.data)
ts.data = rmv.vars(ts.data, 'wnd', ignore.offset=TRUE) #drop old wind speed column
ts.data = merge(ts.data, u10)                          #merge new u10 into big dataset


k600_cole = k.cole(ts.data)

k600_crusius = k.crusius(ts.data)

kd        = tb.data$metadata$averagekd
wnd.z      = 10   #because we converted to u10
atm.press  = 1018
lat       = tb.data$metadata$latitude
lake.area = tb.data$metadata$lakearea

#for k.read and k.macIntyre, we need LW_net.
#Calculate from the observations we have available.

lwnet = calc.lw.net(ts.data, lat, atm.press)
ts.data = merge(ts.data, lwnet)

k600_read = k.read(ts.data, wnd.z=wnd.z, Kd=kd, atm.press=atm.press,
lat=lat, lake.area=lake.area)

k600_soloviev = k.read.soloviev(ts.data, wnd.z=wnd.z, Kd=kd,
atm.press=atm.press, lat=lat, lake.area=lake.area)

k600_macIntyre = k.macIntyre(ts.data, wnd.z=wnd.z, Kd=kd, atm.press=atm.press)
```

k.read.base *Returns a timeseries of gas exchange velocity*

Description

Returns the gas exchange velocity based on the chosen model in units of m/day

Usage

```
k.cole.base(wnd)

k.crusius.base(wnd, method='power')

k.read.base(wnd.z, Kd, lat, lake.area, atm.press, dateTime, Ts, z.aml,
airT, wnd, RH, sw, lwnet)

k.read.soloviev.base(wnd.z, Kd, lat, lake.area, atm.press, dateTime, Ts, z.aml,
airT, wnd, RH, sw, lwnet)

k.macIntyre.base(wnd.z, Kd, atm.press, dateTime, Ts, z.aml, airT, wnd, RH, sw,
lwnet, params=c(1.2,0.4872,1.4784))

k.vachon.base(wnd, lake.area, params=c(2.51,1.48,0.39))

k.heiskanen.base(wnd.z, Kd, atm.press, dateTime, Ts, z.aml, airT, wnd, RH, sw, lwnet)
```

Arguments

wnd.z	Height of wind measurement, (Units: m)
Kd	Light attenuation coefficient (Units: m^-1)
lat	Latitude, degrees north
lake.area	Lake area, m^2
atm.press	Atmospheric pressure, (Units: millibar)
dateTime	datetime (Y-%m-%d %H:%M), (Format: POSIXct)
Ts	Numeric vector of surface water temperature, (Units(deg C)
z.aml	Numeric vector of actively mixed layer depths. Must be the same length as the Ts parameter
airT	Numeric value of air temperature, Units(deg C)
wnd	Numeric value of wind speed, (Units:m/s)
RH	Numeric value of relative humidity, %
sw	Numeric value of short wave radiation, W m^-2
lwnet	Numeric value net long wave radiation, W m^-2

`method`	Only for k.crusius.base. String of valid method . Either "constant", "bilinear", or "power"
`params`	Optional parameter input, only for k.vachon.base and k.macIntyre.base. See details.

Details

Can change default parameters of MacIntyre and Vachon models. Default for Vachon is c(2.51,1.48,0.39). Default for MacIntyre is c(1.2,0.4872,1.4784). Heiskanen et al. (2014) uses MacIntyre model with c(0.5,0.77,0.3) and z.aml constant at 0.15.

Value

Numeric value of gas exchange velocity (k600) in units of m/day. Before use, should be converted to appropriate gas using k600.2.kGAS.

Author(s)

R. Iestyn. Woolway, Hilary Dugan, Luke Winslow, Jordan S Read, GLEON fellows

References

Cole, J., J. Nina, and F. Caraco. *Atmospheric exchange of carbon dioxide in a low-wind oligotrophic lake measured by the addition of SF~ 6*. Limnology and Oceanography 43 (1998): 647-656.

MacIntyre, Sally, Anders Jonsson, Mats Jansson, Jan Aberg, Damon E. Turney, and Scott D. Miller. *Buoyancy flux, turbulence, and the gas transfer coefficient in a stratified lake*. Geophysical Research Letters 37, no. 24 (2010).

Read, Jordan S., David P. Hamilton, Ankur R. Desai, Kevin C. Rose, Sally MacIntyre, John D. Lenters, Robyn L. Smyth et al. *Lake-size dependency of wind shear and convection as controls on gas exchange*. Geophysical Research Letters 39, no. 9 (2012).

Crusius, John, and Rik Wanninkhof. *Gas transfer velocities measured at low wind speed over a lake*. Limnology and Oceanography 48, no. 3 (2003): 1010-1017.

Dominic Vachon and Yves T. Prairie. *The ecosystem size and shape dependence of gas transfer velocity versus wind speed relationships in lakes*. Can. J. Fish. Aquat. Sci. 70 (2013): 1757-1764.

Jouni J. Heiskanen, Ivan Mammarella, Sami Haapanala, Jukka Pumpanen, Timo Vesala, Sally MacIntyre Anne Ojala. *Effects of cooling and internal wave motions on gas transfer coefficients in a boreal lake*. Tellus B 66, no.22827 (2014)

Alexander Soloviev, Mark Donelan, Hans Graber, Brian Haus, Peter Schlussel. *An approach to estimation of near-surface turbulence and CO2 transfer velocity from remote sensing data*. Journal of Marine Systems 66, (2007): 182-194.

See Also

k.cole k.read k.crusius k.macIntyre k.vachon k.heiskanen

Examples

```
wnd.z <- 2
Kd <- 2
lat <- 54
lake.area <- 5000
atm.press <- 1013
dateTime <- as.POSIXct("2013-12-30 14:00")
Ts <- 16.5
z.aml <- 2.32
airT <- 20
wnd <- 6
RH <- 90
sw <- 800
lwnet <- -55
timeStep <- 30

U10 <- wind.scale.base(wnd, wnd.z)

k600_cole <- k.cole.base(U10)

k600_crusius <- k.crusius.base(U10)

k600_read <- k.read.base(wnd.z, Kd, lat, lake.area, atm.press,
dateTime, Ts, z.aml, airT, wnd, RH, sw, lwnet)

k600_soloviev <- k.read.soloviev.base(wnd.z, Kd, lat, lake.area,
atm.press, dateTime, Ts, z.aml, airT, wnd, RH, sw, lwnet)

k600_macInytre <- k.macIntyre.base(wnd.z, Kd, atm.press,
dateTime, Ts, z.aml, airT, wnd, RH, sw, lwnet)
```

---

k600.2.kGAS *Returns the gas exchange velocity for gas of interest w/ no unit conversions*

Description

Returns the gas exchange velocity for gas of interest w/ no unit conversions

Usage

```
k600.2.kGAS.base(k600,temperature,gas="O2")

k600.2.kGAS(ts.data, gas="O2")
```

Arguments

k600	k600 as vector array of numbers or single number

temperature	Water temperature (deg C) as vector array of numbers or single number
gas	gas for conversion, as string (e.g., 'CO2' or 'O2')
ts.data	Object of class data.frame with named columns datetime and k600 and wtr (water temp in deg C). Other columns are ignored

Value

Numeric value of gas exchange velocity for gas

Author(s)

Jordan S. Read

See Also

k.read and k.read.base for functions that calculate k600 estimates

Examples

```
 ## single example
kO2 <- k600.2.kGAS.base(k600=2.4,temperature=20.4,gas='O2')

## Timeseries example
#load data
data.path = system.file('extdata', package="LakeMetabolizer")
sp.data = load.all.data('sparkling', data.path)
ts.data = sp.data$data #pull out just the timeseries data


#calculate U10 and add it back onto the original
u10 = wind.scale(ts.data)
ts.data = rmv.vars(ts.data, 'wnd', ignore.offset=TRUE) #drop old wind speed column
ts.data = merge(ts.data, u10)                          #merge new u10 into big dataset

k600 = k.cole(ts.data)
ts.data = merge(k600, ts.data)

k.gas = k600.2.kGAS(ts.data, 'O2')
```

---

load.all.data *Attemps to load and merge all timeseries data for a given site name*

Description

Loads and returns all the data available in the specified directory for a given site. All timeseries data are merged by "datetime" into a single data.frame. Data are identified by the column header information.

Usage

```
load.all.data(lake.name, data.path, checkMerge=TRUE)
```

Arguments

lake.name	The file prefix to be matched. For example, "sparkling" matches "sparkling.wtr" but not "troutbog.wtr"
data.path	The directory to look for files
checkMerge	Should check merge size before attempting to prevent potential merge problems.

Value

A list with two items

data

metadata

Author(s)

Luke A. Winslow

See Also

load.ts load.meta

---

load.meta *Loads a metadata file from the specified path*

Description

Parses a formatted metadata file. Useful for site-specific metadata that is not contained in the timeseries files.

Usage

```
load.meta(fPath)
```

Arguments

fPath	The file path as a string

Value

A list with the metadata parsed from the file.

Author(s)

Luke A Winslow

See Also

load.ts load.all.data

`metab` *Calculate metabolism*

Description

Returns daily time series of gross primary production (GPP), respiration (R), and net ecosystem production (NEP). Depending on the method used, other information may be returned as well. Calculations are made using one of 5 statistical methods.

Usage

```
metab(data, method, wtr.name="wtr", irr.name="irr", do.obs.name="do.obs", ...)
```

Arguments

`data`	a data.frame whose columns are `"datetime"` = class POSIXct vector `"do.obs"` = numeric vector of oxygen concentration in mg/L `"do.sat"` = numeric vector of saturated oxygen concentration in mg/L `"k.gas"` = numeric vector of gas exchange coefficient values in m/day, should be 0 when depth of do.obs is deeper than z.mix `"z.mix"` = numeric vector of mixing depth values in meters `"irr"` = numeric vector of PAR values, arbitrary units `"wtr"` = numeric vector of water temperature values, arbitrary units Columns that are not used by a particular statistical method do not need to be supplied.
`method`	a character string specifying one of the 5 statistical methods (bayesian, bookkeep, kalman, ols, mle)
`wtr.name`	the name of the column containing temperature at the depth of do.obs (predictor variable for R)
`irr.name`	the name of the column containing irradiance (predictor variable for GPP)
`do.obs.name`	the name of the column in data containing the DO observations (in mg/L) to be used as the response variable
`...`	arguments to be passed on to the metabolism model specified by `method`

Value

A data.frame containing columns for year, doy (day of year, julian day plus fraction of day), GPP, R, and NEP

`year`	integer year
`doy`	numeric, day of year + fraction of day, where the day is the julian day, and a fraction of 0.5 corresponds to noon
`GPP`	numeric, gross primary production, in units of mg O2 per liter per day. By convention, this value is positive.
`R`	numeric, respiration, in units of mg O2 per liter per day. By convention, this value is negative
`NEP`	numeric, net ecosystem production, in units of mg O2 per liter per day. For most methods this equal GPP+R, but this is not necessarily the case for `"method"="bookkeep"`

Note that different models will have different attributes attached to them. See examples.

Author(s)

Ryan D. Batt

See Also

Metabolism models: metab.bookkeep, metab.ols, metab.mle, metab.kalman, metab.bayesian

For smoothing noisy temperature: temp.kalman

To calculate do.sat: o2.at.sat

To calculate k.gas: k600.2.kGAS

To calculate k600 values for k.gas: k.cole, k.crusius, k.macIntyre, k.read

Examples

```
# fake data
datetime <- seq(as.POSIXct("2014-06-16 00:00:00", tz="GMT"),
     as.POSIXct("2014-06-17 23:55:00", tz="GMT"), length.out=288*2)
do.obs <- 2*sin(2*pi*(1/288)*(1:(288*2))+1.1*pi) + 8 + rnorm(288*2, 0, 0.5)
wtr <- 3*sin(2*pi*(1/288)*(1:(288*2))+pi) + 17 + rnorm(288*2, 0, 0.15)
do.sat <- LakeMetabolizer::o2.at.sat.base(wtr, 960)
irr <- (1500*sin(2*pi*(1/288)*(1:(288*2))+1.5*pi) +650 + rnorm(288*2, 0, 0.25)) *
     ifelse(is.day(datetime, 42.3), 1, 0)
k.gas <- 0.4
z.mix <- 1

# plot time series
plot(wtr, type="l", xaxt="n", yaxt="n", xlab="", ylab="")
par(new=TRUE); plot(do.obs, type="l", col="blue", xaxt="n", yaxt="n", xlab="", ylab="")
par(new=TRUE); plot(irr, type="l", col="orange", xaxt="n", yaxt="n", xlab="", ylab="")
abline(v=144, lty="dotted")
abline(v=288)
legend("topleft", legend=c("wtr", "do.obs", "irr"), lty=1,
```

```
     col=c("black", "blue", "orange"), inset=c(0.08, 0.01))

# put data in a data.frame
data <- data.frame(datetime=datetime, do.obs=do.obs, do.sat=do.sat, k.gas=k.gas,
     z.mix=z.mix, irr=irr, wtr=wtr)

# run each metabolism model
m.bk <- metab(data, "bookkeep", lake.lat=42.6)
 m.bk <- metab(data, lake.lat=42.6) # no method defaults to "bookeep"
m.ols <- metab(data, "ols", lake.lat=42.6)
m.mle <- metab(data, "mle", lake.lat=42.6)
m.kal <- metab(data, "kalman", lake.lat=42.6)
## Not run: m.bay <- metab(data, "bayesian", lake.lat=42.6)

# example attributes
names(attributes(m.ols))
attr(m.ols, "mod")

# To get full JAGS model
# including posterior draws:
## Not run: names(attributes(m.bay))
## Not run: attr(m.bay, "model")
```

`metab.bayesian` *Metabolism model based on a bayesian parameter estimation framework*

Description

This function runs the bayesian metabolism model on the supplied gas concentration and other supporting data. This allows for both estimates of metabolism along with uncertainty around the parameters.

Usage

```
metab.bayesian(do.obs, do.sat, k.gas, z.mix, irr, wtr, priors, ...)
```

Arguments

`do.obs`	Vector of dissovled oxygen concentration observations, mg L^-1
`do.sat`	Vector of dissolved oxygen saturation values based on water temperature. Calculate using o2.at.sat
`k.gas`	Vector of kGAS values calculated from any of the gas flux models (e.g., k.cole) and converted to kGAS using k600.2.kGAS
`z.mix`	Vector of mixed-layer depths in meters. To calculate, see ts.meta.depths
`irr`	Vector of photosynthetically active radiation in $\mu mol\ m^{-2}s^{-1}$
`wtr`	Vector of water temperatures in $^{\circ}C$. Used in scaling respiration with temperature

priors	Parameter priors supplied as a named numeric vector (example: c("gppMu"=0, "gppSig2"=1E5, "rMu"=0, "rSig2"=1E5, "kSig2"=NA))
...	additional arguments; currently "datetime" is the only recognized argument passed through ...

Value

A list of length 4 with components:

model	the jags model, including posterior draws (see jags)
params	parameter estimates of interest from model (medians)
metab.sd	standard deviation of metabolism estimates
metab	daily metabolism estimates as a data.frame with columns corresponding to
	GPP numeric estimate of Gross Primary Production, $mgO_2L^{-1}d^{-1}$
	R numeric estimate of Respiration, $mgO_2L^{-1}d^{-1}$
	NEP numeric estimate of Net Ecosystem production, $mgO_2L^{-1}d^{-1}$

Author(s)

Ryan Batt, Luke A. Winslow

References

Holtgrieve, Gordon W., Daniel E. Schindler, Trevor a. Branch, and Z. Teresa A'mar. 2010. *Simultaneous Quantification of Aquatic Ecosystem Metabolism and Reaeration Using a Bayesian Statistical Model of Oxygen Dynamics*. Limnology and Oceanography 55 (3): 1047-1062. doi:10.4319/lo.2010.55.3.1047. http://www.aslo.org/lo/toc/vol_55/issue_3/1047.html.

See Also

metab.mle, metab.bookkeep, metab.kalman

Examples

```
## Not run:
library(rLakeAnalyzer)

doobs = load.ts(system.file('extdata',
                            'sparkling.doobs', package="LakeMetabolizer"))
wtr = load.ts(system.file('extdata',
                          'sparkling.wtr', package="LakeMetabolizer"))
wnd = load.ts(system.file('extdata',
                          'sparkling.wnd', package="LakeMetabolizer"))
irr = load.ts(system.file('extdata',
                          'sparkling.par', package="LakeMetabolizer"))

#Subset a day
mod.date = as.POSIXct('2009-07-08', 'GMT')
doobs = doobs[trunc(doobs$datetime, 'day') == mod.date, ]
```

```
wtr = wtr[trunc(wtr$datetime, 'day') == mod.date, ]
wnd = wnd[trunc(wnd$datetime, 'day') == mod.date, ]
irr = irr[trunc(irr$datetime, 'day') == mod.date, ]

k600 = k.cole.base(wnd[,2])
k.gas = k600.2.kGAS.base(k600, wtr[,3], 'O2')
do.sat = o2.at.sat(wtr[,1:2], altitude=300)

metab.bayesian(irr=irr[,2], z.mix=rep(1, length(k.gas)),
              do.sat=do.sat[,2], wtr=wtr[,2],
              k.gas=k.gas, do.obs=doobs[,2])

## End(Not run)
```

`metab.bookkeep` *Metabolism model based on simple day/night summation NEP-interpreted changes in DO.*

Description

This model is a simple model based on the assumption that movements in DO during the day are due to NEP and gas exchange. Respiration is estimated from night-time decreases. GPP is calculated from the algebraic manipulation of NEP and R. Based on Cole et al 2000.

Usage

```
metab.bookkeep(do.obs, do.sat, k.gas, z.mix, irr, ...)
```

Arguments

`do.obs`	Vector of dissovled oxygen concentration observations, mg L^-1
`do.sat`	Vector of dissolved oxygen saturation values based on water temperature. Calculate using o2.at.sat
`k.gas`	Vector of kGAS values calculated from any of the gas flux models (e.g., k.cole) and converted to kGAS using k600.2.kGAS
`z.mix`	Vector of mixed-layer depths in meters. To calculate, see ts.meta.depths
`irr`	Integer vector of 1's (daytime) and 0's (nighttime), or numeric vector of irradiance that will be converted to boolean 1's and 0's if "datetime" is passed via `...`
`...`	additional arguments to be passed, particularly `POSIXct` class "datetime"

Value

A data.frame with columns corresponding to components of metabolism

GPP numeric estimate of Gross Primary Production, $mgO_2L^{-1}d^{-1}$

R numeric estimate of Respiration, $mgO_2L^{-1}d^{-1}$

NEP numeric estimate of Net Ecosystem production, $mgO_2L^{-1}d^{-1}$

Author(s)

R. Iestyn Woolway, Hilary Dugan, Luke A Winslow, Ryan Batt, Jordan S Read, GLEON fellows

References

Cole, Jonathan J., Michael L. Pace, Stephen R. Carpenter, and James F. Kitchell. 2000. *Persistence of Net Heterotrophy in Lakes during Nutrient Addition and Food Web Manipulations*. Limnology and Oceanography 45 (8): 1718-1730. doi:10.4319/lo.2000.45.8.1718.

See Also

metab.bayesian, metab.mle, metab.kalman

Examples

```
library(rLakeAnalyzer)
Sys.setenv(TZ='GMT')

doobs = load.ts(system.file('extdata',
                            'sparkling.doobs', package="LakeMetabolizer"))
wtr = load.ts(system.file('extdata',
                          'sparkling.wtr', package="LakeMetabolizer"))
wnd = load.ts(system.file('extdata',
                          'sparkling.wnd', package="LakeMetabolizer"))

#Subset a day
mod.date = as.POSIXct('2009-07-08', 'GMT')
doobs = doobs[trunc(doobs$datetime, 'day') == mod.date, ]
wtr = wtr[trunc(wtr$datetime, 'day') == mod.date, ]
wnd = wnd[trunc(wnd$datetime, 'day') == mod.date, ]

k.gas = k600.2.kGAS.base(k.cole.base(wnd[,2]), wtr[,3], 'O2')
do.sat = o2.at.sat.base(wtr[,3], altitude=300)

# Must supply 1 for daytime timesteps and 0 for nighttime timesteps
irr = as.integer(is.day(doobs[,1], 45))

metab.bookkeep(doobs[,2], do.sat, k.gas, z.mix=1, irr, datetime=doobs$datetime)
```

`metab.kalman` *Metabolism calculated from parameters estimated using a Kalman filter*

Description

A state space model accounting for process and observation error, with the maximum likelihood of parameteres estimated using a Kalman filter. Also provides a smoothed time series of oxygen concentration.

Usage

```
metab.kalman(do.obs, do.sat, k.gas, z.mix, irr, wtr, ...)
```

Arguments

do.obs	Vector of dissovled oxygen concentration observations, $mgO[2]L^{-1}$
do.sat	Vector of dissolved oxygen saturation values based on water temperature. Calculate using o2.at.sat
k.gas	Vector of kGAS values calculated from any of the gas flux models (e.g., k.cole) and converted to kGAS using k600.2.kGAS
z.mix	Vector of mixed-layer depths in meters. To calculate, see ts.meta.depths
irr	Vector of photosynthetically active radiation in $\mu mol\ m^{-2}s^{-1}$
wtr	Vector of water temperatures in $^{\circ}C$. Used in scaling respiration with temperature
...	additional arguments; currently "datetime" is the only recognized argument passed through ...

Details

The model has four parameters, c_1, c_2, Q, H, and consists of equations involving the prediction of upcoming state conditional on information of the previous state ($a_{t|t-1}$, $P_{t|t-1}$), as well as updates of those predictions that are conditional upon information of the current state ($a_{t|t}$, $P_{t|t}$). a is the

$$v = k.gas/z.mix$$

$$a_t = c_1 * irr_{t-1} + c_2 * log_e(wtr_{t-1}) + v_{t-1} * do.sat_{t-1}$$

$$beta = e^{-v}$$

$$do.obs_t = a_t/v_{t-1} + -e^{-v_{t-1}} * a_t/v_{t-1} + beta_{t-1} * do.obs_{t-1} + epsilon_t$$

The above model is used during model fitting, but if gas flux is not integrated between time steps, those equations simplify to the following:

$$F_{t-1} = k.gas_{t-1} * (do.sat_{t-1} - do.obs_{t-1})/z.mix_{t-1}$$

$$do.obs_t = do.obs_{t-1} + c_1 * irr_{t-1} + c_2 * log_e(wtr_{t-1}) + F_{t-1} + epsilon_t$$

The parameters are fit using maximum likelihood, and the optimization (minimization of the negative log likelihood function) is performed by `optim` using default settings.

GPP is then calculated as `mean(c1*irr, na.rm=TRUE)*freq`, where `freq` is the number of observations per day, as estimated from the typical size between time steps. Thus, generally `freq==length(do.obs)`.

Similarly, R is calculated as `mean(c2*log(wtr), na.rm=TRUE)*freq`.

NEP is the sum of GPP and R.

Value

A data.frame with columns corresponding to components of metabolism

GPP numeric estimate of Gross Primary Production, $mgO_2L^{-1}d^{-1}$

R numeric estimate of Respiration, $mgO_2L^{-1}d^{-1}$

NEP numeric estimate of Net Ecosystem production, $mgO_2L^{-1}d^{-1}$

Use attributes to access more model output:

`smoothDO`	smoothed time series of oxygen concentration ($mgO[2]L^{-1}$), from Kalman smoother
`params`	parameters estimated by the Kalman filter (c_1, c_2, Q, H)

Note

If observation error is substantial, consider applying a Kalman filter to the water temperature time series by supplying `wtr` as the output from temp.kalman

Author(s)

Ryan Batt, Luke A. Winslow

References

Batt, Ryan D. and Stephen R. Carpenter. 2012. *Free-water lake metabolism: addressing noisy time series with a Kalman filter*. Limnology and Oceanography: Methods 10: 20-30. doi: 10.4319/lom.2012.10.20

See Also

temp.kalman, watts.in, metab, metab.bookkeep, metab.ols, metab.mle, metab.bayesian

Examples

```
library(rLakeAnalyzer)
doobs <- load.ts(system.file('extdata',
                             'sparkling.doobs', package="LakeMetabolizer"))
wtr <- load.ts(system.file('extdata',
                           'sparkling.wtr', package="LakeMetabolizer"))
wnd <- load.ts(system.file('extdata',
                           'sparkling.wnd', package="LakeMetabolizer"))
irr <- load.ts(system.file('extdata',
                           'sparkling.par', package="LakeMetabolizer"))

#Subset a day
Sys.setenv(TZ='GMT')
mod.date <- as.POSIXct('2009-07-08', 'GMT')
doobs <- doobs[trunc(doobs$datetime, 'day') == mod.date, ]
wtr <- wtr[trunc(wtr$datetime, 'day') == mod.date, ]
wnd <- wnd[trunc(wnd$datetime, 'day') == mod.date, ]
irr <- irr[trunc(irr$datetime, 'day') == mod.date, ]

k600 <- k.cole.base(wnd[,2])
```

```
k.gas <- k600.2.kGAS.base(k600, wtr[,3], 'O2')
do.sat <- o2.at.sat.base(wtr[,3], altitude=300)

metab.kalman(irr=irr[,2], z.mix=rep(1, length(k.gas)),
             do.sat=do.sat, wtr=wtr[,2],
             k.gas=k.gas, do.obs=doobs[,2])
```

`metab.mle` *Metabolism calculated from the maximum likelihood estimates of the parameters in a standard linear regression model*

Description

Process-error-only model with parameters fitted via maximum likelihood estimation (MLE). This function runs the maximum likelihood metabolism model on the supplied gas concentration and other supporting data.

Usage

```
metab.mle(do.obs, do.sat, k.gas, z.mix, irr, wtr, error.type = "OE", ...)
```

Arguments

`do.obs`	Vector of dissolved oxygen concentration observations, $mgO[2]L^{-1}$
`do.sat`	Vector of dissolved oxygen saturation values based on water temperature. Calculate using o2.at.sat
`k.gas`	Vector of kGAS values calculated from any of the gas flux models (e.g., k.cole) and converted to kGAS using k600.2.kGAS
`z.mix`	Vector of mixed-layer depths in meters. To calculate, see ts.meta.depths
`irr`	Vector of photosynthetically active radiation in $\mu mol\ m^{-2}s^{-1}$
`wtr`	Vector of water temperatures in $^{\circ}C$. Used in scaling respiration with temperature
`error.type`	Option specifying if model should assume pure Process Error 'PE' or Observation Error 'OE'. Defaults to observation error 'OE'.
`...`	additional arguments; currently "datetime" is the only recognized argument passed through `...`

Details

The model has the three parameters, $c_1, c_2, epsilon$, and has the form

$$v = k.gas/z.mix$$

$$a_t = c_1 * irr_{t-1} + c_2 * log_e(wtr_{t-1}) + v_{t-1} * do.sat_{t-1}$$

$$beta = e^{-v}$$

$$do.obs_t = a_t/v_{t-1} + -e^{-v_{t-1}} * a_t/v_{t-1} + beta_{t-1} * do.obs_{t-1} + epsilon_t$$

The above model is used during model fitting, but if gas flux is not integrated between time steps, those equations simplify to the following:

$$F_{t-1} = k.gas_{t-1} * (do.sat_{t-1} - do.obs_{t-1})/z.mix_{t-1}$$

$$do.obs_t = do.obs_{t-1} + c_1 * irr_{t-1} + c_2 * log_e(wtr_{t-1}) + F_{t-1} + epsilon_t$$

The parameters are fit using maximum likelihood, and the optimization (minimization of the negative log likelihood function) is performed by `optim` using default settings.

GPP is then calculated as `mean(c1*irr, na.rm=TRUE)*freq`, where `freq` is the number of observations per day, as estimated from the typical size between time steps. Thus, generally `freq==length(do.obs)`.

Similarly, R is calculated as `mean(c2*log(wtr), na.rm=TRUE)*freq`.

NEP is the sum of GPP and R.

Value

A data.frame with columns corresponding to components of metabolism

GPP numeric estimate of Gross Primary Production, $mgO_2L^{-1}d^{-1}$

R numeric estimate of Respiration, $mgO_2L^{-1}d^{-1}$

NEP numeric estimate of Net Ecosystem production, $mgO_2L^{-1}d^{-1}$

The maximum likelihood estimates of model parameters can be accessed via `attributes(metab.mle(...))[["params"]]`

Note

Currently, missing values in any arguments will result in an error, so freq must always equal nobs.

Author(s)

Luke A Winslow, Ryan Batt, GLEON Fellows

References

Hanson, PC, SR Carpenter, N Kimura, C Wu, SP Cornelius, TK Kratz. 2008 *Evaluation of metabolism models for free-water dissolved oxygen in lakes*. Limnology and Oceanography: Methods 6: 454:465.

Solomon CT, DA Bruesewitz, DC Richardson, KC Rose, MC Van de Bogert, PC Hanson, TK Kratz, B Larget, R Adrian, B Leroux Babin, CY Chiu, DP Hamilton, EE Gaiser, S Hendricks, V Istvanovics, A Laas, DM O'Donnell, ML Pace, E Ryder, PA Staehr, T Torgersen, MJ Vanni, KC Weathers, G Zhuw. 2013. *Ecosystem Respiration: Drivers of Daily Variability and Background Respiration in Lakes around the Globe*. Limnology and Oceanography 58 (3): 849:866. doi:10.4319/lo.2013.58.3.0849.

See Also

metab, metab.bookkeep, metab.ols, metab.kalman, metab.bayesian

Examples

```
library(rLakeAnalyzer)
doobs = load.ts(system.file('extdata',
                            'sparkling.doobs', package="LakeMetabolizer"))
wtr = load.ts(system.file('extdata',
                          'sparkling.wtr', package="LakeMetabolizer"))
wnd = load.ts(system.file('extdata',
                          'sparkling.wnd', package="LakeMetabolizer"))
irr = load.ts(system.file('extdata',
                          'sparkling.par', package="LakeMetabolizer"))

#Subset a day
mod.date = as.POSIXct('2009-07-08', 'GMT')
doobs = doobs[trunc(doobs$datetime, 'day') == mod.date, ]
wtr = wtr[trunc(wtr$datetime, 'day') == mod.date, ]
wnd = wnd[trunc(wnd$datetime, 'day') == mod.date, ]
irr = irr[trunc(irr$datetime, 'day') == mod.date, ]
z.mix = ts.thermo.depth(wtr)

k600 = k.cole.base(wnd[,2])
k.gas = k600.2.kGAS.base(k600, wtr[,3], 'O2')
do.sat = o2.at.sat.base(wtr[,3], altitude=300)

metab.mle(doobs[,2], do.sat, k.gas, z.mix[,2], irr[,2], wtr[,3])
```

---

metab.ols *Metabolism model based on a ordinary least squares parameter estimation framework.*

Description

This function runs the ordinary least squares metabolism model on the supplied gas concentration and other supporting data. This is a common approach that allows for the concurrent estimation of metabolism paramters from a timeseries.

Usage

```
metab.ols(do.obs, do.sat, k.gas, z.mix, irr, wtr, ...)
```

Arguments

do.obs	Vector of dissolved oxygen concentration observations, mg L^-1
do.sat	Vector of dissolved oxygen saturation values based on water temperature. Calculate using o2.at.sat

k.gas	Vector of kGAS values calculated from any of the gas flux models (e.g., k.cole) and converted to kGAS using k600.2.kGAS
z.mix	Vector of mixed-layer depths in meters. To calculate, see ts.meta.depths
irr	Vector of photosynthetically active radiation in $\mu mol\ m^{-2}s^{-1}$
wtr	Vector of water temperatures in $°C$. Used in scaling respiration with temperature
...	additional arguments; currently "datetime" is the only recognized argument passed through ...

Value

A data.frame with columns corresponding to components of metabolism

GPP numeric estimate of Gross Primary Production, $mgO_2L^{-1}d^{-1}$

R numeric estimate of Respiration, $mgO_2L^{-1}d^{-1}$

NEP numeric estimate of Net Ecosystem production, $mgO_2L^{-1}d^{-1}$

Author(s)

Luke A Winslow, Ryan Batt, GLEON Fellows

See Also

metab, metab.bookkeep, metab.mle, metab.kalman, metab.bayesian,

Examples

```
library(rLakeAnalyzer)
doobs = load.ts(system.file('extdata',
                            'sparkling.doobs', package="LakeMetabolizer"))
wtr = load.ts(system.file('extdata',
                          'sparkling.wtr', package="LakeMetabolizer"))
wnd = load.ts(system.file('extdata',
                          'sparkling.wnd', package="LakeMetabolizer"))
irr = load.ts(system.file('extdata',
                          'sparkling.par', package="LakeMetabolizer"))

#Subset a day
mod.date = as.POSIXct('2009-07-08')
doobs = doobs[trunc(doobs$datetime, 'day') == mod.date, ]
wtr = wtr[trunc(wtr$datetime, 'day') == mod.date, ]
wnd = wnd[trunc(wnd$datetime, 'day') == mod.date, ]
irr = irr[trunc(irr$datetime, 'day') == mod.date, ]
z.mix = ts.thermo.depth(wtr)

k600 = k.cole.base(wnd[,2])
k.gas = k600.2.kGAS.base(k600, wtr[,3], 'O2')
do.sat = o2.at.sat.base(wtr[,3], altitude=300)

metab.ols(doobs[,2], do.sat, k.gas, z.mix[,2], irr[,2], wtr[,3])
```

`o2.at.sat` *Calculates the equilibrium saturation concentration of oxygen in water at the supplied conditions*

Description

Used to calculate the equilibrium concentration of oxygen in water. The equilibration concentration of oxygen in water varies with both temperature, salinity, and the partial pressure of oxygen in contact with the water (calculated from supplied elevation or barometric pressure).

Usage

```
o2.at.sat(ts.data, baro, altitude = 0, salinity = 0, model = "garcia-benson")

o2.at.sat.base(
  temp,
  baro,
  altitude = 0,
  salinity = rep(0, length(temp)),
  model = "garcia-benson"
)
```

Arguments

`ts.data`	Object of class data.frame with two named columns "datetime" and "wtr" (water temp in deg C).
`baro`	barometric pressure in millibars.
`altitude`	a numeric value indicating the elevation above mean sea level in meters. Defaults to mean sea level. An alternative to supplying barometric pressure.
`salinity`	a numeric vector of salinity in PSU. Defaults to zero. Length must be one or equal to length of temperature.
`model`	the empirical model to be used. `"garcia-benson"`, `"garcia"`, `"weiss"` and `"benson"` are the available options. `"garcia-benson"` is our current recommendation. The models correspond to the like-named references described below, where both `"garcia"` and `"garcia-benson"` are from Garcia & Gordon (1992).
`temp`	a numeric vector of water temperature in degrees Celsius.

Details

DO solubility is converted from mL/L to mg/L by multiplying by 1.42905, per USGS memo 2011.03. Corrections for vapor pressure are made according to barometric pressure as in Equations 2&3 of USGS memos 81.11 and 81.15. When barometric pressure is not supplied, it is estimated from altitude by the barometric formula as in Colt (2012).

Value

The equilibration concentration at the supplied conditions in mg/L of oxygen.

Author(s)

Luke A Winslow

References

Colt, John. *1 - Solubility of Atmospheric Gases in Freshwater.* In Computation of Dissolved Gas Concentration in Water as Functions of Temperature, Salinity and Pressure (Second Edition), edited by John Colt, 1-71. London: Elsevier, 2012. http://www.sciencedirect.com/science/article/pii/B9780124159167000012.

Garcia, H., and L. Gordon (1992), *Oxygen solubility in seawater: Better fitting equations*, Limnol. Oceanogr., 37(6).

Benson, B. B. & Krause, D. (1984). *The concentration and isotopic fractionation of oxygen dissolved in freshwater and seawater in equilibrium with the atmosphere.* Limnology and Oceanography, 29(3), 620-632. doi:10.4319/lo.1984.29.3.0620

Staehr, Peter A., Darren Bade, Matthew C. Van de Bogert, Gregory R. Koch, Craig Williamson, Paul Hanson, Jonathan J. Cole, and Tim Kratz. *Lake Metabolism and the Diel Oxygen Technique: State of the Science.* Limnology and Oceanography: Methods 8, no. 11 (November 1, 2010): 628-44. doi:10.4319/lom.2010.8.0628

USGS. *New Tables of Dissolved Oxygen Saturation Values.* Quality of Water Branch, 1981. http://water.usgs.gov/admin/mem

USGS. *New Tables of Dissolved Oxygen Saturation Values; Amendment of Quality of Water Technical Memorandum No. 81.11.* Quality of Water Branch, 1981. http://water.usgs.gov/admin/memo/QW/qw81.15.html.

USGS. *Change to Solubility Equations for Oxygen in Water.* Technical Memorandum 2011.03. USGS Office of Water Quality, 2011.

Weiss, R. (1970). *The solubility of nitrogen, oxygen and argon in water and seawater*. Deep Sea Research and Oceanographic Abstracts, 17(4), 721-735. doi:10.1016/0011-7471(70)90037-9

See Also

water.density, o2.at.sat.base

Examples

```
temp.range = 1:25
sal.range = 1:25

par(mfrow=c(1,2))
plot(temp.range, o2.at.sat.base(temp.range), xlab='Temperature (C)',
ylab='Oxygen Saturation (mg/L)')
plot(o2.at.sat.base(rep(20,25), salinity=sal.range), xlab='Salinity (PSU)', ylab='')
```

`par.to.sw` *Convert PAR to shortwave*

Description

Returns incoming shortwave radiation by converting PAR measuremt.

Usage

```
par.to.sw.base(par, coeff=0.473)

par.to.sw(data, par.col='par', coeff=0.473)
```

Arguments

`data`	Object of class data.frame with column name 'par' (units umol/m^2/sec)
`par.col`	String of alternative name for PAR column
`coeff`	Numerical coefficient to convert PAR (umol/m^2/sec) to SW (W/m^2). Defaults to value from Britton and Dodd (1976).
`par`	Numeric vector of PAR values (umol/m^2/sec)

Value

#For par.to.sw

Object of class data.frame with column name 'sw' and other values from `ts.data`

#For par.to.sw.base

Numeric vector of shortwave values with units W/m^2

Author(s)

LakeMetabolizer

References

Britton, C. M., and J. D. Dodd. *Relationships of photosynthetically active radiation and shortwave irradiance.* Agricultural Meteorology 17, no. 1 (1976): 1-7.

See Also

sw.to.par

Examples

```
par <- 800
par.to.sw.base(par)
```

rmv.vars *subsets data.frame according to header names*

Description

subsets data according to header names. Excludes all matches to var.name

Usage

```
rmv.vars(data, var.name, ignore.missing=TRUE, ignore.offset=FALSE)
```

Arguments

data	Object of class data.frame
var.name	A character vector of names to remove from data
ignore.missing	Boolean, should an error be thrown if no matching data found
ignore.offset	Should the numerical offset be ignored in the match, (e.g. all wtr columns removed, or wtr_0 specifically)

Value

An object of class data.frame

Author(s)

Luke A. Winslow

See Also

has.vars get.vars

sun.rise.set *Calculates the time of sunrise and sunset*

Description

Calculates the time of sunrise and sunset based on latitude and date.

Usage

```
sun.rise.set(datetimes, lat)
```

Arguments

datetimes	Vector of dates as POSIXct or POSIXlt (see DateTimeClasses) format
lat	Single latitude value of site. South should be negative, north positive

Value

A 2-column data frame, first column sunrise, second column sunset, as POSIXct format in standard time. Value is NA when there is no defined sunrise or sunset for that day (winter/summer at high and low latitudes).

Author(s)

Luke A. Winslow

References

Iqbal, Muhammad. 1983. An Introduction to Solar Radiation. Elsevier.

See Also

is.night is.day

Examples

```
sun.rise.set(lat=40.75,datetimes=as.POSIXlt('2013-03-31'))
```

`sw.to.par` *Convert shortwave radiation to PAR*

Description

Returns PAR by converting incoming shortwave radiation measuremt.

Usage

```
sw.to.par(data, sw.col='sw', coeff=2.114)

sw.to.par.base(sw, coeff=2.114)
```

Arguments

`data`	Object of class data.frame with column name `sw` (or specified alternate)
`sw.col`	Name of column containing shortwave data (units must be W/m^2)
`coeff`	Numerical coefficient to convert SW (W/m^2) to PAR (umol/m^2/sec). Defaults to value from Britton and Dodd (1976).
`sw`	Numeric shortwave value in W/m^2

Value

#For sw.to.par

Object of class data.frame with column name 'par' and other values from `ts.data`

#for sw.to.par.base

Numeric vector of PAR values in units umol/m^2/sec

Author(s)

Luke Winslow and others

References

Britton, C. M., and J. D. Dodd. *Relationships of photosynthetically active radiation and shortwave irradiance.* Agricultural Meteorology 17, no. 1 (1976): 1-7.

See Also

par.to.sw

Examples

```
#For base function
sw <- 800
sw.to.par.base(sw)
```

`temp.kalman` *Smooth temperature time series using a Kalman filter/ smoother*

Description

Smoothes a temperature time series uses a Kalman filter/ smoother.

Usage

```
temp.kalman(wtr, watts, ampH=1, ...)
```

Arguments

`wtr`	Vector (regular time series) of water temperature in degrees C
`watts`	estimate of watts entering the layer at each time step, from watts.in
`ampH`	factor by which to artificially amplify the observation error variance, H
`...`	parameters to be passed to optim

Details

basic model process is `x[t] = beta*x[t-1] + c1*watts[t-1]`

Value

a smoothed temperature time series

Author(s)

Ryan Batt

References

Batt, Ryan D. and Stephen R. Carpenter. 2012. *Free-water lake metabolism: addressing noisy time series with a Kalman filter*. Limnology and Oceanography: Methods 10: 20-30. doi: 10.4319/lom.2012.10.20

See Also

watts.in metab.kalman

`var.indx` *finds matching column names in data.frame*

Description

returns index of column matches for `data` according to header names matches with `var.names`.

Usage

```
var.indx(data, var.name)
```

Arguments

`data`	Object of class data.frame
`var.name`	A character vector of names to find matches with `data`

Value

a boolean vector with same length as `var.names`

Author(s)

Luke A. Winslow

See Also

has.vars get.vars rmv.vars

watts.in *Simple estimate of energy gained by a layer of water*

Description

Estimate the amount of energy gained by a layer of water as the difference between energy entering from the top of the layer and energy leaving at the bottom. Energy gained/ lost is calculated from photosynthetically active radiation (PAR, which is then converted to watts) and an estimate of kd (light attenuation coefficient) which is derived from the depth of 1 percent surface light.

Usage

```
watts.in(top, bot, irr, z1perc)
```

Arguments

top	Depth of the top of the layer, in meters
bot	Depth of the bottom of the layer, in meters
irr	PAR in uE/s (umol / m^2 / s)
z1perc	Depth of 1 percent of surface light, in meters

Details

This rough estimate is used in the Kalman filter/ smoother for water temperature. It does not account for a variety of potentially important factors, and is made specifically for use with temp.kalman(), which uses maximum likelihood to fit a linear coefficient that converts this heat gain estimate into temperature change.

Value

numeric vector of estimates of energy gain

Author(s)

Ryan Batt, Luke Winslow

References

Batt, Ryan D. and Stephen R. Carpenter. 2012. *Free-water lake metabolism: addressing noisy time series with a Kalman filter*. Limnology and Oceanography: Methods 10: 20-30. doi: 10.4319/lom.2012.10.20

See Also

temp.kalman metab.kalman

Examples

```
watts.in(3.2, 4, 1200, 4.5)
```

`wind.scale` *Wind Scaling U10 - exponential conversion to 10m wind speed*

Description

Scale wind speed to standard U10 (10 meters) based on height of observations

Usage

```
## Used for timeseries data in a data.frame
wind.scale(ts.data, wnd.z)

## Used for raw numeric data
wind.scale.base(wnd, wnd.z)
```

Arguments

`ts.data`	Object of class data.frame containing a wnd column.
`wnd.z`	height of anemometer (Units: meters)
`wnd`	measured wind speed (Units: typically m s-1, but it is unit agnostic)

Details

This function transforms wind speed to the standard U10, speed at 10 meters, based on the common exponential wind profile assumption. wind.scale defaults to using the supplied wnd.z value. If wnd.z is not supplied, it attempts to determine the anemometer height from the suffix of the header (e.g., a header of wnd_3 would mean an anemometer height of 3 meters).

Value

wind.scale Returns a data frame with columns datetime and wnd_10 and the same number of rows as ts.data

wind.scale.base Returns a vector with the same length as wnd

Author(s)

Aline Jaimes, Luke A. Winslow

References

Saucier, W. 2003. *Principles of Meteorological Analysis*. Dover Publications. New York. p433

See Also

Models of gas flux k.cole, k.crusius, k.macIntyre, & k.read.

Examples

```
wndSpeed <- c(5.1,6.3,6.3,5.2,7,7.2)
wndHeight <- 2

wind.scale.base(wndSpeed, wndHeight)
```

Index