Package ‘respirometry’

March 5, 2021

Type Package

Title Tools for Conducting and Analyzing Respirometry Experiments

Version 1.3.0

Date 2021-03-05

Author Matthew A. Birk

Maintainer Matthew A. Birk <matthewabirk@gmail.com>

Description Provides tools to enable the researcher to more precisely conduct respirometry experiments. Strong emphasis is on aquatic respirometry. Tools focus on helping the researcher setup and conduct experiments. Functions for analysis of resulting respirometry data are also provided. This package provides tools for intermittent, flow-through, and closed respirometry techniques.

Imports birk, graphics, lubridate, marelac, measurements (>= 1.1.0), methods, minpack.lm, seacarb (>= 3.1), segmented (>= 1.0-0), stats, utils

Depends PKNCA

License GPL-3

LazyData TRUE

Encoding UTF-8

RoxygenNote 7.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2021-03-05 20:50:02 UTC

R topics documented:

`calc_alpha` *Calculate the oxygen supply capacity (alpha)*

Description

The oxygen supply capacity (α) is a species- and temperature-specific value quantifying an animal's ability to extract oxygen from the ambient medium to support its metabolism (e.g. umol O2 / g / hr / kPa). This function calculates α based on the single highest $\alpha 0$ (MO2/PO2) value in the dataset. If there are outliers that make this prohibitive, consider setting a threshold MO2 value with `mo2_threshold`.

Usage

```
calc_alpha(po2, mo2, avg_top_n = 1, MR = NULL, mo2_threshold = Inf)
```

Arguments

- `po2` — a vector of PO2 values.
- `mo2` — a vector of metabolic rate values. Must be the same length and corresponding to `po2`.
- `avg_top_n` — a numeric value representing the number of top $\alpha 0$ (MO2/PO2) values to average together to estimate α. Default is 1. When analyzing a trial where the animal was not at MMR the whole time, we recommend no more than 3 to avoid diminishing the α value with sub-maximal observations. If all observations are believed to be at maximal O2 supply capacity, `Inf` can be used to average all observations.
- `MR` — a vector of values for the metabolic rate at which `pcrit_alpha` should be returned. Default is `NULL`. If not specified, then `pcrit_alpha` is not returned and a message is added to the top of the return.
- `mo2_threshold` — a single numeric value above which `mo2` values are ignored. Useful to removing obviously erroneous values. Default is `Inf`.

Value

Returns a list of 1) alpha, 2) a list of the PO2, MO2, and alpha0 value(s) where alpha was reached (the number of observations averaged is set by `avg_top_n`), and 3) the Pcrit at a metabolic rate of `MR`.

Author(s)

Matthew A. Birk, `<matthewabirk@gmail.com>`

References

Seibel, B. A., A. Andres, M. A. Birk, A. L. Burns, C. T. Shaw, A. W. Timpe, C. J. Welsh. 2021. "Oxygen supply capacity breathes new life into the critical oxygen partial pressure (Pcrit)." Journal of Experimental Biology.

See Also

`calc_pcrit`, `plot_pcrit`

Examples

```
mo2_data <- read.csv(system.file('extdata', 'mo2_v_po2.csv', package = 'respirometry'))
calc_alpha(po2 = mo2_data$po2, mo2 = mo2_data$mo2, MR = 1.5) # MR set to 1.5 to capture the
# Pcrit corresponding to some of the lowest MO2 values recorded (something close to SMR).

# extract the alpha0 values that were averaged together
sapply(calc_alpha(po2 = mo2_data$po2, mo2 = mo2_data$mo2,
  MR = 1.5, avg_top_n = 3)$alpha_obs, function(i) i[3])
```

calc_b *Calculate the metabolic scaling coefficient, b*

Description

For most organisms, metabolic rate does not scale linearly, but rather according to a power function: $MO2 = b0 * M^b$. This function estimates the scaling coefficient, b, and normalization constant, b0, given MO2s from different sized individuals.

Usage

```
calc_b(mass, MO2, method = "nls", plot = "linear", b0_start = 1)
```

Arguments

mass	a vector of animal masses.
MO2	a vector of metabolic rates.
method	a string defining which method of calculating scaling coefficients to use. Default is "nls", which utilizes a nonlinear least squares regression. If this does not fit your data well, "lm" may also be used, which calculates a linear regression of log10(MO2) ~ log10(mass) with slope and intercept equivalent to b and 10^b0, respectively.
plot	a string defining what kind of plot to display. "linear" for linear axes, "log" for log10-scale axes, and "none" for no plot. Default is "linear".
b0_start	a single numeric value as the starting point for b0 value determination using the "nls" method. The default is 1 and should work for most situations, but if the "nls" method is not working and you don't want to use the "lm" method, changing the starting b0 value may help. Ignored when method = "lm".

Details

$$MO2 = b0 * M^b$$

where b0 is species-specific normalization constant, M is mass and b is the scaling coefficient.

Value

Returns a list of 1) the b value, 2) a vector of b0 values corresponding to the input MO2 values, and 3) an average b0 that can be used for summarizing the relationship with an equation.

Author(s)

Matthew A. Birk, <matthewabirk@gmail.com>

See Also

scale_MO2, calc_MO2

Examples

```
# Simple example
mass <- c(1, 10, 100, 1000, 40, 4, 400, 60, 2, 742, 266, 983) # made up values
MO2 <- mass ^ 0.65 + rnorm(n = length(mass)) # make up some data
calc_b(mass = mass, MO2 = MO2)

# How about some mass-specific MO2s?
msMO2 <- mass ^ -0.25 + rnorm(n = length(mass), sd = 0.05)
calc_b(mass = mass, MO2 = msMO2)
calc_b(mass = mass, MO2 = msMO2, plot = "log")
```

---

calc_E — *Calculate E temperature coefficient*

Description

An E value is a relatively recent metric to parameterize the temperature-sensitivity of a biological rate (MO2). It is similar conceptually (but not numerically) to Q10.

Usage

```
calc_E(x, temp)
```

Arguments

`x`	a numeric vector of rate values (e.g. MO2).
`temp`	a numeric vector of temperature values (in Celsius).

Details

E is the slope of the relationship between `-ln(x)` and `1/(kB T)`, where `kB` is the Boltzmann constant expressed in eV/K.

Author(s)

Matthew A. Birk, `<matthewabirk@gmail.com>`

References

Deutsch, Curtis et al. 2015. “Climate Change Tightens a Metabolic Constraint on Marine Habitats.” Science 348(6239): 1132–35.

See Also

Q10

Examples

```
calc_E(x = c(1, 2, 3), temp = c(10, 20, 30))
```

---

calc_MO2 *Calculate metabolic rate*

Description

Calculates metabolic rate (MO2) given O2 measurements over time. Oxygen measurements are split into bins and MO2s are calculated from each bin (unless `bin_width` is set to `0`). The `bin_width` parameter defines the width of the bins in timed intervals (e.g. 15 minutes). Linear regressions are fit through each bin and the calculated MO2 is returned as the slope of the change in O2 over time.

Usage

```
calc_MO2(
  duration,
  o2,
  o2_unit = "percent_a.s.",
  bin_width,
  vol,
  temp = 25,
  sal = 35,
  atm_pres = 1013.25,
  time,
  pH,
  good_data = TRUE
)
```

Arguments

`duration`	numeric vector of the timepoint for each observation (minutes).
`o2`	numeric vector of O2 observations.
`o2_unit`	a string describing the unit used to measure `o2`. Default is "percent_a.s." Options are from `conv_o2`.
`bin_width`	numeric or data frame. OPTION 1: A single number defining how long of a period should be binned for each MO2 determination (minutes). If MO2 is to be calculated from one observation to the next (rather than binned observations), set `bin_width` to 0. To calculate a single MO2 value from all observations, set `bin_width` to `Inf`. OPTION 2: A data frame with two numeric columns: "o2" and "width" generated by `make_bins`. Useful for Pcrit calculations or another application where bins of different widths are desired at different PO2s. For each row, set the "width" value to the bin duration (minutes) desired for observations <= the value in the "o2" column.
`vol`	volume of the respirometer (liter).

`temp`	temperature (°C). Default is 25 °C.
`sal`	salinity (psu). Default is 35 psu.
`atm_pres`	atmospheric pressure (mbar). Default is 1013.25 mbar.
`time`	(optional). Numeric vector of timestamp observations.
`pH`	(optional). Numeric vector of pH observations.
`good_data`	logical vector of whether O2 observations are "good" measurements and should be included in analysis. Linear regressions will not be fit over bins that include "bad" data. Bins will be split at bad data points. Default is that all observations are `TRUE`.

Value

A data frame is returned:

DUR_MEAN Mean duration of the bin (minutes).

DUR_RANGE Range of duration timepoints in the bin.

TIME_MEAN Exists only if the parameter `time` has values. Mean timestamp of the bin.

TIME_RANGE Exists only if the parameter `time` has values. Range of timestamps in the bin.

TEMP_MEAN Mean temperature of the bin.

PH_MEAN Exists only if the parameter `pH` has values. Mean pH of the bin. Averaged using `mean_pH()`.

O2_MEAN Mean O2 value of the bin in the unit chosen by `o2_unit`).

O2_RANGE Range of O2 values in the bin.

MO2 Metabolic rate (umol O2 / hour).

R2 Coefficient of determination for the linear regression fit to calculate MO2.

N Number of observations in the bin.

Note

Whole-animal MO2 is returned. If mass-specific MO2 is desired, the output from `calc_MO2` can be divided by the animal's mass. If only beginning and ending O2 observations are known, consider using `closed`. Both functions will work fine, but `closed` is simpler.

Author(s)

Matthew A. Birk, <matthewabirk@gmail.com>

See Also

`make_bins`, `calc_b`, `closed`, `scale_MO2`, `conv_resp_unit`

Examples

```
# get O2 data
file <- system.file('extdata', 'witrox_file.txt', package = 'respirometry')
o2_data <- na.omit(import_witrox(file, split_channels = TRUE)$CH_4)

# calculate MO2
(mo2_5_min <- calc_MO2(duration = o2_data$DURATION, o2 = o2_data$O2,
bin_width = 5, vol = 10, temp = o2_data$TEMP, sal = o2_data$SAL))

# what if measurements from the 10 to 12 minute marks can't be trusted?
bad_data = o2_data$DURATION >= 10 & o2_data$DURATION <= 12
(mo2_5_min <- calc_MO2(duration = o2_data$DURATION, o2 = o2_data$O2,
bin_width = 5, vol = 10, temp = o2_data$TEMP, sal = o2_data$SAL, good_data = !bad_data))

# easily make a Pcrit plot
plot(mo2_5_min$O2_MEAN, mo2_5_min$MO2)

# I want to express MO2 in mg per min instead.
(mo2_5_min$MO2 <- conv_resp_unit(value = mo2_5_min$MO2, from = 'umol_O2 / hr', to = 'mg_O2 / min'))

# just endpoint measurement:
calc_MO2(duration = o2_data$DURATION, o2 = o2_data$O2,
bin_width = Inf, vol = 10, temp = o2_data$TEMP, sal = o2_data$SAL)

# In my trial, observations above 77% air saturation were really noisy, but much less noisy at
# lower O2 values. I want to adjust my bin width based on the PO2 to obtain the best balance of
# resolution and precision throughout the whole trial. Below 77% a.s., use 4 minute bins. Above
# 77% a.s. use 10 minute bins.
bins = data.frame(o2 = c(77, 100), width = c(4, 10))
calc_MO2(duration = o2_data$DURATION, o2 = o2_data$O2,
bin_width = bins, vol = 10, temp = o2_data$TEMP, sal = o2_data$SAL)
```

---

calc_pcrit *Calculate Pcrit*

Description

Calculates Pcrit (commonly understood as the threshold below which oxygen consumption rate can no longer be sustained) based on paired PO2 and MO2 values. Five Pcrit metrics are returned using many of the popular techniques for Pcrit calculation: the traditional breakpoint metric (broken stick regression), the nonlinear regression metric (Marshall et al. 2013), the sub-prediction interval metric (Birk et al. 2019), the alpha-based Pcrit method (Seibel et al. 2021), and the linear low O2 (LLO) method (Reemeyer & Rees 2019). To see the Pcrit values plotted, see plot_pcrit.

Usage

```
calc_pcrit(
  po2,
```

```
  mo2,
  avg_top_n = 1,
  level = 0.95,
  iqr = 1.5,
  NLR_m = 0.065,
  MR = NULL,
  mo2_threshold = Inf,
  return_models = FALSE
)
```

Arguments

- `po2` a vector of PO2 values. Any unit of measurement should work, but the NLR calculation was optimized using kPa. If the NLR metric is giving you trouble, try converting to kPa using `conv_o2`.
- `mo2` a vector of metabolic rate values. Must be the same length and corresponding to `po2`.
- `avg_top_n` applies to the `alpha` metric only. A numeric value representing the number of top $\alpha 0$ (MO2/PO2) values to average together to estimate α. Default is 1. We recommend no more than 3 to avoid diminishing the α value with sub-maximal observations.
- `level` applies to the `Sub_PI` metric only. Percentage at which the prediction interval should be constructed. Default is 0.95.
- `iqr` applies to the `Sub_PI` metric only. Removes `mo2` observations that are this many interquartile ranges away from the mean value for the oxyregulating portion of the trial. If this filtering is not desired, set to infinity. To visualize which observations will be removed by this parameter, use `plot_pcrit`. Default is 1.5.
- `NLR_m` applies to the `NLR` metric only. Pcrit is defined as the PO2 at which the slope of the best fitting function equals `NLR_m` (after the MO2 data are normalized to the 90% quantile). Default is 0.065.
- `MR` applies to the `alpha` and `LLO` metrics only. A numeric value for the metabolic rate at which `pcrit_alpha` and `pcrit_LLO` should be returned. If not supplied by the user, then the mean MO2 of the "oxyregulating" portion of the curve is applied for `pcrit_alpha` and `NA` is returned for `pcrit_LLO`.
- `mo2_threshold` applies to the `alpha` metric only. A single numeric value above which `mo2` values are ignored for `alpha` Pcrit estimation. Useful to removing obviously erroneous values. Default is `Inf`.
- `return_models` logical. Should a list of model parameters be returned along with the converged Pcrit values? Default is `FALSE`.

Details

Alpha Pcrit Alpha is calculated from `calc_alpha` and the Pcrit corresponding to `MR` is returned. This determine's the animal's oxygen supply capacity and calculates the Pcrit at any given metabolic rate of interest. If no `MR` is provided, then it defaults to the mean MO2 value from the oxyregulating portion of the curve (as defined by the broken-stick regression).

Breakpoint Pcrit Data are fit to a broken-stick regression using segmented.

LLO Pcrit A subset of observations are chosen only from those with an MO2 < MR. Then, a linear model is fit through the observations and Pcrit is calculated as the PO2 at which the line reaches MR.

NLR Pcrit Data are fit to the following functions: Michaelis-Menten, Power, Hyperbola, Pareto, and Weibull with intercept. Following the method developed by Marshall et al. 2013, the function that best fits the data (smallest AIC) is chosen and the Pcrit is determined as the PO2 at which the slope of the function is NLR_m (by default = 0.065 following the authors' suggestion).

Sub_PI Pcrit This metric builds off the Breakpoint metric and results in a systematically lower Pcrit value. This is useful for applications where it is important to ensure that Pcrit is not being overestimated. It represents a reasonable lower bounded estimate of the Pcrit value for a given trial. Once the Breakpoint Pcrit is calculated, a 95% prediction interval (can be changed with the level argument) is calculated around the oxyregulating region (i.e. using PO2 values > breakpoint Pcrit). By default, iqr provides some filtering of abberant observations to prevent their influence on the calculated prediction interval. Finally, the Sub_PI Pcrit value is returned at the intersection of the oxyconforming line and the lower limit of the oxyregulating prediction interval.

Value

If return_models is FALSE (default), a named numeric vector of Pcrit values calculated using the Alpha, Breakpoint, LLO, NLR, and Sub_PI metrics. If return_models is TRUE, then a list of converged Pcrit values, along with breakpoint function parameters, the MR value used for calculating Pcrit-alpha, a data frame of the "oxyregulating" portion of the curve, and NLR parameters are returned.

Author(s)

Matthew A. Birk, <matthewabirk@gmail.com>

References

Birk, Matthew A., K.A.S. Mislan, Karen F. Wishner, and Brad A. Seibel. 2019. "Metabolic Adaptations of the Pelagic Octopod Japetella Diaphana to Oxygen Minimum Zones." Deep-Sea Research Part I 148: 123–31.

Marshall, Dustin J., Michael Bode, and Craig R. White. 2013. "Estimating Physiological Tolerances - a Comparison of Traditional Approaches to Nonlinear Regression Techniques." Journal of Experimental Biology 216(12): 2176–82.

Reemeyer, Jessica E., and Bernard B. Rees. 2019. "Standardizing the Determination and Interpretation of Pcrit in Fishes." Journal of Experimental Biology 222(18): jeb210633.

Seibel, B. A., A. Andres, M. A. Birk, A. L. Burns, C. T. Shaw, A. W. Timpe, C. J. Welsh. 2021. "Oxygen supply capacity breathes new life into the critical oxygen partial pressure (Pcrit)." Journal of Experimental Biology.

See Also

plot_pcrit, calc_MO2, conv_o2, calc_alpha

Examples

```
mo2_data <- read.csv(system.file('extdata', 'mo2_v_po2.csv', package = 'respirometry'))
calc_pcrit(po2 = mo2_data$po2, mo2 = mo2_data$mo2)
```

---

closed *Closed respirometry*

Description

Returns the unknown parameter given 3 of 4 parameters to calculate respiration rate in a closed respirometer. This is useful both for basic closed respirometry setups, and also for the closed measurement phase of intermittent respirometry.

Usage

```
closed(MO2, delta_pO2, duration, vol, temp = 25, sal = 35, atm_pres = 1013.25)
```

Arguments

`MO2`	whole-animal oxygen consumption rate (umol O2 / hour).
`delta_pO2`	desired change in pO2 (% air saturation).
`duration`	desired duration to reach `delta_pO2` (minutes).
`vol`	volume of the respirometer (liter).
`temp`	temperature (°C). Default is 25 °C.
`sal`	salinity (psu). Default is 35 psu.
`atm_pres`	atmospheric pressure (mbar). Default is 1013.25 mbar.

Note

If there are more than two O2 observations, consider using `calc_MO2`.

Author(s)

Matthew A. Birk, <matthewabirk@gmail.com>

See Also

`flush_water`, `calc_MO2`

Examples

```
# I've read in the literature that my animal has an SMR of 200 umol/h. How large of a
# respirometer do I want if I want it to breathe down to 80% air saturation in 30 minutes?
closed(MO2 = 200, delta_pO2 = 100 - 80, duration = 30) # returns respirometer volume

# I've read in the literature that my animal has an SMR of 1000 umol/h. How long will it take to
# breathe down a 50 L respirometer by 10% air saturation?
closed(MO2 = 1000, delta_pO2 = 10, vol = 50) # returns the duration to breathe down the O2

# How does animal size affect how long my measurement periods last?
mass_range <- seq(100, 400, 50)
dur_range <- (closed(MO2 = scale_MO2(mass_1 = 100, MO2_1 = 400, mass_2 = mass_range),
 delta_pO2 = 20, vol = 10))
plot(mass_range, dur_range, type = 'b')

# What is the MO2 if O2 drops 0.44 mg/l in 33 minutes when the respirometer volume is 30 L?
closed(delta_pO2 = conv_o2(o2 = 0.44, from = 'mg_per_l', to = 'percent_a.s.'), duration = 33,
 vol = 30)
```

---

co2_add *Calculate CO2 to add to water*

Description

Calculates the moles of CO2 gas to be added to a volume of seawater to achieve the desired pCO2. Useful for ocean acidification experiments where CO2 treatments are desired.

Usage

```
co2_add(
  goal_pco2,
  start_pH,
  vol,
  temp = 25,
  sal = 35,
  TA = NULL,
  atm_pres = 1013.25
)
```

Arguments

`goal_pco2`	the desired pCO2 in the water (uatm).
`start_pH`	pH of the water before CO2 is added (total scale).
`vol`	volume of the water (liter).
`temp`	temperature (°C). Default is 25 °C.
`sal`	salinity (psu). Default is 35 psu. If `sal` < 26 psu, then `TA` must be provided.

TA	(optional) total alkalinity (umol / kg). If undefined TA is estimated from salinity using guess_TA.
atm_pres	atmospheric pressure (mbar). Default is 1013.25 mbar.

Value

moles of CO_2 gas to be added to the seawater.

Note

It is assumed that all of the CO_2 added dissolves and remains in solution. This can be achieved (almost completely) by bubbling CO_2 according to Jokiel et al. 2014.

Author(s)

Matthew A. Birk, <matthewabirk@gmail.com>

References

Jokiel PL, Bahr KD, Rodgers KS. 2014. Low-cost, high-flow mesocosm system for simulating ocean acidification with CO2 gas. Limnol Oceanogr Methods. 12:313–322.

See Also

co2_rate, flush_carb, carb, peri_pump

Examples

```
# I want the 50 L reservoir to have a pCO2 = 1000 uatm. It currently has a pH of 7.88.
# How many moles of CO2 gas should be added to the water to reach my desired pCO2?
co2_add(goal_pco2 = 1000, start_pH = 7.88, vol = 50)
```

---

co2_flush *Calculate CO2 to add to flush reservoir*

Description

Calculates the moles of CO_2 gas to be added to a seawater reservoir before flushing a respirometer to achieve the desired pCO_2 in the respirometer after the flush. Useful for ocean acidification experiments where CO_2 treatments are desired.

Usage

```
co2_flush(
  goal_pco2,
  resp_pH,
  resp_vol,
  flush_pH,
  flush_vol,
  flush_remain = 0,
  temp = 25,
  sal = 35,
  TA = NULL,
  atm_pres = 1013.25
)
```

Arguments

`goal_pco2`	the desired pCO2 in the respirometer after the flush (uatm).
`resp_pH`	pH inside the respirometer before the flush (total scale).
`resp_vol`	volume of the respirometer (liter).
`flush_pH`	pH of the reservoir water used for flushing before CO2 is added (total scale).
`flush_vol`	volume of the flush reservoir (liter).
`flush_remain`	volume of the flush reservoir that will remain after the flush (liter).
`temp`	temperature (°C). Default is 25 °C.
`sal`	salinity (psu). Default is 35 psu. If `sal` < 26 psu, then `TA` must be provided.
`TA`	(optional) total alkalinity (umol / kg). If undefined TA is estimated from salinity using `guess_TA`.
`atm_pres`	atmospheric pressure (mbar). Default is 1013.25 mbar.

Value

moles of CO2 gas to be added to the flush reservoir.

Note

It is assummed that the entire reservoir is drained into the respirometer during the flush. It is also assumed that all of the CO2 added dissolves and remains in solution. This can be achieved (almost completely) by bubbling CO2 according to Jokiel et al. 2014.

Author(s)

Matthew A. Birk, <matthewabirk@gmail.com>

References

Jokiel PL, Bahr KD, Rodgers KS. 2014. Low-cost, high-flow mesocosm system for simulating ocean acidification with CO2 gas. Limnol Oceanogr Methods. 12:313–322.

See Also

co2_add, co2_rate, flush_carb, carb, peri_pump

Examples

```
# I want the respirometer to have a pCO2 = 1000 uatm. It currently has a pH of 7.6 and is 90 L.
# If I have a 200 L reservoir with pH = 7.9 seawater, how much CO2 do I need
# to add to the flush reservoir?
co2_flush(goal_pco2 = 1000, resp_pH = 7.6, resp_vol = 90, flush_pH = 7.9, flush_vol = 200)
```

---

co2_rate *Calculate CO2 to add to a respirometer intake flow*

Description

Calculates the moles of CO2 gas to be added to a respirometer intake seawater flow to achieve the desired pCO2 in the respirometer. Useful for ocean acidification experiments where CO2 treatments are desired. Can be used for acclimation before a trial begins or for use with flow-through respirometry.

Usage

```
co2_rate(
  goal_pco2,
  init_pH,
  flow_rate,
  temp = 25,
  sal = 35,
  TA = NULL,
  atm_pres = 1013.25,
  MO2 = NULL,
  RQ = 1
)
```

Arguments

goal_pco2	the desired pCO2 in the respirometer (uatm).
init_pH	ambient pH of the intake flow (total scale).
flow_rate	rate of water flow into the respirometer (liters / minute).
temp	temperature (°C). Default is 25 °C.
sal	salinity (psu). Default is 35 psu. If sal < 26 psu, then TA must be provided.
TA	(optional) total alkalinity (umol / kg). If undefined TA is estimated from salinity using guess_TA.
atm_pres	atmospheric pressure (mbar). Default is 1013.25 mbar.

MO2	(optional) oxygen consumption rate (umol / hr). If defined, the CO2 to be added is reduced to compensate for the CO2 produced by the organism.
RQ	(optional) respiratory quotient: ratio of CO2 produced / O2 consumed. Only used if MO2 is defined. Default is 1.

Value

moles of CO2 gas to be added to the intake flow per minute.

Note

It is assumed that all of the CO2 added dissolves and remains in solution. This can be achieved (almost completely) by bubbling CO2 according to Jokiel et al. 2014.

Author(s)

Matthew A. Birk, <matthewabirk@gmail.com>

References

Jokiel PL, Bahr KD, Rodgers KS. 2014. Low-cost, high-flow mesocosm system for simulating ocean acidification with CO2 gas. Limnol Oceanogr Methods. 12:313–322.

See Also

co2_add, flush_carb, carb, peri_pump

Examples

```
# I want the respirometer to have a pCO2 = 1000 uatm. How much CO2 per minute do I need
# to add to the intake flow if the ambient pH is 8.1 and it is flowing at 3 LPM?
co2_rate(goal_pco2 = 1000, init_pH = 8.1, flow_rate = 3)
```

---

conv_nh4 *Convert between units of ammonia (NH3) / ammonium (NH4+)*

Description

Ammonia or nitrogen excretion can be measured in a variety of ways. Convert between different measurements.

Usage

```
conv_nh4(n_waste, from = "umol_NH4", to = "all")
```

Arguments

n_waste	a numeric vector of the ammonia or nitrogen value(s).
from	a string describing the unit used to measure `n_waste`. Default is "umol_NH4" Options are: • umol_NH3 • umol_NH4 • mg_NH3 • mg_NH4 • mg_N
to	a single string either describing the unit to which the conversion should be conducted (options are the same as in `from`), or the string "all" to return all units.

Details

The sum of NH4+ and NH3 species are considered. Conversions are based on relationships and values from the package `marelac`.

Author(s)

Matthew A. Birk, `<matthewabirk@gmail.com>`

See Also

`predict_nh3`, `conv_o2`

Examples

```
conv_nh4(n_waste = 100)
conv_nh4(n_waste = 100, from = 'mg_N')
conv_nh4(n_waste = 100, from = 'mg_N', to = 'umol_NH4')
```

---

conv_o2 — *Convert between units of oxygen partial pressure and concentration*

Description

Unfortunately, a consensus on the best way to express how much oxygen is in water has not been formed to date. Until then, this function converts between all commonly used forms of dissolved O2 measurements.

Usage

```
conv_o2(
  o2 = 100,
  from = "percent_a.s.",
  to = "all",
  temp = 25,
  sal = 35,
  atm_pres = 1013.25
)
```

Arguments

`o2`	a numeric vector of the O2 value(s). Default is 100.
`from`	a string describing the unit used to measure `o2`. Default is "percent_a.s." Options are: • percent_a.s. (percent air saturation) • percent_o2 • hPa • kPa • torr • mmHg • inHg • mg_per_l • ug_per_l • umol_per_l • mmol_per_l • ml_per_l • mg_per_kg • ug_per_kg • umol_per_kg • mmol_per_kg • ml_per_kg • volumes_percent
`to`	a single string either describing the unit to which the conversion should be conducted (options are the same as in `from`), or the string "all" to return all units.
`temp`	temperature (°C). Default is 25 °C.
`sal`	salinity (psu). Default is 35 psu.
`atm_pres`	atmospheric pressure (mbar). Default is 1013.25 mbar.

Details

Conversions are based on relationships and values from the package `marelac` which utilizes saturation values from Weiss 1970.

Author(s)

Matthew A. Birk, <matthewabirk@gmail.com>

References

Weiss R. 1970. The solubility of nitrogen, oxygen, and argon in water and seawater. Deep-Sea Research. 17:721-735.

Examples

```
conv_o2(o2 = 50)
conv_o2(o2 = 1:50, from = "umol_per_l", to = "ml_per_l", temp = 10, sal = 0,
atm_pres = 1100)
conv_o2()[c('mmHg','kPa')]
```

`conv_resp_unit` *Convert units related to respirometry*

Description

Converts units of measurement that are joined by " / " or " * ". This function expands upon `conv_multiunit` to incorporate O2 unit conversion and seawater volume-mass conversions.

Usage

```
conv_resp_unit(
  value,
  from,
  to,
  temp = 25,
  sal = 35,
  atm_pres = 1013.25,
  o2_conc_base = "per_l"
)
```

Arguments

`value`	a numeric vector giving the measurement value in its original units.
`from, to`	a string defining the unit with subunits separated by " / " or " * ". See Details for proper notation regarding O2 and seawater mass/volume.
`temp`	temperature (°C). Default is 25 °C.
`sal`	salinity (psu). Default is 35 psu.
`atm_pres`	atmospheric pressure (mbar). Default is 1013.25 mbar.
`o2_conc_base`	(optional) if converting between pO2 and [O2], should concentrations be "per_l" or "per_kg"? Default is "per_l".

Details

The O2 units supported by conv_o2 should be appended with "_O2" (e.g. "kPa_O2"; even "percent_o2_O2") and O2 unit concentrations should drop "per_l" or "per_kg" (e.g. "umol_O2"). To designate seawater mass-volume conversion, append the unit with "_seawater" (e.g. "kg_seawater").

Author(s)

Matthew A. Birk, <matthewabirk@gmail.com>

See Also

conv_multiunit, conv_o2, rho

Examples

```
# I read that an animal's MO2 is 1.92 ml O2/kg/min. What is this MO2 in umol O2/g/h?
conv_resp_unit(value = 1.92, from = "ml_O2 / kg / min", to = "umol_O2 / g / hr")

# Krogh's diffusion coefficient for oxygen through gills can be expressed as ml O2 / mm2 (gill
# surface area) / um (gill thickness) / torr (seawater pO2 - blood pO2) / minute at a given
# temperature.
# To convert to another unit:
conv_resp_unit(value = 1e-6, from = "ml_O2 / mm2 / um / torr / min",
to = "umol_O2 / cm2 / um / kPa / hr", temp = 20)

# Now, with a knowledge of gill morphometrics, seawater pO2, and blood pO2, I can compare
# gill diffusion with whole animal MO2.
```

---

correct_bubble — *Adjust respirometer volume for bubbles*

Description

Given the volume of the respirometer and the volume of bubbles or air space, the moles of O2 in the system are calculated, and the volume of a respirometer holding the same quantity of O2 with only water is returned.

Usage

```
correct_bubble(resp_vol, bubble_vol, temp = 25, sal = 35, atm_pres = 1013.25)
```

Arguments

resp_vol	volume of the respirometer (liter).
bubble_vol	volume of the gas bubbles or headspace (mL).
temp	temperature (°C). Default is 25 °C.
sal	salinity (psu). Default is 35 psu.
atm_pres	atmospheric pressure (mbar). Default is 1013.25 mbar.

Details

Depending on temperature and salinity, air holds 20,000x as much O2 as water per unit volume, thus small air bubbles in a respirometer can dramatically increase the amount of O2 an organism has to consume to lower the pO2 or aqueous [O2]. Thus air bubbles lead to underestimations of MO[2]. To correct for this in MO2 calculations after measurement, the volume of the respirometer can be increased. This function calculates the volume needed for MO2 calculations as a function of the volume of air space. Caution: allowing air bubbles into a respirometer is not recommended, even with this post-measurement adjustment. A small error in bubble volume estimation can lead to a large error in calculated metabolic rate.

Value

The volume of a respirometer holding an equivalent quantity of O2 filled only with water.

Note

Due to the high concentration of O2 in air, very small errors in bubble volume estimates can lead to very large differences in the volume returned. Only trust the returned value if you are very confident of the accuracy of your bubble volume estimate.

Author(s)

Matthew A. Birk, `<matthewabirk@gmail.com>`

See Also

`molvol`

Examples

```
correct_bubble(resp_vol = 50, bubble_vol = 10) # a 10 mL bubble makes a huge difference!

correct_bubble(resp_vol = 50, bubble_vol = 1, temp = 10, sal = 0)
# in calculating MO2, a volume of 63.8 L should be used rather than the true 50 L.
```

`flush_carb` *Estimate carbonate chemistry after a flush*

Description

Given the seawater pH inside the respirometer and in the flush reservoir, the new carbonate parameters (including pH) in the respirometer after the flush are estimated.

Usage

```
flush_carb(
  resp_vol,
  flow_rate,
  duration,
  resp_pH,
  flush_pH,
  temp = 25,
  sal = 35,
  TA = NULL,
  atm_pres = 1013.25
)
```

Arguments

`resp_vol`	volume of the respirometer (liter).
`flow_rate`	rate of water flow into the respirometer (liters / minute).
`duration`	duration of the flush (minutes).
`resp_pH`	pH inside the respirometer before the flush (total scale).
`flush_pH`	pH of the water used for flushing the respirometer (total scale).
`temp`	temperature (°C). Default is 25 °C.
`sal`	salinity (psu). Default is 35 psu. If `sal` < 26 psu, then `TA` must be provided.
`TA`	(optional) total alkalinity (umol / kg). If undefined TA is estimated from salinity using `guess_TA`.
`atm_pres`	atmospheric pressure (mbar). Default is 1013.25 mbar.

Value

A data frame returned by `carb`.

Author(s)

Matthew A. Birk, <matthewabirk@gmail.com>

See Also

`carb`, `flush_water`

Examples

```
flush_carb(resp_vol = 90, flow_rate = 10, duration = 3, resp_pH = 7.8, flush_pH = 8.1)

# What will be the pH in the respirometer after this flush?
flush_carb(resp_vol = 90, flow_rate = 10, duration = 3, resp_pH = 7.8, flush_pH = 8.1)$pH
```

flush_o2 — *Estimate dissolved O2 after a flush*

Description

Calculate the pO2 or [O2] in a respirometer after a flush. Given 5 of the 6 parameters, the 6th parameter is calculated.

Usage

```
flush_o2(resp_vol, flow_rate, duration, resp_o2, flush_o2, final_o2)
```

Arguments

`resp_vol`	volume of the respirometer (liter).
`flow_rate`	rate of water flow into the respirometer (liters / minute).
`duration`	duration of the flush (minutes).
`resp_o2`	O2 inside the respirometer before the flush (units do not matter as long as it is consistant with `flush_o2` and `final_o2`).
`flush_o2`	O2 of the water used for flushing the respirometer (units do not matter as long as it is consistant with `resp_o2` and `final_o2`).
`final_o2`	O2 of the water in the respirometer at the end of the flush (units do not matter as long as it is consistant with `resp_o2` and `flush_o2`).

Author(s)

Matthew A. Birk, <matthewabirk@gmail.com>

See Also

flush_water, flush_carb

Examples

```
# What will be the pO2 in the respirometer after this flush?
flush_o2(resp_vol = 90, flow_rate = 10, duration = 3, resp_o2 = 15, flush_o2 = 21)

# I want to bring the pO2 back up to 95% air saturation. How long do I need to flush?
flush_o2(resp_vol = 20, flow_rate = 2, resp_o2 = 75, flush_o2 = 99, final_o2 = 95)
```

flush_water *Find percent of water exchanged after a flush*

Description

Calculate the proportion of water in a respirometer that is new after a flush. Useful for intermittent respirometry. Given 3 of the first 4 parameters, the 4th parameter is calculated.

Usage

```
flush_water(vol, flow_rate, duration, perc_fresh, plot = FALSE)
```

Arguments

vol	volume of the respirometer (liter).
flow_rate	rate of water flow into the respirometer (liters / minute).
duration	duration of the flush (minutes).
perc_fresh	percent of the respirometer volume that is new flushed water.
plot	logical. Plot the percent exchanged as a function of flow rate and duration to see what effect would result if the rate or duration are changed. All parameters must only have a single value.

Author(s)

Matthew A. Birk, <matthewabirk@gmail.com>

References

Steffensen JF. 1989. Some errors in respirometry of aquatic breathers: How to avoid and correct for them. Fish Physiol Biochem. 6:49–59. Equation 5.

See Also

flush_carb, min_flow

Examples

```
# What proportion of a 90 L respirometer is exchanged after 20 minutes of flow at 2 LPM?
flush_water(vol = 90, flow_rate = 2, duration = 20)

# Would it be worth it to extend the flush another five minutes? How much would that
# improve the exchange?
flush_water(vol = 90, flow_rate = 2, duration = 20, plot = TRUE)
# Another five minutes would increase exchange by nearly 10%.
# Perhaps that's worth the extra time...

# Visualize flushing
```

```
vol = 150
flow_rate = seq(0, 10, by = 0.5)
duration = 0:60
perc_fresh = outer(flow_rate, duration, function(flow_rate, duration){
flush_water(vol = vol, flow_rate = flow_rate, duration = duration)
})
persp(flow_rate, duration, perc_fresh, xlab = 'Flow rate (LPM)', ylab = 'Duration (min)',
zlab = '% exchange', theta = 45, phi = 15, expand = 0.5, ticktype = 'detailed', nticks = 10)
```

---

goal_flush_pH *Calculate goal pH of a flush reservoir to achieve the post-flush goal pCO2*

Description

Calculates the pH of a flush reservoir that is needed to achieve the goal pCO2 after the flush reservoir has been drained into the respirometer.

Usage

```
goal_flush_pH(
  goal_pco2,
  resp_pH,
  resp_vol,
  flush_vol,
  flush_remain = 0,
  temp = 25,
  sal = 35,
  TA = NULL,
  atm_pres = 1013.25
)
```

Arguments

goal_pco2	the desired pCO2 in the respirometer after the flush (uatm).
resp_pH	pH inside the respirometer before the flush (total scale).
resp_vol	volume of the respirometer (liter).
flush_vol	volume of the flush reservoir (liter).
flush_remain	volume of the flush reservoir that will remain after the flush (liter).
temp	temperature (°C). Default is 25 °C.
sal	salinity (psu). Default is 35 psu. If sal < 26 psu, then TA must be provided.
TA	(optional) total alkalinity (umol / kg). If undefined TA is estimated from salinity using guess_TA.
atm_pres	atmospheric pressure (mbar). Default is 1013.25 mbar.

Value

pH needed in the flush reservoir to achieve the goal pCO2 post-flush (total scale).

Author(s)

Matthew A. Birk, <matthewabirk@gmail.com>

See Also

co2_rate, flush_carb, carb, peri_pump

Examples

```
# I want the respirometer to have a pCO2 = 1000 uatm. It currently has a pH of 7.6 and is 90 L.
# If I have a 200 L reservoir which will be drained completely, what do I want
# the pH of the reservoir to be?
goal_flush_pH(goal_pco2 = 1000, resp_pH = 7.6, resp_vol = 90, flush_vol = 200)
```

---

guess_TA — *Estimate total alkalinity from salinity*

Description

Estimate total alkalinity from salinity and temperature of surface seawater according to Lee et al. 2006. Useful when a rough guess of TA is needed because measuring TA is not possible or practical.

Usage

```
guess_TA(temp = 25, sal = 35, region = NULL, extend = TRUE)
```

Arguments

temp	temperature (°C). Default is 25 °C.
sal	salinity (psu). Default is 35 psu. $31 \leq$ sal ≤ 38; may be narrower for some regions.
region	(optional) geographic region. Options are "(Sub)tropics", "Equatorial Upwelling Pacific", "North Atlantic", "North Pacific", and "Southern Ocean". Default is NULL. If undefined, the average from all these regions is used.
extend	logical. If salinity is ≤ 5 psu outside of the bounds defined by Lee et al. 2006 (see Details), should a guess be extrapolated? Default is TRUE.

Details

(Sub)tropics `temp` ≥ 20 and $31 \leq$ `sal` ≤ 38

Equatorial Upwelling Pacific `temp` ≥ 18 and $31 \leq$ `sal` ≤ 36.5

North Atlantic $0 \leq$ `temp` ≤ 20 and $31 \leq$ `sal` ≤ 37

North Pacific `temp` ≤ 20 and $31 \leq$ `sal` ≤ 35

Southern Ocean `temp` ≤ 20 and $33 \leq$ `sal` ≤ 36

Estimates total alkalinity using the equations provided by Lee et al. 2006 (Geophysical Research Letters). While these equations are designed for open ocean environments, they can provide a rough estimate even for coastal environments. For improved estimate accuracy, the geographic region can be provided. The North Pacific region is longitude-dependent so a longitude of 150 °W is assumed which provides a typical value within the range. Only applicable for surface waters, not very accurate for the ocean interior.

Value

An estimate of the total alkalinity (umol / kg). If `NA` or `NaN` are returned, confirm the `temp` and `sal` values are within acceptable ranges for the region of interest.

Author(s)

Matthew A. Birk, `<matthewabirk@gmail.com>`

References

Lee K, Tong LT, Millero FJ, Sabine CL, Dickson AG, Goyet C, Park G-H, Wanninkhof R, Feely RA, Key RM. 2006. Global relationships of total alkalinity with salinity and temperature in surface waters of the world's oceans. Geophys Res Lett. 33:L19605.

See Also

`predict_pH`

Examples

```
guess_TA(temp = 22, sal = 33)
guess_TA(temp = 12, sal = 33, region = "North Atlantic")
guess_TA(temp = 20, sal = 31:35)

guess_TA(sal = 31) # salinity is within bounds
guess_TA(sal = 30) # salinity is outside the bounds and TA is extrapolated
guess_TA(sal = 30, extend = FALSE) # do not extrapolate TA
guess_TA(sal = 25, extend = TRUE) # will not extrapolate with sal > 5 psu out of bounds
```

guess_when *Estimate when the O2 level will reach a defined level*

Description

Estimates the time at which O2 will reach a defined level assuming a linear change in O2 over time.

Usage

```
guess_when(past_o2, past_time, goal_o2, plot = TRUE)
```

Arguments

past_o2	a numeric vector of at least two oxygen measurements previously during the trial.
past_time	a vector of timepoints corresponding to when past_o2 values were recorded. Can be a numeric vector for duration since trial began or a POSIX vector of time values.
goal_o2	a numeric vector or single value describing the O2 level of interest.
plot	logical. Do you want to see a plot to visualize this prediction?

Value

A prediction of the time when O2 will reach goal_o2. If past_time is numeric, then a numeric value(s) will be returned. If POSIX, then POSIX will be returned.

Note

Viewing the plot can be valuable if the O2 consumption or production is not linear.

Author(s)

Matthew A. Birk, <matthewabirk@gmail.com>

See Also

predict_pH, predict_nh3

Examples

```
guess_when(past_o2 = rnorm(n = 10, mean = 100:91), past_time = 1:10, goal_o2 = 75, plot = FALSE)
guess_when(past_o2 = rnorm(n = 10, mean = 100:91, sd = 5), past_time = 1:10, goal_o2 = 75)
# Viewing the plot can be helpful to see how trustworthy the prediction is
# when signal:noise is low.
```

`import_firesting` *Import data from a FireSting O2 transmitter*

Description

Imports the standard txt file output from FireSting O2 transmitters and converts the data into one or more data frames.

Usage

```
import_firesting(
  file,
  o2_unit = "percent_a.s.",
  date = "%m/%d/%Y %X",
  overwrite_sal = NULL,
  keep_metadata = FALSE,
  drop_channels = TRUE,
  split_channels = FALSE
)
```

Arguments

`file`	a character string. The filepath for the file to be read.
`o2_unit`	a character string. The unit of O2 measurement to be output in the data frame. Options are described in `conv_o2`.
`date`	a character string. The date format to be passed to `strptime`.
`overwrite_sal`	Default `NULL`. To overwrite the salinity value(s) from calibration, enter a single numeric value for all channels or a numeric vector with values for each channel. Salinity of water sample (psu).
`keep_metadata`	logical. Should metadata from the file be returned as extra columns in the returned data frame? Default is `FALSE`.
`drop_channels`	logical. Should channels without any O2 data be dropped? Default is `TRUE`.
`split_channels`	logical. Should a list of data frames be returned with a separate data frame for each channel? Default is `FALSE`.

Details

The following FireSting fiber optic O2 transmitters are supported:

- FireStingO2
- FireStingO2 (1st generation)

If you would like support for the Piccolo2, FireStingO2-Mini, TeX4, or any OEM instruments, email me a data file from the device.

Value

A data frame (or list of data frames) is returned.

TIME Date and time, POSIXlt format.

DURATION Duration of measurement trial (minutes).

CH_X_O2 Oxygen measurement in desired unit as determined by `o2_unit`.

CH_X_TEMP Temperature recorded or defined at beginning of measurement trial.

CH_X_SAL Salinity (psu).

... Channel columns (CH_...) are repeated for each channel.

COMMENT Comments from FireSting file.

If `keep_metadata` = `TRUE`, then the following columns are appended to the returned data frame:

ATM_PRES Atmospheric pressure (mbar).

HUMIDITY Relative humidity (% RH).

PROBE_TEMP Probe temperature.

INTERNAL_TEMP Transmitter internal temperature.

ANALOG_IN Voltage input from the extension port (mV).

CH_X_PHASE Phase recorded. Phase is inversely related to O2.

CH_X_INTENSITY Intensity is an indicator of the quality of the signal. A low intensity warning is produced by the transmitter below 10 mV.

CH_X_AMB_LIGHT Ambient light on the sensor. Expressed in mV.

If `split_channels` = `TRUE`, then `"CH_X_"` is removed from the column names and multiple data frames are returned in a named list.

Note

Oxygen conversions are estimates based on the `marelac` package.

Author(s)

Matthew A. Birk, <matthewabirk@gmail.com>

See Also

`import_presens`, `import_witrox`, `conv_o2`

Examples

```
## Not run:
file <- system.file('extdata', 'firesting_file.txt', package = 'respirometry')
import_firesting(file, o2_unit = 'umol_per_l')

# I want each channel as a separate data frame.
data_list <- import_firesting(file, split_channels = TRUE)
data_list$CH_3 # here's the channel 3 data frame.
```

```
## End(Not run)
```

---

import_presens *Import data from a PreSens O2 transmitter*

Description

Imports the standard text file output from most single channel PreSens fiber optic O2 transmitters and converts the data into a data frame.

Usage

```
import_presens(
  file,
  o2_unit = "percent_a.s.",
  date = "%d/%m/%y",
  sal = 35,
  all_cols = FALSE,
  split_channels = FALSE
)
```

Arguments

`file`	a character string. The filepath for the file to be read.
`o2_unit`	a character string. The unit of O2 measurement to be output in the data frame. Options are described in `conv_o2`.
`date`	a character string. The date format to be passed to `strptime`.
`sal`	salinity of water sample (psu). Default is 35 psu. Ignored for Fibox 4 files since salinity is provided by the file.
`all_cols`	logical. For Fibox 4 files only. Should all columns (including calibration data and serial numbers) be output?
`split_channels`	logical. For SDR SensorDish only. Should a list of data frames be returned with a separate data frame for each channel? Default is `FALSE`.

Details

The following PreSens fiber optic O2 transmitters are supported:

- Fibox 4
- Fibox 3
- Fibox 3 trace
- Fibox 3 LCD trace
- Microx TX3

- Microx TX3 trace
- SDR SensorDish Reader

If you would like support for another PreSens O2 meter, email the package maintainer a data file from the device you would like supported. It is very important to note that the PreSens fiber optics O2 transmitters that are supported with this function (except the Fibox 4) DO NOT account for salinity (i.e. they assume salinity = 0 ppt). If the water sample measured was not fresh water, the oxygen concentrations (e.g. mg per liter or umol per liter) are incorrect in the PreSens txt file. This function corrects these O2 concentrations based on the salinity value defined by the `sal` argument. Absolute partial pressures (i.e. hPa and torr) will also be slightly different due to the slight influence of salinity on water's vapor pressure. This difference is typically ~0.05% of the recorded value.

Value

A data frame is returned.

TIME Date and time, POSIXct format.

DURATION Duration of measurement trial (minutes).

O2 Oxygen measurement in desired unit as determined by `o2_unit`.

PHASE Phase recorded. Phase is inversely related to O2. Not included in SDR SensorDish Reader files.

AMPLITUDE Amplitude recorded. Amplitude is an indicator of the quality of the signal. A low amplitude warning is produced by the transmitter below 2500. Not included in SDR SensorDish Reader files.

TEMP Temperature recorded or defined at beginning of measurement trial.

ATM_PRES Atmospheric pressure (mbar).

SAL Salinity (psu).

ERROR_CODE Error code from transmitter. See PreSens user manual for translation of error code. Not included in SDR SensorDish Reader files.

Note

Oxygen conversions are based on `conv_o2` and therefore differ slightly from the conversions provided by PreSens.

Author(s)

Matthew A. Birk, <matthewabirk@gmail.com>

See Also

`import_firesting`, `import_witrox`, `conv_o2`

Examples

```
## Not run:

# Import a Fibox 3 file.
file <- system.file('extdata', 'fibox_3_file.txt', package = 'respirometry')
import_presens(file, o2_unit = 'umol_per_l', sal = 25)

# Import a Fibox 4 file.
file <- system.file('extdata', 'fibox_4_file.csv', package = 'respirometry')
import_presens(file = file, date = '%d-%b-%Y')

# Import an SDR SensorDish Reader file.
file <- system.file('extdata', 'sdr_file.txt', package = 'respirometry')
import_presens(file = file, date = '%d.%m.%y%X')


## End(Not run)
```

`import_witrox` *Import data from a Loligo Systems Witrox O2 transmitter*

Description

Imports the standard txt file output from Loligo Systems Witrox fiber optic O2 transmitters and converts the data into one or more data frames.

Usage

```
import_witrox(
  file,
  o2_unit = "percent_a.s.",
  date = "%m/%d/%Y %I:%M:%S %p",
  overwrite_sal = NULL,
  drop_channels = TRUE,
  split_channels = FALSE
)
```

Arguments

`file`	a character string. The filepath for the file to be read.
`o2_unit`	a character string. The unit of O2 measurement to be output in the data frame. Options are described in `conv_o2`.
`date`	a character string. The date format to be passed to `strptime`.
`overwrite_sal`	Default `NULL`. To overwrite the salinity value(s) from calibration, enter a single numeric value for all channels or a numeric vector with values for each channel. Salinity of water sample (psu).
`drop_channels`	logical. Should channels without any O2 data be dropped? Default is `TRUE`.

`split_channels` logical. Should a list of data frames be returned with a separate data frame for each channel? Default is `FALSE`.

Details

The following Loligo Systems fiber optic O2 transmitters are supported:

- Witrox 4

If you would like support for the Witrox 1, email me a data file from this device.

Value

A data frame (or list of data frames) is returned.

TIME Date and time, POSIXlt format.

DURATION Duration of measurement trial (minutes).

ATM_PRES Atmospheric pressure (mbar).

CH_X_PHASE Phase recorded. Phase is inversely related to O2.

CH_X_TEMP Temperature recorded or defined at beginning of measurement trial.

CH_X_SAL Salinity (psu).

CH_X_O2 Oxygen measurement in desired unit as determined by `o2_unit`.

... Channel columns (CH_...) are repeated for each channel.

If `split_channels` = `TRUE`, then "`CH_X_`" is removed from the column names and multiple data frames are returned in a list.

Author(s)

Matthew A. Birk, <matthewabirk@gmail.com>

See Also

`import_firesting`, `import_presens`, `conv_o2`

Examples

```
## Not run:
file <- system.file('extdata', 'witrox_file.txt', package = 'respirometry')
import_witrox(file, o2_unit = 'umol_per_l')

# Oops. I forgot to change the salinity value when I calibrated
# the instrument. Override the values in the file for 35 psu.
import_witrox(file, o2_unit = 'umol_per_kg', overwrite_sal = 35)

# I want each channel as a separate data frame.
data_list <- import_witrox(file, split_channels = TRUE)
data_list$CH_3 # here's the channel 3 data frame.

## End(Not run)
```

`make_bins` *Make time bins for MO2 calculations*

Description

The width of time bins seems to be an under-appreciated consideration when calculating metabolic rates if PO2 or time are interesting covariates. The wider the bins, the higher the precision of your calculated MO2 value (more observations to average over), but at a loss of resolution of an interesting covariate. The narrower the bins, the higher the resolution of the PO2 or time covariate, but at a cost of lower precision. For Pcrit trials, I have found good success using bins of 1/10th the trial duration at the highest PO2s (where good precision is important) and 1/100th the trial duration at the lowest PO2s (where good resolution is important).

Usage

```
make_bins(
  o2,
  duration,
  good_data = TRUE,
  min_o2_width = 1/100,
  max_o2_width = 1/10,
  n_bins = 10
)
```

Arguments

`o2`	numeric vector of O2 observations.
`duration`	numeric vector of the timepoints for each observation (minutes).
`good_data`	logical vector of whether O2 observations are "good" measurements and should be included in analysis. Default is that all observations are `TRUE`.
`min_o2_width`	Default is 1/100th of the total "good" trial duration.
`max_o2_width`	Default is 1/10th of the total "good" trial duration.
`n_bins`	Default is 10.

Value

A data.frame with two columns is returned.

o2 The O2 value below which the corresponding bin width is applied.

width The bin width at which all data below the corresponding O2 value will be binned.

Author(s)

Matthew A. Birk, `<matthewabirk@gmail.com>`

See Also

calc_MO2

Examples

```
# get O2 data
file <- system.file('extdata', 'witrox_file.txt', package = 'respirometry')
o2_data <- na.omit(import_witrox(file, split_channels = TRUE)$CH_4)

# Total trial duration is 21.783 minutes

make_bins(o2 = o2_data$O2, duration = o2_data$DURATION) # creates the default 10 bins. At the
# highest O2 levels, bin widths are 21.783/10 = 2.1783 mins and at the lowest O2 levels, bin
# widths are 0.21783 mins.

bins <- make_bins(o2 = o2_data$O2, duration = o2_data$DURATION, min_o2_width = 1/20,
max_o2_width = 1/3, n_bins = 5) # creates 5 bins. At the highest O2 levels, bin widths are
# 21.783/3 = 7.261 mins and at the lowest O2 levels, bin widths are 21.783/20 = 1.089 mins.

(mo2 <- calc_MO2(duration = o2_data$DURATION, o2 = o2_data$O2,
bin_width = bins, vol = 10, temp = o2_data$TEMP, sal = o2_data$SAL))
```

---

max_MO2 *Maximum MO2 supported by flow rate*

Description

Calculates the maximum oxygen consumption rate (MO2) supported by a respirometer with a given flow rate. Useful for ensuring an acclimating animal maintains a normoxic environment.

Usage

```
max_MO2(
  flow_rate,
  min_pO2 = 90,
  pO2_in = 100,
  temp = 25,
  sal = 35,
  atm_pres = 1013.25
)
```

Arguments

flow_rate	water flow rate into respirometer (liters / min).
min_pO2	minimum pO2 acceptable in respirometer (% air saturation). Default is 90% air saturation.

`pO2_in`	pO2 of water entering respirometer (% air saturation). Default is 100% air saturation.
`temp`	temperature (°C). Default is 25 °C.
`sal`	salinity (psu). Default is 35 psu.
`atm_pres`	atmospheric pressure (mbar). Default is 1013.25 mbar.

Value

The maximum whole-animal oxygen consumption rate (umol / hr) that can be sustained.

Note

Keep in mind that most organisms are very stressed upon being placed in a respirometer and their MO2 may be much higher than basal MO2.

Author(s)

Matthew A. Birk, <matthewabirk@gmail.com>

References

Steffensen JF. 1989. Some errors in respirometry of aquatic breathers: How to avoid and correct for them. Fish Physiol Biochem. 6:49–59. Equation 8.

See Also

`min_flow`, `flush_water`

Examples

```
max_MO2(flow_rate = 1)

# What is the maximum MO2 organism I can place in my respirometer and still maintain at
# least 75% air saturation when the intake fresh water is 1.5 LPM, 10 °C and 90% air saturated?
(max_mo2 <- max_MO2(flow_rate = 1.5, min_pO2 = 75, pO2_in = 90, temp = 10, sal = 0))

# If a 300 g individual has an MO2 of 2000 umol/hr, how big of an animal can I use?
scale_MO2(mass_1 = 300, MO2_1 = 2000, MO2_2 = max_mo2) # I can almost support a 1 kg individual!
```

mean_pH *Mean pH by [H+]*

Description

Calculates mean pH from a vector of pH values by averaging [H+] rather than numerical pH values.

Usage

```
mean_pH(pH, na.rm = FALSE, ...)
```

Arguments

`pH`	a numeric vector of pH values.
`na.rm`	a logical value indicating whether NA values should be stripped before the computation proceeds.
`...`	further arguments passed to or from other methods.

Details

Since pH is on a logarithmic scale, averaging pH values directly does not provide the true arithmetic mean of what is likely truly important to the organism, [H+] (however, see Boutilier and Shelton 1980). Thus, the pH values are converted to [H+] then averaged and converted back to a mean pH value.

Author(s)

Matthew A. Birk, <matthewabirk@gmail.com>

References

Boutilier RG, Shelton G. 1980. The statistical treatment of hydrogen ion concentration and pH. J Exp Biol. 84:335–339.

Examples

```
mean_pH(c(7, 8)) # 7.26 rather than 7.5!
```

min_flow *Minimum flow rate to support MO2*

Description

Calculates the minimum flow rate into a respirometer required to maintain a high pO2. Useful for ensuring an acclimating animal maintains a normoxic environment. It can also be used to estimate the flow rate needed for a given pO2 decrease desired for flow-through respirometry.

Usage

```
min_flow(
  MO2,
  min_pO2 = 90,
  pO2_in = 100,
  temp = 25,
  sal = 35,
  atm_pres = 1013.25
)
```

Arguments

MO2	whole-animal oxygen consumption rate (umol / hour).
min_pO2	minimum pO2 acceptable in respirometer (% air saturation). Default is 90% air saturation.
pO2_in	pO2 of water entering respirometer (% air saturation). Default is 100% air saturation.
temp	temperature (°C). Default is 25 °C.
sal	salinity (psu). Default is 35 psu.
atm_pres	atmospheric pressure (mbar). Default is 1013.25 mbar.

Value

The flow rate (liters / min) into the respirometer required for the steady state pO2 to be min_pO2.

Note

Keep in mind that most organisms are very stressed upon being placed in a respirometer and their MO2 may be much higher than basal MO2.

Author(s)

Matthew A. Birk, <matthewabirk@gmail.com>

References

Steffensen JF. 1989. Some errors in respirometry of aquatic breathers: How to avoid and correct for them. Fish Physiol Biochem. 6:49–59. Equation 8.

See Also

max_MO2, flush_water

Examples

```
min_flow(MO2 = 1000)

# What is the minimum flow rate required to maintain at least 75% air saturation in a
# respirometer with an organism(s) with an oxygen consumption rate of 1000 umol/h
# when the intake fresh water is 10 °C and 90% air saturated?
min_flow(MO2 = 1000, min_pO2 = 75, pO2_in = 90, temp = 10, sal = 0)
```

---

peri_pump — *Calculate peristaltic pump gaseous flow rate*

Description

Given the number of moles of a gas, calculates the liters to run through a peristaltic pump.

Usage

```
peri_pump(
  mol,
  species = "O2",
  temp = 25,
  reg_pres,
  reg_unit = "psi",
  atm_pres = 1013.25
)
```

Arguments

mol	number of moles to go through a peristaltic pump.
species	character string describing the gas species. Options are available from molvol. Default is "O2".
temp	temperature (°C). Default is 25 °C.
reg_pres	gauge pressure from the gas regulator into the peristaltic pump.
reg_unit	unit used in reg_pres. Default is "psi".
atm_pres	atmospheric pressure (mbar). Default is 1013.25 mbar.

Details

Most mass flow controllers are programmed with a "standard condition" something like 0 *C and 1013 mbar for which they account for the pressure and temperature of an incoming gas source. For setups without expensive mass flow controllers, a more affordable alternative is to use a peristaltic pump. These do not account for variations in incoming gas pressure and temperature and thus, it must be calculated to set the peristaltic pump to the correct RPM.

Author(s)

Matthew A. Birk, <matthewabirk@gmail.com>

See Also

co2_rate, co2_add

Examples

```
peri_pump(mol = 0.5, species = 'O2', temp = 10, reg_pres = 5, reg_unit = "kPa")
# To flow 0.5 moles of O2, then flow 11.1 L.
```

---

plot_pcrit *Plot Pcrit*

Description

Creates a Pcrit plot (the threshold below which oxygen consumption rate can no longer be sustained) based on paired PO2 and MO2 values. Five Pcrit metrics are plotted: the traditional breakpoint metric (broken stick regression, black), the nonlinear regression metric (Marshall et al. 2013, green), the sub-prediction interval metric (Birk et al. 2019, red), the alpha-based Pcrit method (Seibel et al., 2021, blue), and the linear low O2 (LLO) method (Reemeyer & Rees 2019, purple). For details on how the Pcrit values are calculated, see calc_pcrit.

Usage

```
plot_pcrit(
  po2,
  mo2,
  avg_top_n = 1,
  level = 0.95,
  iqr = 1.5,
  NLR_m = 0.065,
  MR = NULL,
  mo2_threshold = Inf,
  showNLRs = FALSE,
  ...
)
```

Arguments

- `po2` a vector of PO2 values. Any unit of measurement should work, but the NLR calculation was optimized using kPa. If the NLR metric is giving you trouble, try converting to kPa using `conv_o2`.
- `mo2` a vector of metabolic rate values. Must be the same length and corresponding to `po2`.
- `avg_top_n` applies to the `alpha` metric only. A numeric value representing the number of top $\alpha 0$ (MO2/PO2) values to average together to estimate α. Default is 1. We recommend no more than 3 to avoid diminishing the α value with sub-maximal observations.
- `level` applies to the `Sub_PI` metric only. Percentage at which the prediction interval should be constructed. Default is 0.95.
- `iqr` applies to the `Sub_PI` metric only. Removes `mo2` observations that are this many interquartile ranges away from the mean value for the oxyregulating portion of the trial. If this filtering is not desired, set to infinity. To visualize which observations will be removed by this parameter, use `plot_pcrit`. Default is 1.5.
- `NLR_m` applies to the `NLR` metric only. Pcrit is defined as the PO2 at which the slope of the best fitting function equals `NLR_m` (after the MO2 data are normalized to the 90% quantile). Default is 0.065.
- `MR` applies to the `alpha` and `LLO` metrics only. A numeric value for the metabolic rate at which `pcrit_alpha` and `pcrit_LLO` should be returned. If not supplied by the user, then the mean MO2 of the "oxyregulating" portion of the curve is applied for `pcrit_alpha` and `NA` is returned for `pcrit_LLO`.
- `mo2_threshold` applies to the `alpha` metric only. A single numeric value above which `mo2` values are ignored for `alpha` Pcrit estimation. Useful to removing obviously erroneous values. Default is `Inf`.
- `showNLRs` logical. Should all the NLR functions be plotted in a second plot? If `FALSE` then only the best fit NLR function will be plotted.
- `...` arguments to be passed to `plot.segmented`.

Details

Alpha Pcrit Alpha is calculated from `calc_alpha` and the Pcrit corresponding to `MR` is returned. This determine's the animal's oxygen supply capacity and calculates the Pcrit at any given metabolic rate of interest. If no `MR` is provided, then it defaults to the mean MO2 value from the oxyregulating portion of the curve (as defined by the broken-stick regression).

Breakpoint Pcrit Data are fit to a broken-stick regression using `segmented`.

LLO Pcrit A subset of observations are chosen only from those with an MO2 < `MR`. Then, a linear model is fit through the observations and Pcrit is calculated as the PO2 at which the line reaches `MR`.

NLR Pcrit Data are fit to the following functions: Michaelis-Menten, Power, Hyperbola, Pareto, and Weibull with intercept. Following the method developed by Marshall et al. 2013, the function that best fits the data (smallest AIC) is chosen and the Pcrit is determined as the PO2 at which the slope of the function is `NLR_m` (by default = 0.065 following the authors' suggestion).

Sub_PI Pcrit This metric builds off the `Breakpoint` metric and results in a systematically lower Pcrit value. This is useful for applications where it is important to ensure that Pcrit is not being overestimated. It represents a reasonable lower bounded estimate of the Pcrit value for a given trial. Once the `Breakpoint` Pcrit is calculated, a 95% prediction interval (can be changed with the `level` argument) is calculated around the oxyregulating region (i.e. using PO2 values > breakpoint Pcrit). By default, `iqr` provides some filtering of abberant observations to prevent their influence on the calculated prediction interval. Finally, the Sub_PI Pcrit value is returned at the intersection of the oxyconforming line and the lower limit of the oxyregulating prediction interval.

Value

A base graphic plot is created. The alpha, breakpoint, LLO, NLR, and sub-PI Pcrit values are shown in the title and on the plot by inverted triangles.

The broken-stick regression is shown by black lines.

The dashed red curves signify the prediction interval used for the sub-PI Pcrit metric.

Black circles represent oxyregulating observations used in the generation of the prediction interval, while transparent circles represent both the oxyconforming observations and those observations outside the IQR threshold (defined by `iqr`).

The gray bands represent the confidence interval (defaults to 95% but will change with `level`).

The green curve represents the best fitting NLR function and the green inverted triangle represents the NLR Pcrit (modified by `NLR_m`)

The blue line represents alpha, which was fit based on the blue circle observation(s).

If `showNLRs = TRUE`, then a second plot is generated which shows all the NLR functions that converged. Vertical lines represent the Pcrit values corresponding to each curve.

Black = Michaelis-Menten

Red = Power

Green = Hyperbola

Blue = Pareto

Cyan = Weibull with intercept.

Author(s)

Matthew A. Birk, `<matthewabirk@gmail.com>`

References

Birk, Matthew A., K.A.S. Mislan, Karen F. Wishner, and Brad A. Seibel. 2019. "Metabolic Adaptations of the Pelagic Octopod Japetella Diaphana to Oxygen Minimum Zones." Deep-Sea Research Part I 148: 123–31.

Marshall, Dustin J., Michael Bode, and Craig R. White. 2013. "Estimating Physiological Tolerances - a Comparison of Traditional Approaches to Nonlinear Regression Techniques." Journal of Experimental Biology 216(12): 2176–82.

Reemeyer, Jessica E., and Bernard B. Rees. 2019. "Standardizing the Determination and Interpretation of Pcrit in Fishes." Journal of Experimental Biology 222(18): jeb210633.

Seibel, B. A., A. Andres, M. A. Birk, A. L. Burns, C. T. Shaw, A. W. Timpe, C. J. Welsh. 2021. "Oxygen supply capacity breathes new life into the critical oxygen partial pressure (Pcrit)." Journal of Experimental Biology.

See Also

calc_pcrit, calc_alpha

Examples

```
mo2_data <- read.csv(system.file('extdata', 'mo2_v_po2.csv', package = 'respirometry'))
plot_pcrit(po2 = mo2_data$po2, mo2 = mo2_data$mo2, avg_top_n = 3, MR = 2.2)

par(mfrow = c(2, 1))
plot_pcrit(po2 = mo2_data$po2, mo2 = mo2_data$mo2, showNLRs = TRUE)
```

---

predict_nh3 *Predict NH3 / NH4+ concentration post-respiration*

Description

Predicts the [NH3] and [NH4+] of seawater after a defined amount of oxygen consumption. Ammonotelic animals excrete the ionized form NH4+ (ammonium) but some of these ions dissociate into unionized NH3 (ammonia) which is toxic for most fishes and crustaceans around 0.4-2.0 mg/L (Boyd 2012).

Usage

```
predict_nh3(
  o2_drop = 10,
  o2_unit = "percent_a.s.",
  o2_nh4_ratio,
  temp = 25,
  sal = 35,
  pH = 8.1,
  atm_pres = 1013.25
)
```

Arguments

o2_drop	a numeric value or vector describing the change in O2. Default is 10.
o2_unit	a string describing the unit used to measure o2_drop. Default is "percent_a.s." Options are from conv_o2.
o2_nh4_ratio	molar ratio of O2 consumed to NH4+ produced.
temp	temperature (°C). Default is 25 °C.
sal	salinity (psu). Default is 35 psu.
pH	seawater pH (total scale). Default is 8.1.
atm_pres	atmospheric pressure (mbar). Default is 1013.25 mbar.

Details

Given a known amount of oxygen consumed and an estimated O2:N ratio, the amount of NH4 produced can be estimated. Production or consumption of ammonium by "background" microbes or conversion of ammonium to nitrite and nitrate is ignored since bacteria in the respirometer are typically sought to be in low levels. The amount of dissociation to produce ammonia is calculated by Kn.

Value

A list containing the predicted NH3, NH4+, and TAN produced in mg/l.

Author(s)

Matthew A. Birk, <matthewabirk@gmail.com>

References

Boyd C. 2012. Water Quality. In "Aquaculture: Farming Aquatic Animals and Plants". Blackwell Publishing, Ltd.

See Also

conv_o2, conv_nh4, Kn

Examples

```
predict_nh3(o2_drop = 25, o2_nh4_ratio = 10)
```

---

predict_pH *Predict pH post-respiration*

Description

Predicts the pH of seawater after a defined amount of oxygen consumption.

Usage

```
predict_pH(
  start_o2 = 100,
  end_o2,
  start_pH,
  temp = 25,
  sal = 35,
  RQ = 1,
  TA = NULL,
  all_carb = FALSE
)
```

Arguments

- `start_o2` pO2 at the start of the measurement (% air saturation). Default is 100% air saturation.
- `end_o2` pO2 at the end of the measurment (% air saturation).
- `start_pH` seawater pH (total scale) at the start of the measurement.
- `temp` temperature (°C). Default is 25 °C.
- `sal` salinity (psu). Default is 35 psu. If `sal` < 26 psu, then `TA` must be provided.
- `RQ` respiratory quotient: ratio of CO2 produced / O2 consumed. Default is 1.
- `TA` (optional) total alkalinity (umol / kg). If undefined TA is estimated from salinity using `guess_TA`.
- `all_carb` logical. Should all carbonate chemistry parameters be returned? Default is FALSE.

Details

Given a known amount of oxygen consumed and an estimated respiratory quotient (see `Q10`), the amount of CO2 produced can be estimated. From this CO2 production estimate, the carbonate chemistry of the seawater can be estimated. Atmospheric pressure is assumed.

Value

If `all_carb` is `FALSE`, then a list of the predicted pH (total scale) at the end of the measurement and the predicted pCO2 (uatm) are returned. If `all_carb` is `TRUE`, then the predicted carbonate chemistry parameters are returned from `carb`.

Author(s)

Matthew A. Birk, <matthewabirk@gmail.com>

See Also

`carb`, `guess_TA`

Examples

```
predict_pH(end_o2 = 75, start_pH = 8.1)
predict_pH(start_o2 = 75, end_o2 = 50, start_pH = 7.96, temp = 15, sal = 33, RQ = 0.88)

# I know pH at the end was 7.8, but what was pH at the beginning?
predict_pH(start_o2 = 75, end_o2 = 100, start_pH = 8.013536) # reverse the order
```

Q10 — *Parameters of Q10 Temperature Coefficient*

Description

Calculates parameters from Q10 temperature coefficient for chemical or biological systems. This function can be used in two ways. 1. if four of the first five parameters are given (`Q10`, `R1`, `R2`, `T1`, `T2`) then the fifth parameter is returned, or 2. if `R_vec` and `T_vec` are given, then the best Q10 for those data is returned.

Usage

```
Q10(Q10, R1, R2, T1, T2, R_vec, T_vec, model = FALSE)
```

Arguments

`Q10`	factor by which rate changes due to 10 °C increase in temperature.
`R1`	rate 1.
`R2`	rate 2.
`T1`	temperature 1 (in °C).
`T2`	temperature 2 (in °C).
`R_vec`	a vector of rate values.
`T_vec`	a vector of temperature values (in °C).
`model`	logical. If `TRUE`, then a list is returned which includes an linear model of `log10(R_vec)` and `T_vec` fit by `stats::lm()`.

Details

$$Q10 = (R2/R1)^(10/(T2 - T1))$$

Author(s)

Matthew A. Birk, `<matthewabirk@gmail.com>`

See Also

`scale_MO2`, `calc_E`

Examples

```
Q10(R1 = 5, R2 = 10, T1 = 10, T2 = 20) # Returns Q10; = 2
Q10(Q10 = 2.66, R1 = 5, T1 = 10, T2 = 20) # Returns R2; = 13.3

# My species has an MO2 of 9.5 umol/g/h at 10 *C. What MO2 should I expect at 13 *C?
Q10(Q10 = 2, R1 = 9.5, T1 = 10, T2 = 13) # expect ~11.7 umol/g/h at 13 *C.

# I measured MO2 at a spectrum of temperatures. What Q10 value best fits my data?
Q10(R_vec = c(1, 2, 5, NA, 18, 33), T_vec = c(0, 10, 20, 30, 40, 50))

# I want to see a plot of my data with a Q10 curve through them.
T_vec = c(5, 13, 13, 20, 27) # dummy data
R_vec = c(1, 3, 4, 9, 20)
curve_x = data.frame(T_vec = seq(5, 30, by = 0.01))
best_fit = Q10(R_vec = R_vec, T_vec = T_vec, model = TRUE)$model
curve_y = predict(best_fit, newdata = curve_x)
plot(T_vec, R_vec)
lines(curve_x$T_vec, curve_y)

# A 100 g individual at 10 *C has an MO2 of 1270 umol/h. How much
# would a 250 g individual likely consume at 14 *C?
Q10(Q10 = 2, R1 = scale_MO2(mass_1 = 100, MO2_1 = 1270, mass_2 = 250), T1 = 10, T2 = 14)

# Visualize MO2 scaling by mass and temperature:
mass <- seq(10, 200, 10)
temp <- 10:25
base_mass <- 50
base_temp <- 20
base_MO2 <- 750
mo2 <- outer(mass, temp, function(mass, temp){
scale_MO2(mass_1 = base_mass, mass_2 = mass, MO2_1 = Q10(Q10 = 2, R1 = base_MO2,
 T1 = base_temp, T2 = temp))
})
persp(mass, temp, mo2, xlab = 'Mass (g)', ylab = 'Temperature (*C)', zlab = 'MO2 (umol / hr)',
 theta = 35, phi = 15, expand = 0.5, ticktype = 'detailed', nticks = 10)
```

---

respirometry — *Tools for Conducting Respirometry Experiments*

Description

Provides tools to enable the researcher to more precisely conduct respirometry experiments. Strong emphasis is on aquatic respirometry. Tools focus on helping the researcher setup and conduct experiments. Analysis of the resulting data is not a focus since analyses are often specific to a particular setup, and thus are better created by the researcher individually. This package provides tools for intermittent, flow-through, and closed respirometry techniques.

Author(s)

Matthew A. Birk, <matthewabirk@gmail.com>

RQ *Calculate respiratory quotient*

Description

Calculates the respiratory quotient (RQ), or ratio of CO2 produced to O2 consumed between observations. To calculate CO2 produced, either DIC or both pH and TA must be provided.

Usage

```
RQ(
  o2,
  o2_unit = "percent_a.s.",
  pH = NULL,
  TA = NULL,
  DIC = NULL,
  temp = 25,
  sal = 35,
  atm_pres = 1013.25
)
```

Arguments

`o2`	a numeric vector of O2 values with a length of at least 2.
`o2_unit`	a string describing the unit used to measure `o2`. Default is "percent_a.s." Options are from `conv_o2`.
`pH`	pH (total scale). Elements must align with `o2` vector.
`TA`	total alkalinity (umol / kg). May be either a vector with length equal to `o2` or a single numeric value.
`DIC`	dissolved inorganic carbon (umol / kg). Elements must align with `o2` vector.
`temp`	temperature (°C). Default is 25 °C.
`sal`	salinity (psu). Default is 35 psu.
`atm_pres`	atmospheric pressure (mbar). Default is 1013.25 mbar.

Value

ratio of CO2 produced to O2 consumed.

Note

If you want a rough estimate of RQ, but only have pH measurements, TA can be estimated from salinity using `guess_TA`.

Author(s)

Matthew A. Birk, <matthewabirk@gmail.com>

See Also

conv_o2, guess_TA

Examples

```
o2_observations <- c(21, 18, 14.5, 7)
pH_observations <- c(8.05, 7.98, 7.86, 7.65)
TA_observations <- c(2222, 2219, 2208, 2214)

RQ(o2 = o2_observations, o2_unit = 'kPa', pH = pH_observations,
TA = TA_observations, temp = 20, sal = 33)

DIC_observations <- c(2222, 2250, 2284, 2355)
RQ(o2 = o2_observations, o2_unit = 'kPa', DIC = DIC_observations)

RQ(o2 = o2_observations, o2_unit = 'kPa', pH = pH_observations, TA = 2032)
```

---

scale_MO2 *Mass-correct metabolic rate*

Description

For most organisms, metabolic rate does not scale linearly, but rather according to a power function. This function estimates MO2 or size of an individual organism given the MO2 and size of another individual of a different size. To mass-correct your MO2 data, plug in your desired mass in mass_2 and the output from calc_b to the b parameter.

Usage

```
scale_MO2(mass_1, MO2_1, mass_2, MO2_2, b = 0.75)
```

Arguments

mass_1	animal mass for MO2_1.
MO2_1	metabolic rate for mass_1.
mass_2	animal mass for MO2_2.
MO2_2	metabolic rate for mass_2.
b	scaling coefficient for MO2. Default is 0.75.

Details

$$(MO2 = b0 * M^b)$$

where `b0` is species-specific normalization constant, `M` is mass and `b` is the scaling coefficient which is around 0.75 for many organisms.

For scaling of **mass-specific** metabolic rates, use something closer to `b = -0.25` rather than `b = 0.75`.

Author(s)

Matthew A. Birk, `matthewabirk@gmail.com`

See Also

`Q10`, `calc_b`

Examples

```
# I know a species has an SMR of 800 umol O2/h at 200 g.
# What would be a likely SMR for a 300 g individual?
scale_MO2(mass_1 = 200, MO2_1 = 800, mass_2 = 300)

# Some squids have a much higher scaling coefficient:
scale_MO2(mass_1 = 200, MO2_1 = 800, mass_2 = 300, b = 0.92)

# A 100 g individual at 10 *C has an MO2 of 1270 umol/h. How much
# would a 250 g individual likely consume at 14 *C?
Q10(Q10 = 2, R1 = scale_MO2(mass_1 = 100, MO2_1 = 1270, mass_2 = 250), T1 = 10, T2 = 14)

# Now I have data from real animals and I want to mass-correct them all to a 10 g animal.
mass = 2:20 # obviously not real but you get the point
mo2 = c(44.8, 41, 36, 35, 35, 33.5, 34.5, 40, 30, 23, 27, 30, 25.6, 27.8, 28, 24, 27, 28, 20)
desired_mass = 10

b = calc_b(mass = mass, MO2 = mo2)
scale_MO2(mass_1 = mass, MO2_1 = mo2, mass_2 = desired_mass, b = b$b)

plot(mass, mo2, ylab = 'Raw MO2') # before
plot(mass, scale_MO2(mass_1 = mass, MO2_1 = mo2, mass_2 = 10, b = b$b),
ylab = 'Mass-corrected MO2') # after


# Visualize MO2 scaling by mass and temperature:
mass <- seq(10, 200, 10)
temp <- 10:25
base_mass <- 50
base_temp <- 20
base_MO2 <- 750
mo2 <- outer(mass, temp, function(mass, temp){
scale_MO2(mass_1 = base_mass, mass_2 = mass, MO2_1 = Q10(Q10 = 2, R1 = base_MO2,
```

```
 T1 = base_temp, T2 = temp))
})
persp(mass, temp, mo2, xlab = 'Mass (g)', ylab = 'Temperature (*C)', zlab = 'MO2 (umol / hr)',
 theta = 35, phi = 15, expand = 0.5, ticktype = 'detailed', nticks = 10)
```

Index