Package 'hellometry'

August 24, 2026

Title Biomass Estimation from Allometric Relationships

Version 1.0.1

Description Estimates body size and biomass of organisms from allometric relationships. It works with any data that follow a few column-naming conventions, using the measurements you supply to impute the ones that are missing.

License GPL (>= 3)

URL https://github.com/pierrerogy/hellometry

BugReports https://github.com/pierrerogy/hellometry/issues

Encoding UTF-8

RoxygenNote 7.3.2

Imports dplyr, magrittr, performance, progress, purrr, rlang, stats, tibble, tidyr, tidyselect, utils

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Pierre Rogy [aut, cre] (ORCID: <https://orcid.org/0000-0002-3002-0059>), Olivier Dézerald [ctb] (ORCID: <https://orcid.org/0000-0002-9987-9865>), Gustavo Q. Romero [ctb] (ORCID: <https://orcid.org/0000-0003-3736-4759>), Fabiola Ospina-Bautista [ctb] (ORCID: <https://orcid.org/0000-0003-2498-1459>), Sarah Abdelazim [ctb] (ORCID: <https://orcid.org/0000-0002-6379-2733>), Nicholas A. C. Marino [ctb] (ORCID: <https://orcid.org/0000-0002-5702-5466>), Diane S. Srivastava [ctb] (ORCID: <https://orcid.org/0000-0003-4541-5595>)

Maintainer Pierre Rogy <pierre.rogy@gmail.com>

Repository CRAN

Date/Publication 2026-08-24 14:50:02 UTC

Contents

`bromeliad_inverts_measurements`
Read the `bromeliad_inverts_measurements` *dataset*

Description

Reads the `bromeliad_inverts_measurements` dataset, a compilation of real invertebrate body size (mm) and body mass (mg) measurements from bromeliad communities. If you use the data outside the package, please cite the package.

Usage

```
bromeliad_inverts_measurements()
```

Value

A tibble with real body length and body size measurements from bromeliad invertebrates. It contains columns for taxonomy ("phylum" to "species"), "stage", "abundance", "body_size_mm", "body_mass_mg", "mass_type" (dry/wet), "data_providers" and "note".

Examples

```
# The measurements the package ships with
bromeliad_inverts_measurements()
```

`data_checker` *Set of checks to perform on the data to make sure it can be handled by* `hellometry()`

Description

Will fail if any condition not met

Usage

```
data_checker(dats, biomass_type, model)
```

Arguments

`dats`	Dataframe to be used for estimation
`biomass_type`	"dry"/"wet". Whether the biomass used in inference should be dry (default) or wet. See `dry_wet()` for details.
`model`	"lm"/"poisson". What kind of allometric model should be computed. See `full_estimation_table()` for details.

Value

An error if any check fails.

Examples

```
# Data named the way the package expects
dats <-
  data.frame(species = c("sp_a", "sp_b"),
             stage = "larva",
             abundance = c(2, 5),
             size_col = c("3.1", "unknown"),
             biomass_col = c(0.4, NA),
             biomass_type = "dry")

# Passes quietly
data_checker(dats, biomass_type = "dry", model = "lm")

# Fails on a column that is not named properly
try(data_checker(dats["species"], biomass_type = "dry", model = "lm"))
```

dry_wet *Parsing dry and wet measurements*

Description

Filter dataframe based on required biomass type (dry or wet).

Usage

```
dry_wet(dats, biomass_type)
```

Arguments

dats	Input data
biomass_type	If biomass should be wet or dry, or most common

Value

Dataframe with filtered biomass type, and chosen one (dry or wet)

Examples

```
# Data holding both kinds of biomass
dats <-
  data.frame(species = c("sp_a", "sp_b"),
             stage = "larva",
             size_col = c(3.1, 2.2),
             biomass_col = c(0.4, 0.2),
             biomass_type = c("dry", "wet"))

# Keep the dry measurements only
dry_wet(dats, biomass_type = "dry")
```

full_estimation_table *Returns size and biomass estimates for every taxon in measurement table*

Description

Returns size and biomass estimates for every taxon in measurement table

Usage

```
full_estimation_table(
  level_vec,
  measurement_table,
  what,
  model = "lm",
  traits = FALSE,
  trait_columns = c(),
  id_col = "species",
  r_square_cutoff_upper = 0.95,
  r_square_cutoff_lower = 0,
  p_val_cutoff = 0.05
)
```

Arguments

`level_vec`	Vector of taxonomic levels over which to iterate estimation
`measurement_table`	A table with the numerical measurements and biomass used to compute allometric lms
`what`	Should size (what = "size_col") or biomass (what = "biomass_col") be estimated?
`model`	What kind of model should be computed, "lm" (default) for a linear model on the log10-log10 scale, or "poisson" for a Poisson glm with a log link
`traits`	Should the table be computed by traits or not? Default FALSE
`trait_columns`	List of traits to match, should be column names in measurement_table
`id_col`	Name of the column holding a unique identifier per species/taxon, used only when `traits = TRUE`. Default "species".
`r_square_cutoff_upper`	Upper cutoff for R2 in allometric models, models with values above it will not be used in estimation. Default is 0.95 to avoid overfit models
`r_square_cutoff_lower`	Lower cutoff for R2 in allometric models, models with values below it will not be used in estimation. Default is 0.
`p_val_cutoff`	Upper cutoff for p-value of allometric models, models with p_value above it will not be used in estimation. Default is 0.05.

Value

Tibble with size estimates or allometric models used in estimation.

Examples

```
# Reference measurements for one family, renamed to the columns the package expects
measurements <-
  bromeliad_inverts_measurements() %>%
  dplyr::filter(family == "Culicidae") %>%
```

```
  dplyr::rename(size_col = body_size_mm,
                biomass_col = body_mass_mg,
                biomass_type = mass_type) %>%
  dplyr::mutate(size_col = as.character(size_col),
                biomass_col = as.numeric(biomass_col))

# Taxa to estimate, with no measurement of their own
communities <-
  trini_communities() %>%
  dplyr::filter(family == "Culicidae") %>%
  dplyr::rename(abundance = n) %>%
  dplyr::mutate(size_col = "unknown",
                biomass_col = NA,
                biomass_type = "dry")

# The two together are what the package works on
dats <-
  dplyr::bind_rows(communities, measurements)
level_vec <-
  c("species", "genus", "family")

measurement_table <-
  make_measurement_table(dats = dats, level_vec = level_vec)

# Every size estimate the data allows
full_estimation_table(level_vec = level_vec,
                      measurement_table = measurement_table,
                      what = "size_col")

# Every allometric model the data allows, on dry biomass
full_estimation_table(level_vec = level_vec,
                      measurement_table = dry_wet(measurement_table,
                                                  biomass_type = "dry"),
                      what = "biomass_col")
```

`hellometry` *Wrapper function to estimate size (mm) and biomass (mg) from an input dataset*

Description

Wrapper function, input your data and get size and measurement. Please name column with number of specimen "abundance", column with measurement "size_col", column with life stage (e.g. larva/adult) "stage", column with biomass type (dry/wet) "biomass_type", and the column with biomass "biomass_col". If you do not have a numerical measurement for a given specimen, the function can do size estimations if you input "small", "medium", "large" or "unknown". In this case, the algorithm will use existing size measurements for the species, and use the size distribution to estimate "small", "medium" and "large" inputs, or a weighted average of all measurements if the input is "unknown". For biomass measurements, just leave NA for those cells you want estimates in.

Usage

```
hellometry(
  dats,
  level_vec,
  biomass_type = "dry",
  model = "lm",
  r_square_cutoff_upper = 0.95,
  r_square_cutoff_lower = 0,
  p_val_cutoff = 0.05
)
```

Arguments

dats	The input data table, please include columns "abundance", "size_col", "biomass_col", "stage" (larva/pupa/adult).
level_vec	A character vector indicating the taxonomic levels present in your data, in increasing order of coarseness.
biomass_type	"dry"/"wet". Should data used in inference be "dry" for just dry biomass (default), or "wet" for just wet biomass. See `dry_wet()` for more information on how this works.
model	What kind of allometric model should be computed, "lm" (default) for a linear model on the log10-log10 scale, or "poisson" for a Poisson glm with a log link.
r_square_cutoff_upper	Upper cutoff for R2 in allometric models, models with values above it will not be used in estimation. Default is 0.95 to avoid overfit models
r_square_cutoff_lower	Lower cutoff for R2 in allometric models, models with values below it will not be used in estimation. Default is 0.
p_val_cutoff	Upper cutoff for p-value of allometric models, models with p_value above it will not be used in estimation. Default is 0.05.

Details

Please make sure that the level_vec argument has levels in increasing order of coarseness, e.g. from species to order.

Value

A list with three tibbles:

- data: the input data with added size and biomass estimates, and new columns with the taxonomic level and name of the taxon at which the estimate was made.
- size_estimates: tibble with size estimates that were used, with columns for taxonomic level, name and size category of the estimation.
- model_estimates: tibble with biomass models that were used, with columns for taxonomic level and name. See `full_estimation_table()` to get all possible size estimates and models for your data

Examples

```
# Reference measurements for one family, renamed to the columns the package expects
measurements <-
  bromeliad_inverts_measurements() %>%
  dplyr::filter(family == "Culicidae") %>%
  dplyr::rename(size_col = body_size_mm,
                biomass_col = body_mass_mg,
                biomass_type = mass_type) %>%
  dplyr::mutate(size_col = as.character(size_col),
                biomass_col = as.numeric(biomass_col))

# Taxa to estimate, with no measurement of their own
communities <-
  trini_communities() %>%
  dplyr::filter(family == "Culicidae") %>%
  dplyr::rename(abundance = n) %>%
  dplyr::mutate(size_col = "unknown",
                biomass_col = NA,
                biomass_type = "dry")

# The two together are what the package works on
dats <-
  dplyr::bind_rows(communities, measurements)
level_vec <-
  c("species", "genus", "family")

res <-
  hellometry(dats = dats, level_vec = level_vec, biomass_type = "dry")

# The estimates, joined back to the data
res$data[, c("genus", "size_col", "biomass_col", "size_level", "model_level")]

# The size estimates and the models they came from
res$size_estimates
res$model_estimates
```

`make_measurement_table` *Build measurement table to be used for biomass estimations*

Description

Compiles the table of numerical measurements used to estimate sizes and fit allometric models. The table is built entirely from the data you supply: every row with a numerical `size_col` is kept (expanded by abundance so each row represents one individual), carrying its taxonomy along.

Usage

```
make_measurement_table(dats, level_vec)
```

Arguments

dats	Dataframe to be used for estimation.
level_vec	Vector of taxonomic levels to be used in the measurement table.

Value

A table with measurements and taxonomy, ready to be used for size and biomass estimation.

Examples

```
# Reference measurements for one family, renamed to the columns the package expects
measurements <-
  bromeliad_inverts_measurements() %>%
  dplyr::filter(family == "Culicidae") %>%
  dplyr::rename(size_col = body_size_mm,
                biomass_col = body_mass_mg,
                biomass_type = mass_type) %>%
  dplyr::mutate(size_col = as.character(size_col),
                biomass_col = as.numeric(biomass_col))

# Taxa to estimate, with no measurement of their own
communities <-
  trini_communities() %>%
  dplyr::filter(family == "Culicidae") %>%
  dplyr::rename(abundance = n) %>%
  dplyr::mutate(size_col = "unknown",
                biomass_col = NA,
                biomass_type = "dry")

# The two together are what the package works on
dats <-
  dplyr::bind_rows(communities, measurements)
level_vec <-
  c("species", "genus", "family")

# One row per measured individual, taxonomy carried along
make_measurement_table(dats = dats, level_vec = level_vec)
```

---

make_trait_table *Prepare data to do estimation with traits in* full_estimation_table

---

Description

Remove columns with taxonomy, and replace them with trait-based groupings

Usage

```
make_trait_table(measurement_table, trait_columns, id_col = "species")
```

Arguments

measurement_table	A table with the numerical measurements and biomass used to compute allometric lms
trait_columns	List of traits to match, should be column names in measurement_table
id_col	Name of the column holding a unique identifier per species/taxon. Default "species".

Value

List of tibbles, one per trait grouping, each carrying the measurements of the species sharing that grouping's traits

Examples

```
# Fuzzy traits of three species, sp_a and sp_b within +/- 1 of each other
measurement_table <-
  data.frame(species = rep(c("sp_a", "sp_b", "sp_c"), each = 4),
             stage = "larva",
             size_col = c(1, 2, 3, 4, 1.2, 2.1, 3.3, 4.2, 8, 9, 10, 11),
             biomass_col = c(0.1, 0.4, 1, 2, 0.2, 0.5, 1.1, 2.2, 12, 15, 20, 26),
             AS1 = rep(c(1, 2, 5), each = 4),
             AS2 = rep(c(3, 3, 9), each = 4))

# The measurements of the species sharing each trait grouping
make_trait_table(measurement_table, trait_columns = c("AS1", "AS2"))
```

---

matcher_of_traits *Finds which species have matching traits*

Description

Evaluates all unique trait groupings and returns species that have the same traits +/- 1

Usage

```
matcher_of_traits(measurement_table, trait_columns, id_col = "species")
```

Arguments

measurement_table	A table with the numerical measurements and biomass used to compute allometric lms
trait_columns	List of traits to match, should be column names in measurement_table
id_col	Name of the column holding a unique identifier per species/taxon. Default "species".

Value

Tibble of four columns: level = "traits", name = focus species, stage of focus species, id = matched species

Examples

```
# Fuzzy traits of three species, sp_a and sp_b within +/- 1 of each other
measurement_table <-
  data.frame(species = rep(c("sp_a", "sp_b", "sp_c"), each = 4),
             stage = "larva",
             size_col = c(1, 2, 3, 4, 1.2, 2.1, 3.3, 4.2, 8, 9, 10, 11),
             biomass_col = c(0.1, 0.4, 1, 2, 0.2, 0.5, 1.1, 2.2, 12, 15, 20, 26),
             AS1 = rep(c(1, 2, 5), each = 4),
             AS2 = rep(c(3, 3, 9), each = 4))

# sp_a and sp_b match each other, sp_c only itself
matcher_of_traits(measurement_table, trait_columns = c("AS1", "AS2"))
```

---

predict_biomass *Estimate biomass from a fitted allometric model*

Description

Wraps `predict()` for every kind of model `full_estimation_table()` can fit, and returns the biomass of a whole row on the scale of the supplied data

Usage

```
predict_biomass(model, size, abundance)
```

Arguments

`model`	A fitted allometric model
`size`	The size to estimate biomass for
`abundance`	The number of individuals the row holds

Details

Both kinds return a confidence interval, i.e. the uncertainty around the mean predicted biomass. The estimate is scaled to the abundance of the row while still on the scale the model was fitted on (log10 for an lm, the log link for a Poisson glm), and then back-transformed.

Value

Three values: the biomass estimate, its lower and its upper bound

Examples

```
# Any allometric model will do
mod <-
  lm(log10(biomass_col) ~ log10(size_col),
     data = data.frame(size_col = c(1, 2, 3, 4, 5),
                       biomass_col = c(0.1, 0.5, 1.2, 2.4, 4.1)))

# Biomass of a row holding ten individuals of size 3, with its bounds
predict_biomass(mod, size = 3, abundance = 10)
```

`r_squared` *Get the R2 of a fitted allometric model*

Description

Wraps `performance::r2()`, which returns the R2 suited to the model it is given, so that any kind of model can be filtered the same way in `full_estimation_table()`

Usage

```
r_squared(model)
```

Arguments

`model`	A fitted allometric model

Value

A single R2 value, NA if none could be computed

Examples

```
# Any allometric model will do
mod <-
  lm(log10(biomass_col) ~ log10(size_col),
     data = data.frame(size_col = c(1, 2, 3, 4, 5),
                       biomass_col = c(0.1, 0.5, 1.2, 2.4, 4.1)))

r_squared(mod)
```

trini_communities *Read the* trini_communities *dataset*

Description

Reads the trini_communities dataset, a set of invertebrate community samples from 17 Trinidadian bromeliads (from @Rogy2024). It is provided as a toy dataset to try out the package, alongside bromeliad_inverts_measurements(). Each taxon also carries a set of discrete (fuzzy) trait columns, matched from a trait database (@Cereghino2018), that can be returned with traits = TRUE to illustrate the trait-based grouping functions.

Usage

```
trini_communities(traits = FALSE)
```

Arguments

traits	Logical; if TRUE, the returned tibble includes the fuzzy trait columns. If FALSE (the default), only the taxonomy, abundance, bromeliad ID and stage columns are returned.

Value

A tibble with one row per taxon per bromeliad, with columns for taxonomy, abundance, and bromeliad ID, and, when traits = TRUE, a set of discrete (fuzzy) trait columns.

Examples

```
# The communities the package ships with
trini_communities()

# The same, with the fuzzy trait columns
trini_communities(traits = TRUE)
```

%notin% *Not In Operator*

Description

Returns logical vector indicating if elements are **not** in a given set.

Usage

```
x %notin% table
```

Arguments

x	vector or NULL: the values to be matched.
table	vector or NULL: the values to be matched against.

Value

A logical vector of the same length as x.

Examples

```
c("a", "b") %notin% c("b", "c")
```

Index