Package 'popbayes'

November 5, 2021

Type Package

Version 1.0

Title Bayesian Model to Estimate Population Trends from Counts Series

Description Infers the trends of one or several animal populations over time from series of counts. It does so by accounting for count precision (provided or inferred based on expert knowledge, e.g. guesstimates), smoothing the population rate of increase over time, and accounting for the maximum demographic potential of species. Inference is carried out in a Bayesian framework. This work is part of the FRB-CESAB working group AfroBioDrivers <https://www.fondationbiodiversite.fr/en/the-frb-in-action/programs-and-projects/le-cesab/afrobiodrivers/>.

URL https://frbcesab.github.io/popbayes/, https://github.com/frbcesab/popbayes

BugReports https://github.com/frbcesab/popbayes/issues

Depends R (>= 3.5)

License GPL (>= 2)

Encoding UTF-8

LazyData true

NeedsCompilation no

SystemRequirements JAGS (>= 4.1)

Imports graphics, grDevices, R2jags, stats, usethis

Suggests knitr, rmarkdown

RoxygenNote 7.1.2

VignetteBuilder knitr

Author Nicolas Casajus [aut, cre, cph] (<https://orcid.org/0000-0002-5537-5294>), Roger Pradel [aut] (<https://orcid.org/0000-0002-2684-9251>)

Maintainer Nicolas Casajus <nicolas.casajus@fondationbiodiversite.fr>

Repository CRAN

Date/Publication 2021-11-05 08:10:08 UTC

R topics documented:

`bugs_to_df` *Extract estimated parameters from a list of BUGS outputs*

Description

From the output of the function `fit_trend()` (or `read_bugs()`), this function extracts estimated parameters into a `data.frame`.

The resulting `data.frame` has no particular use in `popbayes` but it can be useful for users.

Usage

```
bugs_to_df(data)
```

Arguments

`data`	a named `list` of BUGS outputs. The output of `fit_trend()` or `read_bugs()`

Value

A `data.frame`.

Examples

```
## Load Garamba raw dataset ----
file_path <- system.file("extdata", "garamba_survey.csv",
                         package = "popbayes")

garamba <- read.csv(file = file_path)

## Create temporary folder ----
```

```
temp_path <- tempdir()

## Format dataset ----
garamba_formatted <- popbayes::format_data(garamba, path = temp_path)

## Select one serie ----
a_buselaphus <- popbayes::filter_series(garamba_formatted,
                                        location = "Garamba",
                                        species  = "Alcelaphus buselaphus")

## Fit population trends (requires JAGS) ----
a_buselaphus_mod <- popbayes::fit_trend(a_buselaphus, path = temp_path)

## Import BUGS outputs for one count series ----
bugs <- popbayes::read_bugs(series = "garamba__alcelaphus_buselaphus",
                            path   = temp_path)

## Extract estimated parameters ----
popbayes::bugs_to_df(bugs)
```

`diagnostic` *Check if a BUGS model has converged*

Description

From the output of the function `fit_trend()` (or `read_bugs()`), this function checks if the estimation of all parameters of one (or several) BUGS model has converged. This diagnostic is performed by comparing the `Rhat` value of each parameter to a `threshold` (default is `1.1`). If some `Rhat` values are greater than this threshold (no convergence), a message listing problematic models is displayed.

Usage

```
diagnostic(data, threshold = 1.1)
```

Arguments

`data`	a named `list` of BUGS outputs. The output of `fit_trend()` or `read_bugs()`.
`threshold`	a `numeric`.

Value

No return value.

Examples

```
## Load Garamba raw dataset ----
file_path <- system.file("extdata", "garamba_survey.csv",
                         package = "popbayes")

garamba <- read.csv(file = file_path)

## Create temporary folder ----
temp_path <- tempdir()

## Format dataset ----
garamba_formatted <- popbayes::format_data(garamba, path = temp_path)

## Select one serie ----
a_buselaphus <- popbayes::filter_series(garamba_formatted,
                                        location = "Garamba",
                                        species  = "Alcelaphus buselaphus")

## Fit population trends (requires JAGS) ----
a_buselaphus_mod <- popbayes::fit_trend(a_buselaphus, path = temp_path)

## Check for convergence ----
popbayes::diagnostic(a_buselaphus_mod)
```

---

filter_series *Extract the count series corresponding to a location and/or a species*

Description

This function identifies the count series relative to a species and/or a location in a named list like the output of function `format_data()`. If both species and location are provided, the series of counts of the species at the specified location is extracted. Otherwise, all series corresponding to the specified criterion (species or location) are extracted.

Usage

```
filter_series(data, species = NULL, location = NULL)
```

Arguments

`data`	a named `list`. The output of function `format_data()`.
`species`	a `character` string. A species name.
`location`	a `character` string. A site name.

Value

A subset of `data`, i.e. a named `list`.

Examples

```
## Load Garamba raw dataset ----
file_path <- system.file("extdata", "garamba_survey.csv",
                         package = "popbayes")

garamba <- read.csv(file = file_path)

## Create temporary folder ----
temp_path <- tempdir()

## Format dataset ----
garamba_formatted <- popbayes::format_data(garamba, path = temp_path)

## Number of count series ----
length(garamba_formatted)

## Retrieve count series names ----
popbayes::list_series(path = temp_path)

## Get data for Alcelaphus buselaphus (at all sites) ----
x <- popbayes::filter_series(garamba_formatted,
                             species = "Alcelaphus buselaphus")

## Get data at Garamba (for all species) ----
x <- popbayes::filter_series(garamba_formatted,
                             location = "Garamba")

## Get data for Alcelaphus buselaphus at Garamba only ----
x <- popbayes::filter_series(garamba_formatted,
                             location = "Garamba",
                             species  = "Alcelaphus buselaphus")
```

`fit_trend` *Fit a Bayesian model to estimate population size trend*

Description

This function applies a Bayesian model to count series in order to infer the population trend over time. This function only works on the output of `format_data()` or `filter_series()`.

Important: This function uses `R2jags::jags()` and the freeware **JAGS** (`https://mcmc-jags.sourceforge.io/`) must be installed.

There are two types of options: model options (argument `model_opts`) and MCMC options (argument `mcmc_opts`).

A. Model options

- a smoothing factor: the precision (the inverse of variance) of a normal distribution centered on the current relative rate of increase r from which the next candidate relative rate of increase (see below) is drawn. The highest this number, the tighter the link between successive relative rates of increase. The default `100` corresponds to a moderate link.

- a logical indicating whether the population growth must remain limited by the species demographic potential (provided by the argument rmax in format_data()).

The relative rate of increase is the change in log population size between two dates. The quantity actually being modeled is the relative rate of increase per unit of time (usually one date). This quantity reflects more directly the prevailing conditions than the population size itself, which is the reason why it has been chosen.

When the second model option is set to TRUE, the candidate rate of increase is compared to the maximum relative rate of increase (obtained when using format_data()) and replaced by rmax if greater.

B. MCMC options

Classical Markov chain Monte Carlo (MCMC) settings (see argument mcmc_opts below).

Usage

```
fit_trend(
  data,
  model_opts = list(100, TRUE),
  mcmc_opts = list(ni = 50000, nt = 3, nb = 10000, nc = 2),
  path = "."
)
```

Arguments

data	a named list. The output of format_data() or filter_series().
model_opts	a list of two vectors. The model smoothing factor (numeric) and a logical indicating if the parameter **r** must be limited by rmax (TRUE) or not (FALSE). If this second parameter is TRUE, the argument rmax cannot be NULL unless species are listed in popbayes (in species_info).
mcmc_opts	a list containing the number of iterations (ni), the thinning factor (nt), the length of burn in (nb), i.e. the number of iterations to discard at the beginning, and the number of chains (nc).
path	a character string. The directory to save BUGS outputs (the same as in format_data()).

Value

An n-element list (where n is the number of count series). Each element of the list is a BUGS output as provided by JAGS (also written in the folder path).

Examples

```
## Load Garamba raw dataset ----
file_path <- system.file("extdata", "garamba_survey.csv",
                         package = "popbayes")

garamba <- read.csv(file = file_path)

## Create temporary folder ----
```

```
temp_path <- tempdir()

## Format dataset ----
garamba_formatted <- popbayes::format_data(garamba, path = temp_path)

## Get data for Alcelaphus buselaphus at Garamba only ----
a_buselaphus <- popbayes::filter_series(garamba_formatted,
                                        location = "Garamba",
                                        species  = "Alcelaphus buselaphus")


## Fit population trend (requires JAGS) ----
a_buselaphus_mod <- popbayes::fit_trend(a_buselaphus, path = temp_path)

## Check for convergence ----
popbayes::diagnostic(a_buselaphus_mod, threshold = 1.1)

## Plot estimated population trend ----
popbayes::plot_trend(series = "garamba__alcelaphus_buselaphus",
                     path   = temp_path)

## Plot MCMC traceplot ----
R2jags::traceplot(a_buselaphus_mod[[1]], ask = TRUE)
```

`format_data` *Format count series*

Description

This function provides an easy way to get count series ready to be analyzed by the package `popbayes`. It must be used prior to all other functions.

This function formats the count series (passed through the argument `data`) by selecting and renaming columns, checking columns format and content, and removing missing data (if `na_rm = TRUE`). It converts the original data frame into a list of count series that will be analyzed later by the function `fit_trend()` to estimate population trends.

To be usable for the estimation of population trends, counts must be accompanied by information on precision. The population trend model requires a 95% confident interval (CI). If estimates are total counts or guesstimates, this function will construct boundaries of the 95% CI by applying the rules set out in https://frbcesab.github.io/popbayes/articles/popbayes.html. If counts were estimated by a sampling method the user needs to specify a measure of precision. Precision is preferably provided in the form of a 95% CI by means of two fields: `lower_ci` and `upper_ci`. It may also be given in the form of a standard deviation (`sd`), a variance (`var`), or a coefficient of variation (`cv`). If the fields `lower_ci` and `upper_ci` are both absent (or `NA`), fields `sd`, `var`, and `cv` are examined in this order. When one is found valid (no missing value), a 95% CI is derived assuming a normal distribution. The field `stat_method` must be present in `data` to indicate if counts are **total counts** (`'T'`), **sampling** (`'S'`), or **guesstimate** (`'G'`).

If a series mixes aerial and ground counts, a field `field_method` must also be present and must contain either `'A'` (aerial counts), or `'G'` (ground counts). As all counts must eventually refer to the same field method for a correct estimation of trend, a conversion will be performed to homogenize counts. This conversion is based on a **preferred field method** and a **conversion factor** both specific to a species/category. The preferred field method specifies the conversion direction. The conversion factor is the multiplicative factor that must be applied to an aerial count to get an equivalent ground count (note that if the preferred field method is `'A'`, ground counts will be divided by the conversion factor to get the equivalent aerial count).

The argument `rmax` represents the maximum change in log population size between two dates (i.e. the relative rate of increase). It will be used by `fit_trend()` but must be provided in this function.

These three parameters, named `pref_field_method`, `conversion_A2G`, and `rmax` can be present in `data` or in a second `data.frame` (passed through the argument `info`). Alternatively, the package `popbayes` provides their values for some African large mammals.

Note: If the field `field_method` is absent in `data`, counts are assumed to be obtained with one field method.

Usage

```
format_data(
  data,
  info = NULL,
  date = "date",
  count = "count",
  location = "location",
  species = "species",
  stat_method = "stat_method",
  lower_ci = "lower_ci",
  upper_ci = "upper_ci",
  sd = NULL,
  var = NULL,
  cv = NULL,
  field_method = "field_method",
  pref_field_method = "pref_field_method",
  conversion_A2G = "conversion_A2G",
  rmax = "rmax",
  path = ".",
  na_rm = FALSE
)
```

Arguments

`data`	a `data.frame` with at least five columns: `location`, `species`, `date`, `count`, and `stat_method`. The `stat_method` field indicates the method used to estimate counts. It can contain: `T` (total counts), `G` (guesstimate), and/or `S` (sampling). If individual counts were estimated by **sampling**, additional column(s) providing a measure of precision is also required (e.g. `lower_ci` and `upper_ci`, or `sd`, `cv`, `var`). Precision metrics can be different between counts. For instance, some

sampling counts can have a sd value and others lower_ci and upper_ci. In that case three columns are required (lower_ci, upper_ci, and sd). See above section **Description** for further information on the computation of the 95% confident interval of estimates.

If the individuals were counted by different methods, an additional field field_method is also required. It can contain: G (ground counts) and/or A (aerial counts). See above section **Description** for further information on the counts conversion.

Others fields can be present either in data or info (see below).

info — (optional) a data.frame with species in rows and the following columns: species (species name), pref_field_method, conversion_A2G, and rmax. See above section **Description** for further information on these fields. Default is NULL (i.e. these information must be present in data if not available in popbayes).

date — a character string. The column name in data of the date. This column date must be in a numerical form with possibly a decimal part. Default is 'date'.

count — a character string. The column name in data of the number of individuals. This column must be numerical. Default is 'count'.

location — a character string. The column name in data of the site. This field is used to distinguish count series from different sites (if required) and to create an unique series name. Default is 'location'.

species — a character string. The column name in data (and in info if provided) of the species. This field is used to distinguish count series for different species (if required) and to create an unique series name. Default is 'species'.

stat_method — a character string. The column name in data of the method used to estimate individuals counts. It can contain 'T' (total counts), 'G' (guesstimate), and/or 'S' (sampling). If some counts are coded as 'S', precision column(s) must also be provided (see below). Default is 'stat_method'.

lower_ci — (optional) a character string. The column name in data of the lower boundary of the 95% CI of the estimate (i.e. count). If provided, the upper boundary of the 95% CI (argument upper_ci) must be also provided. This argument is only required if some counts have been estimated by a sampling method. But user may prefer use other precision measures, e.g. standard deviation (argument sd), variance (argument var), or coefficient of variation (argument cv). Default is 'lower_ci'.

upper_ci — (optional) a character string. The column name in data of the upper boundary of the 95% CI of the estimate (i.e. count). If provided, the lower boundary of the 95% CI (argument lower_ci) must be also provided. Default is 'upper_ci'.

sd — (optional) a character string. The column name in data of the standard deviation of the estimate. Default is NULL.

var — (optional) a character string. The column name in data of the variance of the estimate. Default is NULL.

cv — (optional) a character string. The column name in data of the coefficient of variation of the estimate. Default is NULL.

field_method — (optional) a character string. The column name in data of the field method used to count individuals. Counts can be ground counts (coded as 'G') or aerial counts (coded as 'A'). This argument is optional if individuals have been

counted by the same method. See above section **Description** for further information on the count conversion. Default is `'field_method'`.

`pref_field_method`
: (optional) a `character` string. The column name in `data` of the preferred field method of the species. This argument is only required is `field_method` is not `NULL` (i.e. individuals have been counted by different methods). Alternatively, this value can be passed in `info` (or internally retrieved if the species is listed in the package). See above section **Description** for further information on the count conversion. Default is `'pref_field_method'`.

`conversion_A2G`
: (optional) a `character` string. The column name in `data` of the count conversion factor of the species. This argument is only required if `field_method` is not `NULL` (i.e. individuals have been counted by different methods). Alternatively this value can be passed in `info` (or internally retrieved if the species is listed in the package). See above section **Description** for further information on the count conversion. Default is `'conversion_A2G'`.

`rmax`
: (optional) a `character` string. The column name in `data` of the species demographic potential (i.e. the relative rate of increase of the population). This is the change in log population size between two dates and will be used later by `fit_trend()`. Default is `'rmax'`.

`path`
: a `character` string. The directory to save formatted data. This directory must exist and can be an absolute or a relative path. Default is the current working directory.

`na_rm`
: a `logical`. If `TRUE`, counts with `NA` values will be removed. Default is `FALSE` (returns an error to inform user if `NA` are detected).

Value

An `n`-elements `list` (where `n` is the number of count series). The name of each element of this list is a combination of location and species. Each element of the list is a `list` with the following content:

- `location` a `character` string. The name of the series site.
- `species` a `character` string. The name of the series species.
- `date` a `numerical` vector. The sequence of dates of the series.
- `n_dates` an `integer`. The number of unique dates.
- `stat_methods` a `character` vector. The different stat methods of the series.
- `field_methods` (optional) a `character` vector. The different field methods of the series.
- `pref_field_method` (optional) a `character` string. The preferred field method of the species (`'A'` or `'G'`).
- `conversion_A2G` (optional) a `numeric`. The conversion factor of the species used to convert counts to its preferred field method.
- `rmax` a `numeric`. The maximum population growth rate of the species.
- `data_original` a `data.frame`. Original data of the series with renamed columns. Some rows may have been deleted (if `na_rm = TRUE`).

- `data_converted` a `data.frame`. Data containing computed boundaries of the 95% CI (`lower_ci_conv` and `upper_ci_conv`). If counts have been obtained by different field methods, contains also converted counts (`count_conv`) based on the preferred field method and conversion factor of the species. This `data.frame` will be used by the function `fit_trend()` to fit population models.

Note: Some original series can be discarded if one of these two conditions is met: 1) the series contains only zero counts, and 2) the series contains only a few dates (< 4 dates).

Examples

```
## Load Garamba raw dataset ----
file_path <- system.file("extdata", "garamba_survey.csv",
                         package = "popbayes")

garamba <- read.csv(file = file_path)

## Create temporary folder ----
temp_path <- tempdir()

## Format dataset ----
garamba_formatted <- popbayes::format_data(garamba, path = temp_path)

## Number of count series ----
length(garamba_formatted)

## Retrieve count series names ----
popbayes::list_series(path = temp_path)

## Print content of the first count series ----
names(garamba_formatted[[1]])

## Print original data ----
garamba_formatted[[1]]$"data_original"

## Print converted data ----
garamba_formatted[[1]]$"data_converted"
```

---

garamba *African mammals survey in the Garamba National Park*

Description

This dataset contains individual counts of 10 African mammal species in the Garamba National Park (Democratic Republic of the Congo) from 1976 to 2017.

Usage

```
garamba
```

Format

A `data.frame` with 141 rows (counts) and the following 8 variables:

location the location of the survey (Garamba)

species the binomial name of the species

date the date of the survey

stat_method the method used to estimate individuals counts. One of `T` (total counts), `G` (guesstimate), and `S` (sampling counts)

field_method the field method used to collect data. One of `A` (aerial counts), and `G` (ground counts)

count number of individuals

lower_ci lower boundary of the 95% confidence interval of the counts (only for sampling counts)

upper_ci upper boundary of the 95% confidence interval of the counts (only for sampling counts)

pref_field_method the preferred field method of the species. One of `A` for Aerial counts, and `G` for Ground counts

conversion_A2G the conversion multiplicative factor (corresponding to the detectability category) used to convert aerial to ground counts

rmax the maximum population growth rate

References

Hillman Smith K & Kalpers J (2015) *Garamba Conservation in Peace & War*. Hillman Smith publisher, 448pp. ISBN: 9789966185105.
Monico M (2014) *Aerial Survey Report March 2014 - Garamba National Park, DRC*. African Parks Network/ICCN/Pan-African Elephant Aerial Survey, 42pp.
Spies K *et al.* (2017) *Aerial Survey Report April 2017 - Garamba National Park, DRC*. African Parks Network/EU/ICCN. 38pp.

Examples

```
data("garamba")
head(garamba, 30)
```

`list_series` *Retrieve the count series names*

Description

This function retrieves the count series names generated by the function `format_data()`.

Usage

```
list_series(path = ".")
```

Arguments

path	a character string. The directory in which count series have been saved by the function format_data().

Value

A vector of count series names (character strings).

Examples

```
## Load Garamba raw dataset ----
file_path <- system.file("extdata", "garamba_survey.csv",
                         package = "popbayes")

garamba <- read.csv(file = file_path)

## Create temporary folder ----
temp_path <- tempdir()

## Format dataset ----
garamba_formatted <- popbayes::format_data(garamba, path = temp_path)

## Retrieve count series names ----
popbayes::list_series(path = temp_path)
```

---

plot_series *Plot original vs converted counts*

Description

This function plots a panel of two graphics for one count series (previously generated by format_data()):

- on the left side, scatter plot overlapping original (black points) and converted counts (grey points);
- on the right side, scatter plot of converted counts with boundaries of the 95% confident interval.

Usage

```
plot_series(series, title = TRUE, path = ".", path_fig = ".", save = FALSE)
```

Arguments

series	a character string. The count series names (can be retrieved by running list_series()).
title	a logical. If TRUE (default) a title (series name) is added.
path	a character string. The directory in which count series have been saved by the function format_data().

path_fig	a character string. The directory where to save the plot (if save = TRUE). This directory must exist and can be an absolute or a relative path.
save	a logical. If TRUE (default is FALSE) the plot is saved in path_fig.

Value

No return value.

Examples

```
## Load Garamba raw dataset ----
file_path <- system.file("extdata", "garamba_survey.csv",
                         package = "popbayes")

garamba <- read.csv(file = file_path)

## Create temporary folder ----
temp_path <- tempdir()

## Format dataset ----
garamba_formatted <- popbayes::format_data(garamba, path = temp_path)

## Get series names ----
popbayes::list_series(path = temp_path)

## Plot for Alcelaphus buselaphus at Garamba ----
popbayes::plot_series("garamba__alcelaphus_buselaphus", path = temp_path)
```

---

plot_trend *Plot estimated population trend*

Description

This function plots a panel of two graphics for one BUGS model (previously generated by fit_trend()):

- on the left side, the population trend estimated by the Bayesian model (blue line) with the 95% CI (gray envelop). Dots (with intervals) represent converted counts passed to the model (with the 95% CI);
- on the right side, a bar plot of estimated relative growth rates (r) by date. Dark bars are real estimated r.

Usage

```
plot_trend(series, title = TRUE, path = ".", path_fig = ".", save = FALSE)
```

Arguments

series	a character string. The count series name (can be retrieved by running list_series()).
title	a logical. If TRUE (default) a title (series name) is added.
path	a character string. The directory in which count series (and BUGS outputs) have been saved by the function format_data() (and by fit_trend()).
path_fig	a character string. The directory where to save the plot (if save = TRUE). This directory must exist and can be an absolute or a relative path.
save	a logical. If TRUE (default is FALSE) the plot is saved in path_fig.

Value

No return value.

Examples

```
## Load Garamba raw dataset ----
file_path <- system.file("extdata", "garamba_survey.csv",
                         package = "popbayes")

garamba <- read.csv(file = file_path)

## Create temporary folder ----
temp_path <- tempdir()

## Format dataset ----
garamba_formatted <- popbayes::format_data(garamba, path = temp_path)

## Select one serie ----
a_buselaphus <- popbayes::filter_series(garamba_formatted,
                                        location = "Garamba",
                                        species  = "Alcelaphus buselaphus")

## Fit population trends (requires JAGS) ----
a_buselaphus_mod <- popbayes::fit_trend(a_buselaphus, path = temp_path)

## Plot estimated population trend ----
popbayes::plot_trend(series = "garamba__alcelaphus_buselaphus",
                     path   = temp_path)
```

---

read_bugs *Import a list of BUGS outputs previously exported*

Description

This function imports a list of BUGS outputs previously exported by fit_trend(). Users can import one, several, or all models.

Usage

```
read_bugs(series = NULL, path = ".")
```

Arguments

series	a vector of character strings. One or several count series names. If NULL (default) BUGS outputs for all count series will be imported. Users can run list_series() to get the correct spelling of count series names.
path	a character string. The directory in which BUGS outputs have been saved by the function fit_trend().

Value

An n-element list (where n is the number of count series). See fit_trend() for further information.

Examples

```
## Load Garamba raw dataset ----
file_path <- system.file("extdata", "garamba_survey.csv",
                         package = "popbayes")

garamba <- read.csv(file = file_path)

## Create temporary folder ----
temp_path <- tempdir()

## Format dataset ----
garamba_formatted <- popbayes::format_data(garamba, path = temp_path)

## Select one serie ----
a_buselaphus <- popbayes::filter_series(garamba_formatted,
                                        location = "Garamba",
                                        species  = "Alcelaphus buselaphus")

## Fit population trends (requires JAGS) ----
a_buselaphus_mod <- popbayes::fit_trend(a_buselaphus, path = temp_path)

## Import BUGS outputs for one count series ----
popbayes::read_bugs(series = "garamba__alcelaphus_buselaphus",
                    path   = temp_path)

## Import BUGS outputs for all count series ----
popbayes::read_bugs(path = temp_path)
```

`read_series` *Import a list of count series previously exported*

Description

This function imports a list of count series data previously exported by `format_data()`. Users can import one, several, or all count series data.

Usage

```
read_series(series = NULL, path = ".")
```

Arguments

`series`	a vector of `character` strings. One or several count series names to be imported. If `NULL` (default), all available count series will be imported.
`path`	a `character` string. The directory in which count series have been saved by the function `format_data()`.

Value

An n-element `list` (where `n` is the number of count series). See `format_data()` for further information.

Examples

```
## Load Garamba raw dataset ----
file_path <- system.file("extdata", "garamba_survey.csv",
                         package = "popbayes")

garamba <- read.csv(file = file_path)

## Create temporary folder ----
temp_path <- tempdir()

## Format dataset ----
garamba_formatted <- popbayes::format_data(garamba, path = temp_path)

## Import all count series ----
count_series <- popbayes::read_series(path = temp_path)

## Import one count series ----
a_bus <- popbayes::read_series(series = "garamba__alcelaphus_buselaphus",
                               path   = temp_path)
```

series_to_df *Extract original/converted count series data from a list*

Description

From the output of the function format_data() (or filter_series()), this function extracts data.frame containing converted counts (converted = TRUE) or original counts (converted = FALSE) for one, several, or all count series.

The resulting data.frame has no particular use in popbayes but it can be useful for users.

Usage

```
series_to_df(data, converted = TRUE)
```

Arguments

data	a named list. The output of format_data() or filter_series().
converted	a logical. If TRUE (default) extracts converted counts, otherwise returns original counts.

Value

A data.frame.

Examples

```
## Load Garamba raw dataset ----
file_path <- system.file("extdata", "garamba_survey.csv",
                         package = "popbayes")

garamba <- read.csv(file = file_path)

## Create temporary folder ----
temp_path <- tempdir()

## Format dataset ----
garamba_formatted <- popbayes::format_data(garamba, path = temp_path)

## Extract converted count data ----
converted_data <- popbayes::series_to_df(garamba_formatted,
                                         converted = TRUE)

## Extract original count data ----
original_data <- popbayes::series_to_df(garamba_formatted,
                                        converted = FALSE)

dim(converted_data)
dim(original_data)
dim(garamba)
```

`species_info` *Species information dataset*

Description

This dataset contains information about 15 African mammal species. It can be used in the function `format_data()` to convert individual counts estimated from a field method to a preferred field method. The field method can be `A` (aerial counts) or `G` (ground counts). See `format_data()` for further information. It also contains the maximum population growth rate (i.e. the maximum change in log population size).

User can take this dataset as a template to add information for missing species. Note that only `species`, `pref_field_method`, `conversion_A2G`, and `rmax` are required.

Usage

```
species_info
```

Format

A `data.frame` with 15 rows (African mammals species) and the following variables:

order the order of the species

family the family of the species

species the species binomial name

english the species English name

french the species French name

category the detectability category of the species. One of `MLB` for Medium-sized Light and Brown species (20-150kg), `LLB` for Large Light and Brown species (>150kg), `LD` for Large Dark (>150kg), `Elephant`, and `Giraffe`

pref_field_method the preferred field method of the species. One of `A` for Aerial counts, and `G` for Ground counts

conversion_A2G the conversion multiplicative factor (corresponding to the detectability category) used to convert aerial to ground counts

rmax the maximum population growth rate

Examples

```
data("species_info")
species_info
```

w_to_rmax *Compute rmax from adult female body mass*

Description

The demographic potential of a species is limited. The intrinsic rate of increase rmax is the maximum increase in log population size that a species can attain in a year. According to Sinclair (2003), it is related to the body mass of adult females by: $1.375 \times W^{-0.315}$

Usage

```
w_to_rmax(w)
```

Arguments

w	a numerical vector. Adult female body mass (in kg).

Value

A numerical vector of rmax values.

References

Sinclair (2013) Mammal population regulation, keystone processes and ecosystem dynamics. *Philosophical Transactions: Biological Sciences*, **358**, 1729-1740.

Examples

```
## Set adult female body mass ----
body_masses <- c(55, 127)

## Add species names ----
names(body_masses) <- c("Impala", "Tiang")

## Compute species rmax ----
w_to_rmax(body_masses)
```

Index