

Package ‘bbsBayes’

July 8, 2020

Type Package

Title Hierarchical Bayesian Analysis of North American BBS Data

Version 2.3.3.2020

Date 2020-07-08

Imports progress, jagsUI, ggrepel, geofacet, ggplot2, stringr,
grDevices, rgdal, dplyr, sf, tools, latticeExtra, rappdirs,
sbttools

Depends R (>= 3.5)

SystemRequirements JAGS 4.3.0
(<https://sourceforge.net/projects/mcmc-jags/>)

URL <https://github.com/BrandonEdwards/bbsBayes>

NeedsCompilation no

Description The North American Breeding Bird Survey (BBS) is a long-running program that seeks to monitor the status and trends of the breeding birds in North America. Since its start in 1966, the BBS has accumulated over 50 years of data for over 500 species of North American Birds. Given the temporal and spatial structure of the data, hierarchical Bayesian models are used to assess the status and trends of these 500+ species of birds. 'bbsBayes' allows you to perform hierarchical Bayesian analysis of BBS data. You can run a full model analysis for one or more species that you choose, or you can take more control and specify how the data should be stratified, prepared for 'JAGS', or modelled. The functions provided here allow you to replicate analyses performed by the United State Geological Survey (USGS, see Link and Sauer (2011) <doi:10.1525/auk.2010.09220>) and Canadian Wildlife Service (CWS, see Smith and Edwards (2020) <doi:10.1101/2020.03.26.010215>).

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

Author Brandon P.M. Edwards [aut, cre],
Adam C. Smith [aut]

Maintainer Brandon P.M. Edwards <edwardsb@uoguelph.ca>

Repository CRAN

Date/Publication 2020-07-08 17:10:03 UTC

R topics documented:

bird_sample	2
fetch_bbs_data	3
generate_indices	4
generate_map	7
generate_trends	8
geofacet_plot	11
get_composite_regions	13
get_final_values	14
get_mcmc_list	15
get_prepared_data	16
get_strata_area	17
load_bbs_data	18
load_sample_data	19
lppd	20
model_to_file	21
plot_indices	22
prepare_jags_data	24
route_sample	26
run_model	27
r_hat	29
species_sample	30
stratify	31
Index	33

bird_sample	<i>Sample point count data per species (Pacific Wren only)</i>
-------------	--

Description

A sample dataset containing 10-stop counts of each bird species seen per route per year. NOTE: This only contains data for Pacific Wren, not the full data set. The full count set is obtained via the function `fetch_bbs_data`. The data is obtained from the United States Geological Survey and is subject to change as new data is added each year. See Details for citation.

Usage

bird_sample

Format

A data frame with 20 rows and 15 variables:

statenum Numerical representation of state or province

Route Numerical representation of the route the species was observed

countrynum Numerical representation of the country

RouteDataID Unique code for each year a route was run

RPID Run Protocol ID

Year Year the species was observed on the route

AOU Numerical representation of the species, designated by American Ornithological Union

Count10 Counts for stops 1-10

Count20 Counts for stops 11-20

Count30 Counts for stops 21-30

Count40 Counts for stops 31-40

Count50 Counts for stops 41-50

StopTotal Count for all stops

SpeciesTotal Total count for the species on the route run

BCR Bird Conservation Region the route was run in

Details

Pardieck, K.L., D.J. Ziolkowski Jr., M. Lutmerding, V. Aponte and M-A.R. Hudson. 2019. North American Breeding Bird Survey Dataset 1966 - 2018, version 2018.0. U.S. Geological Survey, Patuxent Wildlife Research Center. <https://doi.org/10.5066/P9HE8XYJ>.

fetch_bbs_data

Fetch Breeding Bird Survey dataset

Description

fetch_bbs_data uses File Transfer Protocol (FTP) to fetch Breeding Bird Survey data from the United States Geological Survey (USGS) FTP site. This is the raw data that is uploaded to the site before any analyses are performed. A package-specific directory is created on the user's computer (see documentation of `rappdirs::appdir` for details of where this directory lives), and the BBS data is saved to that directory for use by other functions. Before downloading any data, the user must thoroughly read through the terms and conditions of the user of the data and type the word "yes" to agree.

Usage

```
fetch_bbs_data(level = "state", quiet = FALSE, force = FALSE)
```

Arguments

level	A string, either "state" or "stop", specifying which counts to fetch. Defaults to "state", which provides counts beginning in 1966, aggregated in five bins, each of which contains cumulative counts from 10 of the 50 stops along a route. Specifying "stop" provides stop-level data beginning in 1997, which includes counts for each stop along routes individually. Note that stop-level data is not currently supported by the modeling utilities in bbsBayes.
quiet	Logical: should progress bars be suppressed? Defaults to FALSE
force	Logical: if BBS data already exists on computer, should it be overwritten? Defaults to FALSE

Value

None

generate_indices	<i>Generate regional annual indices of abundance continent and strata and optionally for countries, states/provinces, or BCRs from analyses run on the stratifications that support these composite regions</i>
------------------	---

Description

generate_indices creates a data frame of the strata-weighted regional indices by year. This data frame can then be used to generate a population trajectory of the species.

Usage

```
generate_indices(jags_mod = NULL, jags_data = NULL,
  quantiles = c(0.025, 0.05, 0.25, 0.75, 0.95, 0.975),
  regions = c("stratum", "continental"), alternate_n = "n",
  startyear = NULL, drop_exclude = FALSE, max_backcast = NULL,
  alt_region_names = NULL)
```

Arguments

jags_mod	JAGS list generated by run_model
jags_data	data object used in run_model
quantiles	vector of quantiles to be sampled from the posterior distribution Defaults to c(0.025,0.05,0.25,0.5,0.75,0.95,0.975)
regions	vector selecting regional compilation(s) to calculate. Default is "continental","stratum", options also include "national", "prov_state", "bcr", and "bcr_by_country" for the stratifications that include areas that align with those regions.
alternate_n	text string indicating the name of the alternative annual index parameter in a model, Default is "n"

startyear	Optional first year for which to calculate the annual indices if a trajectory for only the more recent portion of the time series is desired. This is probably most relevant if max_backcast is set and so trajectories for different time-periods could include a different subset of strata (i.e., strata removed)
drop_exclude	logical indicating if the strata that exceed the max_backcast threshold should be excluded from the calculations, Default is FALSE (regions are flagged and listed but not dropped)
max_backcast	an optional integer indicating the maximum number of years to backcast the stratum-level estimates before the first year in which the species was observed on any route in that stratum. 5 is used in the CWS national estimates. If the observed data in a given stratum do not include at least one non-zero observation of the species between the first year of the BBS and startyear+max_backcast, the stratum is flagged within the relevant regional summary. Default value, NULL ignores any backcasting limit (i.e., generates annual indices for the entire time series, regardless of when the species was first observed)
alt_region_names	Optional dataframe indicating the strata to include in a custom spatial summary. Generate the basic dataframe structure with the extract_strata_areas function, then modify with an additional column indicating the strata to include in a custom spatial summary

Value

List of 6 objects	
data_summary	dataframe with the following columns
Year	Year of particular index
Region	Region name
Region_alt	Long name for region
Region_type	Type of region including continental, national, Province_State, BCR, bcr_by_country, or stratum
Strata_included	Strata included in the annual index calculations
Strata_excluded	Strata potentially excluded from the annual index calculations because they have no observations of the species in the first part of the time series, see arguments max_backcast and startyear
Index	Strata-weighted count index
additional columns	for each of the values in quantiles quantiles of the posterior distribution
obs_mean	Mean of the observed annual counts of birds across all routes and all years. An alternative estimate of the average relative abundance of the species in the region and year. Differences between this and the annual indices are a function of the model. For composite regions (i.e., anything other than stratum-level estimates) this average count is calculated as an area-weighted average across all strata included

generate_map	<i>Generate a map of trends by strata.</i>
--------------	--

Description

generate_map allows you to generate a colour-coded map of species trends for each strata. Given trends generated by generate_strata_trends, this function will shade in each stratum based on the percent change in that stratum.

Usage

```
generate_map(trend = NULL, select = FALSE, stratify_by = NULL,
            slope = FALSE, species = "")
```

Arguments

trend	Dataframe of strata trends produced by generate_strata_trends or generate_regional_trends(... = "stratum")
select	logical flag to indicate if the stratum data need to be selected out of an trends object that includes continental, national, or other region-types. Default is FALSE
stratify_by	How were the data stratified?
slope	Logical, if TRUE, maps values of the alternative trend metric if slope = TRUE was used in generate_strata_trends, the slope of a log-linear regression through the annual indices. Default FALSE.
species	Text, optional species name to add plot title. if left blank "" no title is added

Value

splot object

Examples

```
# Toy example with Pacific Wren sample data
# First, stratify the sample data
strat_data <- stratify(by = "bbs_cws", sample_data = TRUE)

# Prepare the stratified data for use in a JAGS model.
jags_data <- prepare_jags_data(strat_data = strat_data,
                             species_to_run = "Pacific Wren",
                             model = "firstdiff",
                             min_year = 2009,
                             max_year = 2018)

# Now run a JAGS model.
jags_mod <- run_model(jags_data = jags_data,
                    n_adapt = 0,
                    n_burnin = 0,
```

```

        n_iter = 10,
        n_thin = 1)

# Generate the continental and stratum indices
indices <- generate_indices(jags_mod = jags_mod,
                           jags_data = jags_data)

# Now, generate the trends
trends <- generate_trends(indices = indices)

# Generate the map. Set select = TRUE because we are passing a
# dataframe of trends of more than just the stratum regions
map <- generate_map(trend = trends,
                    stratify_by = "bbs_cws",
                    select = TRUE,
                    species = "Pacific Wren")

```

generate_trends	<i>Generate regional trends continent and strata and optionally for countries, states/provinces, or BCRs from analyses run on the stratifications that support these composite regions</i>
-----------------	--

Description

generate_trends calculates the geometric mean annual changes in population size for composite regions.

Usage

```
generate_trends(indices = NULL, Min_year = NULL, Max_year = NULL,
               quantiles = c(0.025, 0.05, 0.25, 0.75, 0.95, 0.975), slope = FALSE,
               prob_decrease = NULL, prob_increase = NULL)
```

Arguments

indices	regional indices generated by generate_indices
Min_year	Minimum year to calculate trends from (e.g., 1970). Default is NULL, in which case the trend is calculated from the first year of the time-series of the supplied annual_indices file
Max_year	Maximum year to calculate trends to (e.g., 2018). Default is NULL, in which case the trend is calculated up to the last year of the time-series of the supplied annual_indices file
quantiles	vector of quantiles to be sampled from the posterior distribution Defaults to c(0.025,0.05,0.25,0.5,0.75,0.95,0.975)
slope	Logical, if TRUE, calculates an alternative trend metric, the slope of a log-linear regression through the annual indices. Default FALSE

- prob_decrease Optional vector of percent-change values to calculate the posterior probabilities that the population has decreased by at least this much (e.g., prob_decrease = c(50) would result in a calculation of the probability that the population has decreased by more than 50 percent over the period of the trend, i.e., less than half the population remains. Default is NULL, in which case no probability of decrease is calculated.
- prob_increase Optional vector of percent-change values to calculate the posterior probabilities that the population has increased by at least this much (e.g., prob_increase = c(100) would result in a calculation of the probability that the population has increased by more than 100 percent, i.e., doubled, over the period of the trend. Default is NULL, in which case no probability of increase is calculated.

Value

Dataframe with one row for each region included in indices object, and columns including:

Start_year	first year of the trend
End_year	last year of the trend
Region	short name of the region
Region_alt	Long name for region
Region_type	Type of region including continental, national, Province_State, BCR, bcr_by_national, or stratum
Strata_included	Strata included in the trend and annual index calculations
Strata_excluded	Strata potentially excluded from the trend and annual index calculations because they have no observations of the species in the first part of the time series
Trend	Estimated mean annual percent change over the trend time-period (i.e., Start_year - End_year), according to an endpoint comparison of annual index in Start_year and the annual index in End_year
Trend_Q_quantiles	quantiles of the posterior distribution of Trend estimates, matching levels included in the quantiles argument
Percent_Change	Estimated total percent change over the trend time-period
Percent_Change_Q_quantiles	quantiles of the posterior distribution of Percent Change estimates, matching levels included in the quantiles argument
Slope_Trend	Estimated mean annual percent change over the trend time-period, according to the slope of a linear regression through the log-transformed annual indices
Slope_Trend_Q_quantiles	quantiles of the posterior distribution of Percent Change estimates, matching levels included in the quantiles argument
prob_decrease_X_percent	proportion of the posterior distribution of Percent_Change that is below the percentage values supplied in prob_decrease

prob_increase_X_percent	proportion of the posterior distribution of Percent_Change that is above the percentage values supplied in prob_increase
Relative_Abundance	Mean of the annual index values across all years. An estimate of the average relative abundance of the species in the region. Can be interpreted as the predicted average count of the species in an average year on an average route by an average observer, for the years, routes, and observers in the existing data
Observed_Relative_Abundance	Mean of the observed annual counts of birds across all routes and all years. An alternative estimate of the average relative abundance of the species in the region. For composite regions (i.e., anything other than stratum-level estimates) this average count is calculated as an area-weighted average across all strata included
Number_of_Strata	The number of strata included in the region
Width_of_X_percent_Credible_Interval	Width (in percent/year) of the credible interval on the Trend calculation. Calculated for the widest credible interval requested in quantiles argument. Default is 95 percent CI (i.e., Trend_Q0.975 - Trend_Q0.025)
Width_of_X_percent_Credible_Interval_Slope	Width (in percent/year) of the credible interval on the Trend calculation for the slope-based trend. Calculated for the widest credible interval requested in quantiles argument. Default is 95 percent CI (i.e., Slope_Trend_Q0.975 - Slope_Trend_Q0.025)
Number_of_Routes	The number of unique BBS routes included in the trend calculation for this region and species
Mean_Number_of_Routes	The average number of BBS routes across years contributing data for this region and species
backcast_flag	approximate proportion of the included species range*years that are free of extrapolated population trajectories e.g., 1.0 = data cover full time-series, 0.75 = data cover 75 percent of time-series. Only calculated if max_backcast != NULL

Examples

```
# Toy example with Pacific Wren sample data
# First, stratify the sample data
strat_data <- stratify(by = "bbs_cws", sample_data = TRUE)

# Prepare the stratified data for use in a JAGS model.
jags_data <- prepare_jags_data(strat_data = strat_data,
                              species_to_run = "Pacific Wren",
                              model = "firstdiff",
                              min_year = 2009,
                              max_year = 2018)
```

```

# Now run a JAGS model.
jags_mod <- run_model(jags_data = jags_data,
                     n_adapt = 0,
                     n_burnin = 0,
                     n_iter = 10,
                     n_thin = 1)

# Generate the continental and stratum indices
indices <- generate_indices(jags_mod = jags_mod,
                           jags_data = jags_data)

# Now, generate the trends
trends <- generate_trends(indices = indices)

```

geofacet_plot

Generate a geofacet plot of population trajectories by province/state

Description

geofacet_plot allows you to generate a faceted plot of population trajectories for each strata by province/state. Given a model stratified by "state", "bbs_cws", or "bbs_usgs" and indices generated by generate_strata_indices or generate_regional_indices, this function will generate a faceted plot showing the population trajectories. All geofacet plots have one facet per state/province, so if strata-level indices from the "bbs_cws" or "bbs_usgs" are given, the function plots multiple trajectories (one for each of the relevant strata) within each facet.

Usage

```

geofacet_plot(indices_list = NULL, select = FALSE,
              stratify_by = NULL, ci_width = 0.95, multiple = FALSE,
              trends = NULL, slope = FALSE, add_observed_means = FALSE,
              species = "")

```

Arguments

indices_list	Dataframe of strata or state/province indices produced by generate_strata_indices or generate_regional_indices
select	logical flag to indicate if the strata_level data need to be selected out of an indices_list object that includes stratum, national, or other region-types. Default is FALSE
stratify_by	How were the data stratified?
ci_width	quantile to define the width of the plotted credible interval. Defaults to 0.95, lower = 0.025 and upper = 0.975
multiple	Logical, if TRUE, multiple strata-level trajectories are plotted within each prov/state facet

trends	Optional dataframe of matching strata or state/province trends produced by <code>generate_strata_trends</code> or <code>generate_regional_trends</code> . If included trajectories are coloured based on the same colour scale used in <code>generate_map</code>
slope	Optional Logical, if dataframe of trends is included, colours in the plot are based on slope trends, Default = FALSE
add_observed_means	Should the facet plots include points indicating the observed mean counts. Defaults to FALSE. Note: scale of observed means and annual indices may not match due to imbalanced sampling among strata
species	Species name to be added onto the plot

Value

ggplot object

Examples

```
# Toy example with Pacific Wren sample data
# First, stratify the sample data

strat_data <- stratify(by = "bbs_cws", sample_data = TRUE)

# Prepare the stratified data for use in a JAGS model.
jags_data <- prepare_jags_data(strat_data = strat_data,
                              species_to_run = "Pacific Wren",
                              model = "firstdiff",
                              min_year = 2009,
                              max_year = 2018)

# Now run a JAGS model.
jags_mod <- run_model(jags_data = jags_data,
                     n_adapt = 0,
                     n_burnin = 0,
                     n_iter = 10,
                     n_thin = 1)

# Generate the stratum indices
indices <- generate_indices(jags_mod = jags_mod,
                           jags_data = jags_data,
                           regions = c("stratum"))

# Now make the geofacet plot.
gp <- geofacet_plot(indices_list = indices,
                   stratify_by = "bbs_cws",
                   species = "Pacific Wren",
                   multiple = TRUE)

# There is an unfortunate conflict between geofacet function in the geofacet package
# and the S3 +.gg method in other ggplot-extension-packages like ggmcmc
# The geofacet_plot function may fail with the following error message:
```

```
# Error: Don't know how to add e2 to a plot
# If this happens, you can fix the problem by following these steps
# 1 - save your model output
# 2 - restart your R-session
# 3 - reload the bbsBayes package (do not re-load the other conflicting package, e.g., ggmmcmc)
```

get_composite_regions *Get the area of each strata*

Description

get_composite_regions allows you to obtain the dataframe defining the original composite regions for a given stratification type.

Usage

```
get_composite_regions(strata_type = NULL)
```

Arguments

strata_type Stratification type to return the areas of

Value

Data frame with at least the following variables:

region	Name of the stratum/region
area_sq_km	Area of the stratum/region in square kilometres

Examples

```
# Obtain the potential composite regions for each of the 5 stratification types
# Most useful if the user wishes to create an set of custom composite regions
#
# USGS BBS
st_comp_regions <- get_composite_regions(strata_type = "bbs_usgs")
# create new column "Great_Plains"
gpall <- rep("Outside", nrow(st_comp_regions))
gp <- which(st_comp_regions$bcr %in% c(11,17,18,19))
gpall[gp] <- "Inside"
st_comp_regions$Great_Plains <- gpall
# st_comp_regions can now be used as the dataframe input to the argument alt_region_names
# in generate_regional_indices,
# with "Great_Plains" as the value for the argument region

# CWS BBS
st_comp_regions <- get_composite_regions(strata_type = "bbs_cws")
```

```
# BCR
st_comp_regions <- get_composite_regions(strata_type = "bcr")

# State/Province/Territory
st_comp_regions <- get_composite_regions(strata_type = "state")

# Degree block
st_comp_regions <- get_composite_regions(strata_type = "latlong")
```

get_final_values *Get final values of JAGS model*

Description

get_final_values returns the final values of all parameters of the model created by run_model as a list. This function would mostly be used in conjunction with run_model to provide initial values.

Usage

```
get_final_values(model = NULL)
```

Arguments

model Model object returned by run_model

Value

List of final values of monitored parameters.

Examples

```
# Toy example with Pacific Wren sample data
# First, stratify the sample data

strat_data <- stratify(by = "bbs_cws", sample_data = TRUE)

# Prepare the stratified data for use in a JAGS model.
jags_data <- prepare_jags_data(strat_data = strat_data,
                              species_to_run = "Pacific Wren",
                              model = "firstdiff",
                              min_year = 2009,
                              max_year = 2018)

# Now run a JAGS model. For the sake of speed, we've adjusted
# some arguments so that the JAGS model will not run any
# adaptation steps (n_adapt = 0), no burnin steps (n_burnin = 0),
# only 50 iterations per chain (n_iter = 50), and will not
```

```
# thin the chain (n_thin = 1). This will produce several convergence
# warnings, but we can ignore them for the sake of this toy example.

jags_mod <- run_model(jags_data = jags_data,
                    n_adapt = 0,
                    n_burnin = 0,
                    n_iter = 10,
                    n_thin = 1)

# Get the final values
final_values <- get_final_values(model = jags_mod)

# Then, we can use these final values as input for another model run
jags_mod2 <- run_model(jags_data = jags_data,
                    n_adapt = 0,
                    n_burnin = 0,
                    n_iter = 50,
                    n_thin = 1,
                    inits = final_values)
```

get_mcmc_list

Obtain MCMC list from jagsUI object

Description

get_mcmc_list will return both the mcmc.list object and the sims.list object from jagsUI. mcmc.list is a list of the MCMC samples generated by the rjags library, and sims.list is a vectorized version of mcmc.list produced by the jagsUI library.

Usage

```
get_mcmc_list(jags_mod = NULL)
```

Arguments

jags_mod JAGS object returned by run_model

Value

List containing:

mcmc_list MCMC samples produced by rjags

sims_list Vectorized posterior samples produced by jagsUI

Examples

```

# Toy example with Pacific Wren sample data
# First, stratify the sample data

strat_data <- stratify(by = "bbs_cws", sample_data = TRUE)

# Prepare the stratified data for use in a JAGS model.
jags_data <- prepare_jags_data(strat_data = strat_data,
                              species_to_run = "Pacific Wren",
                              model = "firstdiff",
                              min_year = 2009,
                              max_year = 2018)

# Now run a JAGS model. For the sake of speed, we've adjusted
# some arguments so that the JAGS model will not run any
# adaptation steps (n_adapt = 0), no burnin steps (n_burnin = 0),
# only 50 iterations per chain (n_iter = 50), and will not
# thin the chain (n_thin = 1). This will produce several convergence
# warnings, but we can ignore them for the sake of this toy example.

jags_mod <- run_model(jags_data = jags_data,
                     n_adapt = 0,
                     n_burnin = 0,
                     n_iter = 10,
                     n_thin = 1)

# Now, obtain the MCMC list
mcmc_list <- get_mcmc_list(jags_mod = jags_mod)

```

get_prepared_data *Get the prepared species dataset used for JAGS*

Description

get_prepared_data returns a data frame of the data that was used for JAGS. This is the subsetted data based on the selected species to model, with zero counts filled in and any other route/strata filter applied.

Usage

```
get_prepared_data(jags_data = NULL)
```

Arguments

jags_data List of JAGS input data produced by prepare_jags_data

Value

Data frame of 9 variables:

count	Number of species observed for this route run
strat	Numerical factors of the stratum
obser	Numerical factor of the observer
year	Numerical factor of the year
firststyr	1 if this was the observer's first year, 0 otherwise
strat_name	Name of the stratum
route	Route that this count was taken on
rYear	Year this count was conducted
yearscales	Scaled year

Examples

```
# Toy example with Pacific Wren sample data
# First, stratify the sample data

strat_data <- stratify(by = "bbs_cws", sample_data = TRUE)

# Prepare the stratified data for use in a JAGS model. In this
# toy example, we will set the minimum year as 2009 and
# maximum year as 2018, effectively only setting up to
# model 10 years of data. We will use the "first difference
# model.
jags_data <- prepare_jags_data(strat_data = strat_data,
                              species_to_run = "Pacific Wren",
                              model = "firstdiff",
                              min_year = 2009,
                              max_year = 2018)

# Obtain the reassembled data frame for the data sent to JAGS
prepped_data <- get_prepared_data(jags_data = jags_data)
```

get_strata_area *Get the area of each strata*

Description

get_strata_area allows you to obtain the area of each strata for a given stratification type.

Usage

```
get_strata_area(strata_type = NULL)
```

Arguments

strata_type Stratification type to return the areas of

Value

Data frame with the following variables:

region Name of the stratum/region
area_sq_km Area of the stratum/region in square kilometres

Examples

```
# Obtain the strata area for each of the 5 stratification types

# USGS BBS
st_area <- get_strata_area(strata_type = "bbs_usgs")

# CWS BBS
st_area <- get_strata_area(strata_type = "bbs_cws")

# BCR
st_area <- get_strata_area(strata_type = "bcr")

# State/Province/Territory
st_area <- get_strata_area(strata_type = "state")

# Degree block
st_area <- get_strata_area(strata_type = "latlong")
```

load_bbs_data

Load Breeding Bird Survey dataset into R Session

Description

load_bbs_data loads the raw, unstratified BBS data into the current R session. The data must have been previously fetched using the fetch_bbs_data function. Note that this function is not necessary to run a Bayesian analysis of BBS data; calling stratify will return stratified BBS data in a list of data frames.

Usage

```
load_bbs_data()
```

Value

Large list (3 elements) consisting of:

bird	Data frame of sample bird point count data per route, per year
route	Data frame of sample yearly route data
species	Sample list of North American bird species

load_sample_data	<i>Load Sample Breeding Bird Survey dataset into R Session</i>
------------------	--

Description

load_sample_data returns the sample data provided by bbsBayes. The data is returned as a list of data frames, similar to what is returned by load_bbs_data

Usage

```
load_sample_data()
```

Value

Large list (3 elements) consisting of:

bird	Data frame of sample bird point count data per route, per year
route	Data frame of sample yearly route data
species	Sample list of North American bird species

Examples

```
sample_data <- load_sample_data()
```

lppd

*Calculate log posterior predictive density***Description**

lppd Calculate log posterior predictive density (LPPD) for the supplied model.

Usage

```
lppd(jags_data = NULL, jags_mod = NULL, pointwise = FALSE)
```

Arguments

jags_data	Data prepared by prepare_jags_data, used for input to the JAGS model
jags_mod	JAGS list generated by run_model
pointwise	If set to TRUE, a data frame is returned that contains the pointwise LPPD for each count. Defaults to FALSE

Details

NOTE: in order to calculate LPPD, the model MUST track the parameter "lambda". In species that are data-rich, such as Wood Thrush, this produces extremely large JAGS objects, and takes up a considerable amount of memory when simulating with run_model.

Value

Data frame of pointwise LPPD by count if pointwise is set to TRUE. Double precision numerical value of LPPD if pointwise is set to FALSE.

Examples

```
# Toy example with Pacific Wren sample data
# First, stratify the sample data

strat_data <- stratify(by = "bbs_cws", sample_data = TRUE)

# Prepare the stratified data for use in a JAGS model.
jags_data <- prepare_jags_data(strat_data = strat_data,
                              species_to_run = "Pacific Wren",
                              model = "firstdiff",
                              min_year = 2014,
                              max_year = 2018)

# Now run a JAGS model. Make sure to track the lambda parameter here

jags_mod <- run_model(jags_data = jags_data,
                      n_adapt = 0,
                      n_burnin = 0,
```

```

        n_iter = 5,
        n_thin = 1,
        parameters_to_save = c("n",
                               "lambda"))

# Output LPPD
lppd(jags_data = jags_data,
     jags_mod = jags_mod)

```

model_to_file	<i>Save model to text file</i>
---------------	--------------------------------

Description

model_to_file allows you to save any of the preloaded hierarchical Bayesian models to a text file.

Usage

```
model_to_file(model = NULL, filename = NULL, heavy_tailed = FALSE)
```

Arguments

model	Model to be saved. Options are "slope", "firstdiff", "gam", "gamye"
filename	File name to create on disk.
heavy_tailed	Logical indicating whether the extra-Poisson error distribution should be modeled as a t-distribution, with heavier tails than the standard normal distribution. Default is currently FALSE, but recent results suggest users should strongly consider setting this to TRUE, even though it requires much longer convergence times

Value

None

Examples

```

# Save the Slope model to a file called "slope.txt" in temp directory
model_to_file(model = "slope",
              filename = file.path(tempdir(), "slope.txt"))

# Save the First Difference model to a file called "fd.txt" in temp directory
model_to_file(model = "firstdiff",
              filename = file.path(tempdir(), "fd.txt"))

# Save the GAM model to a file called "gam.txt" in temp directory
model_to_file(model = "gam",
              filename = file.path(tempdir(), "gam.txt"))

```

```
# Save the GAM year effects model to a file called "gamy.txt" in temp directory
model_to_file(model = "gamy",
              filename = file.path(tempdir(), "gamy.txt"))
```

plot_indices *Generate plots of index trajectories by stratum*

Description

Generates the indices plot for each stratum modelled.

Usage

```
plot_indices(indices_list = NULL, ci_width = 0.95, min_year = NULL,
             max_year = NULL, species = "", title_size = 20,
             axis_title_size = 18, axis_text_size = 16,
             add_observed_means = FALSE, add_number_routes = FALSE)
```

Arguments

indices_list	List of indices of annual abundance and other results produced by generate_strata_indices
ci_width	quantile to define the width of the plotted credible interval. Defaults to 0.95, lower = 0.025 and upper = 0.975
min_year	Minimum year to plot
max_year	Maximum year to plot
species	Species name to be added onto the plot
title_size	Specify font size of plot title. Defaults to 20
axis_title_size	Specify font size of axis titles. Defaults to 18
axis_text_size	Specify font size of axis text. Defaults to 16
add_observed_means	Should the plot include points indicated the observed mean counts. Defaults to FALSE. Note: scale of observed means and annual indices may not match due to imbalanced sampling among routes
add_number_routes	Should the plot be superimposed over a dotplot showing the number of BBS routes included in each year. This is useful as a visual check on the relative data-density through time because in most cases the number of observations increases over time

Value

List of ggplot objects, each entry being a plot of a stratum indices

```
prepare_jags_data      Wrangle data to use for JAGS input
```

Description

prepare_jags_data subsets raw BBS data by selected species and and wrangles stratified data for use as input to run JAGS models.

Usage

```
prepare_jags_data(strat_data = NULL, species_to_run = NULL,
  model = NULL, heavy_tailed = FALSE, n_knots = NULL,
  min_year = NULL, max_year = NULL, min_n_routes = 3,
  min_max_route_years = 3, min_mean_route_years = 1,
  strata_rem = NULL, quiet = FALSE, ...)
```

Arguments

strat_data	Large list of stratified data returned by stratify()
species_to_run	Character string of the English name of the species to run
model	Character string of model to be used. Options are "slope", "firstdiff", "gam", "gamy".
heavy_tailed	Logical indicating whether the extra-Poisson error distribution should be modeled as a t-distribution, with heavier tails than the standard normal distribution. Default is currently FALSE, but recent results suggest users should strongly consider setting this to TRUE, even though it requires much longer convergence times
n_knots	Number of knots to be used in GAM function
min_year	Minimum year to keep in analysis
max_year	Maximum year to keep in analysis
min_n_routes	Minimum routes per strata where species has been observed. Defaults to 3
min_max_route_years	Minimum number of years with non-zero observations of species on at least 1 route. Defaults to 3
min_mean_route_years	Minimum average of years per route with the species observed. Defaults to 1.
strata_rem	Strata to remove from analysis. Defaults to NULL
quiet	Should progress bars be suppressed?
...	Additional arguments

Value

List of data to be used in JAGS, including:

model	The model to be used in JAGS
heavy_tailed	Logical indicating whether the extra-Poisson error distribution should be modeled as a t-distribution
ncounts	The number of counts containing useful data for the species
nstrata	The number of strata used in the analysis
ymin	Minimum year used
ymax	Maximum year used
nonzeroweight	Proportion of routes in each strata with species observation
count	Vector of counts for the species
strat	Vector of strata to be used in the analysis
osber	Vector of unique observer-route pairings
year	Vector of years for each count
firststyr	Vector of indicator variables as to whether an observer was a first year
month	vector of numeric month of observation
day	vector of numeric day of observation
nobservers	Total number of observer-route pairings
fixedyear	Median of all years (ymin:ymax), included only with slope model
nknots	Number of knots to use for smoothing functions, included only with GAM
X.basis	Basis function for n smoothing functions, included only with GAM

Examples

```
# Toy example with Pacific Wren sample data
# First, stratify the sample data

strat_data <- stratify(by = "bbs_cws", sample_data = TRUE)

# Prepare the stratified data for use in a JAGS model. In this
# toy example, we will set the minimum year as 2009 and
# maximum year as 2018, effectively only setting up to
# model 10 years of data. We will use the "first difference
# model.
jags_data <- prepare_jags_data(strat_data = strat_data,
                              species_to_run = "Pacific Wren",
                              model = "firstdiff",
                              min_year = 2009,
                              max_year = 2018)

# You can also specify the GAM model, with an optional number of
# knots to use for the GAM basis.
# By default, the number of knots will be equal to the floor
# of the total unique years for the species / 4
```

```
jags_data <- prepare_jags_data(strat_data = strat_data,
                              species_to_run = "Pacific Wren",
                              model = "gam",
                              n_knots = 9)
```

 route_sample

Sample route data per year run (Pacific Wren only)

Description

A dataset containing data for each route run per year. NOTE: This only contains data for Pacific Wren, not the full data set. The full count set is obtained via the function `fetch_bbs_data`. The data is obtained from the United States Geological Survey and is subject to change as new data is added each year. See Details for citation.

Usage

```
route_sample
```

Format

A data frame with 20 rows and 32 variables:

countrynum Numerical representation of the country

statenum Numerical representation of state or province

Route Numerical representation of the route the species was observed

RouteName Name of the route, represented as a string

Active Boolean 0 or 1 as to whether the route is currently active

Latitude Latitude of the start of the route

Longitude Longitude of the start of the route

BCR What bird conservation region is the route in

RouteTypeID Type of the route, only 1 is acceptable

RouteTypeDetailID Route type detail ID

RouteDataID Unique code for each year a route was run

RPID Run Protocol ID

Year Year the route was run

Month Month the route was run

Day Day the route was run

ObsN Unique number for the observer on the route

TotalSpp Total species observed on the route

StartTemp Temperature at the start of the route

EndTemp Temperature at the end of the route
TempScale (C)elsius or (F)ahrenheit
StartWind Wind type at the beginning of the route
EndWind Wind type at the end of the route
StartSky Sky conditions at the start of the route
EndSky Sky conditions at the end of the route
StartTime Time the route was started
EndTime Time the route was ended
Assistant Boolean 0 or 1 as to whether an assistant was used
QualityCurrentID Quality current ID
RunType Type of BBS route run. Only acceptable run type is 1
State String representation of state or province
St_Abrev Abbreviated state or province
Country Abbreviated country

Details

Pardieck, K.L., D.J. Ziolkowski Jr., M. Lutmerding, V. Aponte and M-A.R. Hudson. 2019. North American Breeding Bird Survey Dataset 1966 - 2018, version 2018.0. U.S. Geological Survey, Patuxent Wildlife Research Center. <https://doi.org/10.5066/P9HE8XYJ>.

run_model	<i>Run JAGS model for prepared species data</i>
-----------	---

Description

run-model runs a JAGS model as specified by the user for the species of interest

Usage

```
run_model(jags_data = NULL, model_file_path = NULL, inits = NULL,
  parameters_to_save = c("n"), track_n = TRUE, n_chains = 3,
  n_adapt = NULL, n_burnin = 20000, n_thin = 10,
  n_saved_steps = 2000, n_iter = 10000, parallel = FALSE,
  quiet = FALSE, modules = NULL, ...)
```

Arguments

jags_data	List or environment containing the data to model, as output by prepare_jags_data
model_file_path	Path to custom model. Overrides the model variable set by prepare_jags_data
inits	Optional list of initialization values for JAGS model. If none are specified, the JAGS model will generate its own initial values.

parameters_to_save	Character vector of parameters to monitor in JAGS. Defaults to just monitoring "n"
track_n	By default, the parameter "n" will always be tracked, even if the user forgets to specify it. However, if the user is positive they do not want to track "n", this parameter can be set to FALSE. NOTE: you will not be able to generate annual indices if "n" is not tracked.
n_chains	Optional number of chains to run. Defaults to 3.
n_adapt	Optional integer specifying the number of steps to adapt the JAGS model. The default is NULL, which will result in the function running groups of 100 adaptation iterations (to a max of 10,000) until JAGS reports adaptation is sufficient. If you set it manually, 1000 is the recommended minimum value.
n_burnin	Optional integer specifying the number of iterations to burn in the model. Defaults to 20000 per chain.
n_thin	Optional number of steps to thin or discard.
n_saved_steps	Optional number of steps to save per chain. Defaults to 2000.
n_iter	Optional number of iterations per chain. Defaults to 10000.
parallel	Should each chain be run parallel on separate cores? If TRUE, the number of cores used will be the minimum of the n_chains specified and the number of cores on your computer
quiet	Should JAGS output be suppressed?
modules	Character vector of JAGS modules to load before analysis. By default no extra modules are loaded (other than 'basemod' and 'bugs'). To force glm or other modules to load, use modules = "glm". Be warned, our experience suggests that including the glm module may cause problems with the BBS data.
...	Additional arguments

Value

jagsUI object

Examples

```
# Toy example with Pacific Wren sample data
# First, stratify the sample data

strat_data <- stratify(by = "bbs_cws", sample_data = TRUE)

# Prepare the stratified data for use in a JAGS model.
jags_data <- prepare_jags_data(strat_data = strat_data,
                              species_to_run = "Pacific Wren",
                              model = "firstdiff",
                              min_year = 2009,
                              max_year = 2018)

# Now run a JAGS model. For the sake of speed, we've adjusted
```

```
# some arguments so that the JAGS model will not run any
# adaptation steps (n_adapt = 0), no burnin steps (n_burnin = 0),
# only 50 iterations per chain (n_iter = 50), and will not
# thin the chain (n_thin = 1). This will produce several convergence
# warnings, but we can ignore them for the sake of this toy example.

jags_mod <- run_model(jags_data = jags_data,
                     n_adapt = 0,
                     n_burnin = 0,
                     n_iter = 10,
                     n_thin = 1,
                     parameters_to_save = c("n", "strata"))
```

r_hat

Generate Gelman-Rubin's R-Hat statistic

Description

r_hat returns a dataframe of Gelman-Rubin's R-hat statistics for each parameter tracked in the model.

Usage

```
r_hat(jags_mod = NULL, parameter_list = NULL, threshold = NULL)
```

Arguments

jags_mod	JAGS list generated by run_model
parameter_list	Optional list of parameters to subset
threshold	Return only r-hat values greater than OR equal to this threshold (floating point value)

Details

R-hat, also known as the potential scale reduction factor (PSRF) was described by Gelman & Rubin (1992) as a way of calculating convergence of parameters given 2 or more chains. See citation below for details.

Gelman, Andrew; Rubin, Donald B. Inference from Iterative Simulation Using Multiple Sequences. Statist. Sci. 7 (1992), no. 4, 457–472. doi:10.1214/ss/1177011136. <https://projecteuclid.org/euclid.ss/1177011136>

Value

Dataframe consisting of r-hat values per parameter.

Examples

```

# Toy example with Pacific Wren sample data
# First, stratify the sample data

strat_data <- stratify(by = "bbs_cws", sample_data = TRUE)

# Prepare the stratified data for use in a JAGS model.
jags_data <- prepare_jags_data(strat_data = strat_data,
                              species_to_run = "Pacific Wren",
                              model = "firstdiff",
                              min_year = 2009,
                              max_year = 2018)

# Now run a JAGS model.

jags_mod <- run_model(jags_data = jags_data,
                     n_adapt = 0,
                     n_burnin = 0,
                     n_iter = 10,
                     n_thin = 1,
                     parameters_to_track = c("n", "strata"))

# Check convergence for all parameters
convergence <- r_hat(jags_mod = jags_mod)

# Check convergence for only subset of parameters
convergence <- r_hat(jags_mod = jags_mod, parameter_list = "strata")

# Only return R Hat values greater than or equal to specified value
convergence <- r_hat(jags_mod = jags_mod, threshold = 1.1)

```

species_sample

Sample North American bird species list (Pacific Wren only)

Description

A dataset containing species list of North America. NOTE: This only contains data for Pacific Wren, not the full data set. The full count set is obtained via the function `fetch_bbs_data`. The data is obtained from the United States Geological Survey and is subject to change as new data is added each year. See Details for citation.

Usage

```
species_sample
```

Format

A data frame with 20 rows and 10 variables:

seq Sequence - USGS use

aou Numerical representation of the species, designated by American Ornithological Union

english Species name in English

order Taxonomic order

family Taxonomic family

genus Taxonomic genus

species Taxonomic species

sp.bbs Same as aou, no leading 0

Details

Pardieck, K.L., D.J. Ziolkowski Jr., M. Lutmerding, V. Aponte and M-A.R. Hudson. 2019. North American Breeding Bird Survey Dataset 1966 - 2018, version 2018.0. U.S. Geological Survey, Patuxent Wildlife Research Center. <https://doi.org/10.5066/P9HE8XYJ>.

stratify

Stratify raw Breeding Bird Survey data

Description

Assigns each bird count data point and each route a strata based on its geographic location and the stratification as specified by the user.

Usage

```
stratify(by = NULL, sample_data = FALSE, bbs_data = NULL,
         lump_species_forms = TRUE, quiet = FALSE, stratify_by = NULL)
```

Arguments

by	String argument of stratification type. Options are "state", "bcr", "latlong", "bbs_cws", "bbs_usgs"
sample_data	Should just sample data (just Pacific Wren) be used? Defaults to FALSE.
bbs_data	Raw BBS data saved as a list of 3 data frames. Not necessary if you have already run <code>fetch_bbs_data</code>
lump_species_forms	Logical, default is TRUE, indicating that for species with multiple forms, the "unidentified" form is replaced by the sum of observations for all forms (including the original unidentified obs). The underlying BBS database includes separate data for each form, and these separate forms are retained with their original names. The original unidentified category for observations that were not specific to a particular form are replaced by the combined observations. If the user wishes to keep the unidentified form separate, this can be set to FALSE

quiet Should progress bars be suppressed?
stratify_by Deprecated in favour of 'by'

Value

Large list (3 elements) consisting of:

bird_strat Dataframe of stratified bird data
route_strat Dataframe of stratified route data
species_strat Dataframe of stratified species data
by Argument used for stratification

Examples

```
# Toy examples using Pacific Wren sample data

# Stratify by CWS USGS stratifications
data_strat <- stratify(by = "bbs_usgs", sample_data = TRUE)

# Stratify by Bird Conservation Regions only
data_strat <- stratify(by = "bcr", sample_data = TRUE)

# Stratify by CWS BBS stratifications
data_strat <- stratify(by = "bbs_cws", sample_data = TRUE)

# Stratify by State/Province/Territory only
data_strat <- stratify(by = "state", sample_data = TRUE)

# Stratify by blocks of 1 degree of latitude X 1 degree of longitude
data_strat <- stratify(by = "latlong", sample_data = TRUE)

# Requires fetch_bbs_data() to have been run (takes about 10 minutes).

# Stratify the entire data set, may take a minute or so
data_strat <- stratify(by = "bbs_cws")
```


Index

* datasets

- bird_sample, 2
- route_sample, 26
- species_sample, 30

bird_sample, 2

fetch_bbs_data, 3

generate_indices, 4

generate_map, 7

generate_trends, 8

geofacet_plot, 11

get_composite_regions, 13

get_final_values, 14

get_mcmc_list, 15

get_prepared_data, 16

get_strata_area, 17

load_bbs_data, 18

load_sample_data, 19

lppd, 20

model_to_file, 21

plot_indices, 22

prepare_jags_data, 24

r_hat, 29

route_sample, 26

run_model, 27

species_sample, 30

stratify, 31