Package ‘forestecology’

October 2, 2021

Type Package

Title Fitting and Assessing Neighborhood Models of the Effect of Interspecific Competition on the Growth of Trees

Version 0.2.0

Maintainer Albert Y. Kim <`albert.ys.kim@gmail.com`>

Description Code for fitting and assessing models for the growth of trees. In particular for the Bayesian neighborhood competition linear regression model of Allen (2020): methods for model fitting and generating fitted/predicted values, evaluating the effect of competitor species identity using permutation tests, and evaluating model performance using spatial cross-validation.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

URL `https://github.com/rudeboybert/forestecology`

BugReports `https://github.com/rudeboybert/forestecology/issues`

Depends R (>= 3.2.4)

Suggests testthat (>= 3.0.0), knitr, rmarkdown, covr

Imports magrittr, rlang, glue, ggplot2, stringr, dplyr, tidyr, purrr, ggridges, mvnfast, sf, sfheaders, snakecase, tibble, yardstick, blockCV, forcats, patchwork

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation no

Author Albert Y. Kim [aut, cre] (<`https://orcid.org/0000-0001-7824-306X`>), David Allen [aut] (<`https://orcid.org/0000-0002-0712-9603`>), Simon Couch [aut] (<`https://orcid.org/0000-0001-5676-5107`>)

Repository CRAN

Date/Publication 2021-10-02 13:30:05 UTC

R topics documented:

`add_buffer_variable` *Identify trees in the buffer region*

Description

Identify trees in the buffer region

Usage

```
add_buffer_variable(growth_df, direction = "in", size, region)
```

Arguments

`growth_df`	`sf` data frame
`direction`	"in" for buffers that are contained within `region` (default), "out" for buffers that contain `region`.
`size`	Distance to determine which neighboring trees to a focal tree are competitors. The units are assumed to be the same as the `geometry` variables in `growth_df`.
`region`	An `sf` polygon object of region to be buffered

Value

The same `growth_df` data frame but with a new boolean variable `buffer` indicating if a tree is in the study region buffer area. This uses `compute_buffer_region()` to define the boundary of the buffer region.

See Also

Other spatial functions: `compute_buffer_region()`, `focal_vs_comp_distance()`

Examples

```
library(tibble)
library(sfheaders)
library(ggplot2)

# Example square region to be buffered
region <- tibble(
  x = c(0, 0, 1, 1),
  y = c(0, 1, 1, 0)
) %>%
  sf_polygon()

# Example points
study_points <- tibble(
  x = runif(50),
  y = runif(50)
) %>%
  sf_point()

# Size of buffer
size <- 0.05

# Identify whether points are within size of boundary
study_points <- study_points %>%
  add_buffer_variable(direction = "in", size = size, region = region)

# Plot study points coded by whether they are within size of boundary
p <- ggplot() +
  geom_sf(data = region, fill = "transparent") +
  geom_sf(data = study_points, aes(col = buffer))
p

# Additionally, show buffer boundary in red
buffer_boundary <- region %>%
  compute_buffer_region(direction = "in", size = size)
p +
  geom_sf(data = buffer_boundary, col = "red", fill = "transparent")
```

`autoplot.comp_bayes_lm`

Plot Bayesian model parameters

Description

Plot Bayesian model parameters

Usage

```
## S3 method for class 'comp_bayes_lm'
autoplot(object, type = "intercepts", sp_to_plot = NULL, ...)
```

Arguments

`object`	Output of `comp_bayes_lm`
`type`	A single character string for plot type with possible values "intercepts", "dbh_slopes", or "competition".
`sp_to_plot`	Vector of subset of species to plot
`...`	Currently ignored—only included for consistency with generic.

Value

`focal_vs_comp` with new column of predicted `growth_hat`

Examples

```
library(ggplot2)
library(ggridges)

# Load in posterior parameter example
data(comp_bayes_lm_ex)

# Plot beta_0, growth intercepts
autoplot(comp_bayes_lm_ex, type = "intercepts")

# Plot beta_dbh, growth-dbh slope
autoplot(comp_bayes_lm_ex, type = "dbh_slopes")

# Plot lambdas, competition coefficients
autoplot(comp_bayes_lm_ex, type = "competition")
```

`blocks_ex` *Example cross validation grid*

Description

This is an example cross validation grid. This is needed to create the focal versus comp data frame and run cross-validated models

Usage

```
blocks_ex
```

Format

A `sf` polygons

foldID Tree identification number. This identifies an individual tree and can be used to connect trees between the two censuses.

geometry Point location of the individual

See Also

Other example data objects: `census_1_ex`, `census_2008_bw`, `census_2014_bw`, `census_2_ex`, `comp_bayes_lm_ex`, `focal_vs_comp_ex`, `growth_ex`, `growth_spatial_ex`, `growth_toy`, `species_bw`, `study_region_bw`, `study_region_ex`

Examples

```
library(ggplot2)
library(sf)
library(dplyr)

comp_dist <- 1

ggplot(blocks_ex) +
  geom_sf() +
  geom_sf(data = growth_spatial_ex)

focal_vs_comp_ex <- growth_spatial_ex %>%
  mutate(basal_area = 0.0001 * pi * (dbh1 / 2)^2) %>%
 create_focal_vs_comp(comp_dist, blocks = blocks_ex, id = "ID", comp_x_var = "basal_area")
```

`census_1_ex` *Example input census data for package use*

Description

This is example forest census data to be analyzed with this package.

Usage

```
census_1_ex
```

Format

A `tibble`

- **ID** Tree identification number. This identifies an individual tree and can be used to connect trees between the two censuses.
- **sp** Species of the individual
- **gx** x-coordinate meters from reference point
- **gy** y-coordinate meters from reference point
- **date** Date the stem was measured
- **codes** Code for additional information on the stem: M means the main stem of the individual tree and R means the stem was lost, but the tag was moved to another stem greater than DBH cutoff, this stands for resprout.
- **dbh** Diameter at breast height of the tree in cm

See Also

Other example data objects: `blocks_ex`, `census_2008_bw`, `census_2014_bw`, `census_2_ex`, `comp_bayes_lm_ex`, `focal_vs_comp_ex`, `growth_ex`, `growth_spatial_ex`, `growth_toy`, `species_bw`, `study_region_bw`, `study_region_ex`

Examples

```
library(dplyr)
library(stringr)

growth_ex <-
  compute_growth(
    census_1 = census_1_ex %>%
      mutate(sp = to_any_case(sp) %>% factor()),
    census_2 = census_2_ex %>%
      filter(!str_detect(codes, "R")) %>%
      mutate(sp = to_any_case(sp) %>% factor()),
    id = "ID"
  )
```

`census_2008_bw` *Michigan Big Woods research plot data*

Description

The Big Woods data come from three censuses of a 23 ha forest research plots. All free-standing vegetation greater than 1 cm diameter at 1.3 m height (diameter at breast height; DBH) were tagged, identified, spatially mapped and had their DBH measured. The original census took place in 2003 and covered only 12 ha. A second census took place from 2008-2010 and expanded the plot to its current 23 ha. In the first and second censuses trees larger than 3.1 cm DBH were included. Finally a third census took place in 2014. In this census trees larger than 1 cm DBH were included. In the second and third censuses the original trees were found, recorded for survival, remeasured, and new individuals were tagged. This data frame has data from the second census (2008-2010).

Usage

```
census_2008_bw
```

Format

A data frame with 27193 rows and 8 variables:

treeID Tree identification number. This identifies an individual tree and can be used to connect trees between the two censuses.

stemID Stem number for a multi-stemmed individual. For all trees this starts at 1 and continues up from there. To uniquely identify a stem across the plot this value must be combined with `treeID`.

dbh Diameter at breast

sp Code for the species. See `species_bw` for scientific name.

gx x-coordinate meters from reference point

gy y-coordinate meters from reference point

date Approximate date the stem was measured

codes Code for additional information on the stem: M means the main stem of the individual tree; AL means the stem is alive but leaning or completely fallen over; B means the stem is broken and over half the canopy is assumed to be missing; and R means the stem was lost, but the tag was moved to another stem greater than DBH cutoff, this stands for resprout.

Details

This plot is part of the Smithsonian Institution's Forest Global Earth Observatory (ForestGEO) global network of forest research sites. For complete details on this dataset see its Deep Blue Data repository page.

See Also

Other Big Woods data: `census_2014_bw`, `species_bw`, `study_region_bw`

Other example data objects: `blocks_ex`, `census_1_ex`, `census_2014_bw`, `census_2_ex`, `comp_bayes_lm_ex`, `focal_vs_comp_ex`, `growth_ex`, `growth_spatial_ex`, `growth_toy`, `species_bw`, `study_region_bw`, `study_region_ex`

Examples

```
library(ggplot2)
library(sf)

# Convert all 2008 plot stems to sf object
census_2008_bw_sf <- census_2008_bw %>%
  st_as_sf(coords = c("gx", "gy"))

# Plot stems with plot boundary
ggplot() +
  geom_sf(data = census_2008_bw_sf, size = 0.25)
```

`census_2014_bw` *Michigan Big Woods research plot data*

Description

The Big Woods data come from three censuses of a 23 ha forest research plots. All free-standing vegetation greater than 1 cm diameter at 1.3 m height (diameter at breast height; DBH) were tagged, identified, spatially mapped and had their DBH measured. The original census took place in 2003 and covered only 12 ha. A second census took place from 2008-2010 and expanded the plot to its current 23 ha. In the first and second censuses trees larger than 3.1 cm DBH were included. Finally a third census took place in 2014. In this census trees larger than 1 cm DBH were included. In the second and third censuses the original trees were found, recorded for survival, remeasured, and new individuals were tagged. This data frame has data from the third census (2014).

Usage

```
census_2014_bw
```

Format

A data frame with 48371 rows and 8 variables:

treeID Tree identification number. This identifies an individual tree and can be used to connect trees between the two censuses.

stemID Stem number for a multi-stemmed individual. For all trees this starts at 1 and continues up from there. To uniquely identify a stem across the plot this value must be combined with `treeID`.

sp Code for the species. See `species_bw` for scientific name.

dbh Diameter at breast

gx x-coordinate meters from reference point

gy y-coordinate meters from reference point

date Approximate date the stem was measured

codes Code for additional information on the stem: M means the main stem of the individual tree; AL means the stem is alive but leaning or completely fallen over; B means the stem is broken and over half the canopy is assumed to be missing; and R means the stem was lost, but the tag was moved to another stem greater than DBH cutoff, this stands for resprout.

Details

This plot is part of the Smithsonian Institution's Forest Global Earth Observatory (ForestGEO) global network of forest research sites. For complete details on this dataset see its Deep Blue Data repository page.

See Also

Other Big Woods data: `census_2008_bw`, `species_bw`, `study_region_bw`

Other example data objects: `blocks_ex`, `census_1_ex`, `census_2008_bw`, `census_2_ex`, `comp_bayes_lm_ex`, `focal_vs_comp_ex`, `growth_ex`, `growth_spatial_ex`, `growth_toy`, `species_bw`, `study_region_bw`, `study_region_ex`

Examples

```
library(ggplot2)
library(sf)
library(dplyr)

# Convert all 2008 plot stems to sf object
census_2008_bw_sf <- census_2008_bw %>%
  st_as_sf(coords = c("gx", "gy"))

# Plot stems with plot boundary
ggplot() +
  geom_sf(data = census_2008_bw_sf, size = 0.25)

# Species-specific mortality between 2008 and 2014 censuses
census_2008_bw %>%
  left_join(census_2014_bw, by = c("treeID", "stemID"), suffix = c("_2008", "_2014")) %>%
  mutate(mortality = ifelse(is.na(dbh_2014), 1, 0)) %>%
  group_by(sp_2008) %>%
  summarize(mortality = mean(mortality), n = n()) %>%
  arrange(desc(n))
```

`census_2_ex` *Example input census data for package use*

Description

This is an example second census to be analyzed with the package.

Usage

```
census_2_ex
```

Format

A `tibble`

ID Tree identification number. This identifies an individual tree and can be used to connect trees between the two censuses.

sp Species of the individual

gx x-coordinate meters from reference point

gy y-coordinate meters from reference point

date Date the stem was measured

codes Code for additional information on the stem: M means the main stem of the individual tree and R means the stem was lost, but the tag was moved to another stem greater than DBH cutoff, this stands for resprout.

dbh Diameter at breast height of the tree in cm

See Also

Other example data objects: `blocks_ex`, `census_1_ex`, `census_2008_bw`, `census_2014_bw`, `comp_bayes_lm_ex`, `focal_vs_comp_ex`, `growth_ex`, `growth_spatial_ex`, `growth_toy`, `species_bw`, `study_region_bw`, `study_region_ex`

Examples

```
library(dplyr)
library(stringr)

growth_ex <-
  compute_growth(
    census_1 = census_1_ex %>%
      mutate(sp = to_any_case(sp) %>% factor()),
    census_2 = census_2_ex %>%
      filter(!str_detect(codes, "R")) %>%
      mutate(sp = to_any_case(sp) %>% factor()),
    id = "ID"
  )
```

`compute_buffer_region` *Compute buffer to a region.*

Description

Compute buffer to a region.

Usage

```
compute_buffer_region(region, direction = "in", size)
```

Arguments

`region`	An `sf` polygon object of region to be buffered
`direction`	"in" for buffers that are contained within `region` (default), "out" for buffers that contain `region`.
`size`	Distance to determine which neighboring trees to a focal tree are competitors. The units are assumed to be the same as the `geometry` variables in `growth_df`.

Value

An `sf` polygon object of buffer

See Also

Other spatial functions: `add_buffer_variable()`, `focal_vs_comp_distance()`

Examples

```
library(tibble)
library(sfheaders)
library(ggplot2)

# Example square region to be buffered (as sf object)
region <- tibble(
  x = c(0, 0, 1, 1),
  y = c(0, 1, 1, 0)
) %>%
  sf_polygon()

# Size of buffer
size <- 0.05

# Compute "inwards" buffer
inwards_buffer_region <- region %>%
  compute_buffer_region(direction = "in", size = size)

# Compute "outwards" buffer
outwards_buffer_region <- region %>%
```

```
  compute_buffer_region(direction = "out", size = size)

# Plot all three regions:
ggplot() +
  geom_sf(data = outwards_buffer_region, col = "blue", fill = "transparent") +
  geom_sf(data = region, fill = "transparent") +
  geom_sf(data = inwards_buffer_region, col = "orange", fill = "transparent") +
 labs(title = "Regions: original (black), inwards buffer (orange), and outwards buffer (blue)")
```

`compute_growth` *Compute growth of trees*

Description

Based on two tree censuses, compute the average annual growth in dbh for all trees.

Usage

```
compute_growth(census_1, census_2, id)
```

Arguments

`census_1`	A data frame of the first census.
`census_2`	A data frame of the second (later) census
`id`	Name of variable that uniquely identifies each tree common to `census_1` and `census_2` allowing you to join/merge both data frames.

Value

An `sf` data frame with column `growth` giving the average annual growth in dbh.

See Also

Other data processing functions: `create_bayes_lm_data()`, `create_focal_vs_comp()`

Examples

```
library(dplyr)
library(stringr)

growth_ex <-
  compute_growth(
    census_1 = census_1_ex %>%
      mutate(sp = to_any_case(sp) %>% factor()),
    census_2 = census_2_ex %>%
      filter(!str_detect(codes, "R")) %>%
      mutate(sp = to_any_case(sp) %>% factor()),
    id = "ID"
  )
```

`comp_bayes_lm` *Fit Bayesian competition model*

Description

Fit a Bayesian linear regression model with interactions terms where

$$y = X\beta + \epsilon$$

μ	mean hyperparameter vector for β of length $p + 1$
V	covariance hyperparameter matrix for β of dimension $(p + 1)x(p + 1)$
a	shape hyperparameter for $\sigma^2 > 0$
b	scale hyperparameter for $\sigma^2 > 0$

Usage

```
comp_bayes_lm(focal_vs_comp, prior_param = NULL, run_shuffle = FALSE)
```

Arguments

`focal_vs_comp`	data frame from `create_focal_vs_comp()`
`prior_param`	A list of {a_0, b_0, mu_0, V_0} prior hyperparameters. Defaults to `a_0 = 250`, `b_0 = 250`, `mu_0` a vector of zeros of length $p + 1$, `V_0` an identity matrix of dimension $(p + 1)x(p + 1)$
`run_shuffle`	boolean as to whether to run permutation test shuffle of competitor tree species within a particular focal_ID

Value

A list of {a_star, b_star, mu_star, V_star} posterior hyperparameters

Source

Closed-form solutions of Bayesian linear regression doi: 10.1371/journal.pone.0229930.s004

See Also

Other modeling functions: `create_bayes_lm_data()`, `predict.comp_bayes_lm()`, `run_cv()`

Examples

```
library(dplyr)

# Load in focal versus comp
data(focal_vs_comp_ex)
```

```
comp_bayes_lm_ex <- focal_vs_comp_ex %>%
  comp_bayes_lm(prior_param = NULL, run_shuffle = FALSE)
```

`comp_bayes_lm_ex` *Example bayesian competition model fit*

Description

This object contains an example fitted Bayesian competition model outputted by `comp_bayes_lm()`.

Usage

```
comp_bayes_lm_ex
```

Format

A list subclass containing the following elements:

- **prior_params** Prior parameters supplied to `comp_bayes_lm()`
- **post_params** Posterior parameters outputted by `comp_bayes_lm()`
- **terms** The formula object used in model fitting

See Also

`comp_bayes_lm()`

Other example data objects: `blocks_ex`, `census_1_ex`, `census_2008_bw`, `census_2014_bw`, `census_2_ex`, `focal_vs_comp_ex`, `growth_ex`, `growth_spatial_ex`, `growth_toy`, `species_bw`, `study_region_bw`, `study_region_ex`

Examples

```
library(dplyr)
library(yardstick)

# Compare model predictions to observation
predictions <- focal_vs_comp_ex %>%
  mutate(growth_hat = predict(comp_bayes_lm_ex, focal_vs_comp_ex))

predictions %>%
  rmse(truth = growth, estimate = growth_hat) %>%
  pull(.estimate)

# Plot posterior parameters
comp_bayes_lm_ex %>%
  autoplot()
```

`create_bayes_lm_data` *Create input data frame for Bayesian regression*

Description

This function "widens" focal-competitor data frames for use inside of package modeling functions, where each `comp_sp` inside of the `comp` list-column receives its own column with its associated total basal area.

This function is used internally by `comp_bayes_lm()` and `predict.comp_bayes_lm()` exported as a convenience for applications extending this package's functionality.

Usage

```
create_bayes_lm_data(focal_vs_comp, run_shuffle = FALSE)
```

Arguments

`focal_vs_comp`	data frame from `create_focal_vs_comp()`
`run_shuffle`	boolean as to whether to run permutation test shuffle of competitor tree species within a particular focal_ID

Value

Data frame for internal package use.

See Also

Other modeling functions: `comp_bayes_lm()`, `predict.comp_bayes_lm()`, `run_cv()`

Other data processing functions: `compute_growth()`, `create_focal_vs_comp()`

Examples

```
create_bayes_lm_data(focal_vs_comp_ex)
```

`create_focal_vs_comp` *Create focal versus competitor trees data frame*

Description

"Focal versus competitor trees" data frames are the main data frame used for analysis. "Focal trees" are all trees that satisfy the following criteria

1. Were alive at both censuses
2. Were not part of the study region's buffer as computed by `add_buffer_variable()`
3. Were not a resprout at the second census. Such trees should be coded as `"R"` in the `codes2` variable (OK if a resprout at first census)

For each focal tree, "competitor trees" are all trees that (1) were alive at the first census and (2) within `comp_dist` distance of the focal tree.

Usage

```
create_focal_vs_comp(growth_df, comp_dist, blocks, id, comp_x_var)
```

Arguments

`growth_df`	A `compute_growth()` output converted to `sf` object
`comp_dist`	Distance to determine which neighboring trees to a focal tree are competitors.
`blocks`	An sf object of a `blockCV` block output
`id`	A character string of the variable name in `growth_df` uniquely identifying each tree
`comp_x_var`	A character string indicating which numerical variable to use as competitor explanatory variable

Value

`focal_vs_comp` data frame of all focal trees and for each focal tree all possible competitor trees. In particular, for each competitor tree we return `comp_x_var`. Potential examples of `comp_x_var` include basal area or estimate above ground biomass.

Note

In order to speed computation, in particular of distances between all focal/competitor tree pairs, we use the cross-validation `blockCV` object to divide the study region into smaller subsets.

See Also

Other data processing functions: `compute_growth()`, `create_bayes_lm_data()`

Examples

```
library(ggplot2)
library(dplyr)
library(stringr)
library(sf)
library(sfheaders)
library(tibble)
```

```
# Create fold information sf object.
SpatialBlock_ex <-
  tibble(
    # Study region boundary
    x = c(0, 0, 5, 5),
    y = c(0, 5, 5, 0)
  ) %>%
  # Convert to sf object
  sf_polygon() %>%
  mutate(folds = "1")

# Plot example data. Observe for comp_dist = 1.5, there are 6 focal vs comp pairs:
# 1. focal 1 vs comp 2
# 2. focal 2 vs comp 1
# 3. focal 2 vs comp 3
# 4. focal 3 vs comp 2
# 5. focal 4 vs comp 5
# 6. focal 5 vs comp 4
ggplot() +
  geom_sf(data = SpatialBlock_ex, fill = "transparent") +
  geom_sf_label(data = growth_toy, aes(label = ID))

# Return corresponding data frame
growth_toy %>%
  mutate(basal_area = 0.0001 * pi * (dbh1 / 2)^2) %>%
  create_focal_vs_comp(comp_dist = 1.5, blocks = SpatialBlock_ex, id = "ID",
                       comp_x_var = "basal_area")

# Load in growth_df with spatial data
# See ?growth_ex for attaching spatial data to growth_df
data(growth_spatial_ex)
# Load in blocks
data(blocks_ex)

focal_vs_comp_ex <- growth_spatial_ex %>%
  mutate(basal_area = 0.0001 * pi * (dbh1 / 2)^2) %>%
  create_focal_vs_comp(comp_dist = 1, blocks = blocks_ex, id = "ID",
                       comp_x_var = "basal_area")
```

`focal_vs_comp_distance` *Return all pairwise distances between two data frames of trees*

Description

Return all pairwise distances between two data frames of trees

Usage

```
focal_vs_comp_distance(focal_trees, comp_trees)
```

Arguments

`focal_trees`	An `sf` polygon object of the focal trees of interest
`comp_trees`	An `sf` polygon object of the competitor trees

Value

A data frame with three columns: `focal_ID` of focal tree, `comp_dist` of competitor tree, and `dist` of distance between them.

See Also

Other spatial functions: `add_buffer_variable()`, `compute_buffer_region()`

Examples

```
library(tibble)
library(ggplot2)
library(sf)

# Create toy example focal and competitor trees
focal_trees <- tibble(
  focal_ID = c(1, 2, 3),
  x = c(0.3, 0.6, 0.7),
  y = c(0.1, 0.5, 0.7)
) %>%
  st_as_sf(coords = c("x", "y"))

comp_trees <- tibble(
  comp_ID = c(4, 5, 6, 7),
  x = c(0, 0.2, 0.4, 0.6),
  y = c(0.6, 0.7, 1, 0.2)
) %>%
  st_as_sf(coords = c("x", "y"))

# Plot both sets of trees
ggplot() +
  geom_sf_label(data = focal_trees, aes(label = focal_ID), col = "black") +
  geom_sf_label(data = comp_trees, aes(label = comp_ID), col = "orange") +
  labs(title = "Focal trees in black, competitor trees in orange")

# Compute corresponding distances between the 3 focal trees and 4 competitor trees
focal_vs_comp_distance(focal_trees, comp_trees)
```

`focal_vs_comp_ex` *Example focal versus comp data frame*

Description

This is an example focal versus comp data frame. The rows are focal trees which are repeated for all competitor trees within a specified distance from them. In this case that distance is 1. This is the focal versus comp for growth_spatial_ex.

Usage

```
focal_vs_comp_ex
```

Format

A tibble::tbl_df:

focal_ID Tree identification number for the focal tree

focal_sp Species of the focal tree

dbh Diameter at breast height of the focal tree at the first census

foldID The CV-fold that the focal tree is in

geometry The point location of the focal tree

growth The average annual growth of the focal tree between censuses

comp A list-column: characteristics of the relevant competitor trees

The `comp` list-column contains tibble::tbl_dfs with columns:

comp_ID Tree identification number for the competitor tree

dist The distance between the focal and comp tree, this will be less than the max distance specified.

comp_sp Species of the comp tree

comp_x_var Numerical variable associated with comp tree

See Also

`create_focal_vs_comp()`

Other example data objects: `blocks_ex`, `census_1_ex`, `census_2008_bw`, `census_2014_bw`, `census_2_ex`, `comp_bayes_lm_ex`, `growth_ex`, `growth_spatial_ex`, `growth_toy`, `species_bw`, `study_region_bw`, `study_region_ex`

Examples

```
comp_bayes_lm_ex <- focal_vs_comp_ex %>%
  comp_bayes_lm(prior_param = NULL, run_shuffle = FALSE)
```

`forestecology` `forestecology` *package*

Description

Methods and data for forest ecology model selection and assessment

Details

See the README on GitHub

`growth_ex` *Example growth data frame for small example*

Description

This is an example growth data frame formed from two census data frames. In this case it is made by combining census_1_ex and census_2_ex. The individuals alive in both censuses were linked by their tree ID.

Usage

`growth_ex`

Format

A `sf` spatial tibble

ID Tree identification number. This identifies an individual tree and can be used to connect trees between the two censuses.

sp Species of the individual

codes1 Code for additional information on the stem during the first census: M means the main stem of the individual tree and R means the stem was lost, but the tag was moved to another stem greater than DBH cutoff, this stands for resprout.

dbh1 Diameter at breast height of the tree in cm at the first census

dbh2 Diameter at breast height of the tree in cm at the second census

growth Average annual growth between the two censuses in cm per year

codes2 Codes at the second census

geometry Point location of the individual

See Also

`compute_growth()`

Other example data objects: `blocks_ex`, `census_1_ex`, `census_2008_bw`, `census_2014_bw`, `census_2_ex`, `comp_bayes_lm_ex`, `focal_vs_comp_ex`, `growth_spatial_ex`, `growth_toy`, `species_bw`, `study_region_bw`, `study_region_ex`

Examples

```
library(ggplot2)
library(dplyr)
library(sf)
library(sfheaders)
library(blockCV)

growth_ex %>%
  ggplot() +
  geom_sf()

growth_ex %>%
  group_by(sp) %>%
  summarize(mean(growth))

# Add buffer
growth_spatial_ex <- growth_ex %>%
  add_buffer_variable(direction = "in", size = 1, region = study_region_ex)

# Add cross-validation folds
fold1 <- rbind(c(0, 0), c(5, 0), c(5, 5), c(0, 5), c(0, 0))
fold2 <- rbind(c(5, 0), c(10, 0), c(10, 5), c(5, 5), c(5, 0))
blocks <- bind_rows(
  sf_polygon(fold1),
  sf_polygon(fold2)
) %>%
  mutate(foldID = c(1, 2))

SpatialBlock_ex <- spatialBlock(
  speciesData = growth_ex,
  verbose = FALSE,
  k = 2,
  selection = "systematic",
  blocks = blocks
)

# Add foldID to data
growth_spatial_ex <- growth_spatial_ex %>%
  mutate(foldID = SpatialBlock_ex$foldID %>% as.factor())
```

`growth_spatial_ex` *Example growth data frame with spatial data for small example*

Description

This is an example growth data frame formed from two census data frames which has been updated with spatial data. It starts from growth_ex.

Usage

```
growth_spatial_ex
```

Format

A `sf` spatial tibble

ID Tree identification number. This identifies an individual tree and can be used to connect trees between the two censuses.

sp Species of the individual

codes1 Code for additional information on the stem during the first census: M means the main stem of the individual tree and R means the stem was lost, but the tag was moved to another stem greater than DBH cutoff, this stands for resprout.

dbh1 Diameter at breast height of the tree in cm at the first census

dbh2 Diameter at breast height of the tree in cm at the second census

growth Average annual growth between the two censuses in cm per year

codes2 Codes at the second census

geometry Point location of the individual

buffer A boolean variable for whether the individual is in the buffer region or not

foldID Which cross-validation fold the individual is in

See Also

`compute_growth()`

Other example data objects: `blocks_ex`, `census_1_ex`, `census_2008_bw`, `census_2014_bw`, `census_2_ex`, `comp_bayes_lm_ex`, `focal_vs_comp_ex`, `growth_ex`, `growth_toy`, `species_bw`, `study_region_bw`, `study_region_ex`

Examples

```
library(ggplot2)
library(dplyr)
library(sf)

comp_dist <- 1

ggplot() +
  geom_sf(data = growth_spatial_ex, aes(col = buffer), size = 2)

ggplot() +
  geom_sf(data = growth_spatial_ex, aes(col = foldID), size = 2)
```

```
# Create the focal versus comp data frame
focal_vs_comp_ex <- growth_spatial_ex %>%
  mutate(basal_area = 0.0001 * pi * (dbh1 / 2)^2) %>%
 create_focal_vs_comp(comp_dist, blocks = blocks_ex, id = "ID", comp_x_var = "basal_area")
```

`growth_toy` *Example input data for* `create_focal_vs_comp()`

Description

An example `sf` of type generated by `compute_growth()`

Usage

```
growth_toy
```

Format

A `sf` spatial features polygon

See Also

Other example data objects: `blocks_ex`, `census_1_ex`, `census_2008_bw`, `census_2014_bw`, `census_2_ex`, `comp_bayes_lm_ex`, `focal_vs_comp_ex`, `growth_ex`, `growth_spatial_ex`, `species_bw`, `study_region_bw`, `study_region_ex`

`predict.comp_bayes_lm` *Make predictions based on fitted Bayesian model*

Description

Applies fitted model from `comp_bayes_lm()` and returns posterior predicted values.

Usage

```
## S3 method for class 'comp_bayes_lm'
predict(object, newdata, ...)
```

Arguments

`object`	Output of `comp_bayes_lm()`: A list of {a_star, b_star, mu_star, V_star} posterior hyperparameters
`newdata`	A data frame of type `focal_vs_comp` in which to look for variables with which to predict.
`...`	Currently ignored—only included for consistency with generic.

Value

A vector of predictions with length equal to the input data.

Source

Closed-form solutions of Bayesian linear regression doi: 10.1371/journal.pone.0229930.s004

See Also

Other modeling functions: `comp_bayes_lm()`, `create_bayes_lm_data()`, `run_cv()`

Examples

```
library(dplyr)
library(sf)
library(ggplot2)

# Load in posterior parameter example
# and growth data to compare to
data(comp_bayes_lm_ex, growth_ex)

predictions <- focal_vs_comp_ex %>%
  mutate(growth_hat = predict(comp_bayes_lm_ex, focal_vs_comp_ex))

predictions %>%
  ggplot(aes(growth, growth_hat)) +
  geom_point() +
  geom_abline(slope = 1, intercept = 0)
```

`run_cv` *Run the bayesian model with spatial cross validation*

Description

This function carries out the bayesian modeling process with spatial cross-validation as described in Allen and Kim (2020). Given a focal-competitor data frame, it appends a column with predicted growth values.

Usage

```
run_cv(
  focal_vs_comp,
  comp_dist,
  blocks,
  prior_param = NULL,
  run_shuffle = FALSE
)
```

Arguments

`focal_vs_comp`	data frame from `create_focal_vs_comp()`
`comp_dist`	Distance to determine which neighboring trees to a focal tree are competitors.
`blocks`	An sf object of a `blockCV` block output
`prior_param`	A list of {a_0, b_0, mu_0, V_0} prior hyperparameters. Defaults to `a_0 = 250`, `b_0 = 250`, `mu_0` a vector of zeros of length $p + 1$, `V_0` an identity matrix of dimension $(p + 1)x(p + 1)$
`run_shuffle`	boolean as to whether to run permutation test shuffle of competitor tree species within a particular focal_ID

Value

`focal_vs_comp` with new column of predicted `growth_hat`

See Also

Other modeling functions: `comp_bayes_lm()`, `create_bayes_lm_data()`, `predict.comp_bayes_lm()`

Examples

```
run_cv(
  focal_vs_comp_ex,
  comp_dist = 1,
  blocks = blocks_ex
)
```

`species_bw` *Phylogenic groupings and trait based clustering of various tree species*

Description

A date frame mapping the species codes to their common names, scientific names, and families. This also includes a trait-based clustering of the species.

Usage

```
species_bw
```

Format

A data frame with 46 rows and 6 variables:

- **sp** The code for the species. Link to `census_2008_bw` and `census_2014_bw` with `sp` variable.
- **genus** Genus
- **species** Species epithet
- **latin** Scientific name
- **family** Family
- **trait_group** Clustering of species based on three traits rather than their evolutionary relationships. The traits are specific leaf area, maximum height, and wood density

Source

For more information on trait clustering see Allen and Kim 2020 "A permutation test and spatial cross-validation approach to assess models of interspecific competition between trees." doi: 10.1371/journal.pone.0229930PLOS One 15: e0229930.

See Also

Other Big Woods data: `census_2008_bw`, `census_2014_bw`, `study_region_bw`

Other example data objects: `blocks_ex`, `census_1_ex`, `census_2008_bw`, `census_2014_bw`, `census_2_ex`, `comp_bayes_lm_ex`, `focal_vs_comp_ex`, `growth_ex`, `growth_spatial_ex`, `growth_toy`, `study_region_bw`, `study_region_ex`

Examples

```
library(dplyr)

# Original 2008 census data
census_2008_bw

# 2008 census data with additional species information
census_2008_bw %>%
  left_join(species_bw, by = "sp")
```

`study_region_bw` *Bigwoods forest study region boundary*

Description

Boundary region for Bigwoods defined in terms of (x,y) vertices of a polygon.

Usage

```
study_region_bw
```

Format

A `sf` spatial features polygon

See Also

Other Big Woods data: `census_2008_bw`, `census_2014_bw`, `species_bw`

Other example data objects: `blocks_ex`, `census_1_ex`, `census_2008_bw`, `census_2014_bw`, `census_2_ex`, `comp_bayes_lm_ex`, `focal_vs_comp_ex`, `growth_ex`, `growth_spatial_ex`, `growth_toy`, `species_bw`, `study_region_ex`

Examples

```
library(ggplot2)
library(sf)

# Convert all 2008 plot stems to sf object
census_2008_bw_sf <- census_2008_bw %>%
  st_as_sf(coords = c("gx", "gy"))

# Plot stems with plot boundary
ggplot() +
  geom_sf(data = census_2008_bw_sf, size = 0.25) +
  geom_sf(data = study_region_bw, color = "red", fill = "transparent")
```

`study_region_ex` *Study region for example data*

Description

Boundary region for small example data set defined in terms of (x,y) vertices of a polygon.

Usage

```
study_region_ex
```

Format

A `sf` spatial features polygon

See Also

Other example data objects: `blocks_ex`, `census_1_ex`, `census_2008_bw`, `census_2014_bw`, `census_2_ex`, `comp_bayes_lm_ex`, `focal_vs_comp_ex`, `growth_ex`, `growth_spatial_ex`, `growth_toy`, `species_bw`, `study_region_bw`

Examples

```
library(ggplot2)
library(sf)

# Convert stems to sf object
census_1_ex_sf <- census_1_ex %>%
  st_as_sf(coords = c("gx", "gy"))

# Plot stems with plot boundary
ggplot() +
  geom_sf(data = study_region_ex) +
  geom_sf(data = study_region_bw, color = "red", fill = "transparent")
```

Index