The forestecology R package for fitting and assessing models of interspecies competitive effects on the growth of trees

Albert Y. Kim
Program in Statistical & Data Sciences, Smith College
and
David Allen
Biology Department, Middlebury College
and
Simon P. Couch
Mathematics Department, Reed College

February 22, 2021

Abstract

1. Many models for the growth of trees that incorporate the effect of interspecies competition are based on a neighborhood effect assumption whereby all trees within a fixed distance of all focal trees are considered competitors. Methods and tools are needed to quantify this competitive effect and assess the quality of all resulting models
2. We present the `forestecology` package providing methods for both 1) evaluating the effect of competitor species identity using permutation tests and 2) evaluating model performance using spatial cross-validation. Following Allen & Kim (2020), we implement a Bayesian linear regression competition model.
3. We demonstrate the package's functionality using data from the Smithsonian Conservation Biology Institute's large forest dynamics plot, part of the ForestGEO global network of reseach sites. Given ForestGEO's data collection protocols and data formatting standards, the package was designed with cross-site compatibility in mind. We demonstrate that both 1) competitor species identity matters and 2) that not spatially cross-validating leads to error estimates that are overly optimistic.
4. The package features 1) `tidyverse`-like structure whereby verb-named functions can be modularly "piped" in sequence, 2) functions with standardized inputs/outputs of simple features `sf` package class, and 3) R S3 object-oriented implementation of the Bayesian linear regression model. These three facts allow for clear articulation of all the steps in the sequence of analysis and easy wrangling and visualization of

the geospatial forestry data. Furthermore, while the package only has Bayesian linear regression implemented, the package was designed with extensibility to other methods in mind.

Keywords: forest ecology, competition, R, Rstats, tidyverse, sf, cross-validation, spatial statistics

1 Introduction

Repeat-censused forest plots offer excellent data to test neighborhood models of tree competition Allen & Kim (2020) Canham et al. (2006) Uriarte et al. (2004). Here we describe an R package, `forestecology`, to do that. This package implements the methods in Allen & Kim (2020). It provides: a convenient way to specify and fit models of tree growth based on neighborhood competition; a spatial cross validation method to test and compare model fits Roberts et al. (2017); and an ANOVA-like method to assess whether the competitor identity matters in these models. The model is written to work with ForestGEO plot data Anderson-Teixeira et al. (2015), but we envision that it could easily be modified to work with data from other forest plots, e.g. the US Forest Service Forest Inventory and Analysis plots Smith (2002).

The `forestecology` is designed with "tidy" data principles in mind as Wickham et al. (2019).

Given that our data is of geo-spatial nature, we represent our data using the "simple features" `sf` package class of objects Pebesma (2018) whereby. While previously the `sp` package serves such purposes Pebesma & Bivand (2005), the `sf` package is designed to interface with the `tidyverse` suite of packages.

1.1 Model for growth of tree

While there are a littany of models one can consider, in PLOSOne we considered a simple one.

Describe model specifics. Bayesian linear regression model.

Next we fit the following linear model to the DBH of each focal tree. Let $i = 1, \ldots, n_j$ index all n_j trees of "focal" species group j; let $j = 1, \ldots, J$ index all J focal species groups; and let $k = 1, \ldots, K$ index all K "competitor" species groups. We modeled the growth in diameter per year y_{ij} (in centimeters per year) of the i^{th} tree of focal species group j as a linear model f of the following covariates $\vec{x}_{ij}$

$$y_{ij} = f(\vec{x}_{ij}) + \epsilon_{ij} = \beta_{0,j} + \beta_{\text{DBH},j} \cdot \text{DBH}_{ij} + \sum_{k=1}^{K} \lambda_{jk} \cdot \text{BA}_{ijk} + \epsilon_{ij}$$

Link to `https://doi.org/10.1371/journal.pone.0229930.s004`

For this linear model's case, there exists a closed form solution as described here. As such, the `fit_bayesian_model()` function using matrix algebra to obtain all parameter estimates, rather than computationally expensive Monte Carlo approximations. The inputs to this function are a `focal_vs_comp` data frame, `prior_param` a list of priors, and a boolean flag `run_shuffle` on whether or not to run competitor-species identity permutations which we will demonstrate below on the Michigan Big Woods data. This function returns the posterior means of all parameters.

2 Example

We demonstrate the `forestecology` package's functionality on data from the Smithsonian Conservation Biology Institute (SCBI) large forest dynamics plot, located at the Smithsonian's National Zoo and Conservation Biology Institute in Front Royal, VA, USA. The 25.6 ha (640 x 400 m) plot is located at the intersection of three of the major physiographic provinces of the eastern US: the Blue Ridge, Ridge and Valley, and Piedmont provinces and is adjacent to the northern end of Shenandoah National Park. The forest type is typical mature secondary eastern mixed deciduous forest, with a canopy dominated by tulip poplar (*Liriodendron tulipifera*), oaks (*Quercus* spp.), and hickories (*Carya* spp.), and an understory composed mainly of spicebush (*Lindera benzoin*), paw-paw (*Asimina triloba*), American hornbeam (*Carpinus caroliniana*), and witch hazel (*Hamamelis virginiana*) Bourg et al. (2013).

The `forestecology` package attempts has the following two ecological goals: 1) to evaluate the effect of competitor species identity using permutation tests and 2) to evaluate model performance using spatial cross-validation. To achieve these two goals, we outline a basic analysis sequence comprising of these four main steps:

1. Compute the growth of trees based on census data.
2. Add spatial information:
    1. Define a buffer region of trees.
    2. Add spatial cross-validation block information.

3. Identify all focal trees and their competitors.
4. Apply model, which includes:
    1. Fit model.
    2. Compute fitted/predicted values.
    3. Visualize posterior distributions.

We start by loading all necessary packages.

```r
library(tidyverse)
library(lubridate)
library(sf)
library(patchwork)
library(forestecology)
library(blockCV)
```

2.1 Step 1: Compute the growth of trees based on census data

The first step in the our analysis sequence is to compute the growth of trees using data from two censuses. The `compute_growth()` function computes growth assuming census data that roughly follows ForestGEO standards. Despite such standards, minor variations will still exist between sites, thereby necessitating some data wrangling and checking. For example, the SCBI site records all DBH's in millimeters, whereas the Michigan Big Woods site records them in centimeters Anderson-Teixeira et al. (2015) Allen et al. (2020).

We load both 2008 and 2014 SCBI census data `.csv` files as they existed on GitHub on November 20, 2020. After selecting only the relevant variables, we perform a few additional data wrangling steps: convert the character variable with the date of measurement to be of explicit type `date`, convert DBH to be in centimeters[1], convert the `sp` variable containing

[1]A rule of thumb to determine the units of DBH is to check if the smallest non-zero and non-missing measurement is 1 or 10. If the former, then centimeters. If the later, then millimeters. This is because ForestGEO protocols state that only trees with DBH greater or equal to 1cm should be included in censuses.

species information from type `chr` character to `fct` factor.[2] Furthermore, in order to speed up computation for purposes of this example, we only consider a 9 ha subsection of the 25.6 ha of the SCBI site: `gx` from 0–300 instead of 0–400 and `gy` from 300–600 instead of 0–640.

```r
census_2013_scbi <- read_csv("scbi.stem2.csv") %>%
  select(stemID, sp, date = ExactDate, gx, gy, dbh, codes, status) %>%
  mutate(
    date = mdy(date),
    dbh = as.numeric(dbh)/10,
    sp = factor(sp)
  ) %>%
  filter(gx < 300, between(gy, 300, 600))

census_2018_scbi <- read_csv("scbi.stem3.csv") %>%
  select(stemID, sp, date = ExactDate, gx, gy, dbh, codes, status) %>%
  mutate(
    date = mdy(date),
    dbh = as.numeric(dbh)/10,
    sp = factor(sp)
  ) %>%
  filter(gx < 300, between(gy, 300, 600))
```

These two data frames are then used as arguments to the `compute_growth()` function, along with the `id` argument that specifies the variable that uniquely identifies each tree-stem. Note furthermore that we discard all resprouts in the later census (those with `code == R`).

[2] In our spatial cross-validation algorithm in Section 2.6 issues can occur when rare species do not occur in the training set, but then are encountered in the test set. This risk is mitigated by representing `sp` as a factor variable, which has a complete list of all levels of the categorical variable.

```
growth_scbi <-
  compute_growth(
    census_1 = census_2013_scbi,
    census_2 = census_2018_scbi %>% filter(!str_detect(codes, "R")),
    id = "stemID"
  )
growth_scbi
## Simple feature collection with 7954 features and 8 fields
## geometry type:  POINT
## dimension:      XY
## bbox:           xmin: 0.2 ymin: 300 xmax: 299.9 ymax: 600
## CRS:            NA
## # A tibble: 7,954 x 9
##    stemID sp     dbh1 codes1 status  dbh2 codes2  growth     geometry
##     <dbl> <fct> <dbl> <chr>  <chr>  <dbl> <chr>    <dbl>      <POINT>
##  1      4 nysy  13.6  M      A       14.2 M        0.103  (14.2 428.5)
##  2      5 havi   8.8  M      A        9.6 M;P      0.150   (9.4 436.4)
##  3      6 havi   3.25 NULL   A        4   M        0.140     (1.3 434)
##  4     77 qual  65.2  M      A       66   M        0.141  (34.7 307.2)
##  5     79 tiam  47.7  M      A       46.8 M       -0.161    (40 381.1)
##  6     80 caca   5.15 M      A        6.5 M        0.253  (38.7 421.7)
##  7     96 libe   2.3  J;M    A        3.7 M        0.262      (60 310)
##  8    100 caca   5.09 NULL   A       NA   DN      NA      (52.5 476.3)
##  9    101 litu  65.4  M      A       68.4 M        0.552  (47.1 567.3)
## 10    102 astr   1.99 NULL   A        2.5 M        0.0954 (40.8 575.5)
## # ... with 7,944 more rows
```

The output `growth_scbi` is a single data frame of class `sf` that includes `growth`, the average annual growth in DBH (in cm) for all trees that were alive at both time points, as well a `geometry` variable encoding each tree's geolocation. Furthermore, the variables that should remain unchanged between censuses appear only once, such as location variables `gx`

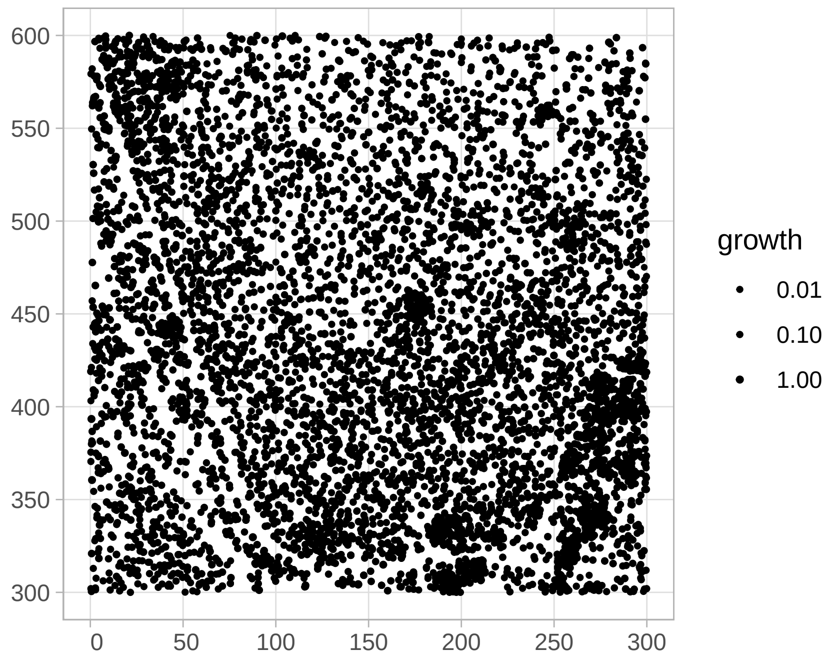

Figure 1: Compute growth of trees based on census data: Map with growth of all trees from a 9 ha subsection of the Smithsonian Conservation Biology Institute (SCBI) forest plot.

and `gy`; as well as species-related variables. Variables that should change between censuses are suffixed with `1` and `2` indicating the earlier and later censuses, such as `dbh1/dbh2` and `codes1/codes2`.

The data format of other sites may be such that our `compute_growth()` function doesn't work at all. However, in the end all that matters is that the growth of all trees is saved in a data frame of class `sf` whereby the geolocation of each tree is presented in a `geometry` variable of type <`POINT`> and at a minimum the data contains the variables above.

Given that `growth_scbi` is of class `sf`, it can be easily plotted in `ggplot2` using the `geom_sf()` geometry as seen in Figure 1.

```r
# TODO: Rescale points in this plot
ggplot() +
  geom_sf(data = growth_scbi, aes(size = growth)) +
  scale_size(breaks = c(0.01, 0.1, 1), range = c(0.1, 1))
```

2.2 Step 2: Add spatial information

The next step in our analysis sequence is to add spatial information to our main `growth_scbi` data frame. The first element of spatial information we add is a "buffer region" to the periphery of the study region. Since some of our model's explanatory variables are cumulative (such as competitor basal area), we must ensure that all trees being modeled are not biased to have different neighbor structures. This is of concern for trees at the boundary of study regions, which will not have the same number of neighbors as trees in the internal part of the study region. In order to account for such edge effects, only trees that are not part of this buffer region, i.e. are part of the interior of the study region, will have their growths modeled Waller & Gotway (2004).

Our model of interspecific competition relies on a spatial definition of who the competitor trees are for focal trees of interest: all trees within a distance `comp_dist` of a focal tree are considered its competitors (assuming the same units as the `gx` and `gy` location variables). In our case we set this value at 7.5m, a value informed by Canham et al. (2004) Uriarte et al. (2004) Canham et al. (2006). Using this value along with a manually constructed `sf` object representation of the study region's boundary, we apply the `add_buffer_variable()` to our `growth_scbi` data frame to add a `buffer` boolean variable: all trees who have `buffer` set to `FALSE` will be our focal trees whose growths are modeled, whereas those with `TRUE` will only be considered as competitor trees whose growth will not be modeled.

```r
# Define buffer region using competitive distance range
comp_dist <- 7.5


study_region_scbi <- tibble(
  x = c(0, 300, 300, 0, 0),
  y = c(300, 300, 600, 600, 300)
) %>%
  sf_polygon()


growth_scbi <- growth_scbi %>%
  add_buffer_variable(size = comp_dist, region = study_region_scbi)
```

The second element of spatial information are blocks corresponding to folds of a spatial cross-validation algorithm used to estimate out-of-sample model error. Conventional cross-validation algorithms assign observations to folds by randomly resampling individual observations. However, an assumption that the observations are independent underlies any cross-validation algorithm. In the case of forest census data, observations exhibit spatial autocorrelation. We therefore incorporate this spatial dependence into the cross-validation algorithm by randomly resampling spatial blocks of trees Roberts et al. (2017) Pohjankukka et al. (2017).

In the example below, we first manually define four folds that partition the study region as an `sf` object. We then use the output of the `spatialBlock()` function from the `blockCV` package to associate each tree in `growth_scbi` to the correct fold (saved in the `foldID` variable) Valavi et al. (2019). [3]

```
# Manually define spatial blocks to act as folds
n_fold <- 4
fold1 <- rbind(c(0, 300), c(150, 300), c(150, 450), c(0, 450))
fold2 <- rbind(c(150, 300), c(300, 300), c(300, 450), c(150, 450))
fold3 <- rbind(c(0, 450), c(150, 450), c(150, 600), c(0, 600))
fold4 <- rbind(c(150, 450), c(300, 450), c(300, 600), c(150, 600))

blocks_scbi <- bind_rows(
  sf_polygon(fold1), sf_polygon(fold2), sf_polygon(fold3), sf_polygon(fold4)
) %>%
  mutate(folds = c(1:n_fold) %>% factor())

# Associate each observation to a fold
SpatialBlock_scbi <- spatialBlock(
```

[3] In the Appendix we present an example where the folds themselves are also created using the `spatialBlock()` function given a specified `cv_block_size`.

```
  speciesData = growth_scbi, k = n_fold, selection = "systematic",
  blocks = blocks_scbi, showBlocks = FALSE, verbose = FALSE
)

growth_scbi <- growth_scbi %>%
  mutate(foldID = SpatialBlock_scbi$foldID %>% factor())
```

Figure 2 illustrates the net effect of adding these two elements of information to the `growth_scbi` data frame. The location of each tree is marked with an integer indicating which fold it belongs to, where the folds are marked with solid lines. The color of each digit indicates whether the tree is part of the buffer region (and thus will only be considered as a competitor tree in our model) or is part of the interior of the study region (and thus is a focal tree whose growth is of modeled interest).

```
ggplot() +
  geom_sf(data = blocks_scbi, fill = "transparent", linetype = "dashed") +
  geom_sf_text(data = growth_scbi %>% sample_n(1000), aes(label = foldID, col = buffer)
```

2.3 Step 3: Identify all focal and corresponding competitor trees

The next step in our analysis sequence is to identify all focal trees and their corresponding competitor trees. The `create_focal_vs_comp()` functions performs these tasks and returns a new data frame of type `sf`. On top of the previous arguments `comp_dist` defining the competition neighborhood and `id` indicating which variable uniquely identifies each tree-stem, this function also requires an `sf` object representation of the spatial cross-validation blocks/folds; in our case, this was manually encoded in the `blocks_scbi` in Section 2.2 while in our Appendix we present an example where this was performed using `spatialBlock()` from the `blockCV` package. We present the resulting data frame below with the `foldID` variable omitted for compactness of presentation.

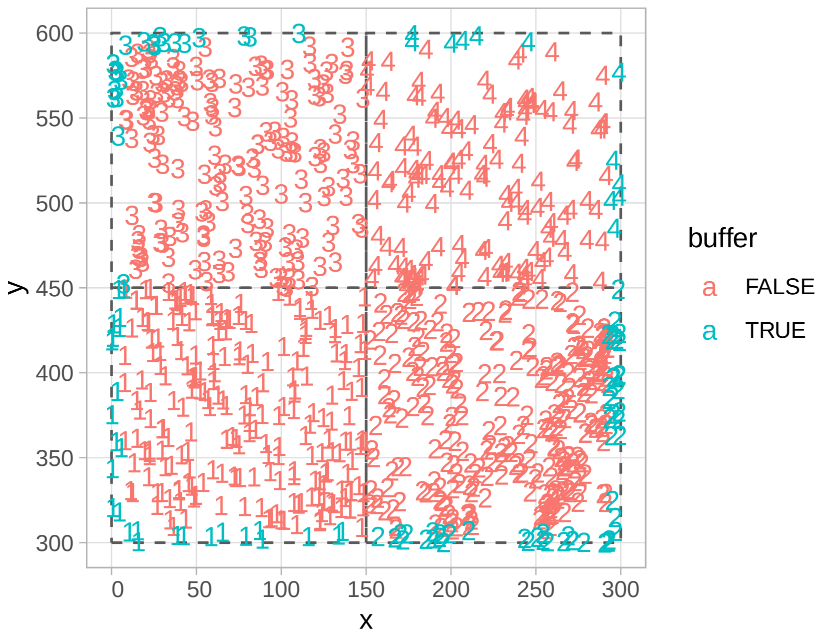

Figure 2: Add spatial information: Buffer region and spatial cross-validation blocks (1 through 4). All trees in the interior of the study region (i.e. not part of buffer) will be the focal trees whose growth will be modeled.

```
# TODO: Below reconcile the number of rows as they off by one from growth_scbi %>%
# filter(!is.na(growth) & !buffer). Perhaps by removing NA's in the growth_scbi stage?
focal_vs_comp_scbi <- growth_scbi %>%
  create_focal_vs_comp(comp_dist, cv_grid_sf = blocks_scbi, id = "stemID")
focal_vs_comp_scbi %>%
  select(-foldID)
## # A tibble: 6,296 x 6
##    focal_ID focal_sp   dbh     geometry  growth comp
##       <dbl> <fct>    <dbl>      <POINT>   <dbl> <list>
##  1        4 nysy     13.6  (14.2 428.5)  0.103  <tibble [20 x 4]>
##  2        5 havi      8.8   (9.4 436.4)  0.150  <tibble [32 x 4]>
##  3       79 tiam     47.7    (40 381.1) -0.161  <tibble [20 x 4]>
##  4       80 caca      5.15 (38.7 421.7)  0.253  <tibble [12 x 4]>
##  5       96 libe      2.3      (60 310)  0.262  <tibble [14 x 4]>
##  6      101 litu     65.4   (47.1 567.3)  0.552  <tibble [19 x 4]>
##  7      102 astr      1.99  (40.8 575.5)  0.0954 <tibble [44 x 4]>
##  8      126 cato     37.4   (60.6 400.2)  0.165  <tibble [16 x 4]>
##  9      127 caca      8.72  (72.7 514.1)  0.0370 <tibble [14 x 4]>
## 10      139 astr      1.71  (96.7 315.1)  0.0549 <tibble [48 x 4]>
## # ... with 6,286 more rows
```

The resulting data frame `focal_vs_comp_scbi` has 6296 rows, representing the subset of the 7954 trees in `growth_scbi` that will be considered as focal trees. Two new variables `focal_ID` and `focal_sp` relate to tree-stem identification and species information. Most notably however is a new variable `comp` which contains information on all competitor trees for a given focal tree, saved in `tidyr` package list-column format Wickham (2020). For example, we drill-down on the tree with `focal_ID` 4, which has 20 competitor trees each described by 4 variables as indicated by the fact that `comp` is a <`tibble [20 × 4]`>.

```
focal_vs_comp_scbi %>%
  filter(focal_ID == 4) %>%
  select(focal_ID, dbh, comp)
## # A tibble: 1 x 3
##   focal_ID   dbh comp
##      <dbl> <dbl> <list>
## 1        4  13.6 <tibble [20 x 4]>
```

The spatial distribution of these trees is visualized in Figure 3: the dashed circle extends 7.5 m away from the focal tree while all 20 competitor trees are within this circle.

Using the `unnest()` function from the `tidyr` package, we can flatten list-column into regular columns. We observe that for the same focal tree, we have information on all 20 competitor trees whose `dist` distance to the focal tree is ≤ 7.5: their unique tree-stem ID number, their species, and their basal area (in m^2) calculated as $\frac{\pi \times (\mathrm{DBH}/2)^2}{10000}$ where DBH is the value from the earlier of the two censuses in cm. Saving our focal versus competitor information in list-column minimizes redundancy since we do not repeat information on the focal tree 20 times.

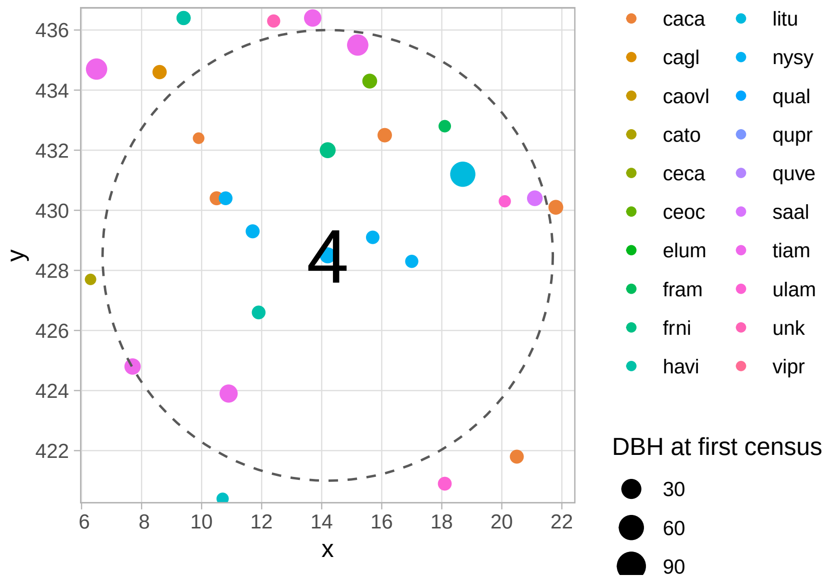

Figure 3: Identify all focal and corresponding competitor trees: All 20 competitor trees of focal tree 4.

```
focal_vs_comp_scbi %>%
  filter(focal_ID == 4) %>%
  select(focal_ID, dbh, comp) %>%
  unnest(cols = "comp")
```

```
## # A tibble: 20 x 6
##   focal_ID   dbh comp_ID  dist comp_sp comp_basal_area
##      <dbl> <dbl>   <dbl> <dbl> <fct>             <dbl>
## 1        4  13.6    1836  7.48 tiam            0.0176
## 2        4  13.6    1847  2.81 nysy            0.00332
## 3        4  13.6    1848  1.62 nysy            0.00396
## 4        4  13.6    1849  2.62 nysy            0.00535
## 5        4  13.6    1850  2.98 havi            0.00472
## # ... with 15 more rows
```

2.4 Step 4: Fit model

Now that we've identified all focal and corresponding competitor trees and saved this information in a data frame of type `focal_vs_comp`, the final step in our analysis sequence is to fit a model for the growth of all focal trees. Currently the `forestecology` package can only fit the competition Bayesian linear regression model outlined in Section 1.1 using the `comp_bayes_lm()` function. However, any model implemented in a function that similarly takes an input data frame of type `focal_vs_comp` as an argument can be easily swapped in. For our specific competition Bayesian linear regression model, we also specify prior distributions on all parameters of interest (here chosen to be the defaults as specified in `?comp_bayes_lm`).

```
# TODO: Add information about default priors in ?comp_bayes_lm()
comp_bayes_lm_scbi <- focal_vs_comp_scbi %>%
  comp_bayes_lm(prior_param = NULL)
```

The returned `comp_bayes_lm_scbi` output is an object of S3 class type `comp_bayes_lm` which contains the posterior values of all parameters in our competition Bayesian linear regression. This class of object includes generic methods implemented for `print()`, `predict()`, and `ggplot2::autoplot()`. First the generic for `print()` displays the names of all prior & posterior parameters along with the model formula:

```
comp_bayes_lm_scbi
## Bayesian linear regression model parameters with a multivariate Normal likelihood.
##
##   parameter_type            prior posterior
## 1 Inverse-Gamma on sigma^2 a_0   a_star
## 2 Inverse-Gamma on sigma^2 b_0   b_star
## 3 Multivariate t on beta   mu_0  mu_star
## 4 Multivariate t on beta   V_0   V_star
##
## Model formula:
## growth ~ sp + dbh + dbh * sp + acne * sp + acpl * sp + acru * sp + acsp * sp + aia
```

Next, the generic for `predict()` takes as inputs the posterior parameter values in `comp_bayes_lm_scbi` and the predictor variables in `newdata` and outputs a vector of fitted/predicted values $\widehat{y}$ of the DBH for each focal tree computed from the posterior predictive distribution.

```
focal_vs_comp_scbi <- focal_vs_comp_scbi %>%
  mutate(growth_hat = predict(comp_bayes_lm_scbi, newdata = focal_vs_comp_scbi))
```

```
focal_vs_comp_scbi
## # A tibble: 6,296 x 8
##    focal_ID focal_sp   dbh foldID                 geometry  growth
##       <dbl> <fct>    <dbl> <fct>                   <POINT>   <dbl>
##  1        4 nysy     13.6  1                  (14.2 428.5)  0.103
##  2        5 havi      8.8  1                   (9.4 436.4)  0.150
##  3       79 tiam     47.7  1                    (40 381.1) -0.161
##  4       80 caca      5.15 1                  (38.7 421.7)  0.253
##  5       96 libe      2.3  1                      (60 310)  0.262
##  6      101 litu     65.4  3                  (47.1 567.3)  0.552
##  7      102 astr      1.99 3                  (40.8 575.5)  0.0954
##  8      126 cato     37.4  1                  (60.6 400.2)  0.165
##  9      127 caca      8.72 3                  (72.7 514.1)  0.0370
## 10      139 astr      1.71 1                  (96.7 315.1)  0.0549
## # ... with 6,286 more rows, and 2 more variables: comp <list>,
## #   growth_hat <dbl>
```

We then compare the observed and fitted/predicted growths to compute the root mean squared error (RMSE) of our model fit.

```
model_rmse <- focal_vs_comp_scbi %>%
  rmse(truth = growth, estimate = growth_hat) %>%
  pull(.estimate)
model_rmse
## [1] 0.1281398
```

Lastly, the generic for `ggplot2::autoplot()` allows us to plot the posterior distribution of all parameters in Figure 4 (for compactness we only show posteriors for 3 species).

```
# Plot posteriors for only a subset of species
sp_to_plot <- c("litu", "quru", "cagl")

plot1 <- autoplot(comp_bayes_lm_scbi, type = "intercepts", sp_to_plot = sp_to_plot)
plot2 <- autoplot(comp_bayes_lm_scbi, type = "dbh_slopes", sp_to_plot = sp_to_plot)
plot3 <- autoplot(comp_bayes_lm_scbi, type = "competition", sp_to_plot = sp_to_plot)

# Combine plots using patchwork
(plot1 | plot2) / plot3
```

TODO: Discuss meaning of results.

2.5 Evaluate the effect of competitor species identity using permutation tests

In order to evaluate the effect of competitor species identity, we use the four steps of our analysis sequence answer along with a permutation test: Under a null hypothesis where competitor species identity does not matter, we can permute/shuffle this variable within each focal tree, compute the RMSE (the test statistic of interest), repeat this process several times to construct a null distribution of the RMSE, and compare it to the observed RMSE to assess significance. Going back to our example in Section 2.3 of focal tree with `focal_ID` 4 and its 20 competitors, the permutation test randomly resamples the `comp_sp` variable with replacement, leaving all other variables intact. The resampling with replacement is nested

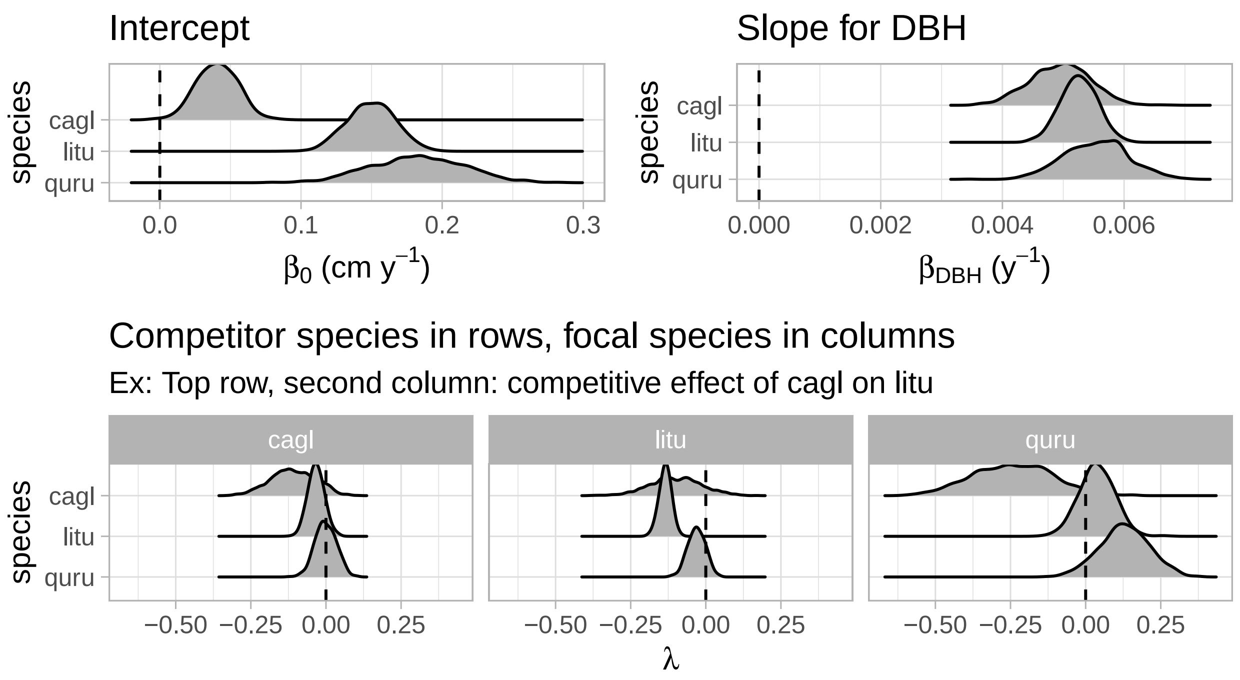

Figure 4: Fit model: Posterior distributions of all parameters for three species.

within each focal tree in order to preserve the neighborhood structure of our competition model. To run the permutation test, we use the same`comp_bayes_lm()` function as in Section 2.4, but with a `run_shuffle = TRUE` argument.

```
comp_bayes_lm_scbi_shuffle <- focal_vs_comp_scbi %>%
  comp_bayes_lm(prior_param = NULL, run_shuffle = TRUE)

focal_vs_comp_scbi <- focal_vs_comp_scbi %>%
  mutate(growth_hat_shuffle = predict(comp_bayes_lm_scbi_shuffle, newdata = focal_vs_cor

model_rmse_shuffle <- focal_vs_comp_scbi %>%
  rmse(truth = growth, estimate = growth_hat_shuffle) %>%
  pull(.estimate)
model_rmse_shuffle
## [1] 0.131083
```

The resulting RMSE of 0.131083 based on the permutation test is larger than the earlier

RMSE of 0.1281398, suggesting that models that do incorporate competitor species identity better fit the data. We conduct a fuller simulation in Section below.

2.6 Evaluate model performance using spatial cross-validation

We answer the second of our two questions: how can we obtain an accurate estimate of model performance/error? The model fits and predictions in Section 2.4 all suffer from a common failing: they use the same data to both fit the model and to assess the model's performance using the RMSE. As argued by Roberts et al. (2017), this can lead to overly optimistic assessments of model quality as the models can be overfit, in particular in situations where spatial-autocorrelation is present. To mitigate the effects of such overfitting, we use a spatially block cross-validation algorithm.

To this end, we use the `foldID` variable defined in Section 2.2 whereby all focal trees are assigned to one of 4 spatially contiguous blocks that act as folds in our cross-validation routine. Figure 5 presents a schematic illustrating this scheme for fold 1 (bottom-left) as the test set and folds 2, 3, and 4 as the training sets. We fit the model to all focal trees in the training set, apply the model to all focal trees in the test set to compute fitted/predicted values, and compute the RMSE of the observed versus predicted growths. We repeat this procedure 3 more times with each of the three remaining folds acting as the test set and then average all four resulting RMSE's. Furthermore, in order to maintain spatial independence between the test and training set, a buffer that extend outwards from the boundary of the test set is computed; all trees falling within this buffer are excluded from the training set.

This algorithm is implemented in the `run_cv()` function, which is a wrapper function to the `comp_bayes_lm()` function that fits the model and the `predict()` generic that returns fitted/predicted values. We compare these values to the observed growth values to again compute our RMSE.

```
focal_vs_comp_scbi <- focal_vs_comp_scbi %>%
  run_cv(comp_dist = comp_dist, cv_grid = blocks_scbi)
```

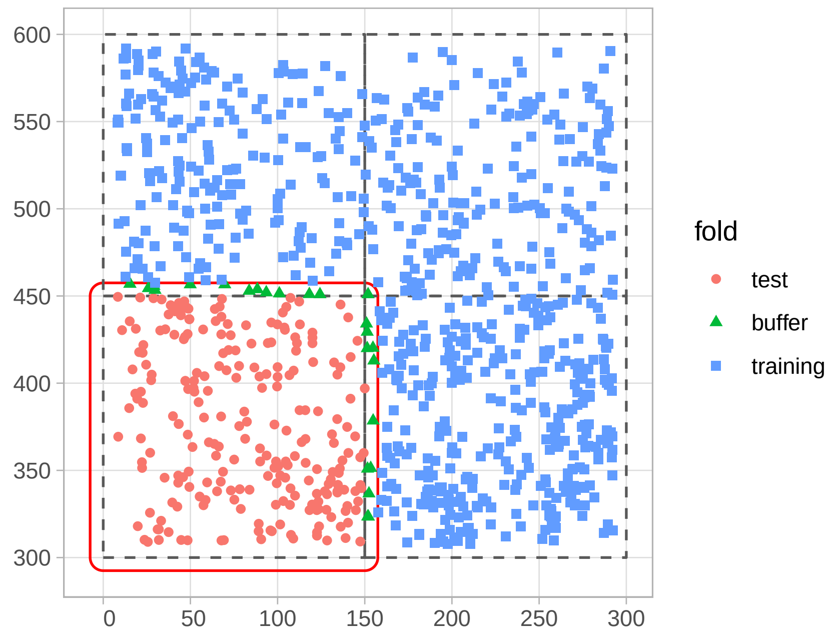

Figure 5: Schematic of spatial cross-validation: Using the k = 1 fold as the test set, assigning each focal tree to training set, test set, and buffer.

```
model_rmse_cv <- focal_vs_comp_scbi %>%
  rmse(truth = growth, estimate = growth_hat) %>%
  pull(.estimate)
model_rmse_cv
## [1] 0.1402209
```

The resulting RMSE of 0.1402209 computed using cross-validation is larger than the earlier RMSE of 0.1281398, suggesting that models that do not take the inherent spatial autocorrelation of the data into account generate error estimates that are overly optimistic; in our case RMSE's that are too low.

3 Discussion

- Run full simulation on SCBI data
- run time considerations

4 Acknowledgments

Sophie Li for her feedback on package interface.

References

Allen, D., Dick, C., Burnham, R. J., Perfecto, I. & Vandermeer, J. (2020), 'The michigan big woods research plot at the edwin s. george, pinckney, mi, usa', *Miscellaneous Publications of the Museum of Zoology, University of Michigan* **207**.
URL: *http://hdl.handle.net/2027.42/156251*

Allen, D. & Kim, A. Y. (2020), 'A permutation test and spatial cross-validation approach to assess models of interspecific competition between trees', *PLOS ONE* **15**(3), e0229930. Publisher: Public Library of Science.
URL: *https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0229930*

Anderson-Teixeira, K. J., Davies, S. J., Bennett, A. C., Gonzalez-Akre, E. B., Muller-Landau, H. C., Wright, S. J., Salim, K. A., Zambrano, A. M. A., Alonso, A., Baltzer, J. L., Basset, Y., Bourg, N. A., Broadbent, E. N., Brockelman, W. Y., Bunyavejchewin, S., Burslem, D. F. R. P., Butt, N., Cao, M., Cardenas, D., Chuyong, G. B., Clay, K., Cordell, S., Dattaraja, H. S., Deng, X., Detto, M., Du, X., Duque, A., Erikson, D. L., Ewango, C. E. N., Fischer, G. A., Fletcher, C., Foster, R. B., Giardina, C. P., Gilbert, G. S., Gunatilleke, N., Gunatilleke, S., Hao, Z., Hargrove, W. W., Hart, T. B., Hau, B. C. H., He, F., Hoffman, F. M., Howe, R. W., Hubbell, S. P., Inman-Narahari, F. M., Jansen, P. A., Jiang, M., Johnson, D. J., Kanzaki, M., Kassim, A. R., Kenfack, D., Kibet, S., Kinnaird, M. F., Korte, L., Kral, K., Kumar, J., Larson, A. J., Li, Y., Li, X., Liu, S., Lum, S. K. Y., Lutz, J. A., Ma, K., Maddalena, D. M., Makana, J.-R., Malhi, Y., Marthews, T., Serudin, R. M., McMahon, S. M., McShea, W. J., Memiaghe, H. R., Mi, X., Mizuno, T., Morecroft, M., Myers, J. A., Novotny, V., Oliveira, A. A. d., Ong, P. S., Orwig, D. A., Ostertag, R., Ouden, J. d., Parker, G. G., Phillips, R. P., Sack, L., Sainge, M. N., Sang, W., Sri-ngernyuang, K., Sukumar, R., Sun, I.-F., Sungpalee, W., Suresh, H. S., Tan, S., Thomas, S. C., Thomas, D. W., Thompson, J., Turner, B. L.,

Uriarte, M., Valencia, R., Vallejo, M. I., Vicentini, A., Vrška, T., Wang, X., Wang, X., Weiblen, G., Wolf, A., Xu, H., Yap, S. & Zimmerman, J. (2015), 'CTFS-ForestGEO: a worldwide network monitoring forests in an era of global change', *Global Change Biology* **21**(2), 528–549.
URL: *http://onlinelibrary.wiley.com/doi/abs/10.1111/gcb.12712*

Bourg, N. A., McShea, W. J., Thompson, J. R., McGarvey, J. C. & Shen, X. (2013), 'Initial census, woody seedling, seed rain, and stand structure data for the SCBI SIGEO Large Forest Dynamics Plot', *Ecology* **94**(9), 2111–2112.
URL: *http://esajournals.onlinelibrary.wiley.com/doi/abs/10.1890/13-0010.1*

Canham, C. D., LePage, P. T. & Coates, K. D. (2004), 'A neighborhood analysis of canopy tree competition: effects of shading versus crowding', *Canadian Journal of Forest Research* **34**(4). Publisher: NRC Research Press Ottawa, Canada.
URL: *https://cdnsciencepub.com/doi/abs/10.1139/x03-232*

Canham, C. D., Papaik, M. J., Uriarte, M., McWilliams, W. H., Jenkins, J. C. & Twery, M. J. (2006), 'Neighborhood Analyses Of Canopy Tree Competition Along Environmental Gradients In New England Forests', *Ecological Applications* **16**(2), 540–554. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1890/1051-0761%282006%29016%5B0540%3ANAOCTC%5D2.0.CO%3B2.

Pebesma, E. (2018), 'Simple Features for R: Standardized Support for Spatial Vector Data', *The R Journal* **10**(1), 439–446.
URL: *https://journal.r-project.org/archive/2018/RJ-2018-009/index.html*

Pebesma, E. J. & Bivand, R. S. (2005), 'Classes and methods for spatial data in R', *R News* **5**(2), 9–13.
URL: *https://CRAN.R-project.org/doc/Rnews/*

Pohjankukka, J., Pahikkala, T., Nevalainen, P. & Heikkonen, J. (2017), 'Estimating the prediction performance of spatial models via spatial k-fold cross validation', *International Journal of Geographical Information Science* **31**(10), 2001–2019.

Roberts, D. R., Bahn, V., Ciuti, S., Boyce, M. S., Elith, J., Guillera-Arroita, G., Hauenstein, S., Lahoz-Monfort, J. J., Schröder, B., Thuiller, W., Warton, D. I., Wintle, B. A., Hartig, F. & Dormann, C. F. (2017), 'Cross-validation strategies for data with temporal, spatial, hierarchical, or phylogenetic structure', *Ecography* **40**(8), 913–929.
URL: *http://onlinelibrary.wiley.com/doi/abs/10.1111/ecog.02881*

Smith, W. B. (2002), 'Forest inventory and analysis: a national inventory and monitoring program', *Environmental pollution* **116**, S233–S242.

Uriarte, M., Condit, R., Canham, C. D. & Hubbell, S. P. (2004), 'A spatially explicit model of sapling growth in a tropical forest: does the identity of neighbours matter?', *Journal of Ecology* **92**(2), 348–360. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.0022-0477.2004.00867.x.
URL: *http://besjournals.onlinelibrary.wiley.com/doi/abs/10.1111/j.0022-0477.2004.00867.x*

Valavi, R., Elith, J., Lahoz-Monfort, J. J. & Guillera-Arroita, G. (2019), 'blockCV: An r package for generating spatially or environmentally separated folds for k-fold cross-validation of species distribution models', *Methods in Ecology and Evolution* **10**(2), 225–232. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/2041-210X.13107.
URL: *http://besjournals.onlinelibrary.wiley.com/doi/abs/10.1111/2041-210X.13107*

Waller, L. A. & Gotway, C. A. (2004), *Applied Spatial Statistics for Public Health Data*, John Wiley & Sons, Incorporated, Hoboken, UNITED STATES.
URL: *http://ebookcentral.proquest.com/lib/smith/detail.action?docID=214360*

Wickham, H. (2020), *tidyr: Tidy Messy Data*. R package version 1.1.2.
URL: *https://CRAN.R-project.org/package=tidyr*

Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., François, R., Grolemund, G., Hayes, A., Henry, L., Hester, J., Kuhn, M., Pedersen, T. L., Miller, E., Bache, S. M., Müller, K., Ooms, J., Robinson, D., Seidel, D. P., Spinu, V., Takahashi, K., Vaughan, D., Wilke, C., Woo, K. & Yutani, H. (2019), 'Welcome to the Tidyverse',

Journal of Open Source Software **4**(43), 1686.
URL: *https://joss.theoj.org/papers/10.21105/joss.01686*