`forestecology` package for modeling interspecies competition between trees

Albert Y. Kim [*]
Program in Statistical & Data Sciences, Smith College
and
David Allen
Biology Department, Middlebury College

January 8, 2021

Abstract

Move abstract below here after completed.

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

[*]Albert Y. Kim is Assistant Professor, Statistical & Data Sciences, Smith College, Northampton, MA 01063 (e-mail: `akim04@smith.edu`).

1 Abstract (350 words)

1. set the context for and purpose of the work
2. indicate the approach and the methods
3. outline the main results
4. identify the conclusions and wider implications

2 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 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).

3 Example

We demonstrate the `forestecology` package's features on two data sets, both based on inventory censuses of two sites from the Smithsonian Institution's ForestGEO international network of 72 long-term forest dynamics research sites Anderson-Teixeira et al. (2015). First, the Michigan Big Woods Forest Dynamics Plot located at the Edwin S. George Reserve in Pinckney, MI, USA. The 23 ha plot is situated in mature oak-hickory forest. The canopy is dominated by white oak (Quercus alba), northern red oak (Quercus rubra), black oak (Quercus velutina), shagbark hickory (Carya ovata) and pignut hickory (Carya glabra). The most common understory tree is witch-hazel (Hamamelis virginiana) Allen et al. (2020). In the example below, we will preface any data frames from this plot in with bw_.

Second, 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). In the example below, we will preface any data frames from this plot in with scbi_.

We load all the necessary packages.

```r
library(forestecology)

# Load tidyverse packages:
library(tidyverse)
library(lubridate)

# Load spatial packages:
# devtools::install_github("rvalavi/blockCV")
library(blockCV)
library(sf)
library(sfheaders)

# Load other packages:
library(snakecase)
library(yardstick)
```

3.1 Preprocess census data

We start by preprocessing the census data for both sites. While ForestGEO data protocols ensure a high degree of standardization between site, minor variations still exist Anderson-Teixeira et al. (2015). While the Big Woods data comes pre-loaded in the `forestecology` package, we load the SCBI data as they are saved in .csv files in the SCBI-ForestGEO-Data repository on GitHub Gonzalez-Akre et al. (2020). In both cases, we load the census data as R as "tibble" data frames thereby ensuring a standardized input/output format that can be used across all `tidyverse` packages Wickham et al. (2019).

Furthermore, we ensure that the different variables have the correct names, types (`dbl`, `data`, `factor`).

3.1.1 Big Woods

We load census data from 2008 and 2014 saved in the package, then merge species data (genus, species, linnean classification, family, etc).

```r
data(bw_census_2008, bw_census_2014, bw_species)

# Append additional species data
bw_census_2008 <- bw_census_2008 %>%
  left_join(bw_species, by = "sp") %>%
  select(-c(genus, species, latin))
```

3.1.2 SCBI

We load census data from 2008 and 2014 from `.csv` files saved from GitHub on November 20, 2020. Furthermore, we perform two additional pre-processing steps. First, 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. Second, in order to standardize comparisons between Big Woods and SCBI, we convert the units of dbh from mm to cm. [1]

```
scbi_2013 <- read_csv("scbi.stem2.csv") %>%
  select(treeID, stemID, sp, ExactDate, gx, gy, dbh, codes, status) %>%
  mutate(
    date = ExactDate,
    dbh = as.numeric(dbh),
    date = mdy(date)
  ) %>%
  filter(gx < 300, between(gy, 300, 600)) %>%
  mutate(dbh = dbh / 10)

scbi_2018 <- read_csv("scbi.stem3.csv") %>%
  select(treeID, stemID, sp, ExactDate, gx, gy, dbh, codes, status) %>%
  mutate(
    date = ExactDate,
    dbh = as.numeric(dbh),
    date = mdy(date)
  ) %>%
  filter(gx < 300, between(gy, 300, 600)) %>%
  mutate(dbh = dbh / 10)
```

3.2 Compute annual growth

For each plot we then compute average annual growth between the two censuses using the `compute_growth()` function. This function takes the two census data frames as well as a character indicating which variable in both data frames uniquely identifies each stem. This function returns a single data frame that includes a numerical variable `growth` reflecting the average annual dbh growth (in cm) of all trees alive at both time points. Furthermore,

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

variables that (in theory) remain unchanged between censuses appear only once, such as location variables `gx` 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`. Here the resulting data frames are named with some variation of `growth_df`.

After computing the average annual growth for each tree, we ensure to convert all variables denote species from type character to factors; this is to ensure that issues of rare species being accounted for in both training and test sets in our upcoming cross-validation step (see Section REF)

3.2.1 Big Woods

In the case of Big Woods data, we first remove all trees that were re-sprouts in the later (2014) census. Additionally, we have included three classification of tree species: `species`, `family`, and `trait_group`. DESCRIBE THESE

```r
bw_census_2014 <- bw_census_2014 %>%
  filter(!str_detect(codes, "R"))

bw_growth_df <-
  compute_growth(bw_census_2008, bw_census_2014, id = "treeID") %>%
  # Convert all variables denoting species to factors
  mutate(
    sp = sp %>% to_any_case() %>% as.factor(),
    species = sp,
    family = as.factor(family),
    trait_group = as.factor(trait_group)
  ) %>%
  # Drop unnecessary variables
  select(-stemID)
```

3.2.2 SCBI

```r
scbi_growth_df <-
  compute_growth(scbi_2013, scbi_2018, "stemID") %>%
  # Convert all variables denoting species to factors
  mutate(sp = as.factor(sp))
```

3.2.3 Comparison

Figure 1 displays histograms comparing the distribution of average annual growth at both sites. Observe that average annual growth appears higher at the Big Woods site.

```r
# Both Big Woods & SCBI
bind_rows(
  bw_growth_df %>% st_drop_geometry() %>% select(growth) %>% mutate(Site = "Big Woods")
  scbi_growth_df %>% st_drop_geometry() %>% select(growth) %>% mutate(Site = "SCBI")
) %>%
  ggplot(aes(x = growth, y = ..density.., fill = Site)) +
  geom_histogram(alpha = 0.5, position = "identity", binwidth = 0.05) +
  labs(x = "Average annual growth in dbh (cm per yr)") +
  coord_cartesian(xlim = c(-0.5, 1))
```

3.3 Add spatial information

We now encode spatial information to the `growth_df` data frames. First, in order to control for study region edge effects, we add “buffers” to the periphery of the study region (cite Waller?). Our model of interspecific competition relies on a spatial definition of who the competitor trees are for focal trees of interest. Since certain explanatory variables such as biomass are cumulative, we must ensure that all trees being modeled are not biased to have different neighbor structures. This is a particular 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.

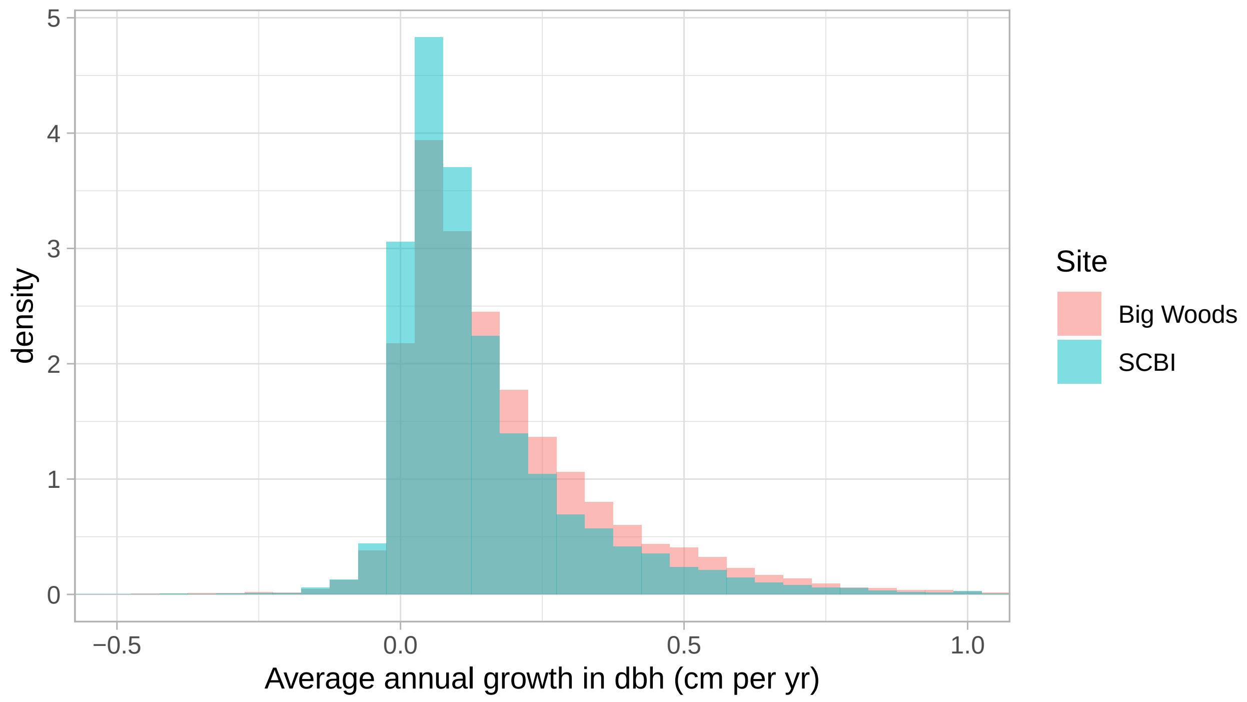


Figure 1: Distribution of average annual growth in DBH for both sites.

Second, our ultimate method for model assessment will rely on estimates of model error as generated by cross-validation. Conventional cross-validation schemes assign observations to folds by resampling individual observations at random. However, underlying this scheme is an assumption that the observations are independent. In the case of forest census data, observations exhibit spatial autocorrelation, and thus this dependence must be incorporated in our resampling scheme in spatial cross-validation **?** **?**. We will therefore associate portions of the study region to spatial folds.

To these two ends, we define two constants, both of which are in the same units as the `gx` and `gy` variables (most often meters).

```
max_dist <- 7.5
cv_fold_size <- 100
```

The first constant is `max_dist` which defines the maximum distance for a tree's competitive neighborhood. Trees within this distance of each other are assumed to compete while those farther than this distance apart do not. Put differently, all trees within `max_dist` of a focal tree will be considered its competitors (see below). Other studies have estimated the value of `max_dist`; we use an average of estimated values Canham et al. (2004), Uriarte et al. (2004), Tatsumi et al. (2013), Canham et al. (2006).

Furthermore, `max_dist` will define the size of all buffers considered, which will be encoded as a binary variable `buffer` as computed by the `add_buffer_variable()` function. This function takes as input the main `growth_df` data frame, the `size` of the buffer which we set as `max_dist`, and the boundary of the study region encoded as a simple features polygon Pebesma (2018). DESCRIBE SF PACKAGE. In the Big Woods example below we will use a pre-loaded simple features polygon while for the SCBI example we present example code on how to manually construct one.

The second constant is `cv_fold_size` which defines the length and width of the spatial folds (note that for now the spatial folds are restricted be squares). We will then use this constant to associate each observed tree to one of k folds in the respective study region. In the Big Woods example below we will use the `blockCV` R package that has implemented spatial cross-validation while for the SCBI we will do this manually **?**.

3.3.1 Big Woods

First, we indicate which trees are part of the buffer. This necessitates information about the study region boundary. In this case, we use a `sf_polygon` object `bw_study_region` which comes pre-loaded in the `forestecology` packages. After loading `bw_study_region`, we illustrate the results of the `add_buffer_variable()` function in Figure 2. Trees on the periphery denote with lighter colors are part of the buffer and will not be considered as "focal" trees of interest going forward; they will only be considered as competitor trees.

```
data(bw_study_region)

bw_growth_df <- bw_growth_df %>%
  add_buffer_variable(direction = "in", size = max_dist, region = bw_study_region)

ggplot() +
  geom_sf(data = bw_growth_df %>% sample_frac(0.2), aes(col = buffer), size = 0.5)
```

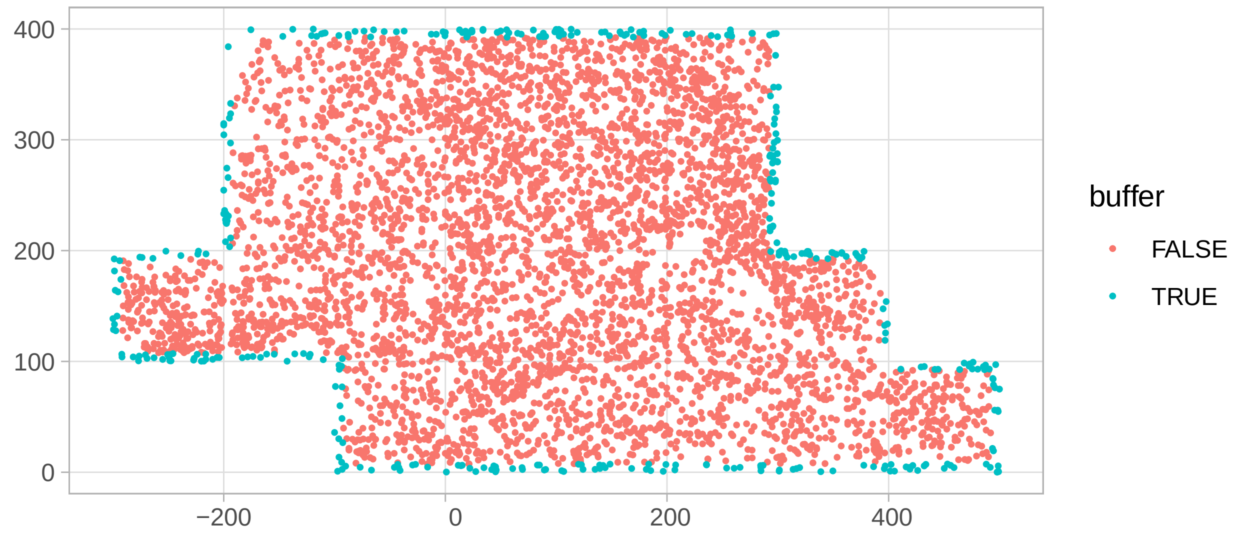


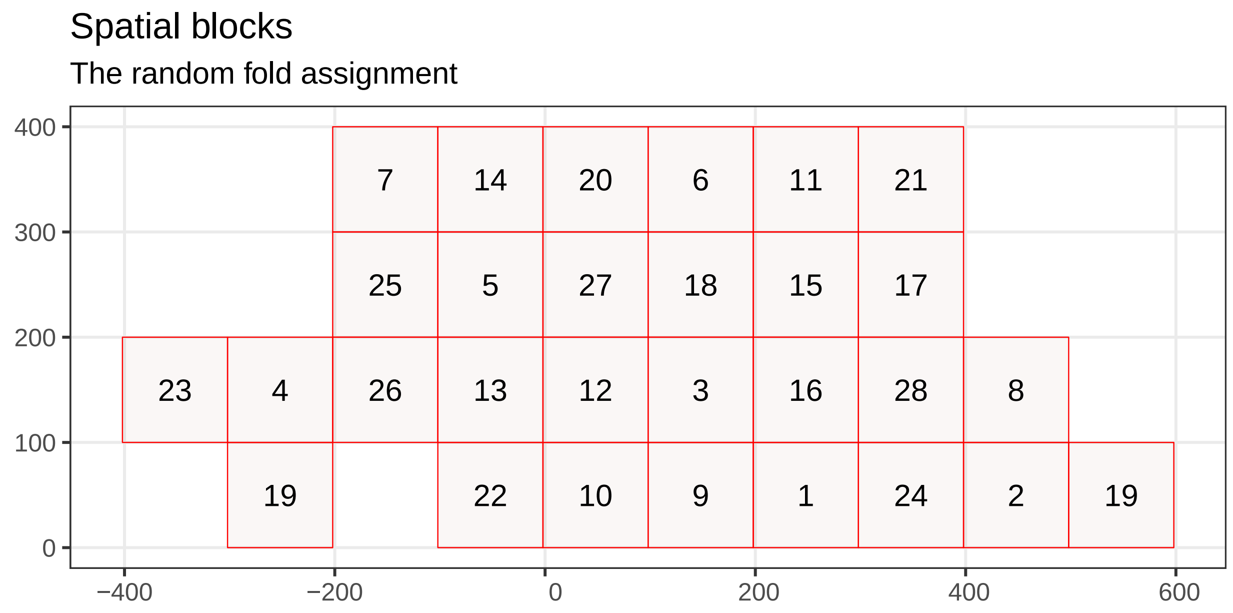
Figure 2: Buffer region for Big Woods study region.

Second, we associate each tree to spatial cross validation folds. In this case, we use the `spatialBlock()` function from the `blockCV` package to define the spatial grid which

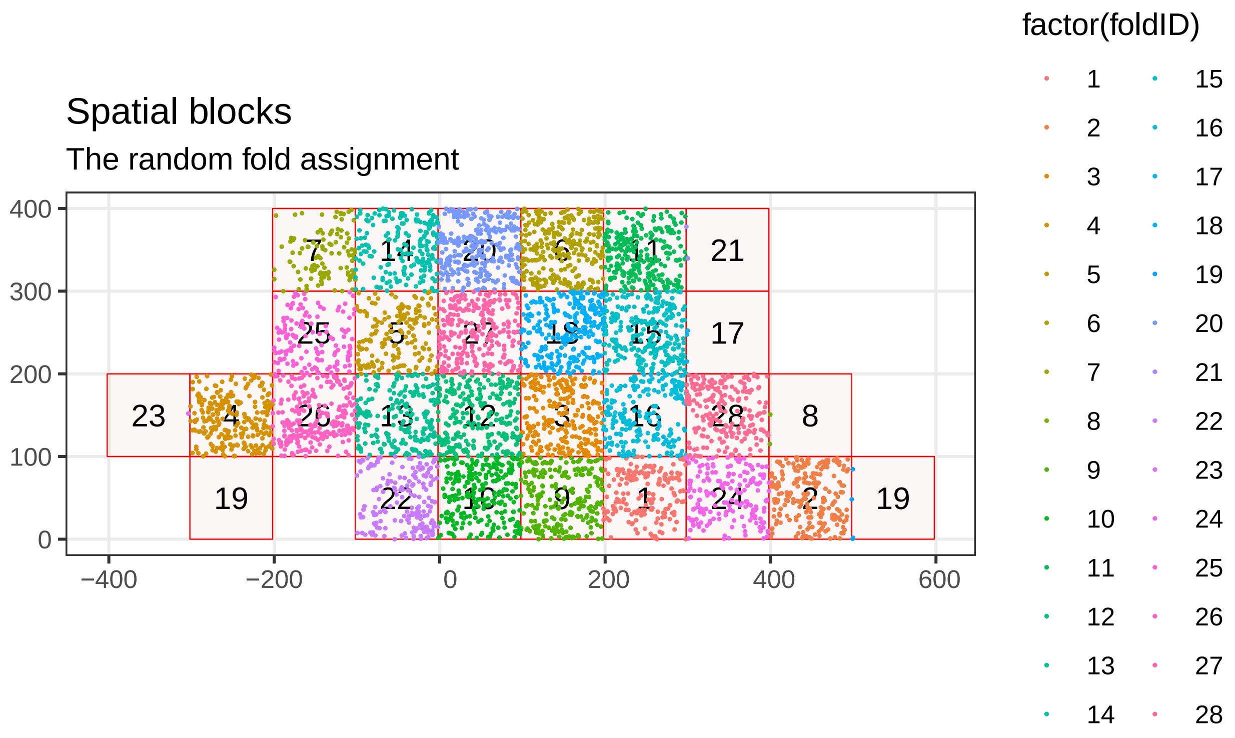
THIS IS A MESS. We use the Valavi et al. (2019), whose elements will act as the folds in our leave-one-out (by "one" we mean "one grid block") cross-validation scheme. The upshot here is we add `foldID` to `growth_df` which identifies which fold each individual is in, and the creation of a `cv_grid_sf` object which gives the geometry of the cross validation grid.

```r
set.seed(76)
bw_spatialBlock <- spatialBlock(
  speciesData = bw_growth_df, theRange = cv_fold_size,
  k = 28, xOffset = 0.5, yOffset = 0,
  verbose = FALSE
)

# Add foldID to each tree
bw_growth_df <- bw_growth_df %>%
  mutate(foldID = bw_spatialBlock$foldID)
```

```
# Visualize grid. Why does fold 19 repeat?
bw_spatialBlock$plots +
  geom_sf(data = bw_growth_df %>% sample_frac(0.2), aes(col = factor(foldID)), size = 0

# Remove empty folds
bw_growth_df <- bw_growth_df %>%
  filter(!foldID %in% c(19, 23, 21, 17, 8, 19)) %>%
  mutate(foldID = as.character(foldID))
```



133



Separately, we save the spatial cross-validation grid as an `sf_polygon` object `bw_cv_grid`

```
bw_cv_grid <- bw_spatialBlock$blocks %>%
  st_as_sf()
```

3.3.2 SCBI

First, we indicate which trees are part of the buffer. In this case however we manually define the study region boundary based on the subregion we defined in Section 3.1.2 and create an `sf_polygon` object using the `sf_polygon()` function from the `sfheaders` package. Figure 3 displays the resulting buffer trees.

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

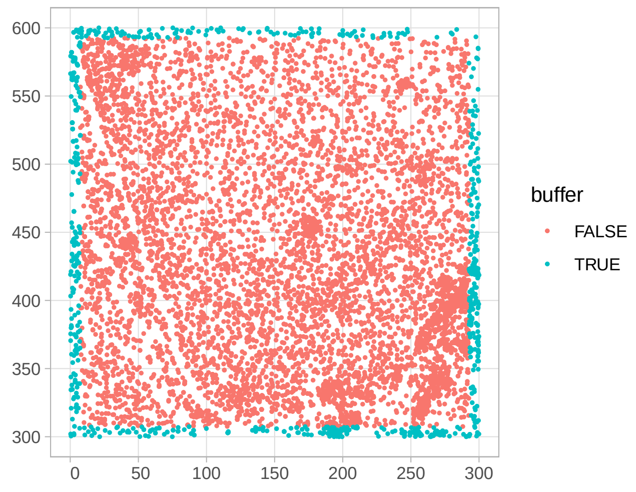


Figure 3: Buffer region for SCBI study region.

```r
scbi_growth_df <- scbi_growth_df %>%
  add_buffer_variable(direction = "in", size = max_dist, region = scbi_study_region)

ggplot() +
  geom_sf(data = scbi_growth_df, aes(col = buffer), size = 0.5)
```

Second, we associate each tree to spatial cross validation folds. In this case we manually define a spatial crossvaliation grid. Figure 4 displays the resulting cross-validation folds along with the buffer from Figure 3.

Here we manually define the spatial cross-validation grid as an `sf_polygon` object `scbi_cv_grid`

```r
fold1 <- rbind(c(0, 300), c(150, 300), c(150, 600), c(0, 600), c(0, 300)) %>%
  sf_polygon() %>%
  mutate(folds = 1)
fold2 <- rbind(c(150, 300), c(300, 300), c(300, 600), c(150, 600), c(150, 300)) %>%
  sf_polygon() %>%
  mutate(folds = 2)
scbi_cv_grid <- bind_rows(fold1, fold2)
```

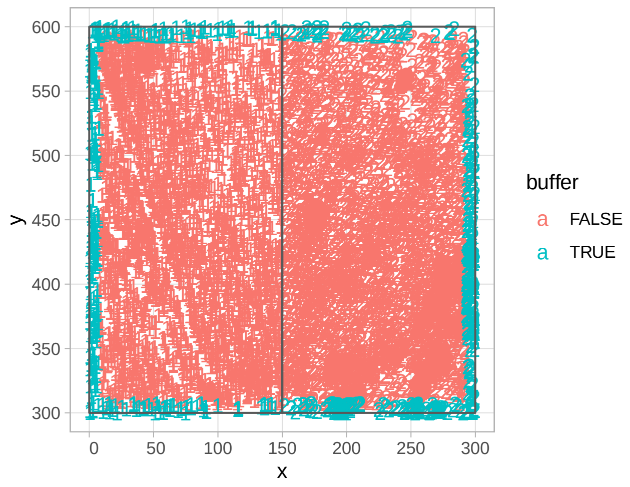


Figure 4: Buffer region for SCBI study region.

```r
scbi_spatialBlock <- spatialBlock(
  speciesData = scbi_growth_df,
  k = 2,
  verbose = FALSE,
  showBlocks = FALSE,
  # Note new arguments:
  selection = "systematic",
  blocks = scbi_cv_grid
)

# Add foldID to each tree
scbi_growth_df <- scbi_growth_df %>%
  mutate(foldID = scbi_spatialBlock$foldID)

ggplot() +
  geom_sf_text(data = scbi_growth_df, aes(label = foldID, col = buffer)) +
  geom_sf(data = scbi_cv_grid, fill = "transparent")
```

3.4 Define focal versus competitor trees

Next we define `focal_vs_comp` data frames which connects each focal tree in the `growth_df` data frames to the trees in its competitive neighborhood range as defined by the `max_dist` constant. So for example, if `growth_df` consisted of two focal trees with two and three neighbors with `max_dist` respectively, `focal_vs_comp` would be a data frame of 5 rows connecting each focal tree to it's competitors. The `create_focal_vs_comp()` function makes this connection taking as inputs the `growth_df` data frame; the `max_dist` constant defining competitive range; `cv_grid_sf`, giving the cross validation grid; and the `id` variable.

3.4.1 Big Woods

```
focal_vs_comp_bw <- bw_growth_df %>%
  create_focal_vs_comp(max_dist, cv_grid_sf = bw_cv_grid, id = "treeID")
```

TODO: Figure out how to show this data frame's contents.

```
head(focal_vs_comp_bw)
## # A tibble: 6 x 10
##   focal_ID focal_sp   dbh foldID                   geometry growth
##      <dbl> <fct>    <dbl> <chr>                     <POINT>  <dbl>
## 1        1 white_o~  41.2 12                  (8.7 107.5)  0.404
## 2        1 white_o~  41.2 12                  (8.7 107.5)  0.404
## 3        1 white_o~  41.2 12                  (8.7 107.5)  0.404
## 4        1 white_o~  41.2 12                  (8.7 107.5)  0.404
## 5        1 white_o~  41.2 12                  (8.7 107.5)  0.404
## 6        1 white_o~  41.2 12                  (8.7 107.5)  0.404
## # ... with 4 more variables: comp_ID <dbl>, dist <dbl>,
## #   comp_sp <fct>, comp_basal_area <dbl>
```

3.4.2 SCBI

```
focal_vs_comp_scbi <- scbi_growth_df %>%
  create_focal_vs_comp(max_dist, cv_grid_sf = scbi_cv_grid, id = "stemID")
```

TODO: Figure out how to show this data frame's contents.

```
head(focal_vs_comp_scbi)
## # A tibble: 6 x 10
##   focal_ID focal_sp   dbh foldID               geometry growth
##      <dbl> <fct>    <dbl>  <int>                <POINT>  <dbl>
## 1        4 nysy      13.6      1          (14.2 428.5)  0.103
## 2        4 nysy      13.6      1          (14.2 428.5)  0.103
## 3        4 nysy      13.6      1          (14.2 428.5)  0.103
## 4        4 nysy      13.6      1          (14.2 428.5)  0.103
## 5        4 nysy      13.6      1          (14.2 428.5)  0.103
## 6        4 nysy      13.6      1          (14.2 428.5)  0.103
## # ... with 4 more variables: comp_ID <dbl>, dist <dbl>,
## #   comp_sp <fct>, comp_basal_area <dbl>
```

3.5 Fit model and make predictions

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 $\vec{x}_{ij}$

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

We estimate the model's parameters using Bayesian linear regression implemented in the `fit_bayesian_model()` function. TODO: define all parameters

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.

Using these posterior means, we then use the posterior predictive distribution to obtain fitted/predicted values $\widehat{y}$ of the dbh for each focal tree using the `predict_bayesian_model()`. These $\widehat{y}$ can then be compared to the observed y dbh's to compute the root mean-square error, a measure of a model's predictive error which has the same units as the observed data y.

3.5.1 Big Woods

For the Michigan Big Woods data we present two use cases of the model fitting and prediction scheme. The first use case is the simplest where we assess the fit of the model using root mean squared error. The second use case then answers the question of whether species competitor identity matters using permutation test.

For the first use case, we fit the linear model specified in Equation XXX to our data frame of type `focal_vs_comp`. This input/outputs of the `fit_bayesian_model()` function are lists of the prior/posterior means of parameters of the linear regression specified in XXX. Generally speaking, there are two classes of regression parameters: β main effects and λ competitive effects. In the upcoming Section 3.7, we will present code visualizing this posterior distributions.

```
posterior_param_bw <- focal_vs_comp_bw %>%
  fit_bayesian_model(prior_param = NULL)
```

This output of posterior parameters for the specified competition model are then used along with the posterior predictive distribution encoded in `predict_bayesian_model()` to return predicted growths for each individual tree. We join these predicted growths to the original growth data frame.

```r
predictions <- focal_vs_comp_bw %>%
  predict_bayesian_model(posterior_param = posterior_param_bw) %>%
  right_join(bw_growth_df, by = c("focal_ID" = "treeID"))
```

We then use the `rmse()` function from the `yardstick` package to obtain the root mean squared error of the observed versus fitted values of growth.

```r
predictions %>%
  yardstick::rmse(truth = growth, estimate = growth_hat) %>%
  pull(.estimate)
## [1] 0.148145
```

The second use case is near identical to the first, but with a small change in the code to test whether the identity of the competitor matters. By adding a `run_shuffle = TRUE` argument to `fit_bayesian_model()`, for each focal tree its competitor trees' species identity will be "shuffled" randomly much like in a permutation test. By shuffling these species labels we are effectively fitting the model under a null model that competitor species identity does not matter. If the "shuffled" RMSE's are consistently lower than the unshuffled RMSE corresponding to the observed data, then we have evidence to suggest that competitor identity matters to competitive interactions.

```r
posterior_param_bw_shuffle <- focal_vs_comp_bw %>%
  fit_bayesian_model(prior_param = NULL, run_shuffle = TRUE)
```

```r
predictions_shuffle <- focal_vs_comp_bw %>%
  predict_bayesian_model(posterior_param = posterior_param_bw_shuffle) %>%
  right_join(bw_growth_df, by = c("focal_ID" = "treeID"))
```

```r
predictions_shuffle %>%
  rmse(truth = growth, estimate = growth_hat) %>%
  pull(.estimate)
## [1] 0.1505383
```

The RMSE is fact lower for the non-shuffled version, indicative of a better model fit. This gives support for the idea that competitor identity does matter for competitive interactions. In Allen & Kim (2020) we run this shuffle a large number of times to construct a full permutation distribution to show that this difference is robust to resampling variation.

3.5.2 SCBI

In the case of the SCBI data, we once again perform the same model fitting and computing of fitted growths as with the Big Woods data, but this time we map the residuals of the observed minus fitted values to look for spatial patterns.

```r
posterior_param_scbi <- focal_vs_comp_scbi %>%
  fit_bayesian_model(prior_param = NULL, run_shuffle = FALSE)
```

```r
scbi_growth_df_noCV <- focal_vs_comp_scbi %>%
  predict_bayesian_model(posterior_param = posterior_param_scbi) %>%
  right_join(scbi_growth_df, by = c("focal_ID" = "stemID"))
```

```r
scbi_growth_df_noCV %>%
  rmse(truth = growth, estimate = growth_hat) %>%
  pull(.estimate)
## [1] 0.1280644
```

In Figures 5 and 6 we present the residuals.

```r
ggplot(scbi_growth_df_noCV, aes(x = growth, y = growth_hat)) +
  geom_point(size = 0.5, color = rgb(0, 0, 0, 0.25)) +
  stat_smooth(method = "lm") +
```

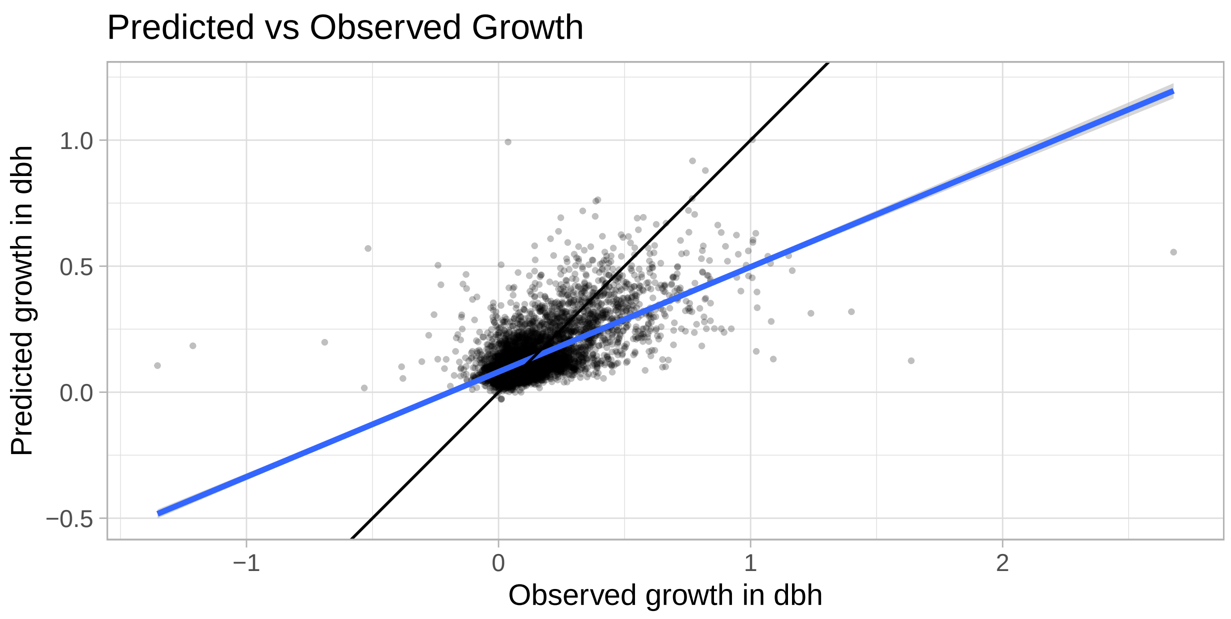


Figure 5: Spatial distribution of residuals for model applied to SCBI data.

```
  geom_abline(slope = 1, intercept = 0) +
  coord_fixed() +
  labs(
    x = "Observed growth in dbh", y = "Predicted growth in dbh",
    title = "Predicted vs Observed Growth"
  )
```

```
scbi_growth_df_noCV %>%
  st_as_sf() %>%
  # TODO: Need to investigate missingness
  filter(!is.na(growth_hat)) %>%
  mutate(
    error = growth - growth_hat,
    error_bin = cut_number(error, n = 5),
    error_compress = ifelse(error < -0.75, -0.75, ifelse(error > 0.75, 0.75, error))
  ) %>%
  ggplot() +
```

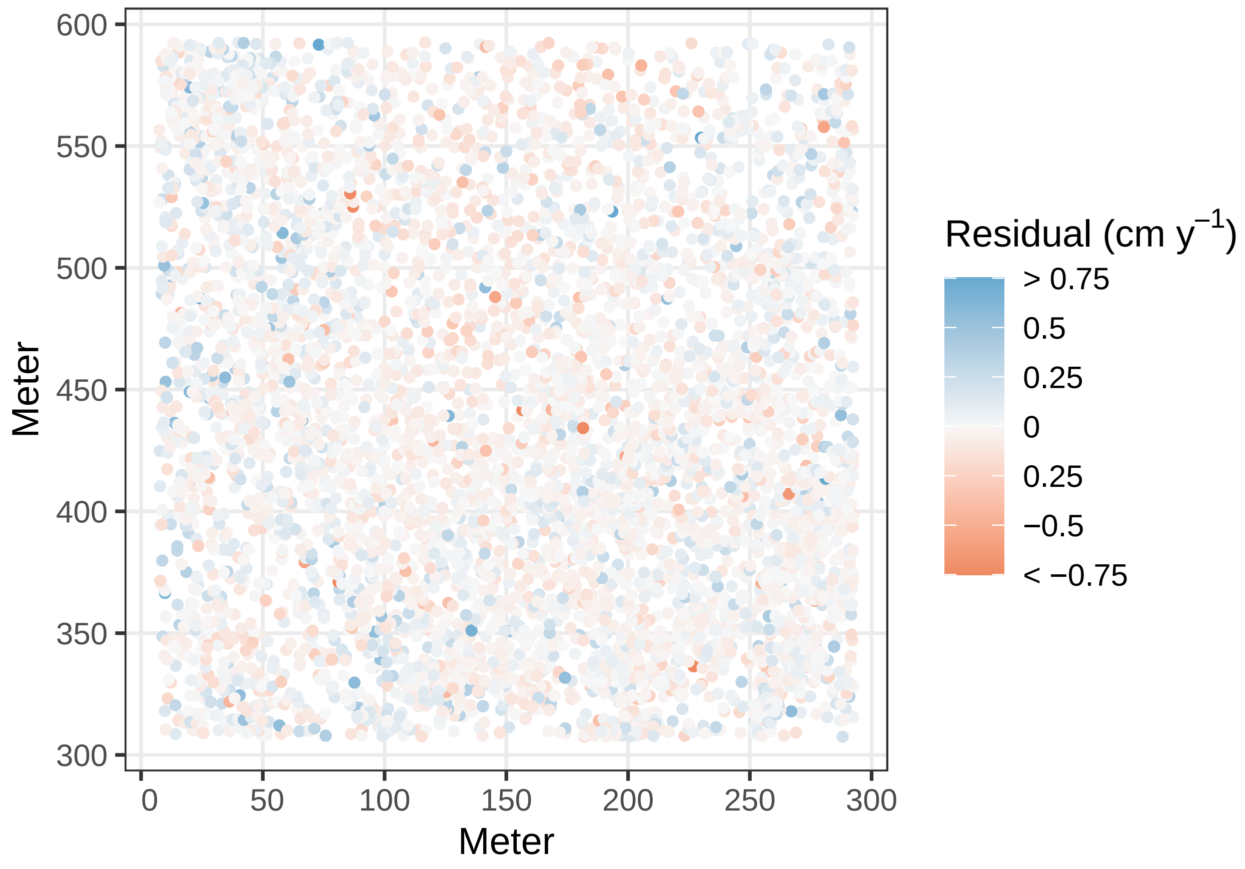


Figure 6: Spatial distribution of residuals for model applied to SCBI data part 2.

```
geom_sf(aes(col = error_compress), size = 1) +
theme_bw() +
scale_color_gradient2(
  low = "#ef8a62", mid = "#f7f7f7", high = "#67a9cf",
  name = expression(paste("Residual (cm ", y^{-1}, ")")),
  breaks = seq(from = -0.75, to = 0.75, by = 0.25),
  labels = c("< -0.75", "-0.5", "0.25", "0", "0.25", "0.5", "> 0.75")
) +
labs(x = "Meter", y = "Meter")
```

3.6 Run spatial cross-validation

The model fits and predictions in Section 3.5 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 implemented in the `run_cv()`. This function at its core uses the same model fitting implemented in the `fit_bayesian_model()` function, however trains the model on $k - 1$ spatial folds of the train and returns fitted values for the test data. Recall that the spatial blocking scheme wass encoded in Section 3.3.

3.6.1 Big Woods

Applying this spatially cross-validated model fit yields an RMSE is higher than that when the model is fit without cross validation. In other words, our model fits in 3.5 were overly optimistic in the model's fitting power, whereas a cross-validated results yield an estimate that is closer to the truth. See Allen & Kim (2020) for more discussion of this.

```
cv_bw <- focal_vs_comp_bw %>%
  run_cv(max_dist = max_dist, cv_grid = bw_cv_grid) %>%
  right_join(bw_growth_df, by = c("focal_ID" = "treeID"))

cv_bw %>%
  rmse(truth = growth, estimate = growth_hat) %>%
  pull(.estimate)
## [1] 0.1533511
```

3.6.2 SCBI

Observe once again that this RMSE is much higher than that for the above SCBI model fit without cross-validation.

```
cv_scbi <- focal_vs_comp_scbi %>%
  run_cv(max_dist = max_dist, cv_grid = scbi_cv_grid) %>%
  right_join(scbi_growth_df, by = c("focal_ID" = "treeID"))

cv_scbi %>%
  rmse(truth = growth, estimate = growth_hat) %>%
  pull(.estimate)
## [1] 0.1494775
```

3.7 Visualize posterior distributions

Lastly, we return to the model fits from Section 3.5 and present tools to visually explore the posterior distributions of all parameters in our model. There are two main groups of parameters to consider. The β coefficients tell us about how fast each species grows and how this depends on DBH while the full matrix of λ values describe the competitive effects between pairs of species. There is a rich literature on this matrix (cite).

DO WE NEED TO DESCRIBE MECHANICS? Because of the structure of the `bw_fit_model` object we cannot simply draw these curves based on the posterior distribution. `bw_fit_model()` gives the parameters *compared* to a baseline. This is not of direct interest. So to display these parameters, as we care about them, we have to sample from the baseline distribution and from the comparison one to get the posterior distribution of interest.

3.7.1 Big Woods

Here we re-run the model fit to the Big Woods data from Section 3.5, but this time use "family" as the group for comparison which has. This makes the posterior distributions easier to follow. Also, surprisingly, grouping by family performed just as well as grouping by species Allen & Kim (2020). First we re-run `create_focal_vs_comp()` and `fit_bayesian_model()` with no permutation shuffling with the grouping variable as family.

```r
focal_vs_comp_bw <- bw_growth_df %>%
  mutate(sp = family) %>%
  create_focal_vs_comp(max_dist = max_dist, cv_grid_sf = bw_cv_grid, id = "treeID")

posterior_param_bw <- focal_vs_comp_bw %>%
  fit_bayesian_model(prior_param = NULL, run_shuffle = FALSE)
```

Now the posterior parameter outputs of `fit_bayesian_model()` are passed to `plot_posterior_paramet` to generate visualizations of the posterior parameters. These visualizations are displayed in Figure 5 of Allen & Kim (2020). For simplicity we only plot a subset of the species families.

```r
posterior_plots <- plot_posterior_parameters(
  posterior_param = posterior_param_bw,
  sp_to_plot = c("cornaceae", "fagaceae", "hamamelidaceae", "juglandaceae",
                 "lauraceae", "rosaceae", "sapindaceae", "ulmaceae")
)
```

The output is a list with three plots stored. Figure 7 The element `beta_0` gives the baseline growth intercept β_0, i.e., how fast an individual of each group grows independent of DBH).

```r
posterior_plots[["beta_0"]]
```

Figure 8 Next `beta_dbh` gives the slope for DBH slope $\beta_{dbh,i}$ for each group.

```r
posterior_plots[["beta_dbh"]]
```

Finally Figure 9 `lambda` gives the competition coefficients λ.

```r
posterior_plots[["lambda"]]
```

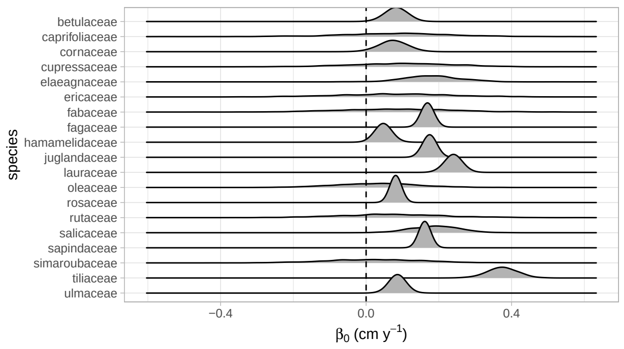


Figure 7: Posterior distribution of beta0.

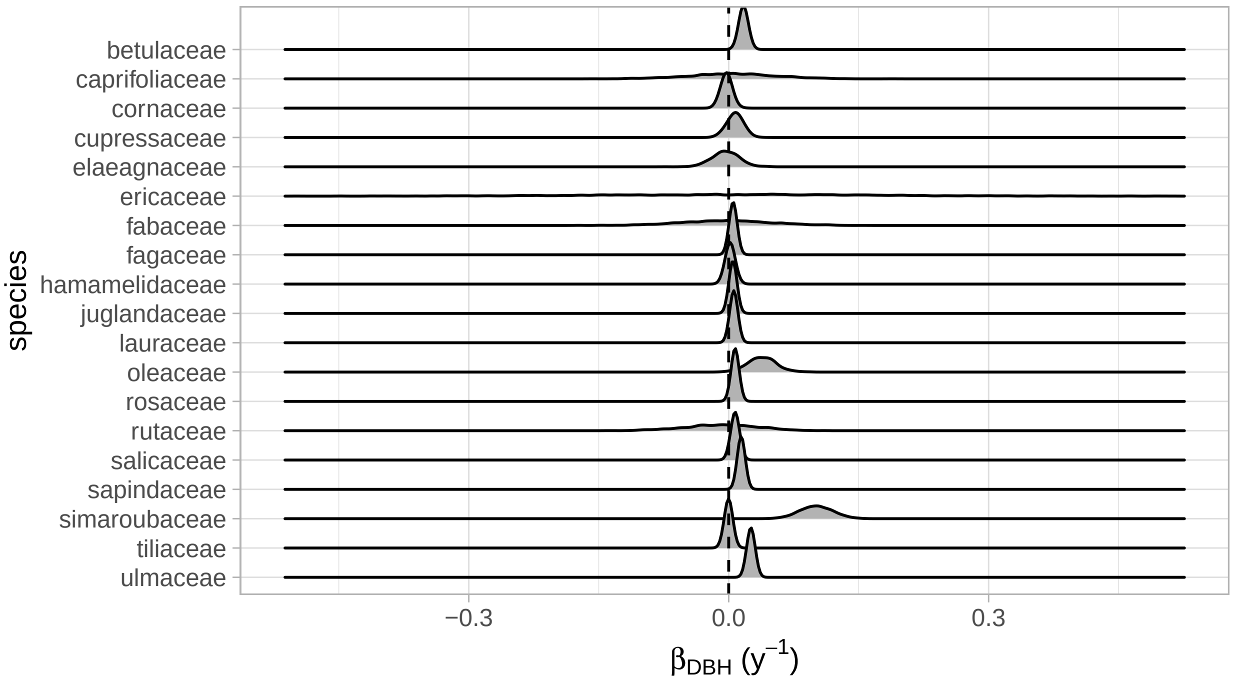


Figure 8: Posterior distribution of betadbh.

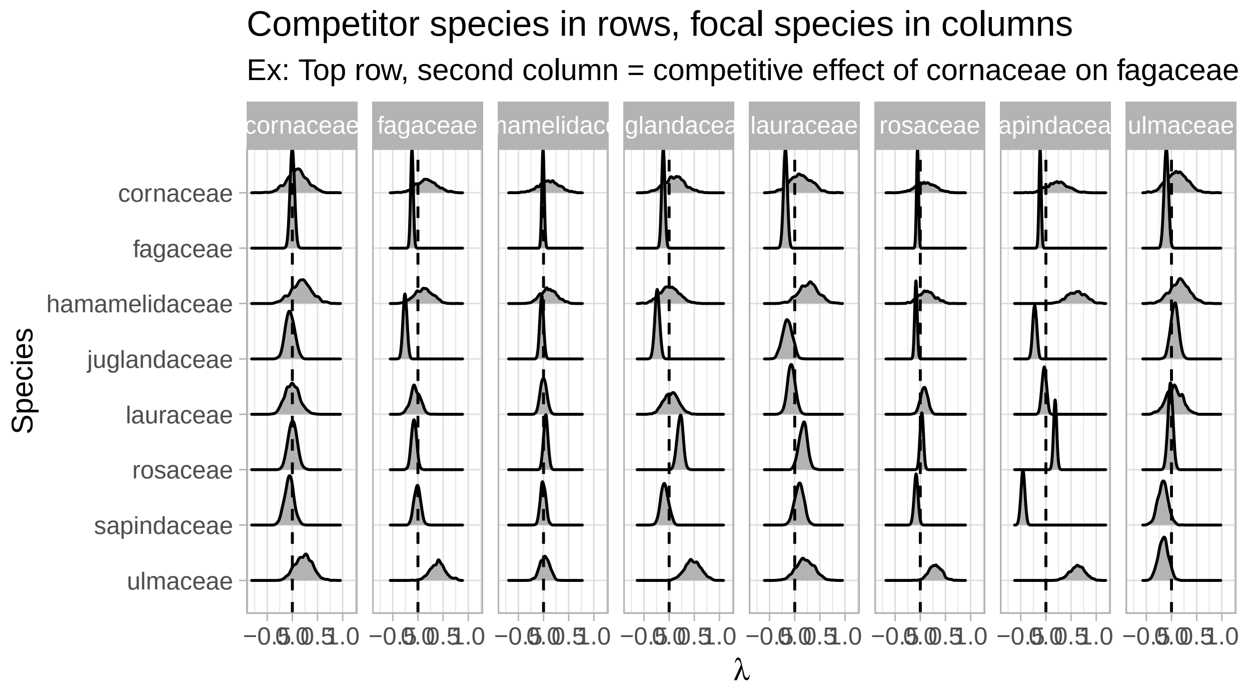


Figure 9: Posterior distribution of lambda.

3.7.2 SCBI

We revisit the posterior parameters for the SCBI from Section {model-fit-predict}, but this time only focus on the λ competition coefficients.

```r
posterior_plots_scbi <- plot_posterior_parameters(
  posterior_param = posterior_param_scbi,
  sp_to_plot = c("quru", "litu", "cagl", "cato")
)
```

```r
posterior_plots_scbi[["lambda"]]
```

Add explanation here.

HEY BERT PICK IT UP HERE

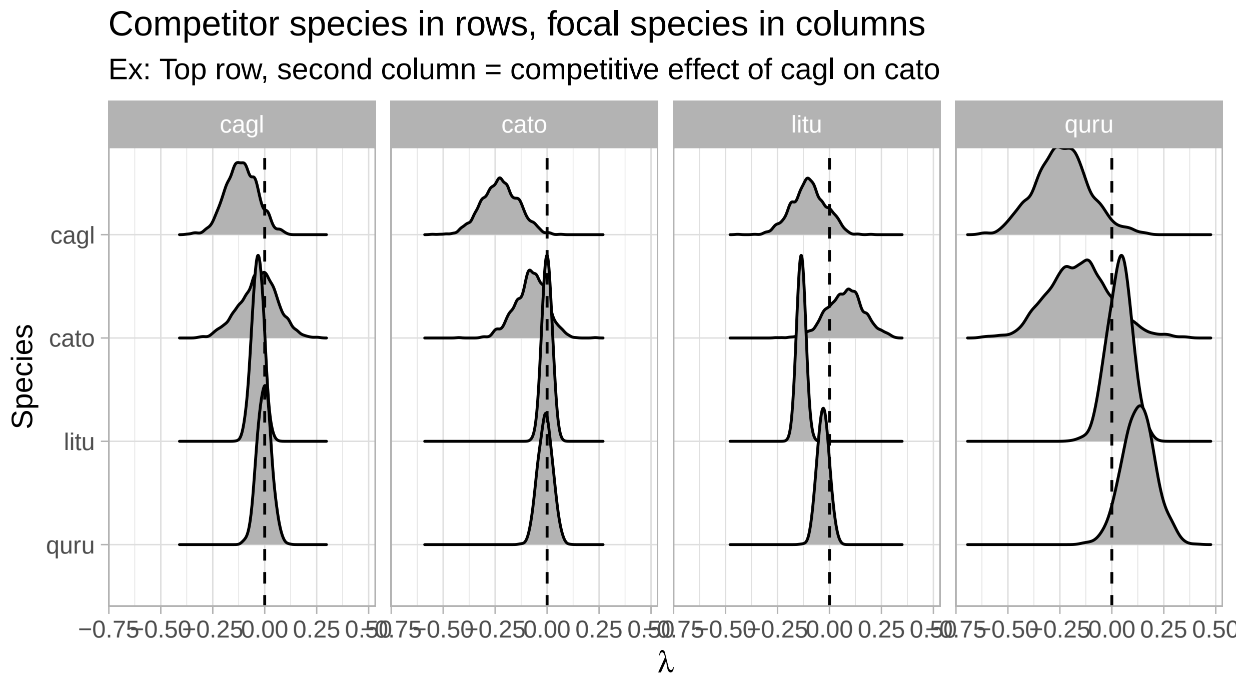


Figure 10: Posterior distribution of lambda.

4 Discussion

5 Acknowledgments

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.

Gonzalez-Akre, E., McGregor, I., Anderson-Teixeira, K., Dow, C., Herrmann, V., Terrell, A., Kim, A. Y., Vinod, N. & Helcoski, R. (2020), 'SCBI-ForestGEO/SCBI-ForestGEO-Data: 2020 update'.
URL: *https://doi.org/10.5281/zenodo.4041595*

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*

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.

Tatsumi, S., Owari, T., Ohkawa, A. & Nakagawa, Y. (2013), 'Bayesian modeling of neighborhood competition in uneven-aged mixed-species stands', *Formath* **12**, 191–209.

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*

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*