The forestecology R package for fitting and assessing neighborhood models of the effect of interspecific competition on the growth of trees

# Abstract

1. Neighborhood competition models are powerful tools to measure the effect of interspecific competition. Statistical methods to ease the application of these models are currently lacking.
2. We present the forestecology package providing methods to i) specify neighborhood competition models, ii) evaluate the effect of competitor species identity using permutation tests, and iii) measure model performance using spatial cross-validation. Following Allen and Kim ([2020](#ref-allen_permutation_2020)), we implement a Bayesian linear regression neighborhood 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 research sites. Given ForestGEO’s data collection protocols and data formatting standards, the package was designed with cross-site compatibility in mind. We highlight the importance of spatial cross-validation when interpreting model results.
4. The package features i) tidyverse-like structure whereby verb-named functions can be modularly “piped” in sequence, ii) functions with standardized inputs/outputs of simple features sf package class, and iii) an 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 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, interspecific competition, neighborhood competition, tree growth, R, ForestGEO, spatial cross-validation

Running head: forestecology R package

# 1 Introduction

Repeat-censused forest plots offer excellent opportunities to test neighborhood models of the effect of competition on the growth of trees (Canham, LePage, and Coates [2004](#ref-canham_neighborhood_2004)). Neighborhood models of competition have been used to: test whether the species identity of a competitor matters [Uriarte et al. ([2004](#ref-uriarte_spatially_2004)); measure species-specific competition coefficients (Das [2012](#ref-das_effect_2012); Tatsumi, Owari, and Mori [2016](#ref-tatsumi_estimating_2016)); test competing models to see what structures competitive interactions, e.g. traits or phylogeny (Allen and Kim [2020](#ref-allen_permutation_2020); Uriarte et al. [2010](#ref-uriarte_trait_2010)); and inform selective logging practices (Canham et al. [2006](#ref-canham_neighborhood_2006)). Although these are well-described methods, few methods are currently available for easy application.

We address this shortcoming with the forestecology R package providing methods and data for forest ecology model fitting and assessment, available on CRAN (<https://cran.r-project.org/package=forestecology>) and on GitHub (<https://github.com/rudeboybert/forestecology>). The package is written to model stem diameter growth between two censuses based on neighborhood competition, largely following the methods in Allen and Kim ([2020](#ref-allen_permutation_2020)).

Let index all trees of “focal” species ; let index all focal species; and let index all “competitor” species. The average annual growth in diameter at breast height (DBH) (in centimeters/year) of the tree of focal species is modeled as

where is the diameter-independent growth rate of species ; is the DBH of the focal tree at the earlier census and the slope of that species’s diameter-growth relationship; is the sum of some numerical explanatory variable of all trees of competitor species , and quantifies the corresponding change in growth for individuals of species from these competitors; and is a random error term distributed .

Allen and Kim ([2020](#ref-allen_permutation_2020)) use the sum of the basal area of all trees of competitor species as . Furthermore, they estimate all parameters via Bayesian linear regression, while exploiting Normal/Inverse Gamma conjugacy to derive closed-form solutions to all posterior distributions[[1]](#footnote-24). These closed-form solutions are not as computationally expensive as approximations from Markov Chain Monte Carlo algorithms.

To evaluate whether competitor species identity matters, Allen and Kim ([2020](#ref-allen_permutation_2020)) run a permutation test where a null hypothesis of no species grouping-specific effects of competition is assumed, thus the species identity of all competitors can be permuted:

Furthermore, to account for the spatial autocorrelation in their estimates of out-of-sample model error, Allen and Kim ([2020](#ref-allen_permutation_2020)) use spatial cross-validation. Estimates of model error that do not account for this dependence tend to underestimate the true model error (Roberts et al. [2017](#ref-roberts_cross-validation_2017)).

The package is designed with “tidy” design principles in mind (Wickham et al. [2019](#ref-wickham_welcome_2019)). Much like all tidyverse packages, forestecology has verb-named functions that can be modularly composed using the pipe %>% operator to sequentially complete all necessary analysis steps (Bache and Wickham [2020](#ref-bache_pipe_2020)).

Furthermore, the inputs and outputs of most functions use the same “simple features for R” data structures for spatial data from the sf package (Pebesma [2018](#ref-pebesma_simple_2018)). Previously sp package classes were commonly used for storing spatial data and interfacing with geospatial libraries (Bivand, Pebesma, and Gomez-Rubio [2013](#ref-bivand_sp_2013)); the sf package aims to improve on the sp package by:

1. Using simple feature access as the base standard for representing and encoding spatial data, rather than shapefiles (Herring [2011](#ref-herring_2011)).
2. Leveraging improvements in external libraries for reading and writing spatial data (GDAL) and for geometrical operations (GEOS) (Warmerdam [2008](#ref-warmerdam_geospatial_2008); Team [2017](#ref-geos_2017)).
3. Integrating closely with the popular tidyverse suite of packages for data science (Wickham et al. [2019](#ref-wickham_welcome_2019)).

By using the sf package classes to represent spatial data rather than the sp package, the implementation and use of the forestecology package’s spatial algorithms was greatly simplified.

# 2 forestecology workflow: a case study

We present a case-study of forestecology’s functionality on data from the Smithsonian Conservation Biology Institute (SCBI) large forest dynamics plot in Front Royal, VA, USA, part of the ForestGEO global network of research sites (Bourg et al. [2013](#ref-bourg_initial_2013); Anderson‐Teixeira et al. [2015](#ref-andersonteixeira_ctfs-forestgeo_2015); Davies et al. [2021](#ref-davies_forestgeo_2021)). 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 package has the following goals: to evaluate i) the effect of competitor species identity using permutation tests and ii) model performance using spatial cross-validation. We outline the four-step basic analysis sequence:

1. Compute the growth of stems based on two censuses.
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 predicted values.
   3. Visualize posterior distributions.

We start by loading all packages.

library(tidyverse)  
library(lubridate)  
library(sf)  
library(patchwork)  
library(forestecology)  
library(blockCV)  
  
# Resolve conflicting functions  
filter <- dplyr::filter  
select <- dplyr::select

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

We first compute the growth of trees using data from two censuses. compute\_growth() computes the average annual growth based on census data that roughly follows ForestGEO standards. Despite such standards, minor variations will still exist between sites, thereby necessitating some data wrangling. For example, the SCBI site records all DBH values in millimeters (Bourg et al. [2013](#ref-bourg_initial_2013)), whereas the Michigan Big Woods site used in Allen and Kim ([2020](#ref-allen_permutation_2020)) records them in centimeters (Allen et al. [2020](#ref-allen_michigan_2020)).

We load both 2008 and 2014 SCBI census .csv files as they existed on GitHub on 2021/08/02 and perform minor data wrangling (Gonzalez-Akre, McGregor, et al. [2020](#X934ea31ad7769370076365c5e4b0bbb4a041a66)). We then only consider a 9 ha subsection of the 25.6 ha of the site to speed up computation for this example: gx from 0–300 instead of 0–400 and gy from 300–600 instead of 0–640.

census\_2013\_scbi <- read\_csv("scbi.stem2.csv") %>%  
 select(stemID, sp, date = ExactDate, gx, gy, dbh, codes, status) %>%  
 mutate(  
 # Convert date from character to date  
 date = mdy(date),  
 # Convert dbh to be in cm  
 dbh = as.numeric(dbh) / 10  
 ) %>%  
 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  
 ) %>%  
 filter(gx < 300, between(gy, 300, 600))

These two data frames are then used as inputs to compute\_growth(), along with id specifying the variable that uniquely identifies each tree-stem. We also discard all resprouts with code == R in the later census, since we are only interested in the growth of surviving, and not resprouted, stems.

growth\_scbi <-  
 compute\_growth(  
 census\_1 = census\_2013\_scbi,  
 census\_2 = census\_2018\_scbi %>% filter(!str\_detect(codes, "R")),  
 id = "stemID"  
 )  
growth\_scbi %>%  
 select(stemID, sp, dbh1, dbh2, growth, geometry)  
## Simple feature collection with 7954 features and 5 fields  
## Geometry type: POINT  
## Dimension: XY  
## Bounding box: xmin: 0.2 ymin: 300 xmax: 300 ymax: 600  
## CRS: NA  
## # A tibble: 7,954 × 6  
## stemID sp dbh1 dbh2 growth geometry  
## <dbl> <fct> <dbl> <dbl> <dbl> <POINT>  
## 1 4 nysy 13.6 14.2 0.103 (14.2 428)  
## 2 5 havi 8.8 9.6 0.150 (9.4 436)  
## 3 6 havi 3.25 4 0.140 (1.3 434)  
## 4 77 qual 65.2 66 0.141 (34.7 307)  
## 5 79 tiam 47.7 46.8 -0.161 (40 381)  
## # … with 7,949 more rows

The output growth\_scbi is a data frame of class sf that includes among other variables the species variable sp converted to a factor, the average annual growth in DBH (cm y-1) for all stems that were alive at both time points, and the sf package’s encoding of geolocations of geometry type <POINT>. Given that growth\_scbi is of class sf, it can be easily plotted in ggplot2 using geom\_sf() as seen in Figure 2.1.

ggplot() +  
 geom\_sf(data = growth\_scbi %>% sample\_n(500), aes(size = growth)) +  
 scale\_size\_binned(limits = c(0.1, 1)) +  
 labs(size = expression(paste(Growth, " (cm ", y^{-1}, ")")))

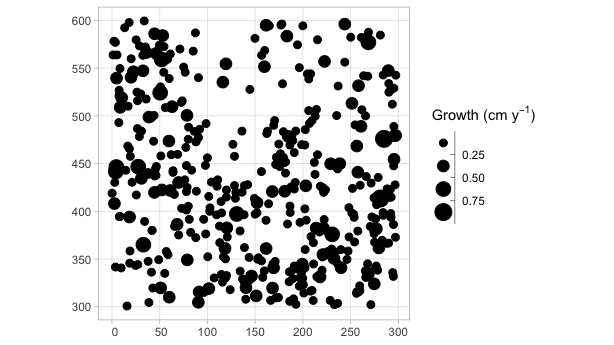


Figure 2.1: Step 1 - Compute growth of trees based on census data. A map of the growth of a random sample of 500 trees from a 9 ha subsection of the Smithsonian Conservation Biology Institute (SCBI) forest plot.

We also load species information as it existed on GitHub on 2021/08/02, which includes family, genus, and species information; as well as classifications of the canopy position (canopy, canopy emergent, understory, shrub layer), drought tolerance (intolerant, resistant), and other characteristics of the species.

sp\_info <- read\_csv("SCBI\_ForestGEO\_sp\_ecology.csv") %>%  
 select(  
 sp = spcode, family, genus, species, canopy\_position,  
 drought\_tolerance  
 )  
sp\_info  
## # A tibble: 65 × 6  
## sp family genus species canopy\_position drought\_toleran…  
## <chr> <chr> <chr> <chr> <chr> <chr>   
## 1 acne Sapindaceae Acer negundo understory drought-resista…  
## 2 acpl Sapindaceae Acer platanoides canopy drought-resista…  
## 3 acru Sapindaceae Acer rubrum canopy drought-resista…  
## 4 aial Simaroubaceae Ailanthus altissima canopy drought-resista…  
## 5 amar Rosaceae Amelanchier arborea understory drought-resista…  
## # … with 60 more rows

We join this species information to our growth\_scbi data frame and convert the species variable to a factor.

growth\_scbi <- growth\_scbi %>%  
 left\_join(sp\_info, by = "sp") %>%  
 mutate(sp = as.factor(sp))

Furthermore, we compute two potential competitor explanatory variables from Equation (1.1). First, the basal area of each tree as a function of its DBH in the earlier census. Second, the above ground biomass as estimated by allometric equations encoded in the get\_biomass() function from the allodb package (Gonzalez-Akre, Piponiot, et al. [2020](#ref-allodb_package)); this function has DBH, species, and geographic coordinates as arguments.

# Install development version of allodb using:  
# remotes::install\_github("forestgeo/allodb")  
library(allodb)  
growth\_scbi <- growth\_scbi %>%  
 mutate(  
 # Compute basal area:  
 basal\_area = 0.0001 \* pi \* (dbh1 / 2)^2,  
 # Compute above ground biomass:  
 agb = get\_biomass(  
 dbh = dbh1,  
 genus = genus,  
 species = species,  
 coords = c(-78.2, 38.9)  
 )  
 )

## 2.2 Step 2: Add spatial information

We then add spatial information to growth\_scbi. We first add a “buffer region” to the periphery of the study region. Since some of our model’s explanatory variables are cumulative, 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 the study region who will not have all their neighbors included in the census stems. 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 growth modeled (Waller and Gotway [2004](#ref-waller_applied_2004)).

Our model of interspecific competition relies on a spatial definition of who competitor trees are: all trees within a distance comp\_dist of a focal tree. Here we set comp\_dist to 7.5m, a value informed by other studies (Canham, LePage, and Coates [2004](#ref-canham_neighborhood_2004); Uriarte et al. [2004](#ref-uriarte_spatially_2004); Canham et al. [2006](#ref-canham_neighborhood_2006)), but the package could also be used to compare multiple distances and see which is best supported (see Appendix 6). We use comp\_dist and a manually constructed sf representation of the study region’s boundary as inputs to add\_buffer\_variable() to add a buffer boolean variable to growth\_scbi. All trees with buffer equal to FALSE will be our focal trees whose growth will be modeled, whereas those with TRUE will only act as competitor trees.

# Define competitive distance range  
comp\_dist <- 7.5  
  
# Manually construct study region boundary  
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 we add are blocks corresponding to folds of a spatial cross-validation algorithm. Conventional cross-validation algorithms assign individual observations to folds by randomly resampling them all while assuming they are statistically independent. In the case of forest census data however, observations exhibit spatial autocorrelation. We therefore incorporate this dependence into the cross-validation algorithm by resampling spatial blocks of trees (Roberts et al. [2017](#ref-roberts_cross-validation_2017); Pohjankukka et al. [2017](#ref-pohjankukka_estimating_2017)).

We first manually define an sf object defining four folds that partition the study region. We then use the output of the spatialBlock() function from the blockCV package to associate each tree in growth\_scbi to the correct foldID (Valavi et al. [2019](#ref-valavi_blockcv_2019)). This foldID variable will be used in Section 2.6.

Figure 2.2 illustrates the net effect of adding these two elements of spatial information to growth\_scbi.

# Manually define spatial blocks to act as folds  
n\_fold <- 4  
fold1 <- cbind(c(0, 150, 150, 0), c(300, 300, 450, 450))  
fold2 <- cbind(c(150, 300, 300, 150), c(300, 300, 450, 450))  
fold3 <- cbind(c(0, 150, 150, 0), c(450, 450, 600, 600))  
fold4 <- cbind(c(150, 300, 300, 150), c(450, 450, 600, 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  
spatial\_block\_scbi <-  
 spatialBlock(  
 speciesData = growth\_scbi, k = n\_fold,  
 selection = "systematic", blocks = blocks\_scbi,  
 showBlocks = FALSE, verbose = FALSE  
 )  
  
growth\_scbi <- growth\_scbi %>%  
 mutate(foldID = spatial\_block\_scbi$foldID %>% factor())

ggplot() +  
 geom\_sf(  
 data = blocks\_scbi,   
 fill = "transparent", linetype = "dashed"  
 ) +  
 geom\_sf\_text(  
 data = growth\_scbi %>% sample\_n(1000),  
 aes(label = foldID, col = buffer)  
 )

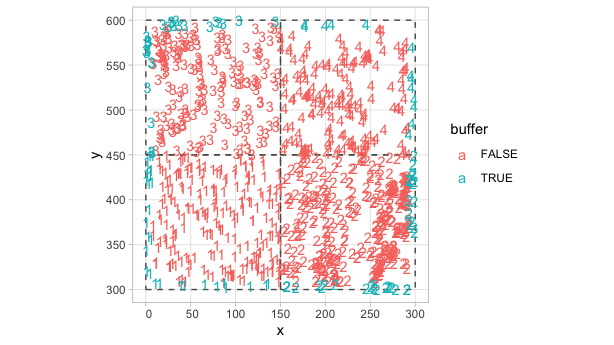


Figure 2.2: Step 2 - Add spatial information. A buffer region and spatial cross-validation blocks 1 through 4. The location of each tree is marked with its fold number where the folds are delineated with solid lines. The color of each digit indicates whether the tree is part of the buffer region (thus will only be considered as a competitor tree) or is part of the interior of the study region (thus is a focal tree whose growth is of modeled interest).

## 2.3 Step 3: Identify all focal and corresponding competitor trees

We then identify all focal trees and their corresponding competitor trees. More specifically, identify all trees that are not part of the buffer region, have a valid growth measurement, and have at least one neighbor within 7.5m. We do this using create\_focal\_vs\_comp(), which takes the previously detailed comp\_dist and id arguments, the sf representation of the spatial cross-validation blocks blocks\_scbi, and a specification comp\_x\_var of the basal\_area variable we use as the competitor explanatory variable from Equation (1.1). This function returns a new data frame focal\_vs\_comp\_scbi.

focal\_vs\_comp\_scbi <- growth\_scbi %>%  
 create\_focal\_vs\_comp(  
 comp\_dist, blocks = blocks\_scbi, id = "stemID",   
 comp\_x\_var = "basal\_area"  
 )  
  
focal\_vs\_comp\_scbi %>%  
 select(focal\_ID, focal\_sp, geometry, growth, comp)  
## # A tibble: 6,296 × 5  
## focal\_ID focal\_sp geometry growth comp   
## <dbl> <fct> <POINT> <dbl> <list>   
## 1 4 nysy (14.2 428) 0.103 <tibble [20 × 4]>  
## 2 5 havi (9.4 436) 0.150 <tibble [32 × 4]>  
## 3 79 tiam (40 381) -0.161 <tibble [20 × 4]>  
## 4 80 caca (38.7 422) 0.253 <tibble [12 × 4]>  
## 5 96 libe (60 310) 0.262 <tibble [14 × 4]>  
## # … with 6,291 more rows

The resulting focal\_vs\_comp\_scbi has 6296 rows, representing the subset of the 7954 trees in growth\_scbi that will be considered as focal trees. The variables focal\_ID and focal\_sp relate to tree-stem identification and species information. Most notably however is the variable comp, which contains information on all competitor trees saved in tidyr package list-column format (Wickham [2020](#ref-tidyr_package)). To inspect this information, we flatten the comp list-column for the tree with focal\_ID 4 in the first row, here a tibble [20 × 4], into regular columns using unnest() from the tidyr package.

focal\_vs\_comp\_scbi %>%  
 filter(focal\_ID == 4) %>%  
 select(focal\_ID, dbh, comp) %>%  
 unnest(cols = "comp")  
## # A tibble: 20 × 6  
## focal\_ID dbh comp\_ID dist comp\_sp comp\_x\_var  
## <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

We observe 4 variables describing 20 competitor trees: the unique tree-stem ID, the distance to the focal tree (all 7.5m), the species, and the basal area (in m) calculated as for the DBH in cm from the earlier census. Saving competitor information in list-column format minimizes redundancy since we do not need to repeat information on the focal tree 20 times. We visualize the spatial distribution of these trees in Figure 2.3.

Here we use basal area as the continuous competitor explanatory variable but the package is flexible to allow the user to specify any competitor explanatory variable (basal area, biomass, tree height, a soil nutrient value, ). The package can also be used to compare competitor explanatory variables and see which best explains tree growth, see Appendix 7 for an example comparing basal area and above ground biomass. Similarly, the package can use any categorical variable as an explanatory variable and compare between different categorical variables. For example in Allen and Kim ([2020](#ref-allen_permutation_2020)) we compare grouping individuals based on species, family, and based on trait-based groups. In Appendix 8 we give another example and compare grouping individuals by species or by potential canopy position (canopy, understory, shrub layer).

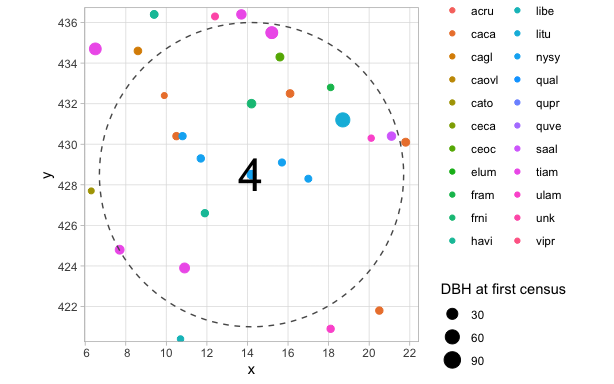


Figure 2.3: Step 3 - Identify all focal and corresponding competitor trees. The dashed circle extends 7.5m away from the focal tree 4 while all 20 competitor trees are within this circle.

## 2.4 Step 4: Fit model

Lastly, we fit the competition Bayesian linear regression model for tree growth outlined in Equation (1.1) using comp\_bayes\_lm(). This function has an option to specify prior distributions of all parameters, chosen here to be the defaults detailed in ?comp\_bayes\_lm.

comp\_bayes\_lm\_scbi <- focal\_vs\_comp\_scbi %>%  
 comp\_bayes\_lm(prior\_param = NULL)

The resulting comp\_bayes\_lm\_scbi is an object of S3 class type comp\_bayes\_lm containing the posterior values of all parameters. Furthermore, this class includes generics for three methods. First, the generic for print() displays the names of all prior and posterior parameters and the model formula:

comp\_bayes\_lm\_scbi  
## Bayesian linear regression model parameters with a multivariate Normal  
## likelihood. See ?comp\_bayes\_lm for details:  
##   
## 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 + acru \* sp + amar \* sp + astr  
## \* sp + caca \* sp + caco \* sp + cade \* sp + cagl \* sp + caovl \* sp + cato  
## \* sp + ceca \* sp + ceoc \* sp + chvi \* sp + cofl \* sp + crpr \* sp + crsp  
## \* sp + divi \* sp + elum \* sp + fagr \* sp + fram \* sp + frni \* sp + frpe  
## \* sp + havi \* sp + ilve \* sp + juci \* sp + juni \* sp + libe \* sp + litu  
## \* sp + nysy \* sp + pist \* sp + pivi \* sp + ploc \* sp + prav \* sp + prse  
## \* sp + qual \* sp + quco \* sp + qufa \* sp + qumi \* sp + qupr \* sp + quru  
## \* sp + quve \* sp + rops \* sp + saal \* sp + saca \* sp + tiam \* sp + ulam  
## \* sp + ulru \* sp + unk \* sp + vipr \* sp

Next, the generic for predict() takes the posterior parameter values in comp\_bayes\_lm\_scbi and a newdata data frame, and outputs a vector growth\_hat of predicted DBH values 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 %>%  
 select(focal\_ID, focal\_sp, dbh, growth, growth\_hat)  
## # A tibble: 6,296 × 5  
## focal\_ID focal\_sp dbh growth growth\_hat  
## <dbl> <fct> <dbl> <dbl> <dbl>  
## 1 4 nysy 13.6 0.103 0.0809  
## 2 5 havi 8.8 0.150 0.112   
## 3 79 tiam 47.7 -0.161 0.229   
## 4 80 caca 5.15 0.253 0.121   
## 5 96 libe 2.3 0.262 0.142   
## # … with 6,291 more rows

We can now compare the observed and predicted growths to compute the root mean squared error (RMSE) of our model:

model\_rmse <- focal\_vs\_comp\_scbi %>%  
 rmse(truth = growth, estimate = growth\_hat) %>%  
 pull(.estimate)  
model\_rmse  
## [1] 0.128

Lastly, the generic for ggplot2::autoplot() allows us to visualize all posterior distributions, as seen in Figure 2.4. Setting type to "intercepts" and "dbh\_slopes" returns species-specific posterior distributions for and respectively, while setting type = "competition" returns competition coefficients .

# 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 the patchwork package  
(plot1 | plot2) / plot3

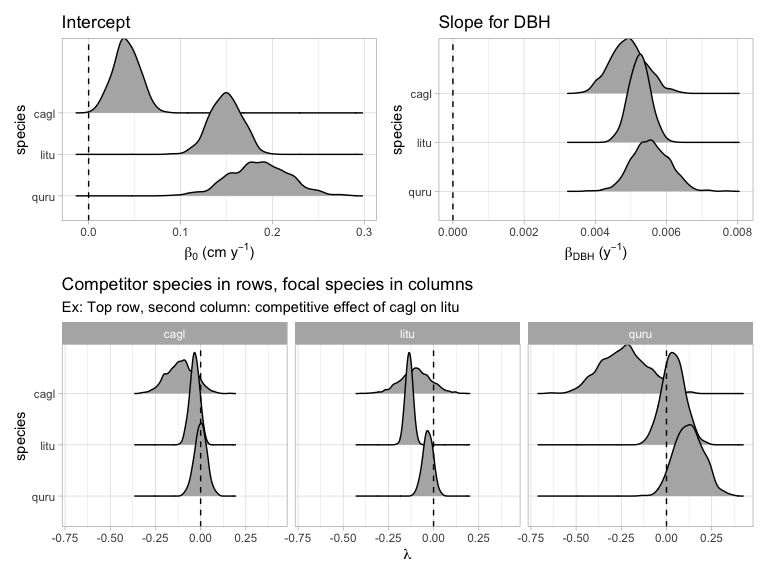


Figure 2.4: Step 4 - Fit model. Posterior distributions of all parameters. For compactness we include only three species.

For many users the visualizations of will be of particular interest as they provide insight into species-specific competitive interactions, where negative values indicate a competitor species which slows the growth of a focal species. Here, for example, we see that tulip poplars (litu) have a strong negative effect on the growth of conspecifics but relatively lesser effect on pignut hickory (cagl) and red oak (quru) neighbors.

Currently the forestecology package can only fit the competition Bayesian linear regression model in Equation (1.1). However, it can be extended to any model as long as it is implemented in a function similar to comp\_bayes\_lm().

## 2.5 Evaluate the effect of competitor species identity using permutation tests

To evaluate the effect of competitor species identity, we use the above four steps along with the permutation test in Equation (1.2). Under a null hypothesis where competitor species identity does not matter, we can permute the competitor species identities within each focal tree, compute the RMSE test statistic, repeat this process several times to construct a null distribution, 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 only randomly resamples the comp\_sp variable without replacement, leaving all other variables intact. This resampling is nested within each focal tree in order to preserve neighborhood structure. We perform this permutation test once again using comp\_bayes\_lm() but by setting run\_shuffle = TRUE.

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\_comp\_scbi)  
 )

model\_rmse\_shuffle <- focal\_vs\_comp\_scbi %>%  
 rmse(truth = growth, estimate = growth\_hat\_shuffle) %>%  
 pull(.estimate)  
model\_rmse\_shuffle  
## [1] 0.131

The resulting permutation test RMSE of 0.131 is larger than the earlier RMSE of 0.128, suggesting that models that do incorporate competitor species identity better fit the data.

## 2.6 Evaluate model performance using spatial cross-validation

To evaluate model performance, we use spatial cross-validation. The model fit in Section 2.4 uses the same data to both fit and assess model performance. Given the spatial-autocorrelation of our data, this can potentially lead to overfit models (Roberts et al. [2017](#ref-roberts_cross-validation_2017)). To mitigate this risk, we use the spatial cross-validation blocking scheme encoded in the foldID variable from Section 2.2 and visualized in Figure 2.2.

At each iteration of the cross-validation, one fold acts as the test set and the remaining three act as the training set. We fit the model to all focal trees in the training set, apply the model to all focal trees in the test set, compute predicted values, and compute the RMSE. Furthermore, to maintain spatial independence between the test and training sets, a “fold buffer” that extends 7.5m outwards from the boundary of the test set is considered; all trees within this “fold buffer” are excluded from the training set (see Figure 2.5).

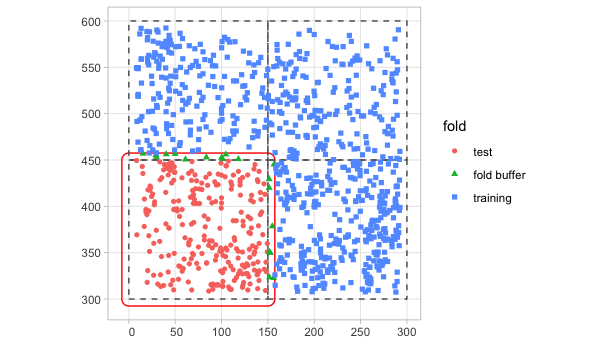


Figure 2.5: Schematic of spatial cross-validation. Using the k = 1 fold (bottom-left) as the test set, k = 2 through 4 as the training set, along with a fold buffer extending outwards from the test set to maintain spatial independence between it and the training set.

This process is repeated for each of the four folds acting as the test set, then the four RMSE’s are averaged to provide a single estimate of model error. This algorithm is implemented in run\_cv(), which acts as a wrapper function to both comp\_bayes\_lm() that fits the model and predict() that returns predicted values.

focal\_vs\_comp\_scbi <- focal\_vs\_comp\_scbi %>%  
 run\_cv(comp\_dist = comp\_dist, blocks = blocks\_scbi)

model\_rmse\_cv <- focal\_vs\_comp\_scbi %>%  
 rmse(truth = growth, estimate = growth\_hat) %>%  
 pull(.estimate)  
model\_rmse\_cv  
## [1] 0.14

The resulting RMSE of 0.14 computed using cross-validation is larger than the earlier RMSE of 0.128, suggesting that models that do not account for spatial autocorrelation generate model error estimates that are overly optimistic, i.e. RMSE values that are too low.

# 3 Importance of spatial cross-validation

run\_cv() also accepts the run\_shuffle argument in order to permute competitor species identity as described in Section 2.5. Figure 3.1 compares model performance for 49 permutations of competitor species and RMSE calculations, both with and without cross-validation. Without cross-validation, competitor species identity does matter as the observed RMSE was significantly lower than the permutation null distribution of RMSE. However, once we incorporate spatial cross-validation, this improvement disappears. These results suggest that in this 9 ha subplot of the SCBI plot, competitive interactions do not depend on the identity of the competitor, which is the opposite of what has been observed in other locations (Allen and Kim [2020](#ref-allen_permutation_2020); Uriarte et al. [2004](#ref-uriarte_spatially_2004)). This provides a striking example of the importance of cross-validation, as without it the over-fit model gives rise to an incorrect conclusion.

![Figure 3.1: Comparison of root mean squared error of models for standard, permuted, and spatially cross-validated error estimates. The dotted lines show observed RMSE while the histograms show the null distribution of RMSE for 49 permutations under the null hypothesis of no competitor species identity effects. The colors indicate whether spatial cross-validation was used or not.](data:application/pdf;base64,)

Figure 3.1: Comparison of root mean squared error of models for standard, permuted, and spatially cross-validated error estimates. The dotted lines show observed RMSE while the histograms show the null distribution of RMSE for 49 permutations under the null hypothesis of no competitor species identity effects. The colors indicate whether spatial cross-validation was used or not.

# 4 Conclusion and future work

The forestecology package provides an accessible way to fit and test models of neighborhood competition. The package follows the tidy data design principles, leverages the sf package for spatial data, and S3 open-oriented model implementation structure (Pebesma [2018](#ref-pebesma_simple_2018)). We hope that the package will increase the use of neighborhood competition models to better understand what structures plant competition.

While the package is designed with ForestGEO plot data in mind, we envision that it can be modified to work on any single large, mapped forest plot in which at least two measurements of each individual have been taken. Furthermore, we hope that future versions of the package will be flexible to other plot layouts, for example inventory plot-structure with many spatially separated plots like the US Forest Service Forest Inventory and Analysis plots (Smith [2002](#ref-smith_forest_2002)).

We also hope to extend the forestecology package’s functionality to account for a larger variety of models for tree growth. One clear future direction would be to allow competition based on species trait values rather than species identity. There is evidence that traits predict competitive outcomes (Kunstler et al. [2012](#ref-kunstler_competitive_2012); Lasky et al. [2014](#ref-lasky_trait-mediated_2014); Uriarte et al. [2010](#ref-uriarte_trait_2010)). Thus an extension of the model would allow values from Equation (1.1) to be a function of the traits of competing species.

Lastly, the forestecology current uses the blockCV package behind the scenes to create the spatial blocks acting as folds for our spatial cross-validation algorithm detailed in Sections 2.2 and 2.6 (Valavi et al. [2019](#ref-valavi_blockcv_2019)). This back-end functionality could be substituted with the spatialsample package for spatial resampling infrastructure; a tidymodels package under active development as of 2021 (Silge [2021](#ref-spatialsample_package); Kuhn and Wickham [2020](#ref-tidymodels_package)).

Lastly, currently the package only implements the Bayesian linear regression model detailed in Equation (1.1). As we demonstrate in Section 2.4 however, the fitting of this model is self-contained in a single function comp\_bayes\_lm() which returns an object of S3 class type comp\_bayes\_lm. This class has generic methods implemented to print, make predictions, and plot all results. Therefore the package can be modularly extended to fit other models as long as they are coded similarly to comp\_bayes\_lm() and have equivalent generic methods implemented.

# Acknowledgements

We thank Sophie Li for their feedback on the package interface.

# Conflict of Interest

We declare no competing interests.

# Author’s contributions

**Albert Y. Kim**: Conceptualization (equal); Methodology (equal); Software (equal); Writing-original draft (lead); Writing-review & editing (equal). **David N. Allen**: Conceptualization (equal); Methodology (equal); Software (equal); Writing-original draft (supporting); Writing-review & editing (equal). **Simon P. Couch**: Software (supporting); Writing-review & editing (supporting).

# Data accessibility

All data and source code for this manuscript on GitHub at <https://github.com/rudeboybert/forestecology> and are archived on Zenodo at <https://doi.org/10.5281/zenodo.5348060>. The example Smithsonian Conservation Biology Institute census and species information data are available on GitHub at <https://github.com/SCBI-ForestGEO/SCBI-ForestGEO-Data/tree/master/> and are archived on Zenodo at <https://doi.org/10.5281/zenodo.2649301> (Gonzalez-Akre, McGregor, et al. [2020](#X934ea31ad7769370076365c5e4b0bbb4a041a66)).

# 5 References

Allen, David, Christopher Dick, Robyn J Burnham, Ivette Perfecto, and John Vandermeer. 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. <http://hdl.handle.net/2027.42/156251>.

Allen, David, and Albert Y. Kim. 2020. “A Permutation Test and Spatial Cross-Validation Approach to Assess Models of Interspecific Competition Between Trees.” *PLOS ONE* 15 (3): e0229930. <https://doi.org/10.1371/journal.pone.0229930>.

Anderson‐Teixeira, Kristina J., Stuart J. Davies, Amy C. Bennett, Erika B. Gonzalez‐Akre, Helene C. Muller‐Landau, S. Joseph Wright, Kamariah Abu Salim, et al. 2015. “CTFS-ForestGEO: A Worldwide Network Monitoring Forests in an Era of Global Change.” *Global Change Biology* 21 (2): 528–49. <https://doi.org/10.1111/gcb.12712>.

Bache, Stefan Milton, and Hadley Wickham. 2020. *Magrittr: A Forward-Pipe Operator for R*. <https://CRAN.R-project.org/package=magrittr>.

Bivand, Roger S., Edzer Pebesma, and Virgilio Gomez-Rubio. 2013. *Applied Spatial Data Analysis with R, Second Edition*. Springer, NY. <https://asdar-book.org/>.

Bourg, Norman A., William J. McShea, Jonathan R. Thompson, Jennifer C. McGarvey, and Xiaoli Shen. 2013. “Initial Census, Woody Seedling, Seed Rain, and Stand Structure Data for the SCBI SIGEO Large Forest Dynamics Plot.” *Ecology* 94 (9): 2111–2. <https://doi.org/10.1890/13-0010.1>.

Canham, Charles D., Philip T. LePage, and K. Dave Coates. 2004. “A Neighborhood Analysis of Canopy Tree Competition: Effects of Shading Versus Crowding.” *Canadian Journal of Forest Research* 34 (4): 778–87. <https://doi.org/10.1139/x03-232>.

Canham, Charles D., Michael J. Papaik, María Uriarte, William H. McWilliams, Jennifer C. Jenkins, and Mark J. Twery. 2006. “Neighborhood Analyses of Canopy Tree Competition Along Environmental Gradients in New England Forests.” *Ecological Applications* 16 (2): 540–54. [https://doi.org/10.1890/1051-0761(2006)016[0540:NAOCTC]2.0.CO;2](https://doi.org/10.1890/1051-0761(2006)016%5B0540:NAOCTC%5D2.0.CO;2).

Das, Adrian. 2012. “The Effect of Size and Competition on Tree Growth Rate in Old-Growth Coniferous Forests.” *Canadian Journal of Forest Research* 42: 1983–95.

Davies, Stuart J., Iveren Abiem, Kamariah Abu Salim, Salomón Aguilar, David Allen, Alfonso Alonso, Kristina Anderson-Teixeira, et al. 2021. “ForestGEO: Understanding Forest Diversity and Dynamics Through a Global Observatory Network.” *Biological Conservation* 253: 108907. <https://doi.org/https://doi.org/10.1016/j.biocon.2020.108907>.

Gonzalez-Akre, Erika, Ian McGregor, Kristina Anderson-Teixeira, Cameron Dow, Valentine Herrmann, Alyssa Terrell, Albert Y. Kim, Nidhi Vinod, and Ryan Helcoski. 2020. “SCBI-ForestGEO/SCBI-ForestGEO-Data: 2020 Update.” Zenodo. <https://doi.org/10.5281/zenodo.4041595>.

Gonzalez-Akre, Erika, Camille Piponiot, Mauro Lepore, and Kristina Anderson-Teixeira. 2020. *Allodb: An R Database for Biomass Estimation at Extratropical Forest Plots*. <https://github.com/forestgeo/allodb>.

Herring, J. R. 2011. “OpenGIS Implementation Standard for Geographic Information-Simple Feature Access Part 1: Common Architecture.” In, 211. Open Geospatial Consortium Inc.

Kuhn, Max, and Hadley Wickham. 2020. *Tidymodels: A Collection of Packages for Modeling and Machine Learning Using Tidyverse Principles.* <https://www.tidymodels.org>.

Kunstler, Georges, Sébastien Lavergne, Benoît Courbaud, Wilfried Thuiller, Ghislain Vieilledent, Niklaus E. Zimmermann, Jens Kattge, and David A. Coomes. 2012. “Competitive Interactions Between Forest Trees Are Driven by Species’ Trait Hierarchy, Not Phylogenetic or Functional Similarity: Implications for Forest Community Assembly.” *Ecology Letters* 15 (8): 831–40. <https://doi.org/https://doi.org/10.1111/j.1461-0248.2012.01803.x>.

Lasky, Jesse R., María Uriarte, Vanessa K. Boukili, and Robin L. Chazdon. 2014. “Trait-Mediated Assembly Processes Predict Successional Changes in Community Diversity of Tropical Forests.” *Proceedings of the National Academy of Sciences* 111 (15): 5616–21. <https://doi.org/10.1073/pnas.1319342111>.

Pebesma, Edzer. 2018. “Simple Features for R: Standardized Support for Spatial Vector Data.” *The R Journal* 10 (1): 439–46. <https://doi.org/https://doi.org/10.32614/RJ-2018-009>.

Pohjankukka, Jonne, Tapio Pahikkala, Paavo Nevalainen, and Jukka Heikkonen. 2017. “Estimating the Prediction Performance of Spatial Models via Spatial K-Fold Cross Validation.” *International Journal of Geographical Information Science* 31 (10): 2001–19.

Roberts, David R., Volker Bahn, Simone Ciuti, Mark S. Boyce, Jane Elith, Gurutzeta Guillera‐Arroita, Severin Hauenstein, et al. 2017. “Cross-Validation Strategies for Data with Temporal, Spatial, Hierarchical, or Phylogenetic Structure.” *Ecography* 40 (8): 913–29. <https://doi.org/10.1111/ecog.02881>.

Silge, Julia. 2021. *Spatialsample: Spatial Resampling Infrastructure*. <https://CRAN.R-project.org/package=spatialsample>.

Smith, David Martyn. 1986. “The Practice of Silviculture.”

Smith, W Brad. 2002. “Forest Inventory and Analysis: A National Inventory and Monitoring Program.” *Environmental Pollution* 116: S233–S242.

Tatsumi, Shinichi, Toshiaki Owari, and Akira S. Mori. 2016. “Estimating Competition Coefficients in Tree Communities: A Hierarchical Bayesian Approach to Neighborhood Analysis.” *Ecosphere* 7: e01273.

Team, GEOS Development. 2017. “GEOS - Geometry Engine, Open Source.” In, 211. Open Source Geospatial Foundation. <https://trac.osgeo.org/geos/>.

Uriarte, María, Richard Condit, Charles D. Canham, and Stephen P. Hubbell. 2004. “A Spatially Explicit Model of Sapling Growth in a Tropical Forest: Does the Identity of Neighbours Matter?” *Journal of Ecology* 92 (2): 348–60. <https://doi.org/https://doi.org/10.1111/j.0022-0477.2004.00867.x>.

Uriarte, María, Nathan G. Swenson, Robin L. Chazdon, Liza S. Comita, W. John Kress, David Erickson, Jimena Forero-Montaǹa, Jess K. Zimmeran, and Jill Thompson. 2010. “Trait Similarity, Shared Ancestry and the Structure of Neighbourhood Interactions in a Subtropical Wet Forest: Implications for Community Assembly.” *Ecology Letters* 13: 1503–14.

Valavi, Roozbeh, Jane Elith, José J. Lahoz‐Monfort, and Gurutzeta Guillera‐Arroita. 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): 225–32. <https://doi.org/https://doi.org/10.1111/2041-210X.13107>.

Waller, Lance A., and Carol A. Gotway. 2004. *Applied Spatial Statistics for Public Health Data*. Hoboken, UNITED STATES: John Wiley & Sons, Incorporated. <http://ebookcentral.proquest.com/lib/smith/detail.action?docID=214360>.

Warmerdam, Frank. 2008. “The Geospatial Data Abstraction Library.” In *Open Source Approaches in Spatial Data Handling*, edited by G. Brent Hall and Michael G. Leahy, 87–104. Advances in Geographic Information Science. Berlin, Heidelberg: Springer. <https://doi.org/10.1007/978-3-540-74831-1_5>.

Wickham, Hadley. 2020. *Tidyr: Tidy Messy Data*. <https://CRAN.R-project.org/package=tidyr>.

Wickham, Hadley, Mara Averick, Jennifer Bryan, Winston Chang, Lucy D’Agostino McGowan, Romain François, Garrett Grolemund, et al. 2019. “Welcome to the Tidyverse.” *Journal of Open Source Software* 4 (43): 1686. <https://doi.org/10.21105/joss.01686>.

# 6 Appendix: Compare different competitive distances

For all the above analyses we set the cut off distance (comp\_dist) for two stems to compete as 7.5m. This distance has been estimated in previous neighborhood competition studies in forests (Canham, LePage, and Coates [2004](#ref-canham_neighborhood_2004); Uriarte et al. [2004](#ref-uriarte_spatially_2004); Canham et al. [2006](#ref-canham_neighborhood_2006)). We used 7.5m in Allen and Kim ([2020](#ref-allen_permutation_2020)) as an average of the values estimated in other studies. But our package can be used to find which distance is best supported by the data. Here we provide an example using another section of the SCBI plot to provide an additional example of the cross-validation block layout. To speed computation we do not consider species differences in competitive effects and treat all species as the same.

We observe in Figure 6.1 that a cut-off distance of approximately 6m minimizes the cross-validation estimated RMSE.

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  
 ) %>%  
 filter(gx < 400, gy < 400)  
  
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  
 ) %>%  
 filter(gx < 400, gy < 400)  
  
growth\_scbi <-  
 compute\_growth(  
 census\_1 = census\_2013\_scbi,  
 census\_2 = census\_2018\_scbi %>% filter(!str\_detect(codes, "R")),  
 id = "stemID"  
 ) %>%  
 # make all species the same, needs to be factor with at least   
 # two levels  
 mutate(sp = factor("A", levels = c("A", "B"))) %>%  
 # Compute basal area:  
 mutate(basal\_area = 0.0001 \* pi \* (dbh1 / 2)^2)  
  
study\_region\_scbi <- tibble(  
 x = c(0, 400, 400, 0, 0),  
 y = c(0, 0, 400, 400, 0)  
) %>%  
 sf\_polygon()  
  
n\_fold <- 4  
fold1 <- cbind(c(0, 200, 200, 0), c(0, 0, 200, 200))  
fold2 <- cbind(c(200, 400, 400, 200), c(0, 0, 200, 200))  
fold3 <- cbind(c(0, 200, 200, 0), c(200, 200, 400, 400))  
fold4 <- cbind(c(200, 400, 400, 200), c(200, 200, 400, 400))  
  
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  
spatial\_block\_scbi <-  
 spatialBlock(  
 speciesData = growth\_scbi, k = n\_fold,  
 selection = "systematic", blocks = blocks\_scbi,  
 showBlocks = FALSE, verbose = FALSE  
 )  
  
growth\_scbi <- growth\_scbi %>%  
 mutate(foldID = spatial\_block\_scbi$foldID %>% factor())  
  
mult\_dist\_comp <- tibble(  
 dist = c(5, 6.25, 7.5, 8.75, 10),  
 rmse = 0  
)  
  
for (i in 1:length(mult\_dist\_comp$dist)) {  
 comp\_dist <- mult\_dist\_comp$dist[i]  
   
 growth\_scbi <- growth\_scbi %>%  
 add\_buffer\_variable(size = comp\_dist, region = study\_region\_scbi)  
   
 focal\_vs\_comp\_scbi <- growth\_scbi %>%  
 create\_focal\_vs\_comp(  
 comp\_dist = comp\_dist, blocks = blocks\_scbi, id = "stemID",   
 comp\_x\_var = "basal\_area"  
 ) %>%  
 run\_cv(comp\_dist = comp\_dist, blocks = blocks\_scbi)  
   
 mult\_dist\_comp$rmse[i] <- focal\_vs\_comp\_scbi %>%  
 rmse(truth = growth, estimate = growth\_hat) %>%  
 pull(.estimate)  
}

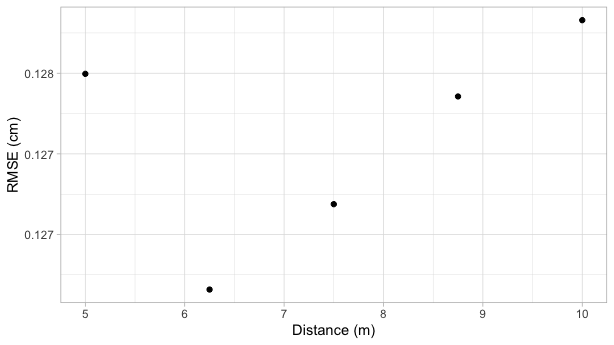


Figure 6.1: Cross-validated RMSE estimates for 5 competitive distances.

# 7 Appendix: Compare competitor explanatory variables

In the above code we use the basal area of an individual as a continuous competitor explanatory variable. But the package allows the user to specify any competitor explanatory variable in the comp\_x\_var argument of create\_focal\_vs\_comp function. Here we use the cross-validated model comparison to see which of two possible competitor explanatory variables computed in Section 2.1, basal area or above ground biomass, best explains growth.

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  
 ) %>%  
 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  
 ) %>%  
 filter(gx < 300, between(gy, 300, 600))  
  
growth\_scbi <-  
 compute\_growth(  
 census\_1 = census\_2013\_scbi,  
 census\_2 = census\_2018\_scbi %>% filter(!str\_detect(codes, "R")),  
 id = "stemID"  
 ) %>%  
 left\_join(sp\_info, by = "sp") %>%  
 mutate(  
 sp = as.factor(sp),  
 basal\_area = 0.0001 \* pi \* (dbh1 / 2)^2,  
 agb = get\_biomass(  
 dbh = dbh1,  
 genus = genus,  
 species = species,  
 coords = c(-78.2, 38.9)  
 )  
 )  
  
study\_region\_scbi <- tibble(  
 x = c(0, 300, 300, 0, 0),  
 y = c(300, 300, 600, 600, 300)  
) %>%  
 sf\_polygon()  
  
n\_fold <- 4  
fold1 <- cbind(c(0, 150, 150, 0), c(300, 300, 450, 450))  
fold2 <- cbind(c(150, 300, 300, 150), c(300, 300, 450, 450))  
fold3 <- cbind(c(0, 150, 150, 0), c(450, 450, 600, 600))  
fold4 <- cbind(c(150, 300, 300, 150), c(450, 450, 600, 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  
spatial\_block\_scbi <-  
 spatialBlock(  
 speciesData = growth\_scbi, k = n\_fold,  
 selection = "systematic", blocks = blocks\_scbi,  
 showBlocks = FALSE, verbose = FALSE  
 )  
  
growth\_scbi <- growth\_scbi %>%  
 mutate(foldID = spatial\_block\_scbi$foldID %>% factor())  
  
comp\_dist <- 7.5  
  
growth\_scbi <- growth\_scbi %>%  
 add\_buffer\_variable(size = comp\_dist, region = study\_region\_scbi)  
  
focal\_vs\_comp\_ba <- growth\_scbi %>%  
 create\_focal\_vs\_comp(  
 comp\_dist = comp\_dist,  
 blocks = blocks\_scbi,  
 id = "stemID",  
 comp\_x\_var = "basal\_area"  
 ) %>%  
 run\_cv(comp\_dist = comp\_dist, blocks = blocks\_scbi)  
  
focal\_vs\_comp\_agb <- growth\_scbi %>%  
 create\_focal\_vs\_comp(  
 comp\_dist = comp\_dist,  
 blocks = blocks\_scbi,  
 id = "stemID",  
 comp\_x\_var = "agb"  
 ) %>%  
 run\_cv(comp\_dist = comp\_dist, blocks = blocks\_scbi)  
  
focal\_vs\_comp\_ba %>%  
 rmse(truth = growth, estimate = growth\_hat) %>%  
 pull(.estimate)  
## [1] 0.14  
  
focal\_vs\_comp\_agb %>%  
 rmse(truth = growth, estimate = growth\_hat) %>%  
 pull(.estimate)  
## [1] 2.08

Here we observe that basal area is a better competitor explanatory variable competitor explanatory variables from Equation (1.1) than above ground biomass as suggested by the lower estimated RMSE.

# 8 Appendix: Compare grouping variables

The package also allows the user to specify the categorical explanatory grouping variable. Here we compare two different such variables: species and the potential canopy position of that species. If we had individual-level crown classes (Smith ([1986](#ref-smith_practice_1986)) dominant, codominant, intermediate and suppressed) that could also be used.

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  
 ) %>%  
 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  
 ) %>%  
 filter(gx < 300, between(gy, 300, 600))  
  
growth\_scbi\_sp <-  
 compute\_growth(  
 census\_1 = census\_2013\_scbi,  
 census\_2 = census\_2018\_scbi %>% filter(!str\_detect(codes, "R")),  
 id = "stemID"  
 ) %>%  
 mutate(  
 sp = as.factor(sp),  
 basal\_area = 0.0001 \* pi \* (dbh1 / 2)^2  
 )  
  
growth\_scbi\_can\_pos <-  
 compute\_growth(  
 census\_1 = census\_2013\_scbi,  
 census\_2 = census\_2018\_scbi %>% filter(!str\_detect(codes, "R")),  
 id = "stemID"  
 ) %>%  
 left\_join(sp\_info, by = "sp") %>%  
 mutate(  
 canopy\_position = str\_replace(canopy\_position, " ", "\_"),  
 canopy\_position = str\_replace(canopy\_position, ",", ""),  
 canopy\_position = ifelse(is.na(canopy\_position), "shrub\_layer",   
 canopy\_position),  
 sp = as.factor(canopy\_position),  
 basal\_area = 0.0001 \* pi \* (dbh1 / 2)^2  
 )  
  
study\_region\_scbi <- tibble(  
 x = c(0, 300, 300, 0, 0),  
 y = c(300, 300, 600, 600, 300)  
) %>%  
 sf\_polygon()  
  
n\_fold <- 4  
fold1 <- cbind(c(0, 150, 150, 0), c(300, 300, 450, 450))  
fold2 <- cbind(c(150, 300, 300, 150), c(300, 300, 450, 450))  
fold3 <- cbind(c(0, 150, 150, 0), c(450, 450, 600, 600))  
fold4 <- cbind(c(150, 300, 300, 150), c(450, 450, 600, 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  
spatial\_block\_scbi <-  
 spatialBlock(  
 speciesData = growth\_scbi, k = n\_fold,  
 selection = "systematic", blocks = blocks\_scbi,  
 showBlocks = FALSE, verbose = FALSE  
 )  
  
growth\_scbi\_sp <- growth\_scbi\_sp %>%  
 mutate(foldID = spatial\_block\_scbi$foldID %>% factor())  
growth\_scbi\_can\_pos <- growth\_scbi\_can\_pos %>%  
 mutate(foldID = spatial\_block\_scbi$foldID %>% factor())  
  
comp\_dist <- 7.5  
  
growth\_scbi\_sp <- growth\_scbi\_sp %>%  
 add\_buffer\_variable(size = comp\_dist, region = study\_region\_scbi)  
growth\_scbi\_can\_pos <- growth\_scbi\_can\_pos %>%  
 add\_buffer\_variable(size = comp\_dist, region = study\_region\_scbi)  
  
focal\_vs\_comp\_sp <- growth\_scbi\_sp %>%  
 create\_focal\_vs\_comp(  
 comp\_dist = comp\_dist,  
 blocks = blocks\_scbi,  
 id = "stemID",  
 comp\_x\_var = "basal\_area"  
 ) %>%  
 run\_cv(comp\_dist = comp\_dist, blocks = blocks\_scbi)  
  
focal\_vs\_comp\_can\_pos <- growth\_scbi\_can\_pos %>%  
 create\_focal\_vs\_comp(  
 comp\_dist = comp\_dist,  
 blocks = blocks\_scbi,  
 id = "stemID",  
 comp\_x\_var = "basal\_area"  
 ) %>%  
 run\_cv(comp\_dist = comp\_dist, blocks = blocks\_scbi)  
  
focal\_vs\_comp\_sp %>%  
 rmse(truth = growth, estimate = growth\_hat) %>%  
 pull(.estimate)  
## [1] 0.14  
  
focal\_vs\_comp\_can\_pos %>%  
 rmse(truth = growth, estimate = growth\_hat) %>%  
 pull(.estimate)  
## [1] 0.144

We find that species identity has a lower RMSE, so does a better job. We still however plot the competition posteriors for the canopy position groupings in Figure 8.1. Unsurprisingly we see that canopy and canopy emergent competitors generally have negative effects on their neighbors, while shrubs and understory competitors have neutral or even positive effects.

fit\_mod\_can\_pos <- growth\_scbi\_can\_pos %>%  
 create\_focal\_vs\_comp(  
 comp\_dist = comp\_dist,  
 blocks = blocks\_scbi,  
 id = "stemID",  
 comp\_x\_var = "basal\_area"  
 ) %>%  
 comp\_bayes\_lm(prior\_param = NULL)

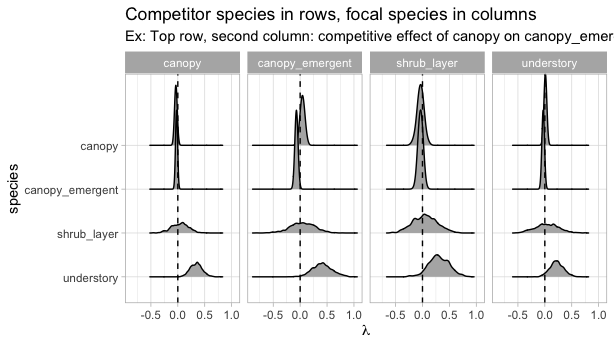


Figure 8.1: Posterior distributions of all competition parameters.

# 9 Appendix: Replicate RMSE comparison

This code replicates Figure 3.1: A comparison of root mean squared error of models for standard, permuted, and spatial cross-validated error estimates.

library(tidyverse)  
library(lubridate)  
library(here)  
library(sf)  
library(viridis)  
library(forestecology)  
library(blockCV)  
library(tictoc)  
  
  
# Compute growth of trees based on census data ------------------------  
census\_2013\_scbi <- here("paper/scbi.stem2.csv") %>%  
 read\_csv() %>%  
 select(stemID, sp, date = ExactDate, gx, gy, dbh, codes, status) %>%  
 mutate(  
 date = mdy(date),  
 dbh = as.numeric(dbh) / 10  
 ) %>%  
 filter(gx < 300, between(gy, 300, 600))  
  
census\_2018\_scbi <- here("paper/scbi.stem3.csv") %>%  
 read\_csv() %>%  
 select(stemID, sp, date = ExactDate, gx, gy, dbh, codes, status) %>%  
 mutate(  
 date = mdy(date),  
 dbh = as.numeric(dbh) / 10  
 ) %>%  
 filter(gx < 300, between(gy, 300, 600))  
  
growth\_scbi <-  
 compute\_growth(  
 census\_1 = census\_2013\_scbi,  
 census\_2 = census\_2018\_scbi %>% filter(!str\_detect(codes, "R")),  
 id = "stemID"  
 ) %>%  
 # Compute basal area:  
 mutate(basal\_area = 0.0001 \* pi \* (dbh1 / 2)^2)  
  
  
# Add spatial information ---------------------------------------------  
# 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)  
  
# Manually define spatial blocks to act as folds  
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))  
n\_fold <- 4  
  
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(  
 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())  
  
# Compute focal versus competitor tree information --------------------  
focal\_vs\_comp\_scbi <- growth\_scbi %>%  
 create\_focal\_vs\_comp(  
 comp\_dist, blocks = blocks\_scbi, id = "stemID",   
 comp\_x\_var = "basal\_area"  
 )  
  
  
# Fit model and make predictions --------------------------------------  
# Number of permutation shuffles:  
num\_shuffle <- 49  
  
# Save results here  
run\_time <- 0  
observed\_RMSE <- 0  
observed\_RMSE\_CV <- 0  
shuffle\_RMSE <- vector("list", 1)  
shuffle\_RMSE\_CV <- vector("list", 1)  
filename <- here("paper/simulation\_results/") %>%  
 str\_c("2021-03-03\_scbi\_", num\_shuffle, "\_shuffles")  
  
# Run all simulations  
# 0. Setup simulation for this species type ----  
# Start clock  
tic()  
  
# 1. Compute observed test statistic: RMSE with no cross-validation ---  
# Fit model (compute posterior parameters)  
comp\_bayes\_lm\_scbi <- focal\_vs\_comp\_scbi %>%  
 comp\_bayes\_lm(prior\_param = NULL, run\_shuffle = FALSE)  
  
# Make predictions and compute RMSE  
observed\_RMSE <- focal\_vs\_comp\_scbi %>%  
 mutate(  
 growth\_hat =  
 predict(comp\_bayes\_lm\_scbi, focal\_vs\_comp\_scbi)  
 ) %>%  
 rmse(truth = growth, estimate = growth\_hat) %>%  
 pull(.estimate)  
  
# 2. Compute observed test statistic: RMSE with cross-validation ------  
observed\_RMSE\_CV <- focal\_vs\_comp\_scbi %>%  
 run\_cv(comp\_dist = comp\_dist, blocks = blocks\_scbi) %>%  
 rmse(truth = growth, estimate = growth\_hat) %>%  
 pull(.estimate)  
  
# 3. Permutation distribution: RMSE with no cross-validation ----------  
# Compute num\_shuffle permutation test statistics  
shuffle\_RMSE <- numeric(length = num\_shuffle)  
for (j in 1:num\_shuffle) {  
 # Fit model (compute posterior parameters) with shuffling  
 comp\_bayes\_lm\_scbi <- focal\_vs\_comp\_scbi %>%  
 comp\_bayes\_lm(prior\_param = NULL, run\_shuffle = TRUE)  
   
 # Make predictions and compute RMSE  
 shuffle\_RMSE[j] <- focal\_vs\_comp\_scbi %>%  
 mutate(  
 growth\_hat = predict(comp\_bayes\_lm\_scbi, focal\_vs\_comp\_scbi)  
 ) %>%  
 rmse(truth = growth, estimate = growth\_hat) %>%  
 pull(.estimate)  
}  
  
# 4. Permutation distribution: RMSE with cross-validation -------------  
# Compute num\_shuffle permutation test statistics  
shuffle\_RMSE\_CV <- numeric(length = num\_shuffle)  
  
# Compute num\_shuffle permutation test statistics  
for (j in 1:num\_shuffle) {  
 # Compute and save RMSE  
 shuffle\_RMSE\_CV[j] <- focal\_vs\_comp\_scbi %>%  
 run\_cv(  
 comp\_dist = comp\_dist, blocks = blocks\_scbi,  
 run\_shuffle = TRUE  
 ) %>%  
 rmse(truth = growth, estimate = growth\_hat) %>%  
 pull(.estimate)  
   
 # Status update  
 str\_c("Shuffle with permutation ", j, " at ", Sys.time()) %>%   
 print()  
}  
  
# 5. Save results ----  
clock <- toc(quiet = TRUE)  
run\_time <- clock$toc - clock$tic  
  
model\_comp\_tbl <- tibble(  
 run\_time = run\_time,  
 observed\_RMSE = observed\_RMSE,  
 observed\_RMSE\_CV = observed\_RMSE\_CV,  
 shuffle\_RMSE = shuffle\_RMSE,  
 shuffle\_RMSE\_CV = shuffle\_RMSE\_CV,  
)  
save(model\_comp\_tbl, file = filename %>% str\_c(".RData"))  
  
  
# Visualize results ---------------------------------------------------  
model\_comp <- bind\_rows(  
 model\_comp\_tbl %>%  
 select(run\_time,  
 observed = observed\_RMSE,  
 shuffle = shuffle\_RMSE  
 ) %>%  
 mutate(CV = FALSE),  
 model\_comp\_tbl %>%  
 select(run\_time,  
 observed = observed\_RMSE\_CV,  
 shuffle = shuffle\_RMSE\_CV  
 ) %>%  
 mutate(CV = TRUE)  
) %>%  
 gather(type, RMSE, -c(run\_time, CV))  
  
model\_comp\_observed <- model\_comp %>%  
 filter(type == "observed") %>%  
 unnest(cols = c(RMSE))  
model\_comp\_shuffle <- model\_comp %>%  
 filter(type == "shuffle") %>%  
 unnest(cols = c(RMSE))  
  
cv\_plot <- ggplot() +  
 geom\_vline(  
 data = model\_comp\_observed,  
 aes(xintercept = RMSE, col = CV),  
 linetype = "dashed", show.legend = F  
 ) +  
 geom\_histogram(  
 data = model\_comp\_shuffle,  
 aes(x = RMSE, fill = CV), bins = 50  
 ) +  
 labs(  
 fill = "Cross-validated?",  
 x = expression(paste("RMSE (cm ", y^{-1}, ")"))  
 ) +  
 scale\_color\_viridis(discrete = TRUE, option = "D") +  
 scale\_fill\_viridis(discrete = TRUE) +  
 theme\_light()  
cv\_plot  
  
filename %>%  
 str\_c(".pdf") %>%  
 ggsave(plot = cv\_plot, width = 16 / 2, height = 9 / 2)

1. See S1 Appendix of Allen and Kim ([2020](#ref-allen_permutation_2020)), available at <https://doi.org/10.1371/journal.pone.0229930.s004> [↑](#footnote-ref-24)