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

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

March 4, 2021

Abstract

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

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

the Bayesian linear regression model. These three facts allow for clear articulation of all the steps in the sequence of analysis and easy wrangling and visualization of the geospatial forestry data. Furthermore, while the package only has Bayesian linear regression implemented, the package was designed with extensibility to other methods in mind.

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

1 Introduction

BERT: Dave could you write a few sentences (and add references where appropriate) starting with the big picture of modeling tree growth, zooming in to neighhorhood models, and then connecting to the "Allen & Kim (2020) considers the following model" I write below?

Repeat-censused forest plots offer excellent data to test neighborhood models of the effect of competition on the growth of trees Canham et al. (2006) Uriarte et al. (2004).

Allen & Kim (2020) considers the following model: 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 model the average annual growth in diameter y_{ij} (in centimeters per year) of the i^{th} tree of focal species group j as a linear model f of the covariates $\vec{x}_{ij}$

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

where $\beta_{0,j}$ is the diameter-independent growth rate for group j; dbh_{ij} is the diameter at breast height (in centimeters) of the focal tree at the earlier census; $\beta_{\text{dbh},j}$ is the amount of the growth rate changed depending on diameter for group j; BA_{ijk} is the sum of the basal area of all trees of competitor species group k; λ_{jk} is the change in growth for individuals of group j from nearby competitors of group k; and ϵ_{ij} is a random error term distributed Normal$(0, \sigma^2)$. They estimate all parameters via Bayesian linear regression while exploiting Normal/Inverse Gamma conjugacy to derive closed-form solutions to all posterior distributions via linear algebra[1]. These closed-form solutions for the posterior distributions are in contrast to approximations of all posteriors via computationally expensive Markov Chain Monte Carlo algorithms.

In order to evaluate whether competitor species identity matters, Allen & Kim (2020)

[1]See S1 Appendix of Allen & Kim (2020), available at `https://doi.org/10.1371/journal.pone.0229930.s004`

run a permutation test where under the null hypothesis the species identity of all competitors of a focal tree can be permuted/shuffled:

$$H_0 : \lambda_{jk} = \lambda_j \text{ for all } k = 1, \ldots, K \tag{2}$$

$$\text{vs.} \qquad H_A : \text{at least one } \lambda_{jk} \text{ is different} \tag{3}$$

where the null hypothesis H_0 reflects a hypothesis of no species grouping-specific effects of competition while the alternative hypothesis H_A reflects a hypothesis of species grouping-specific effects of competition. Furthermore, in order to account for the spatial autocorrelation inherent to forest data in their estimates of out-of-sample model error, Allen & Kim (2020) use spatial cross-validation. Estimates of model error that do not account for this spatial dependency tend to underestiamte the true model error (Roberts et al. 2017).

We introduce the `forestecology` R package providing methods and data for forest ecology model fitting and assessment, available on CRAN (`https://cran.r-project.org/web/packages/forestecology/index.html`) with the corresponding source code available on GitHub (`https://github.com/rudeboybert/forestecology`). The package implements all aspects of the model in Equation 1: model fitting and generating fitted/predicted values, evaluating the effect of competitor species identity using permutation tests, and evaluating model performance using spatial cross-validation.

The package designed with "tidy" design principles in mind (Wickham et al. 2019). Much like many of the `tidyverse` component packages, `forestecology` is designed with verb-named functions that can be modularly composed in sequence using the pipe `%>%` operator (Bache & Wickham 2020). As we articulate in Section 2, these functions delineate the key steps in our analysis sequence. Furthermore, the inputs and outputs of nearly all of our functions use the same "simple features for R" data structures as implemented in

the `sf` package for standardized support for spatial vector data (Pebesma 2018). The `sf` package is a `tidyverse`-friendly evolution of the `sp` package of classes and methods for spatial data in R (Pebesma & Bivand 2005). As such, wrangling and visualization spatial data such as ours becomes much easier.

Currently the package only implements the Bayesian linear regression model of tree growth based on neighborhood competition detailed in Equation 1. As we demonstrate in Section 2.4 however, the fitting of this model is self-contained in a single function `comp_bayes_lm()`. This function returns an object of S3 class type `comp_bayes_lm` with generic methods implemented for `print()` to inspect the output, `predict()` to generate fitted/predicted values, and `ggplot2::autoplot()` to visualize all results. Therefore the package can be modularly extended to fit other models as long as they are coded into a function similar type as `comp_bayes_lm()` as has equivalent generic methods implemented.

We present a case-study of the `forestecology` package's use on data from the Smithsonian Conservation Biology Institute's (SCBI) large forest dynamics plot in Front Royal, Virginia, USA in Section 2, which is part of the ForestGEO global network of research sites (Bourg et al. 2013, Anderson-Teixeira et al. (2015)). The package is designed with ForestGEO plot data in mind, 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 or more generally to model interactions of any community of mapped sessile organisms (Smith 2002).

2 forestecology workflow: a case study

We demonstrate the `forestecology` package's functionality on data from the Smithsonian Conservation Biology Institute (SCBI) large forest dynamics plot, located at the Smithsonian's National Zoo and Conservation Biology Institute in Front Royal, VA, USA (Bourg et al. 2013). The 25.6 ha (640 x 400 m) plot is located at the intersection of three of

the major physiographic provinces of the eastern US—the Blue Ridge, Ridge and Valley, and Piedmont provinces—and is adjacent to the northern end of Shenandoah National Park. The forest type is typical mature secondary eastern mixed deciduous forest, with a canopy dominated by tulip poplar (*Liriodendron tulipifera*), oaks (*Quercus* spp.), and hickories (*Carya* spp.), and an understory composed mainly of spicebush (*Lindera benzoin*), paw-paw (*Asimina triloba*), American hornbeam (*Carpinus caroliniana*), and witch hazel (*Hamamelis virginiana*) (Bourg et al. 2013).

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

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 fitted/predicted values.
    3. Visualize posterior distributions.

We start by loading all necessary packages.

```
library(tidyverse)
library(lubridate)
library(sf)
```

```
library(patchwork)
library(forestecology)
library(blockCV)
```

2.1 Step 1: Compute the growth of trees based on census data {compute-growth}

The first step in the our analysis sequence is to compute the growth of trees using data from two censuses. The `compute_growth()` function computes average annual growth assuming census data that roughly follows ForestGEO standards. Despite such standards, minor variations will still exist between sites, thereby necessitating some data wrangling and checking. For example, the SCBI site records all diameters at breast height (DBH) in millimeters (Bourg et al. 2013), whereas the Michigan Big Woods site records them in centimeters (Allen et al. 2020).

We load both 2008 and 2014 SCBI census data `.csv` files as they existed on GitHub on November 20, 2020 (Gonzalez-Akre et al. 2020). After selecting only the relevant variables, we perform a few additional data wrangling steps: convert the character variable with the date of measurement to be of explicit type `date`, convert DBH to be in centimeters[2] Furthermore, in order to speed up computation for purposes of this example, we only consider a 9 ha subsection of the 25.6 ha of the SCBI site: `gx` from 0–300 instead of 0–400 and `gy` from 300–600 instead of 0–640.

```
census_2013_scbi <- read_csv("scbi.stem2.csv") %>%
  select(stemID, sp, date = ExactDate, gx, gy, dbh, codes, status) %>%
  mutate(
```

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

```
    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))
```

These two data frames are then supplied as arguments to the `compute_growth()` function, along with the `id` argument that specifies the variable that uniquely identifies each tree-stem. Note furthermore that we discard all resprouts in the later census (those with `code == R`), since we are only interested in the diameter 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
## Simple feature collection with 7954 features and 8 fields
## geometry type:  POINT
```

```
## dimension:       XY
## bbox:            xmin: 0.2 ymin: 300 xmax: 300 ymax: 600
## CRS:             NA
## # A tibble: 7,954 x 9
##   stemID sp     dbh1 codes1 status  dbh2 codes2 growth
##    <dbl> <fct> <dbl> <chr>  <chr>  <dbl> <chr>   <dbl>
## 1      4 nysy  13.6  M      A       14.2 M       0.103
## 2      5 havi   8.8  M      A        9.6 M;P     0.150
## 3      6 havi   3.25 NULL   A        4   M       0.140
## 4     77 qual  65.2  M      A       66   M       0.141
## 5     79 tiam  47.7  M      A       46.8 M      -0.161
## # ... with 7,949 more rows, and 1 more variable: geometry <POINT>
```

The output `growth_scbi` is a single data frame of class `sf` that includes variables `growth`, the average annual growth in DBH (cm y^{-1}) for all stems that were alive at both time points, and `geometry`, the `sf` package's encoding of geolocations of type <`POINT`>. In addition the species variable `sp` is converted to a factor if it wasn't already by `compute_growth()`.[3] Furthermore, the variables that should 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`.

Site data that does not align with this convention will need to be transformed for use with the `compute_growth()` function. However, in the end, all that matters is that the growth of all stems is saved in a data frame of class `sf` and, at a minimum, contains the variable uniquely identifying each stem, `sp`, `dbh1`, `growth`, `geometry`.

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

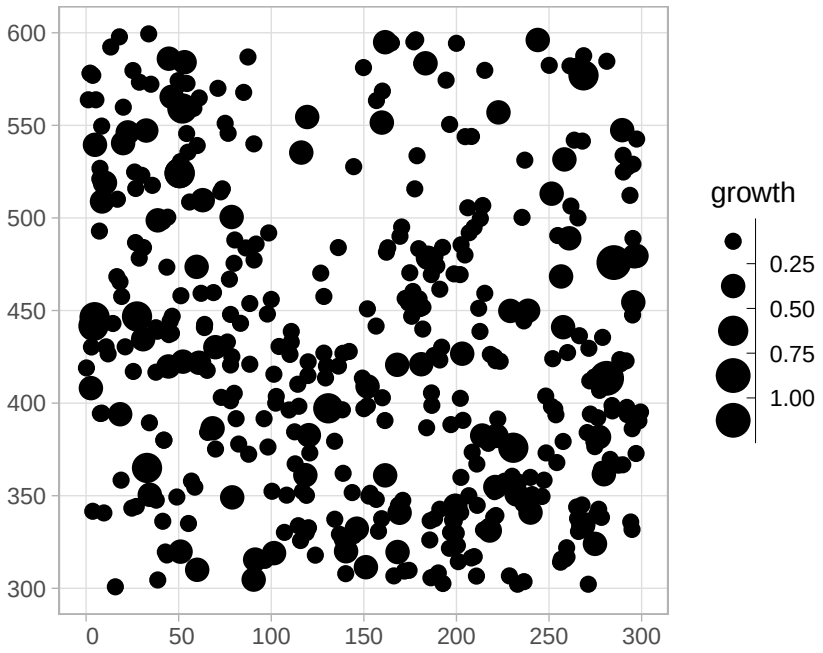

Figure 1: Compute growth of trees based on census data: Map with growth of a random sample of 500 trees from a 9 ha subsection of the Smithsonian Conservation Biology Institute (SCBI) forest plot.

Given that `growth_scbi` is of class `sf`, it can be easily plotted in `ggplot2` using the `geom_sf()` geometry as seen in Figure 1 where we plot a random sample of 500 out of the 7954 trees.

```
ggplot() +
  geom_sf(data = growth_scbi %>% sample_n(500), aes(size = growth)) +
  scale_size_binned(limits = c(0.1, 1))
```

2.2 Step 2: Add spatial information

The next step in our analysis sequence is to add additional spatial information to our main `growth_scbi` data frame. The first element of spatial information we add is a "buffer region" to the periphery of the study region. Since some of our model's explanatory variables are cumulative (such as competitor basal area), we must ensure that all trees being modeled

are not biased to have different neighbor structures. This is of concern for trees at the boundary of study regions, for which all neighbors will not be included in the censused stems. In order to account for such edge effects, only trees that are not part of this buffer region, i.e. are part of the interior of the study region, will have their growth modeled (Waller & Gotway 2004).

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

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


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


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

The second element of spatial information are blocks corresponding to folds of a spatial cross-validation algorithm used to estimate out-of-sample model error. Conventional cross-validation algorithms assign observations to folds by randomly resampling individual observations. However, many of these algorithms assume that the observations are independent of each other. In the case of forest census data, observations exhibit spatial autocorrelation. We therefore incorporate this spatial dependence into the cross-validation algorithm with our spatial blocks of trees (Roberts et al. 2017, Pohjankukka et al. (2017)) In the example below, we first manually define four folds that partition the study region as an `sf` object. We then use the output of the `spatialBlock()` function from the `blockCV` package to associate each tree in `growth_scbi` to the correct fold (saved in the `foldID` variable) (Valavi et al. 2019). [4]

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

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

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

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

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


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

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

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

2.3 Step 3: Identify all focal and corresponding competitor trees

The next step in our analysis sequence is to identify all focal trees and their corresponding competitor trees. More specifically, it identifies all trees that are not part of the buffer region, have a valid `growth` measurement, and have at least one neighbor within 7.5 of it. The `create_focal_vs_comp()` functions performs these tasks and returns a new data frame of type `sf`. On top of the previous arguments `comp_dist` defining the competition neighborhood and `id` indicating which variable uniquely identifies each tree-stem, this function also requires an `sf` object representation of the spatial cross-validation blocks/folds. In this example, the blocks were manually encoded in `blocks_scbi` by specifying it's vertices in

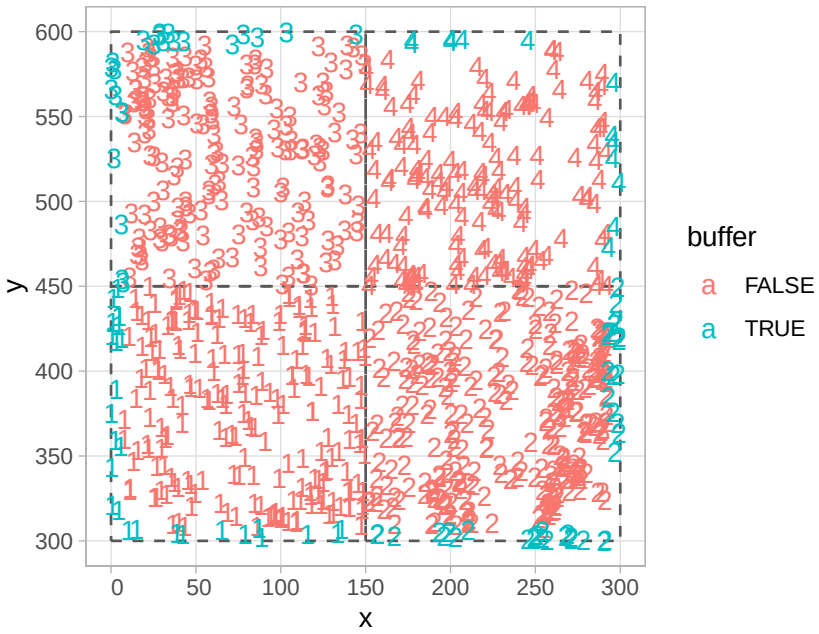

Figure 2: Add spatial information: Buffer region and spatial cross-validation blocks (1 through 4). All trees in the interior of the study region (i.e. not part of buffer) will be the focal trees whose growth will be modeled.

Section 2.2[5]. We present the resulting data frame below with the `foldID` variable omitted for compactness of presentation.

```
focal_vs_comp_scbi <- growth_scbi %>%
  create_focal_vs_comp(comp_dist, blocks = blocks_scbi, id = "stemID")
focal_vs_comp_scbi %>%
  select(-foldID)
## # A tibble: 6,296 x 6
##   focal_ID focal_sp   dbh   geometry growth comp
##      <dbl> <fct>    <dbl>    <POINT>  <dbl> <list>
## 1        4 nysy      13.6  (14.2 428)  0.103 <tibble [20 x 4]>
## 2        5 havi       8.8   (9.4 436)  0.150 <tibble [32 x 4]>
## 3       79 tiam      47.7    (40 381) -0.161 <tibble [20 x 4]>
```

[5]We present an alternative method for defining spatial cross-validation blocks is using the `spatialBlock()` function from the `blockCV` package in the Supporting Information.

```
## 4       80 caca      5.15 (38.7 422)  0.253 <tibble [12 x 4]>
## 5       96 libe      2.3    (60 310)  0.262 <tibble [14 x 4]>
## # ... with 6,291 more rows
```

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

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

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

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

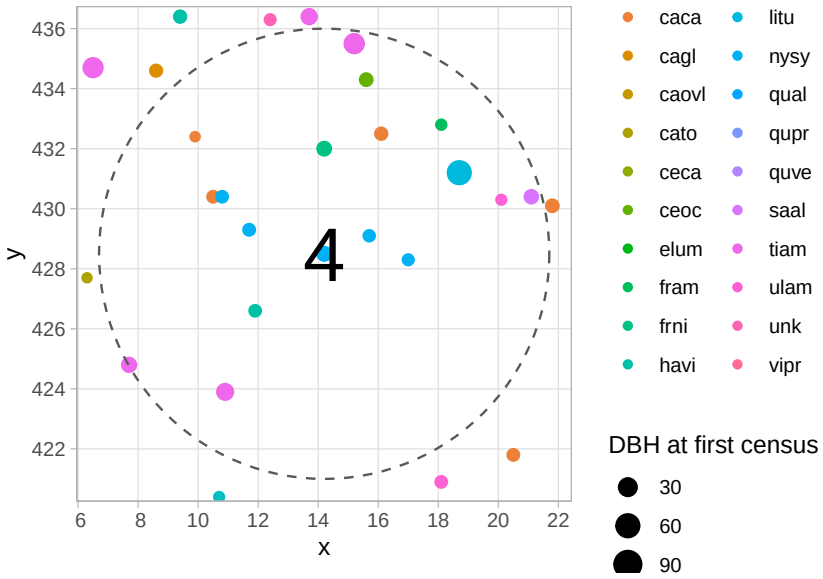

Figure 3: Identify all focal and corresponding competitor trees: All 20 competitor trees of focal tree 4.

the focal tree 20 times.

```
focal_vs_comp_scbi %>%
  filter(focal_ID == 4) %>%
  select(focal_ID, dbh, comp) %>%
  unnest(cols = "comp")
## # A tibble: 20 x 6
##   focal_ID   dbh comp_ID  dist comp_sp comp_basal_area
##      <dbl> <dbl>   <dbl> <dbl> <fct>             <dbl>
## 1        4  13.6    1836  7.48 tiam            0.0176
## 2        4  13.6    1847  2.81 nysy            0.00332
## 3        4  13.6    1848  1.62 nysy            0.00396
## 4        4  13.6    1849  2.62 nysy            0.00535
## 5        4  13.6    1850  2.98 havi            0.00472
## # ... with 15 more rows
```

2.4 Step 4: Fit model

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

```
comp_bayes_lm_scbi <- focal_vs_comp_scbi %>%
  comp_bayes_lm(prior_param = NULL)
```

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

```
comp_bayes_lm_scbi
## Bayesian linear regression model parameters with a multivariate Normal likelihood.
##
##   parameter_type           prior posterior
## 1 Inverse-Gamma on sigma^2 a_0   a_star
## 2 Inverse-Gamma on sigma^2 b_0   b_star
## 3 Multivariate t on beta   mu_0  mu_star
## 4 Multivariate t on beta   V_0   V_star
##
```

```
## Model formula:
## growth ~ sp + dbh + dbh * sp + acne * sp + acru * sp + amar * sp + astr * sp + caco
```

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

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

```
focal_vs_comp_scbi
## # A tibble: 6,296 x 8
##   focal_ID focal_sp   dbh foldID   geometry growth comp
##      <dbl> <fct>    <dbl> <fct>     <POINT>  <dbl> <list>
## 1        4 nysy      13.6  1      (14.2 428)  0.103 <tibble [20 x 4]>
## 2        5 havi       8.8  1       (9.4 436)  0.150 <tibble [32 x 4]>
## 3       79 tiam      47.7  1        (40 381) -0.161 <tibble [20 x 4]>
## 4       80 caca       5.15 1      (38.7 422)  0.253 <tibble [12 x 4]>
## 5       96 libe       2.3  1        (60 310)  0.262 <tibble [14 x 4]>
## # ... with 6,291 more rows, and 1 more variable: growth_hat <dbl>
```

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

```
model_rmse <- focal_vs_comp_scbi %>%
  rmse(truth = growth, estimate = growth_hat) %>%
  pull(.estimate)
```

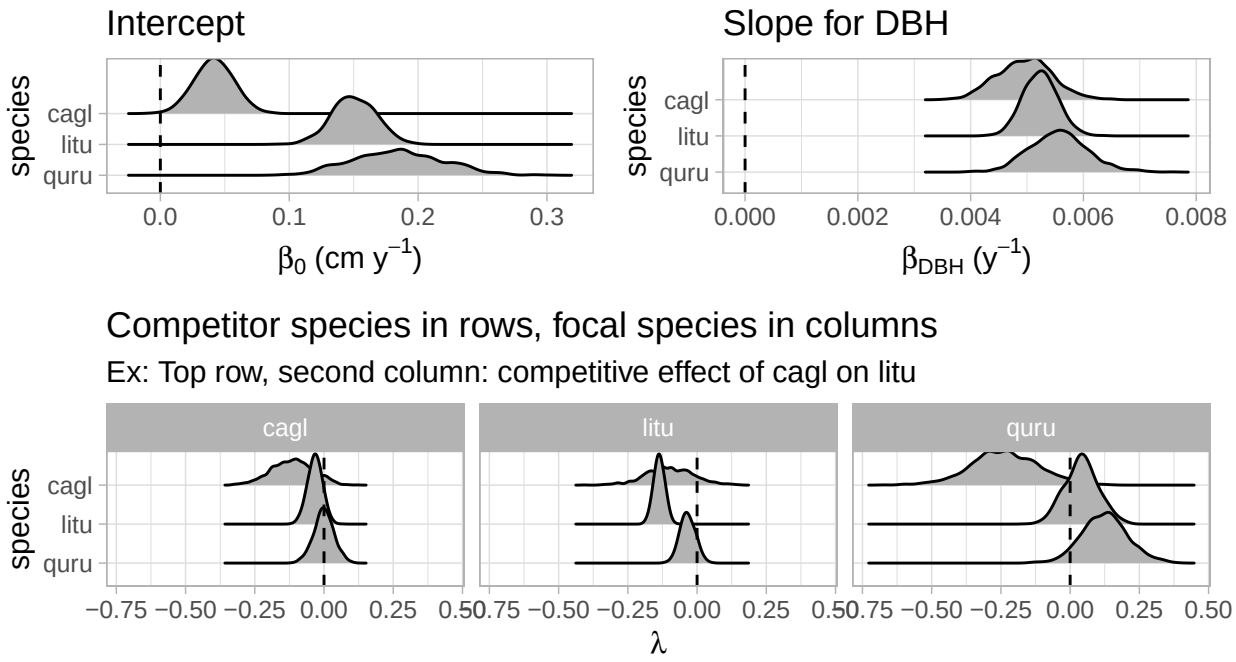

Figure 4: Fit model: Posterior distributions of all parameters for three species.

```
model_rmse
```

```
## [1] 0.128
```

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

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


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


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

These plots give the posterior distributions of parameters from Equation 1. For many

package users they will be of interest because they give insight into the species-specific competitive interactions. Setting `type = "intercepts"` gives posterior distributions for $\beta_{0,j}$ and `type = "dbh_slopes"` for $\beta_{dbh,j}$. These give species specific growth independent of competition. The values of more interest are plotted with `type = 'competition'` which gives the posterior distribution for $\lambda_{j,k}$ species-specific competition coefficients (i.e., the λ-matrix). Negative values indicate a competitor species which slows the growth of a focal species. Here, for example, we see that tulip trees (`litu`) have a strong negative effect on the growth of conspecifics but relatively little effect on neighbors of the other two species.

2.5 Evaluate the effect of competitor species identity using permutation tests

In order to evaluate the effect of competitor species identity, we use the four steps of our analysis sequence answer along with a permutation test: Under a null hypothesis where competitor species identity does not matter, we can permute/shuffle this variable within each focal tree, compute the RMSE (the test statistic of interest), repeat this process several times to construct a null distribution of the RMSE, and compare it to the observed RMSE to assess significance. Going back to our example in Section 2.3 of focal tree with `focal_ID` 4 and its 20 competitors, the permutation test randomly resamples the `comp_sp` variable with replacement, leaving all other variables intact. The resampling with replacement is nested within each focal tree in order to preserve the neighborhood structure of our competition model. To run the permutation test, we use the same `comp_bayes_lm()` function as in Section 2.4, but with a `run_shuffle = TRUE` argument.

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


focal_vs_comp_scbi <- focal_vs_comp_scbi %>%
```

```
  mutate(growth_hat_shuffle = predict(comp_bayes_lm_scbi_shuffle, newdata = focal_vs_cor
```

```
model_rmse_shuffle <- focal_vs_comp_scbi %>%
  rmse(truth = growth, estimate = growth_hat_shuffle) %>%
  pull(.estimate)
model_rmse_shuffle
## [1] 0.131
```

The resulting RMSE of 0.131 based on the permutation test is larger than the earlier RMSE of 0.128, suggesting that models that do incorporate competitor species identity better fit the data. We conduct an analysis of the full SCBI plot in Section 3 below.

2.6 Evaluate model performance using spatial cross-validation

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

To this end, we use the `foldID` variable defined in Section 2.2 whereby all focal trees are assigned to one of 4 spatially contiguous blocks that act as folds in our cross-validation routine. Figure 5 presents a schematic illustrating this scheme for fold 1 (bottom-left) as the test set and folds 2, 3, and 4 as the training sets. We fit the model to all focal trees in the training set, apply the model to all focal trees in the test set to compute fitted/predicted values, and compute the RMSE of the observed versus predicted growths. We repeat this procedure 3 more times with each of the three remaining folds acting as

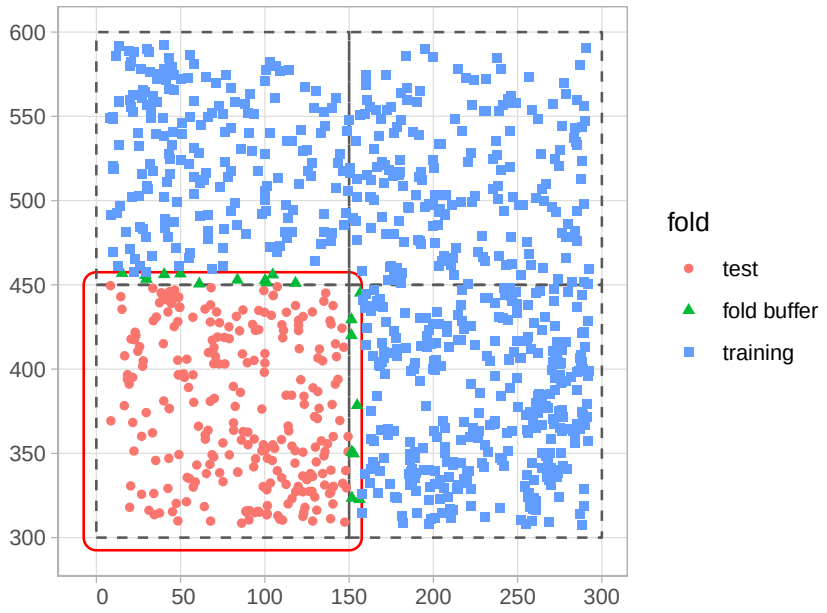

Figure 5: Schematic of spatial cross-validation: Using the k = 1 fold as the test set, assigning each focal tree to training set, test set, and fold buffer.

the test set and then average all four resulting RMSE's. Furthermore, in order to maintain spatial independence between the test and training set, a fold buffer that extend outwards from the boundary of the test set is computed; all trees falling within this fold buffer are excluded from the training set.

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

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

```
model_rmse_cv <- focal_vs_comp_scbi %>%
  rmse(truth = growth, estimate = growth_hat) %>%
  pull(.estimate)
model_rmse_cv
```

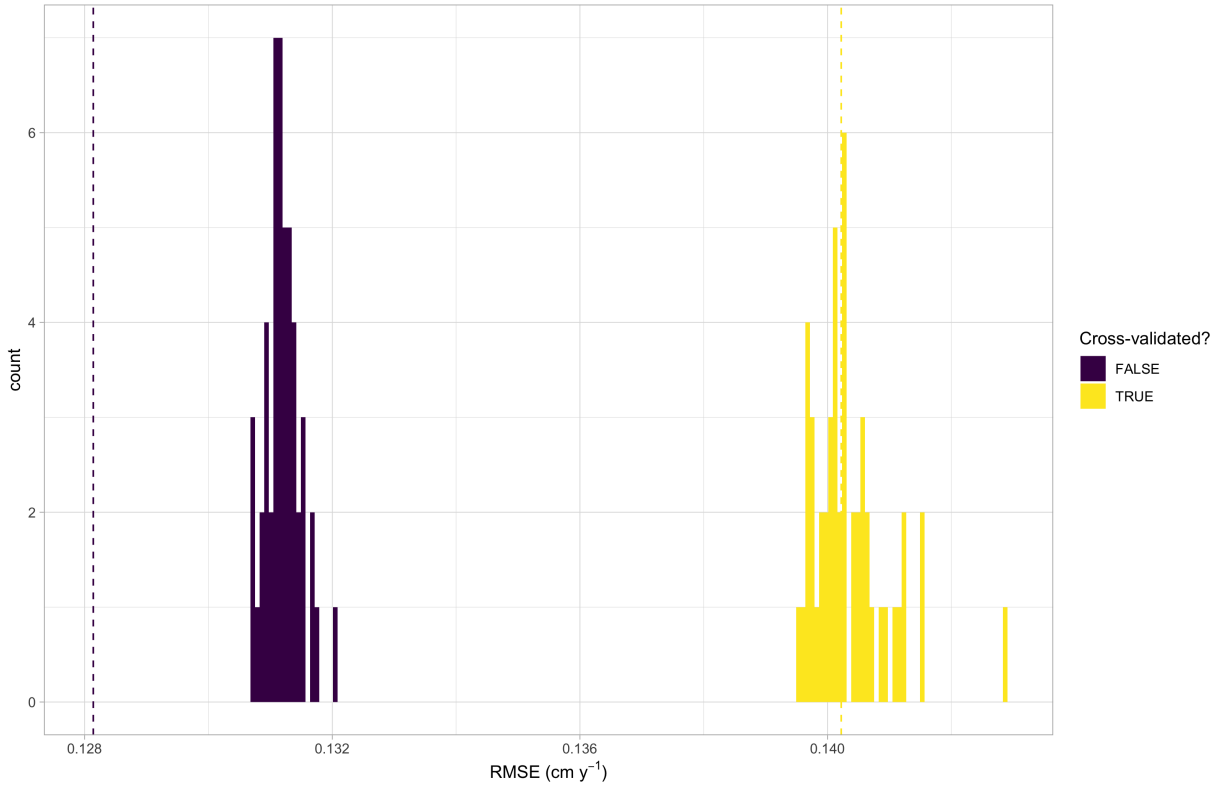

Figure 6: Root mean squared error of models for standard, permuted, and spatial cross-validated error estimates.

```
## [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 take the inherent spatial autocorrelation of the data into account generate error estimates that are overly optimistic; in our case RMSE's that are too low. We conduct an analysis of the full SCBI plot in Section 3 below.

3 Results

Figure 6

4 Discussion

The `forestecology` package provides

5 Acknowledgments

The authors thank Ryan Giordano and Jonathan Che for their help with the statistical methodology and Sophie Li for their feedback on package interface. The authors declare no conflicts of interest.

6 Author's contributions

AYK and DNA conceived the ideas and coded most of the package in the early design stages. AYK led the writing of the manuscript. SPC refactored much of the package's code in the later design stages to align with R and "tidy" programming best practices (Wickham et al. 2019). All authors contributed critically to the drafts and gave final approval for publication.

7 Data accessibility

We intend to archive all data and source code for the `forestecology` package as well as this manuscript on GitHub at `https://github.com/rudeboybert/forestecology`. This repository will be versioned and archived on Zenodo upon acceptance. The 2008 and 2014 Smithsonian Conservation Biology Institute census data loaded in Section **??** and saved in `scbi.stem2.csv` and `scbi.stem3.csv` are available on GitHub at `https://github.com/SCBI-ForestGEO/SCBI-ForestGEO-Data/tree/master/tree_main_census/data/census-csv-files` and have been versioned and archived on Zenodo at `https://doi.org/10.5281/zenodo.2649301` (Gonzalez-Akre et al. 2020).

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*

Bache, S. M. & Wickham, H. (2020), *magrittr: A Forward-Pipe Operator for R*. R package version 2.0.1.
URL: *https://CRAN.R-project.org/package=magrittr*

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*

Pebesma, E. J. & Bivand, R. S. (2005), 'Classes and methods for spatial data in R', *R News* **5**(2), 9–13.

URL: *https://CRAN.R-project.org/doc/Rnews/*

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

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

URL: *http://onlinelibrary.wiley.com/doi/abs/10.1111/ecog.02881*

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

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

URL: *http://besjournals.onlinelibrary.wiley.com/doi/abs/10.1111/j.0022-0477.2004.00867.x*

Valavi, R., Elith, J., Lahoz-Monfort, J. J. & Guillera-Arroita, G. (2019), 'blockCV: An r package for generating spatially or environmentally separated folds for k-fold cross-validation of species distribution models', *Methods in Ecology and Evolution* **10**(2), 225–

232. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/2041-210X.13107.

URL: *http://besjournals.onlinelibrary.wiley.com/doi/abs/10.1111/2041-210X.13107*

Waller, L. A. & Gotway, C. A. (2004), *Applied Spatial Statistics for Public Health Data*, John Wiley & Sons, Incorporated, Hoboken, UNITED STATES.

URL: *http://ebookcentral.proquest.com/lib/smith/detail.action?docID=214360*

Wickham, H. (2020), *tidyr: Tidy Messy Data*. R package version 1.1.2.

URL: *https://CRAN.R-project.org/package=tidyr*

Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., François, R., Grolemund, G., Hayes, A., Henry, L., Hester, J., Kuhn, M., Pedersen, T. L., Miller, E., Bache, S. M., Müller, K., Ooms, J., Robinson, D., Seidel, D. P., Spinu, V., Takahashi, K., Vaughan, D., Wilke, C., Woo, K. & Yutani, H. (2019), 'Welcome to the Tidyverse', *Journal of Open Source Software* **4**(43), 1686.

URL: *https://joss.theoj.org/papers/10.21105/joss.01686*