Occupancy Models and the Importance of Accounting for Imperfect Detection When Estimating the Probability of Blister Rust Infection in Whitebark Pine Trees in the Greater Yellowstone Ecosystem

Rachel Rebecca Wyand

Department of Mathematical Sciences
Montana State University

December 14, 2019

A writing project submitted in partial fulfillment
of the requirements for the degree

Master of Science in Statistics

APPROVAL

of a writing project submitted by

Rachel Rebecca Wyand

This writing project has been read by the writing project advisor and has been found to be satisfactory regarding content, English usage, format, citations, bibliographic style, and consistency, and is ready for submission to the Statistics Faculty.

12/14/19 Katie Banner

Date

Katharine M. Banner
Writing Project Advisor

Date

Mark C. Greenwood
Writing Project Coordinator

CONTENTS

ABSTRACT

The motivation for this project involves an ongoing study of monitoring whitebark pine trees (*Pinus albicaulis*) for blister rust infection in in the Greater Yellowstone Ecosystem (GYE). Occupancy, whether or not a tree is infected with blister rust, is one indicator that has been used to monitor whitebark pine trees. One objective during this study investigated the relationship between the probability of infection and elevation for each 4-year time period surveyed (2004 – 2007, 2008 – 2011, and 2012 – 2015) using occupancy models, which accounted for imperfect detection in the form of false negatives (failing to detect blister rust infection when the tree really was infected) when estimating the probability of infection. It was found that the probability of infection decreased as elevation increased, when all other covariates were at their average values. The focus of this project concerns only the data collected for the most recent panel of data (2016 – 2018) with the aim of investigating whether the relationship between elevation and infection probability has changed. The occupancy model used in the previous analysis required the use of the Bayesian framework, which is beyond the scope of this paper. Therefore, the results presented herein are naïve and represent the beginning of our investigation into the research question of interest. Another aim of this project includes explaining and describing occupancy models, incorporating information about their study design, assumptions, and the statistical model itself. In turn, this gives rise to a simulation study to highlight the importance of accounting for imperfect detection, a defining feature of occupancy models.

ACKNOWLEDGMENTS

We thank Erin Shanahan and Kathi Irvine for providing the motivating idea for this project as well as the data and answering any questions related to the whitebark pine monitoring program and data. We also thank the Greater Yellowstone Inventory and Monitoring Network field staff who spent numerous hours collecting the data used in the analysis.

1 INTRODUCTION

White pine blister rust is a disease caused by the fungal pathogen *Cronartium ribicola*, a fungus indigenous to Asia that attacks North American white pines. Despite the strict conditions required in order for it to successfully propagate, it has been very successful in its introduction to the Pacific Northwest and tree species in this region have faced high mortality as a result (GYWPMWG, 2011). The fungal spores of blister rust establish themselves and then the fungus infects a tree either by entering the stomata of the pine needles or through a lesion of some sort (as shown in Figure 1 to the right; Shanahan et al., 2016). From where it has established itself in the tree, it continues to grow into the branch and damages through the cambium layer of the bark causing cankers to form and preventing the flow of nutrients through these areas (see Figure 1). Depending on where the infection occurs on the tree, the branch of a tree may be the only casualty it suffers. However, if the infection is able to continue to spread and a canker forms on the trunk, the most common results are the death of the tree and/or the prevention of the tree from producing cones. The time it takes for a blister rust infection to be lethal can depend on a multitude of factors that are related to the severity of the infection (GYWPMWG, 2011).

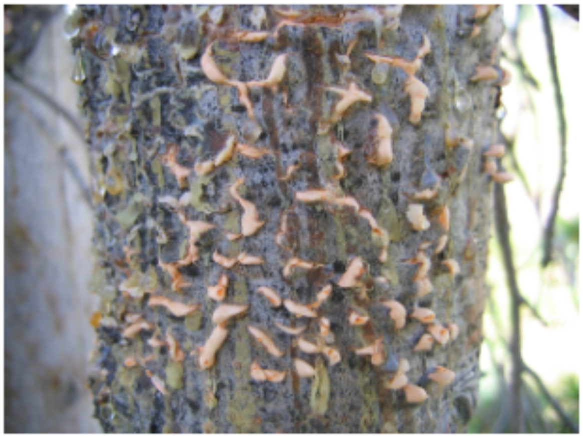

Figure 1. Photo of a Blister Rust Infected Whitebark Pine Tree. Photo provided by Erin Shanahan.

Whitebark pine (*Pinus albicaulis*) receives particular attention because it is a keystone species that is vulnerable to blister rust. Keystone species have the ability to potentially change local climate attributes allowing for other species to move in. Whitebark pine thrive in harsh environments and bring species diversity to areas where conditions were previously too harsh. Moreover, the seeds of whitebark pine trees are valuable to various wildlife species since they are a high-fat food source. In the Greater Yellowstone Ecosystem (GYE), it has been shown that higher survival and better fitness of grizzly bears (*Ursus arctos horribilis*) is anticipated when the production of whitebark pine cones is more plentiful (GYWPMWG, 2011). It is for these reasons that a monitoring program has been put into place to monitor whitebark pine in the GYE. Multiple agencies (Greater Yellowstone Coordinating Committee, USDA Forest Service, USDI National Park Service – NPS, USDI Geological Survey, and Montana State University) are a part of this program but this paper will focus on the goals of the NPS, where the main overarching goal is to remain aware of how its infection status and potential drivers of infection are changing over time (Wright and Irvine, 2017).

One of the ecological indicator variables of interest for how whitebark pine will be monitored is occupancy, defined in this case as whether or not a tree is infected with blister rust. A challenge for estimating occupancy is that the data are detection/non-detection, and not presence/absence. In other words, the observation process is not perfect. For instance, if an observer classifies a tree as not being infected with blister rust, that does not necessarily mean the tree is not infected; it could simply mean that the observer failed to detect the infection. Therefore, including the probability of a tree being *detected* as infected given the tree is infected (denoted p) is a crucial component to the modeling process so that reliable estimates of occupancy probability (denoted ψ) are obtained. Wright and Irvine (2017) used occupancy models for many objectives, two of which were: 1) They estimated the prevalence of blister rust infection in whitebark pine trees in the GYE for 3 different time periods, where a full survey of the GYE takes 1 time period of 4 years (2004 – 2007, 2008 – 2011, and 2012 – 2015). 2) They investigated the relationship between elevation and probability of infection within each time period. They found that the probability of infection decreased as elevation increased for the three different time periods analyzed.

Motivation for this paper comes from researchers wanting to see if this relationship has changed due to the changing climatic conditions in recent years. Thus, we set out to investigate the relationship between the probability of infection of blister rust and elevation in the most recent panel of data (2016 – 2018 as 2019 data are not available yet), with the goal of making comparisons to what was found in the previous three panels. For the sake of comparisons, the ultimate aim is to use the same models from the initial analysis with the new data to address this question. Only naïve estimates were able to be obtained for this paper due to some of the methods needed being beyond the scope of this paper. Regardless, addressing the above research question required learning about occupancy models. Therefore, an equally important goal of this paper is to clearly explain the nuances of occupancy models, which includes highlighting the importance of accounting imperfect detection.

2 OCCUPANCY MODELS

2.1 STUDY DESIGN

In order to estimate occupancy, this first requires determining the level at which occupancy is to be determined –the sample unit– referred to as a site or patch (MacKenzie et al., 2002). The probability of occupancy is the probability that a species is present at the i^{th} site, denoted ψ_i (where i = 1, 2, …, n). The true status or latent state variable of occupancy (denoted as Z_i) represents presence (or absence) at site i: $Z_i = 1$ if a site is occupied (i.e., infection present), and if a site is not occupied, $Z_i = 0$ (Royle and Dorazio, 2008). Sites are defined as spatial units within a larger area (population) of interest (Mackenzie et al., 2005), where sampled sites would ideally be based on a probabilistic sample from a sampling frame that includes all sites within the region of interest. This is because this allows for results to be generalized to all sites within the region of interest as opposed to just the sites surveyed.

Sites also need to be defined in such a way that makes sense for the species of interest. This requires to first think about whether it is more appropriate to define and survey fixed areas (i.e., same size) or to survey areas that occur naturally, meaning sites can differ in size. For instance, if an individual tree is defined to represent a site, the researcher has no control over what the size of each tree is and so the size of the sampling unit will vary from tree-to-tree, which could then be accounted for in the model. The method the researcher decides will logically depend on the research objectives of a study, what is most meaningful in terms of the biology of the species of interest, and how reasonable model assumptions will be (see Subsection 2.3). It is important to consider a balance between specifying the size of each sampling unit so that it could be occupied by the species of interest while being small enough that the eventual estimate of ψ_i will be meaningful for the research objectives (i.e., that model assumptions are not severely violated). In addition, it is important to consider the surveyors and what are practical areas for them to survey. Considerations related to model assumptions remain even when a site is naturally occurring and will be discussed further in Subsection 2.3.

2.2 STATISTICAL MODEL

In order to estimate ψ_i, at least one site is visited multiple times within a sampling season. The sampling season is again determined based on the biology of the species such that the population can be assumed to remain closed within a season. What this means is that Z_i (the true occupancy status at site i) remains constant within visits to the same site. We assume Z_i follows a Bernoulli distribution with probability of success, ψ_i. Then, ψ_i can be related to site-level covariates and, in turn, can be estimated using the logit link (or probit link function), $logit(\psi_i) = \boldsymbol{\beta} \boldsymbol{X}_i$, where $\boldsymbol{\beta}$ is a vector of parameters and $\boldsymbol{X}_i$ is a matrix of site-level covariates that relate to ψ_i. If sampling is done probabilistically, these relationships can be used to predict site-occupancy probability at unsampled sites. However, occupancy models are not just simple logistic regression models. What makes an occupancy model different from logistic regression is its incorporation of another source of variation, detectability (Makenzie et al., 2002). Just because a site is occupied does not mean that the species will be detected. In fact, imperfect detection is incredibly common in ecological studies. Imperfect detection needs to be considered to allow for the possibility that a species could be present but missed by the observer (i.e., a false-negative detection error) or a species may not be present but the observer recorded the species as present (i.e., a false-positive detection error). It is intuitive to think about how those errors could impact probability of occupancy estimates. For instance, ψ_i will tend to be overestimated with false positives and underestimated with false negatives. It is important to note that only false negatives are considered for the remainder of this paper. Although there are plenty of scenarios where false positives are an issue, and methods for dealing with them exist (refer to Mackenzie et al., 2005 and/or Royle and Dorazio, 2008), it is assumed false positives occur with negligible frequency for the motivating application of this paper.

This idea of imperfect detection is what gives rise to a hierarchical modeling framework. To account for imperfect detection means to be aware that the true occupancy state (Z_i) may not agree with the observed occupancy state, denoted by the random variable Y_{ij} (j represents the number of visits where j = 1, 2, 3, …, J). Therefore, this introduces a new parameter, detection probability, which is the probability of a detection given the site is occupied (i.e., $P(Y_{ij} = 1 | Z_i = 1)$), denoted p_{ij}. At least one (usually many) of the sites must be visited at least twice within a season in order for it to be possible to estimate this parameter. Thus, a detection history matrix is created for each site, where each row represents a site and there are J columns for the recorded/observed detection at each. One row of this matrix would look something like $Y_{ij} = [y_{i1}, y_{i2}, y_{i3}, \dots, y_{iJ}]$ where each y_{ij} is filled in with a 1 (species was detected), 0 (species was not detected), or NA (site was not visited).

Another way to think about all of this is that there is a state process model for Z_i (described above) and an observation process model for $Y_{ij}|Z_i$. That is, the state process model has ψ_i as a parameter and the observation process model has p_{ij} as a parameter. In addition, the observation process model incorporates the number of visits to a site (J_i) as a fixed piece of the study design. This means that p_{ij} is specifically the probability the species of interest is detected during visit j to site i, *given* that it is truly there ($Z_i = 1$). Going off of the model described above, $Y_{ij}|Z_i$ follows a Binomial distribution with parameters J_i and $p_{ij} * Z_i$, where $P(Y_{ij} = 1|Z_i = 0) = 0$ (i.e., no false positives). Once again, p_{ij} is modeled and estimated using the logit link, $logit(p_{ij}) = \boldsymbol{a}\boldsymbol{W}_{ij}$, where $\boldsymbol{a}$ is a vector of parameters and $\boldsymbol{W}_{ij}$ is a matrix of site-level and/or observer-level covariates related to p_{ij}.

2.3 MODEL ASSUMPTIONS

The assumptions that go along with occupancy models are directly related to the distributional assumptions of Z_i and $Y_{ij}|Z_i$. It is important to note that assumptions are never perfectly met, and care needs to be incorporated into the design-phase of the study so that violations are minimized and are not severe. The first assumption is that the population is closed, meaning that during a sampling period, or season, the occupancy status remains the same. In other words, Z_i remains constant within visits to the same site, as described at the beginning of

Subsection 2.2 when discussing sampling seasons. It is also assumed that observations are independent of one another, which is related to visits within the same site (i.e., $Y_{ij} \perp Y_{ij'}$, where $j \neq j'$) as well as between sites (i.e., $\boldsymbol{Y}_{ij} \perp \boldsymbol{Y}_{i'\cdot}$, where $i \neq i'$). This is why it is important that the researcher, especially when arbitrarily defining what a site is, makes sure to consider the biology of the species they are working with so that sites are reasonably sized and samples do not severely violate this assumption. Also, ψ_i and p_{ij} are both considered constant after accounting for any covariates relating to them. Lastly, as discussed earlier, since the whitebark pine study has negligible issues with false positives but considerable issues with false negatives, the standard occupancy model assumes that the probability of detecting a species given that the site is not occupied is 0.

3 IMPERFECT DETECTION SIMULATION STUDY

As stated previously, ignoring imperfect detection can result in biased estimates of occupancy probability. The purpose of this study was to explore what happens to these estimates when using a naïve logistic regression model that ignores imperfect detection versus the standard occupancy model, which accounts for false-negative errors only. In this section, we describe the details of our simulation study (commented code is provided in Appendix A).

3.1 STUDY DESIGN

The study was set-up to explore how the estimator for ψ from two different models (i.e., $\hat{\psi}$ from occupancy and logistic regression models) responds to different values of the true occupancy probability (ψ) and true detection probability (p), for a fixed number of sites (n) and visits to each site (J). The code allows for the exploration of different values for n and J as well. The results that will be discussed below set the simulation to have J = 3, since the blister rust study had at most 3 visits to each whitebark pine tree, and n = 50 as a conservative estimate of sample size related to the average number of trees in a stand. We investigated a total of 9 treatment combinations for a range of values of ψ and p (low = 0.2, medium = 0.5, and high = 0.8).

With this investigation, we assumed that the probability of detecting a species given the species is truly not present is 0. In other words, false-positive errors could not occur. It was also assumed that the probability of detection is constant within a site (among visits) as well as among sites. Lastly, ψ is constant among sites. Although this last assumption is most likely a stretch, as there are usually covariates that influence this parameter in real world applications, it is reasonable for the purpose of this simulation study to get a general sense of how estimates of ψ are affected by imperfect detection (specifically, false-negative errors).

Since the purpose of this study is to show how biased estimates of ψ can occur when imperfect detection is ignored, this requires fitting the generated data to an intercept-only logistic regression model (which ignores imperfect detection) and a standard intercept-only occupancy model (which accounts for imperfect detection) to compare results. For both models, functions were written (described in detail in Appendix A) so that each iteration keeps track of the estimate for ψ, its estimated confidence interval, and whether or not the confidence interval captures the true, user-specified value of ψ. The previous information is also kept track of for p only when fitting the occupancy model since detection is not a component of logistic regression models. Finally, a function was written to calculate the average values of ψ, average lower and upper limits of the confidence intervals, and the proportion of confidence intervals that include the true value of ψ over all iterations of the simulation for both of the models separately. Again, this was done for p in the occupancy model only.

3.2 DATA GENERATION & MODEL FITTING

Data were generated assuming they arose exactly as specified by the occupancy model described in Section 2. This involved generating data from a binomial distribution twice since both the true occupancy state (Z_i) and observed occupancy state (Y_{ij}) are binomial random variables where Y_{ij} is conditional on Z_i. In addition, this function included inputs for p, ψ, n, and J in order to allow one to experiment with inputting different values to see how they affect the bias of the parameter estimates. First, a total of n z_i true binary occupancy state values were generated from a binomial distribution where $Z_i \sim IID\ Binomial(1, \psi)$. The data for the true occupancy state were generated first since those values impact what the probability of detection is (i.e., $p = 0$ if $z_i = 0$ and $p = p$ if $z_i = 1$) when generating the matrix of observed occupancy data, also referred to as detection histories. The dimensions of this matrix were n by J where each of the J observations in the i^{th} row had $p * z_i$ as the probability of detection.

The next step was to formulate response data for both models. For the occupancy model, the `unmarked` package (Fiske and Chandler, 2011) in R (R Core Team, 2018) was loaded and the matrix of observed occupancy data was inputted to the `unmarkedFrameOccu` function as the argument for the response matrix. Since having multiple visits to the same site is unique to occupancy models, as that is what allows for being able to account for imperfect detection, the response data for the logistic regression model were in a vector as opposed to a matrix. To create this vector of responses, the values were summed across the visits and if the sum was greater than 0, that observation was assigned a 1. Similarly, if the sum was 0, that observation was assigned a 0.

For each iteration, the `model_compare` function was written to generate an estimate of ψ with associated confidence interval for both an intercept-only logistic regression model and an occupancy model with no site-level or visit-level covariates. The percentage of confidence intervals using both models that captured the true value of ψ (again, specified at the start of the data simulation) was also calculated. For the occupancy model, the same process was completed for p. Lastly, the `compare_sim` function was written to compute the average ψ value and associated average confidence interval over all iterations for each model separately as well as the average value of p and associated average confidence interval over all iterations for the occupancy model only (output shown in Figure 2 below).

3.3 DESCRIPTION OF RESULTS

After the code for obtaining parameter and confidence interval estimates, the `plot_compare` function was written using `ggplot2` (Wickham, 2016) to plot all of that information and provide a visual representation that illustrates the purpose of this study. For a given plot, the values specified for ψ and p during the data-generating process are shown at the top of the plot and indicated by black tick marks in the panel. The number of iterations run gives the total number of confidence intervals for estimating p as well as for estimating ψ for each of the models, which all appear on the plot as thin horizontal lines in the color of the capture probability discussed below. The teal vertical tick mark and teal text that appears for p as well as for both estimates of ψ represents the average estimate that was calculated over the total number of iterations and the pink horizontal strip represents the estimated average confidence interval over the total number of iterations. Also appearing below each set of confidence intervals is black text that provides the bias associated with each estimate. Finally, a legend corresponding to the capture probability indicates the proportion of confidence intervals that succeeded in capturing the true, specified parameter value. To make the correspondence clearer, this capture probability was added in text above each set of confidence intervals.

3.4 DISCUSSION OF RESULTS

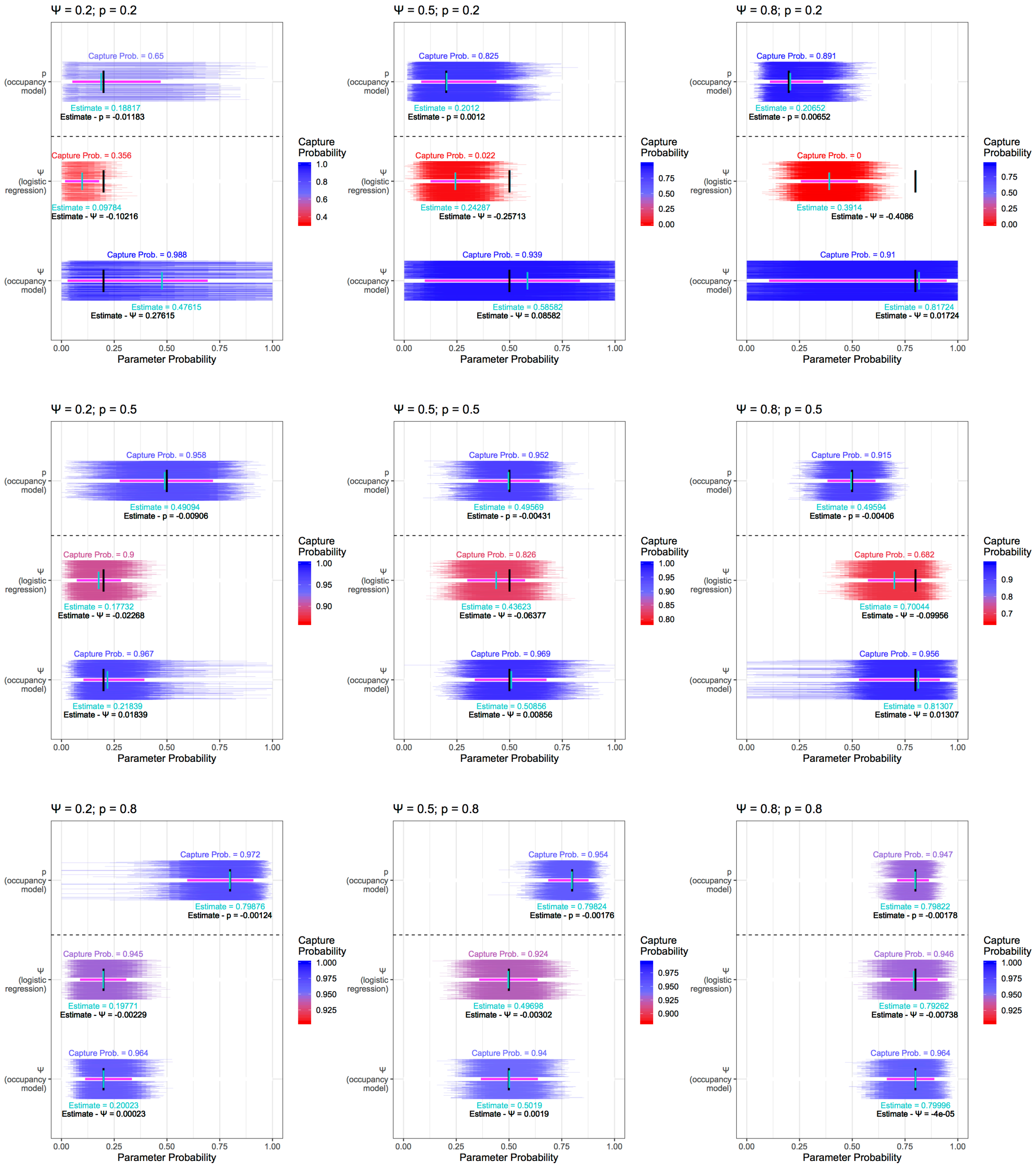

Figure 2. Plots showing results over 2000 iterations* with $n = 50$ and $J = 3$ for combinations of low (top row), medium (middle row), and high (bottom row) values of p with low (left column), medium (middle column), and high (right column) values of ψ.

*plot with low values for both p and ψ (top-left plot) had only 500 iterations.

The 9 plots above show all possible combinations between low (0.2), medium (0.5), and high (0.8) values of ψ with low (0.2), medium (0.5), and high (0.8) values of p. The plot could only be generated for 500 iterations when both ψ and p were 0.2 since the optimization had trouble converging. This is due to there not being enough visits and/or there not being enough sites, which also explains why the bias is so large with this particular occupancy model estimate of ψ. The other 8 plots show that the bias for the estimate obtained fitting a naïve logistic regression model is consistently worse than the bias when fitting the occupancy model. Overall, any bias from fitting the occupancy model is fairly negligible for these other 8 plots as well with the exception of when $p = 0.2$ and $\psi = 0.5$. This bias would also become negligible if J were to increase, as the optimal number of visits is 9 for this particular combination of parameter values (see Figure 3 below). In general, for a given value of ψ and when ignoring imperfect detection, the downward bias that occurs when estimating ψ gets worse as p decreases. In addition, for a given value of p, the bias gets increasingly worse as ψ increases. Intuitively, this all makes sense. We would expect for the estimate of ψ to suffer from more bias if the ability to detect a species gets worse. Moreover, ignoring imperfect detection is going to be more problematic if a species is common (ψ is larger) than if a species is rare. Although the bias is not terrible with the logistic regression model when $p = 0.8$, we can see that it still does slightly worse than the occupancy model in terms of capture probability. The capture probabilities also show a sharp decline as p decreases when ignoring imperfect detection. When $p = 0.2$, the probability of the true value of ψ falling within a confidence interval generated using logistic regression is either 0 or very close to it. The results from this study clearly illustrate the importance of accounting for imperfect detection by using occupancy models in order to obtain reliable estimates of ψ.

Interestingly enough, different combinations of p and ψ indicate whether it is more advantageous to increase the total number of sites that are surveyed versus thinking about increasing the number of visits to each site (Mackenzie et al., 2005). As ψ increases for a given value of p, the number of visits should be increased, being a larger increase when the given value of p is on the smaller side (see Figure 3 below). Additionally, in general, if ψ is low, it is going to be more important to make sure to survey a greater number of sites.

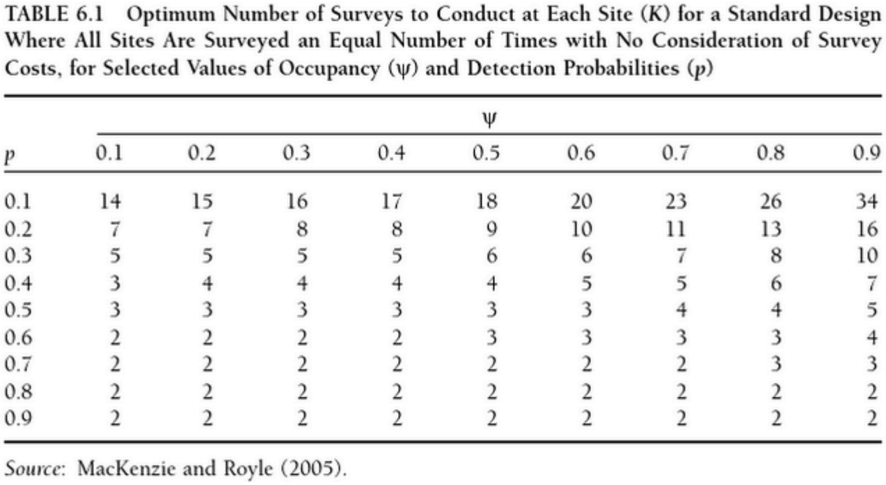

TABLE 6.1 Optimum Number of Surveys to Conduct at Each Site (K) for a Standard Design Where All Sites Are Surveyed an Equal Number of Times with No Consideration of Survey Costs, for Selected Values of Occupancy (ψ) and Detection Probabilities (p)

	ψ								
p	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0.1	14	15	16	17	18	20	23	26	34
0.2	7	7	8	8	9	10	11	13	16
0.3	5	5	5	5	6	6	7	8	10
0.4	3	4	4	4	4	5	5	6	7
0.5	3	3	3	3	3	3	4	4	5
0.6	2	2	2	2	3	3	3	3	4
0.7	2	2	2	2	2	2	2	3	3
0.8	2	2	2	2	2	2	2	2	2
0.9	2	2	2	2	2	2	2	2	2

Source: MacKenzie and Royle (2005).

Figure 3. Table of optimum number of visits for different combinations of ψ and p.

4 APPLICATION

Here, we return to the blister rust study and examine how occupancy models will be used to estimate the relationship between blister rust and elevation for the final time period (2016 – 2019). This will involve discussing both work that has been accomplished as well as considering future work that goes beyond the scope of this paper but that will inevitably be required in order to adequately address this research question.

4.1 STUDY DESIGN

The data collection for this monitoring program has been ongoing since 2004 and focuses on a random sample of 150 whitebark pine stands in the GYE (Wright and Irvine, 2017). These were sampled from a total of 10,770 whitebark pine stands that exist in the GYE, with a stand being at least 2 hectares of contiguous forest in size (Shanahan et al., 2016). Within stands, a total of 176 10 meter by 50 meter transects were delineated. There were 26 stands with 2 transects and the remaining 124 stands contained 1 transect. Any tree exceeding 1.4 meters in height within a transect was either visited by one observer, two observers, or at most three observers during each time period. In other words, each tree had anywhere between one and three observations associated with it.

To relate this back to occupancy modeling terminology, a tree represents a site and an observer represents a visit. All of the stands and transects within stands were surveyed in a rotating panel design, spanning 4 time periods; the first time period ran from 2004 – 2007, second time period from 2008 – 2011, third time period from 2012 – 2015, and the most recent time period spanning from 2016 – 2018 (2019 data are not available yet). There were 4 panels defined by a partition of the panels into approximately equal sized groups and each panel is sampled every four years. To date, there have been a total of four time periods. Table 1 below shows a visual example of how this four-panel sample design works (GYWPMWG, 2011). Since the research question is, “has the relationship between elevation and prevalence of blister rust changed?”, this paper focuses on what needs to be done to analyze that last time period of data so that the results obtained can be compared to the results obtained from those three previous time periods.

Sampling Panel	Number of Transects Sampled	Sampling Year
1	43	2008
2	45	2009
3	44	2010
4	44	2011
1	43	2012
2	45	2013
3	44	2014
4	44	2015
1	43	2016
2	45	2017
3	44	2018
4	44	2019

Table 1. Rotating-panel sample design where all 176 transects are sampled every 4 years.

4.2 MODELS

4.2.1 FULL OCCUPANCY MODEL

In order to make a fair comparison to address the research question, the eventual models to be fitted will be the same as the models described in Wright and Irvine (2017). It will be addressed in Subsection 5.2 below what is required in order to be able to fit the following occupancy model:

(1) $logit(\psi_{tij}) = \beta_0^t + \beta_1^t DBH_{tij} + \beta_2^t elev_{ti} + \beta_3^t topo_{ti} + \beta_4^t trees_{ti} + b_{stand_{ti}}^t$

(2) $logit(p_{tijk}) = \alpha_0 + \alpha_1 DBH_{tij} + \alpha_2 date_{ti} + \alpha_3 slope_{ti} + \alpha_4 hike_{ti} + \alpha_5 trees_{ti} +$

$\alpha_6 trees_{ti}^2 + \alpha_7 exp_{tijk} + a_{obs_{tijk}}$

The superscript/subscript t refers to the time period, which was relevant for them since they dealt with three different time periods. However, t will specifically refer only to the most recent time period for this analysis so will be omitted. Beyond this, there are transect-level (i), tree-level (j) and observer-level (k) covariates in these models. The only tree-level covariate is the diameter at breast height (DBH), which is thought to influence both ψ and p based on previous research. At the site- or tree-level of the occupancy model (Equation (1)), elevation ($elev$), topography ($topo$), number of trees ($trees$), and a random effect for stand ($stand$) were all recorded at the transect-level. At the observer- or visit-level of the occupancy model (Equation (2)), survey date ($date$), slope ($slope$), hiking time ($hike$), and number of trees ($trees$) were recorded at the transect-level. In addition, the two observer-level covariates included a dummy variable for whether it was an observer's first field season or not (exp) and a random effect for observer (a_{obs}; Equation (2)). The coefficients, the βs and the αs, relate the covariates to ψ_{tij} and p_{tijk}, respectively, with the logit-link function after accounting for the other covariates in the model.

4.2.2 NAÏVE OCCUPANCY MODEL

Since Bayesian methods (discussed in Subsection 5.2 below) are required in order to incorporate random effects, which are beyond the scope of this paper, we fit simplified models as a place to begin this investigation. In addition, exp was omitted from Equation (2) in Sub-subsection 4.2.1 at the observer-level of the occupancy model below (Equation (2)) due to challenges faced incorporating covariates that were recorded at the observer-level (also discussed in Subsection 5.2 below). The following naïve occupancy model was fit using the `occu` function in `unmarked` and results are reported in Subsection 4.3:

(1) $logit(\psi_{ij}) = \beta_0 + \beta_1 DBH_{ij} + \beta_2 elev_i + \beta_3 topo_i + \beta_4 trees_i$

(2) $logit(p_{ij}) = \alpha_0 + \alpha_1 DBH_{ij} + \alpha_2 date_i + \alpha_3 slope_i + \alpha_4 hike_i + \alpha_5 trees_i + \alpha_6 trees_i^2$

Refer to Sub-subsection 4.2.1 above for a reminder as to what each covariate above represents. It is important to note that these covariates were standardized. Therefore, covariates are at their average values when their corresponding coefficients are set to 0 with standard deviation 1.

4.3 RESULTS

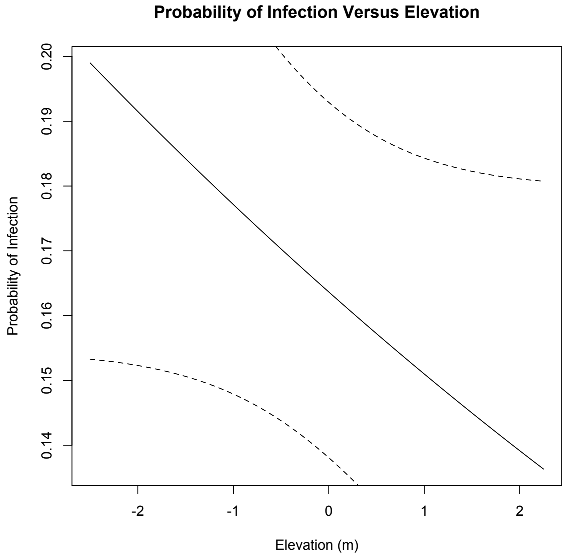

Figure 5. Plot showing the relationship between probability of infection and elevation including 95% confidence bands where DBH, $topo$, and $trees$ were at their average values.

We observed decreasing probability of infection as elevation increased, after accounting for DBH, $topo$, and $trees$ being at their average values in Equation (1) from Sub-subsection 4.2.2 (Figure 5). Although this is consistent with the results from Wright and Irvine (2017), these results should really not be trusted as they come from a naïve model that we do not believe is realistic. A random effect for stand is needed at the transect-level in order to account for the dependency among trees as without it, we would be assuming that trees are independent in terms of ψ_{ij}, which is a severely violated assumption. Moreover, certain challenges arose when trying to incorporate a couple of the observer-level covariates that are needed to accurately replicate the models from Wright and Irvine (2017). See Subsections 5.1 and 5.2 below for more of a discussion about these issues.

5 DISCUSSION

5.1 IMPLEMENTATION CHALLENGES

Before being able to get into actually using the data, quite a bit of data manipulation and cleaning had to be done in order to omit irrelevant information as well as get the data in a usable format. Here, we provide a brief description of what was done but refer to Appendix A for commented code. The first obstacle encountered was that not all of the data were in one place, so 4 different data sets had to be merged to combine all of the necessary information. Secondly, lots of data were included that were not needed and so sub-setting the data in various ways was necessary. Lastly, based on the covariates needed for the eventual models to be fit, some variables had to be created as well as manipulated in order to get them into the correct format.

Once it came to using the data, a few challenges arose there as well. There were issues when trying to incorporate observer-level covariates for modeling detection probability due to the fact that trees had a varying number of observers. The occu function for fitting a model in `unmarked` expects the covariates in the state process model to be of dimension $n \times q$, where q is the total number of covariates at this level of the model and the covariates in the observation process model to be of dimension $(n \times J) \times r$, where r is the total number of covariates at this level of the model (refer to Section 2). The issue is that J needed to be able to vary (i.e., need J_i) since each tree could either have j = 1, j = 2, or j = 3 but when merging the detection history matrix with the other covariate data, this did not allow for that without affecting the dimensions of covariates at the state process level of the model. In addition, since we did not have the tools to incorporate random effects, although skeptical, we attempted to still incorporate the transect-level stand covariate ($stand_i$) as a fixed effect. As anticipated, the optimization did not converge due to the numerous levels that this created. Although results still would not have been 100% trustworthy in terms of making a comparison to the previous analysis, they would at least have potentially been able to provide some useful and reasonable insight into what we may expect the actual relationship to be between elevation and ψ_{ij}, assuming the other covariates in the model are at their average values. This is because there would have been a term in the model that accounted for trees belonging to the same transect, which is important when it comes to the assumption of independence among trees.

5.2 FUTURE WORK

Since we cannot trust the estimates of ψ_{ij} in order to make a justifiable comparison to the results of what was observed previously, it is necessary to discuss what future work will entail in order to adequately address the research question. As mentioned in Subsection 4.3, observer-level covariates as well as random effects need to be incorporated as the results to compare these results to used random effects and observer-level covariates with the 2004 – 2015 data in the occupancy model. Work still needs to be done to figure out how to keep the observer-level covariates (whether it was an observer's first field season or not, exp_{ijk}, as well as a random effect for observer, obs_{ijk}), which appear in the observer process model from the previous analysis (see Equation (2) in Sub-subsection 4.2.1), while making sure the other covariate information that corresponds to those trees does not get accounted for more than once. When it comes to incorporating random effects into hierarchical models such as occupancy models, this requires using a Bayesian framework, which was beyond the scope of this paper but the last piece of the puzzle to be able to fully address the research question.

6 CONCLUSION

In conclusion, occupancy models are extremely useful due to their ability to model imperfect detection, whether that be the result of human-error (i.e., the species of interest are wolves and the observer mistakes a coyote for a wolf) or impossibility (i.e., the species of interest is present but incredibly elusive, making detection extremely difficult). The simulation study showed that estimates for ψ tend to be downwardly-biased when imperfect detection is ignored in terms of false negatives and demonstrated the necessity that occupancy models play when it comes to dealing with that issue through being an unbiased analysis method. Unless trusted and valid results are obtained, the purposes they serve are ultimately meaningless. Relating this idea to the context of the blister rust study, depending on what the results show, it could provide valuable information about where to focus management efforts for this project in the future. For instance, if there is evidence that shows the relationship between the probability of infection and elevation has changed, which would mean that ψ_{ij} is increasing as elevation increases, insight into potential drivers of this change will be of upmost importance as well as coming up with solutions to help mitigate the consequences at higher elevations. If the results came from a logistic regression model, lots of effort and time could be wasted by focusing on the wrong things due to misleading results, not to mention more damage being done. Although the simulation study in and of itself is eye-opening in terms of what ignoring imperfect detection can do, thinking about what was seen there in terms of a real-life application truly puts it into perspective.

7 REFERENCES

Fiske, I., & Chandler, R. B. (2011). *unmarked*: An R Package for Fitting Hierarchical Models of Wildlife Occurrence and Abundance. *Journal of Statistical Software*, **43**(10), 1-23. URL http://www.jstatsoft.org/v43/i10/.

Greater Yellowstone Whitebark Pine Monitoring Working Group (GYWPMWG). (2011). *Interagency Whitebark Pine Monitoring Protocol for the Greater Yellowstone Ecosystem*, Version 1.1. Greater Yellowstone Coordinating Committee, Bozeman, MT.

MacKenzie, D. I., Bailey, L. L., Hines, J. E., Nichols, J. D., Pollock, K. H., & Royle, J. A. (2005). *Occupancy Estimation and Modeling: Inferring Patterns and Dynamics of Species Occurrence*. Retrieved from https://ebookcentral.proquest.com.

MacKenzie, D. I., Nichols, J. D., Lachman, G. B., Droege, S., Royle, J. A., & Langtimm, C. A. (2002). Estimating Site Occupancy Rates When Detection Probabilities Are Less Than One. *Ecology, 83*(8), 2248-2255. doi:10.2307/3072056.

R Core Team (2018). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Royle, J. A., & Dorazio, R. M. (2008). *Hierarchical Modeling and Inference in Ecology: The Analysis of Data from Populations, Metapopulations and Communities* (1st ed.). Amsterdam; Burlington, MA: Academic.

Shanahan, E., Irvine, K. M., Thoma, D., Wilmoth, S., Ray, A., Legg, K., & Shovic, H. (2016). Whitebark Pine Mortality Related to White Pine Blister Rust, Mountain Pine Beetle Outbreak, and Water Availability. *Ecosphere, 7*(12) doi:http://dx.doi.org.proxybz.lib.montana.edu/10.1002/ecs2.1610.

Wickham, H. (2016). *Ggplot2: Elegant Graphics for Data Analysis.* Springer-Verlag New York.

Wickham, H., François, R., Henry, L., & Müller, K. (2019). *dplyr: A Grammar of Data Manipulation*. R package version 0.8.3. URL https://CRAN.R-project.org/package=dplyr.

Wright, W. J., & Irvine, K. M. (2017). *Assessment of Imperfect Detection of Blister Rust in Whitebark Pine Within the Greater Yellowstone Ecosystem*. Natural Resource Report NPS/GRYN/NRR—2017/1457. National Park Service, Fort Collins, Colorado.

APPENDIX A

SIMULATION STUDY CODE

```
#########################
### PACKAGES REQUIRED ###
#########################

library(dplyr)
library(ggplot2)
library(unmarked)

##############################
### INVERSE LOGIT FUNCTION ###
##############################

exp.it <- function(x){

  out <- exp(x)/(1 + exp(x))
  # Backtransforms x from logit scale back to probability scale

  return(out)

}

#################################
### FUNCTION TO GENERATE DATA ###
#################################

data_sim <- function(p, psi, n_site, n_visit){

  z <- rbinom(n = n_site, size = 1, prob = psi)
  # Random generation of presence/absence data from Binomial distribution

  y_det <- matrix(NA, nrow = n_site, ncol = n_visit)
  # Creates matrix of NAs

  for(i in 1:n_site){

    y_det[i, ] <- rbinom(n = n_visit, size = 1, prob = p*z[i])
    # Fills in NA matrix with random generation of multiple visit
  }  # detection/non-detection data from Binomial distribution

  y_occu <- unmarkedFrameOccu(y_det)
  # Turns y_det matrix into unmarked data frame for the occupancy fit

  sum_y <- matrix(NA, nrow = n_site, ncol = 1)
  # Creates single column matrix of NAs
  for(i in 1:n_site){

    sum_y[i, ] <- sum(y_det[i, ])
    # Fills in each row of single column NA matrix with
  }  # sum of values from each row of y_det matrix

  y_log <- ifelse(sum_y > 0, 1, 0)
  # Makes sum_y values binary to create response data for the logistic regression fit

  out <- list(y_occu = y_occu, y_logistic = y_log, psi = psi, p = p)

  return(out)

}

############################################################################
### FUNCTION TO OBTAIN/COMPARE MODEL ESTIMATES FROM INDIVIDUAL ITERATIONS ###
############################################################################

model_compare <- function(data){

  y_occu <- data$y_occu

  y_log <- data$y_logistic

  psi <- data$psi

  p <- data$p

  fit_occu <- try(occu(~ 1 ~ 1, data = y_occu))
  # Tries to fit an intercept-only occupancy model using y_occu data frame

  if(class(fit_occu) == "try-error"){

    est_psi <- est_det <- cap_psi <- cap_det <- NA
    # If get "try-error," assigns NA to psi/p estimates and whether psi/p are within CIs

    ci_psi <- ci_det <- matrix(c(NA, NA), nrow = 1, ncol = 2)
    # If get "try-error," assigns NA to lower and upper values of CIs for psi and p
  }

  else{

    est_psi <- backTransform(fit_occu, type = "state")@estimate
    # If no error, backTransform function generates estimate of psi on probability scale

    ci_psi <- exp.it(confint(fit_occu, type = "state"))
    # If no error, inverse logit function generates CI for psi on probability scale

    cap_psi <- as.numeric(ci_psi[, 1] <= psi & psi <= ci_psi[, 2])
    # If no error, generates a 1 if psi is within CI and a 0 if psi is not within CI
    est_det <- backTransform(fit_occu, type = "det")@estimate
    # If no error, backTransform function generates estimate of p on probability scale

    ci_det <- exp.it(confint(fit_occu, type = "det"))
    # If no error, inverse logit function generates CI for p on probability scale

    cap_det <- as.numeric(ci_det[, 1] <= p & p <= ci_det[, 2])
    # If no error, generates a 1 if p is within CI and a 0 if p is not within CI
  }

  out_psi <- data.frame(par = paste(expression('\u03A8'), "\n(occupancy\nmodel)"),
                        lower = ci_psi[, 1], upper = ci_psi[, 2],
                        est = est_psi, truth = psi, cap = cap_psi)

  out_det <- data.frame(par = "p\n(occupancy\nmodel)",
                        lower = ci_det[, 1], upper = ci_det[, 2],
                        est = est_det, truth = p, cap = cap_det)

  fit_logi <- try(glm(y_log ~ 1))
  # Tries to fit an intercept-only logistic regression model

  if(class(fit_logi)[1] == "try-error"){

    est_psil <- cap_logi <- NA
    # If get "try-error," assigns NA to psi estimate and whether psi is within CI

    ci_psil <- c(NA, NA)
    # If get "try-error," assigns NA to lower and upper values of CI for psi
  }

  else{

    est_psil <- fit_logi$coefficients
    # If no error, generates estimate of psi on probability scale

    ci_psil <- confint(fit_logi)
    # If no error, generates CI for psi on probability scale

    cap_logi <- as.numeric(ci_psil[1] <= psi & psi <= ci_psil[2])
    # If no error, generates a 1 if psi is within CI and a 0 if psi is not within CI
  }

  out_logi <- data.frame(par = paste(expression('\u03A8'), "\n(logistic\nregression)"),
                         lower = ci_psil[1], upper = ci_psil[2],
                         est = est_psil, truth = psi, cap = cap_logi)

  out_psi <- rbind(out_psi, out_logi)

  out <- list(ydata = data, psi_compare = out_psi, detection = out_det)

  return(out)

}
```

```
########################################################################################
### FUNCTION TO OBTAIN/COMPARE AVG. MODEL ESTIMATES ALONG WITH INDIVIDUAL ITERATIONS ###
########################################################################################

compare_sim <- function(nsim, p, psi, n_visit, n_site){

  psi_compare <- data.frame()

  p_compare <- data.frame()

  for(i in 1:nsim){

    y_data <- data_sim(p = p, psi = psi, n_visit = n_visit, n_site = n_site)

    sim_list <- model_compare(y_data)

    psi_compare <- rbind(psi_compare, sim_list$psi_compare)

    p_compare <- rbind(p_compare, sim_list$detection)

  }

  psi_comp_avg <- psi_compare %>% group_by(par) %>%
    summarise(est_avg = mean(est, na.rm = T), l_avg = mean(lower, na.rm = T),
              u_avg = mean(upper, na.rm = T), truth = mean(truth, na.rm = T),
              cap = mean(cap, na.rm = T), num_na = sum(is.na(lower) & is.na(upper)),
              converge_error = sum(is.na(cap)))
  # Computes average estimate of psi, average CI for psi, and average capture value for
   # psi over all iterations for both the occupancy model and logistic regression model
   # as well as displays the true value of psi and computes the total number of all the
   # iterations that generated NAs for CI endpoint values/capture values for each model

  p_comp_avg <- p_compare %>% group_by(par) %>%
    summarise(est_avg = mean(est, na.rm = T), l_avg = mean(lower, na.rm = T),
              u_avg = mean(upper, na.rm = T), truth = mean(truth, na.rm = T),
              cap = mean(cap, na.rm = T), num_na = sum(is.na(lower) & is.na(upper)),
              converge_error = sum(is.na(cap)))
  # Computes average estimate of p, average CI for p, and average capture
   # value for p over all iterations for the occupancy model only as well
   # as displays the true value of p and computes the total number of all
   # iterations that generated NAs for CI endpoint values/capture values

  df_avg <- rbind(psi_comp_avg, p_comp_avg)
  # Data frame binding the average psi information from all iterations for
   # both the occupancy and logistic regression models separately with the
   # average p information from all iterations for the occupancy model only

  df_psi <- left_join(psi_compare, psi_comp_avg, by = c("par", "truth"))
  # Data frame adding the average psi information from all iterations
   # to every individual iteration of the psi information for both
   # the occupancy model and the logistic regression model separately
```

```r
  df_p <- left_join(p_compare, p_comp_avg, by = c("par", "truth"))
  # Data frame adding the average p information from all iterations to every
   # individual iteration of the p information for the occupancy model only

  df_all <- rbind(df_psi, df_p)
  # Data frame binding all individual and average information regarding psi for
   # the occupancy model and the logistic regression model separately with all
   # individual and average information regarding p for the occupancy model only

  out <- list(df_avg = df_avg, df_all = df_all)

  return(out)

}

################################################################################
### FUNCTION TO GENERATE PLOT AND SUMMARY OUTPUT OF AVERAGE PARAMETER VALUES ###
################################################################################

plot_compare <- function(df_avg, df_all){

  low_cap <- max(0, min(df_avg$cap) - 0.04)
  # Lowest value for capture probability legend

  high_cap <- min(1, max(df_avg$cap) + 0.04)
  # Highest value for capture probability legend

  mp <- (low_cap + high_cap)/2
  # Middle value for capture probability legend

  df_all$lower <- ifelse(df_all$lower < 0, 0, df_all$lower)
  # Makes it so lower value can't go below 0 on probability scale

  df_all$upper <- ifelse(df_all$upper > 1, 1, df_all$upper)
  # Makes it so upper value can't go above 1 on probability scale

  plot_titlea <- paste(expression('\u03A8'), "=", df_avg$truth[1])
  # Creates plot title of true value of psi

  plot_titleb <- paste("p =", df_avg$truth[3])
  # Creates plot title of true value of p

  title <- paste(plot_titlea, plot_titleb, sep = "; ")
  # Combines true values of psi and p into one plot title

  bias <- df_avg$est_avg - df_avg$truth
  # Computes difference between parameter estimate and true parameter value

  p <- suppressWarnings(ggplot(df_all) +
  # Plot code

      geom_point(aes(x = est_avg, y = par),
                 pch = "|", size = 0, col = "darkturquoise", data = df_avg) +
      geom_errorbarh(aes(x = est, y = jitter(as.numeric(par)),
                         xmin = lower, xmax = upper, col = cap.y),
                     alpha = 0.25, height = 0, lwd = 0.25) +

      geom_segment(aes(x = 0, y = par, xend = 1, yend = par),
                   size = 2.5, col = "white", data = df_avg) +

      geom_segment(aes(x = l_avg, y = par, xend = u_avg, yend = par),
                   size = 1.5, col = "#FF33FF", data = df_avg) +

      geom_point(aes(x = truth, y = par),
                 pch = "|", size = 11, hjust = 0.5, col = "black", data = df_avg) +

      geom_point(aes(x = est_avg, y = par),
                 pch = "|", size = 8.5, hjust = 0.5,
                 col = "darkturquoise", data = df_avg) +

      geom_text(aes(x = (l_avg + u_avg)/2, y = par,
                    label = paste("Capture Prob. =", round(cap, 3)), col = cap),
                hjust = 0.5, vjust = -4, data = df_avg) +

      geom_text(aes(x = est_avg, y = par, label = paste("Estimate =", round(est_avg, 5))),
                hjust = 0.5, vjust = 5, col = "darkturquoise", data = df_avg) +

      geom_text(aes(x = (est_avg[1] + truth[1])/2, y = par[1],
                    label = paste("Estimate -", expression('\u03A8'),
                                  "=", round(bias[1], 5))),
                hjust = 0.5, vjust = 6.5, col = "black", data = df_avg) +

      geom_text(aes(x = (est_avg[2] + truth[2])/2, y = par[2],
                    label = paste("Estimate -", expression('\u03A8'),
                                  "=", round(bias[2], 5))),
                hjust = 0.5, vjust = 6.5, col = "black", data = df_avg) +

      geom_text(aes(x = (est_avg[3] + truth[3])/2, y = par[3],
                    label = paste("Estimate - p =", round(bias[3], 5))),
                hjust = 0.5, vjust = 6.5, col = "black", data = df_avg) +

      scale_colour_gradient2(limits = c(low_cap, high_cap), midpoint = mp,
                             low = "#FF0000", high = "#0000FF", mid = "#7171FF",
                             guide = guide_colourbar(title = "Capture\nProbability")) +

      xlab("Parameter Probability") +

      ylab(" ") +

      geom_hline(yintercept = 2.45, lty = 2) +

      theme_bw(base_size = 14) +

      ggtitle(title))

  out <- list(summary = df_avg)
```

```r
  print(p)

  return(out)

}

HL <- compare_sim(nsim = 2000, p = 0.8, psi = 0.2, n_site = 50, n_visit = 3)
HL.plot <- plot_compare(HL$df_avg, HL$df_all)
# High p, low psi

HM <- compare_sim(nsim = 2000, p = 0.8, psi = 0.5, n_site = 50, n_visit = 3)
HM.plot <- plot_compare(HM$df_avg, HM$df_all)
# High p, medium psi

HH <- compare_sim(nsim = 2000, p = 0.8, psi = 0.8, n_site = 50, n_visit = 3)
HH.plot <- plot_compare(HH$df_avg, HH$df_all)
# High p, high psi

ML <- compare_sim(nsim = 2000, p = 0.5, psi = 0.2, n_site = 50, n_visit = 3)
ML.plot <- plot_compare(ML$df_avg, ML$df_all)
# Medium p, low psi

MM <- compare_sim(nsim = 2000, p = 0.5, psi = 0.5, n_site = 50, n_visit = 3)
MM.plot <- plot_compare(MM$df_avg, MM$df_all)
# Medium p, medium psi

MH <- compare_sim(nsim = 2000, p = 0.5, psi = 0.8, n_site = 50, n_visit = 3)
MH.plot <- plot_compare(MH$df_avg, MH$df_all)
# Medium p, high psi

LL <- compare_sim(nsim = 2000, p = 0.2, psi = 0.2, n_site = 50, n_visit = 3)
LL.plot <- plot_compare(LL$df_avg, LL$df_all)
# Low p, low psi

LM <- compare_sim(nsim = 2000, p = 0.2, psi = 0.5, n_site = 50, n_visit = 3)
LM.plot <- plot_compare(LM$df_avg, LM$df_all)
# Low p, medium psi

LH <- compare_sim(nsim = 2000, p = 0.2, psi = 0.8, n_site = 50, n_visit = 3)
LH.plot <- plot_compare(LH$df_avg, LH$df_all)
# Low p, high psi
```

APPLICATION DATA CLEANING CODE

```
#########################
### PACKAGES REQUIRED ###
#########################

library(unmarked)
library(dplyr)

####################################
### IMPORTING ORIGINAL DATA SETS ###
####################################

SO <- read.csv("/Users/rachel.wyand/Downloads/SingleObs.csv")
# Single observer data

MO <- read.csv("/Users/rachel.wyand/Downloads/MultiObs.csv")
# Multi-observer data

FD <- read.csv("/Users/rachel.wyand/Downloads/FirstData.csv")
# First data with important covariate data

HD <- read.csv("/Users/rachel.wyand/Downloads/hikingdist.csv")
# Hiking time/distance data

################################################
### CREATING VARIABLES IN ORIGINAL DATA SETS ###
################################################

FD$Slope_Rad <- (FD$slope_degrees)*(pi/180)
# Variable for slope in radians

FD$Aspect_Rad <- (FD$aspect_degrees)*(pi/180)
# Variable for aspect in radians

FD$topo <- sin(FD[, 105])*cos(FD[, 106])
# Variable for topography using slope_rad and aspect_rad to calculate

#################################################
### SUBSETTING FOR RELEVANT ROWS OF DATA SETS ###
#################################################

MO.new <- subset(MO, SurveyYear %in% c(2016, 2017))
# Years 2016 and 2017 only (no multi-observer data for 2018)

FD.new <- subset(FD, SurveyYear %in% c(2016, 2017, 2018))
# Years 2016, 2017, and 2018 only

SO.new <- subset(SO, SurveyYear %in% c(2016, 2017, 2018))
# Years 2016, 2017, and 2018 only

SO.L <- subset(SO.new, Status %in% c("L"))
# Live trees only
```

```
FD.WBP <- subset(FD.new, TreeSpecies %in% c("PIAL"))
# Whitebark pine trees only

FD.WBP.L <- subset(FD.WBP, TreeStatus %in% c("L"))
# Live trees only

#################################################
### CREATING VARIABLES IN SUBSETTED DATA SETS ###
#################################################

FD.WBP.L$SiteID <- paste(as.character(FD.WBP.L$Stand_ID),
                         as.character(FD.WBP.L$Transect_ID), sep = ".")
# Variable for SiteID

##########################################
### SUBSETTING FOR RELEVANT COVARIATES ###
##########################################

MO.names <- which(names(MO.new) %in%c("SiteID", "TreeID", "SurveyYear",
                                      "SurveyDate", "ObserverName",
                                      "InfectionPresent", "DBH_cm_MostRecent"))
MO.new <- MO.new[, MO.names]

FD.names <- which(names(FD.WBP.L) %in% c("SiteID", "MultiObserver", "LatestDBH",
                                         "SurveyDate", "TreeID", "SurveyYear",
                                         "elev_meters", "Slope_Rad", "topo"))
FD.WBP.L <- FD.WBP.L[, FD.names]

SO.names <- which(names(SO.L) %in% c("SiteID", "TreeID", "SurveyYear", "SurveyDate",
                                     "Observer", "InfectionPresent", "DBH_cm_MostRecent"))
SO.L <- SO.L[ , SO.names]

HD.names <- which(names(HD) %in% c("Site", "Time"))

HD.new <- HD[, HD.names]

##################################
### RENAMING COVARIATE HEADERS ###
##################################

names(MO.new) <- c("SiteID", "TreeID", "SurveyYear2", "SurveyDate2",
                   "Observer", "LatestDBH2", "InfectionPresent")
names(SO.L) <- c("SiteID", "TreeID", "SurveyYear2","SurveyDate2",
                 "Observer", "LatestDBH2", "InfectionPresent")
names(HD.new) <- c("SiteID", "Time")

###################################
### COMBINING/MERGING DATA SETS ###
###################################

MO.SO <- rbind(MO.new, SO.L)
# Adding rows from subsetted single observer data set to subsetted multi-observer data set
```

```
MO.SO_FD.WBP.L <- merge(MO.SO, FD.WBP.L)
# Merging combined subsetted single/multi-observer data set with subsetted FD data set

MOST.DATA <- merge(MO.SO_FD.WBP.L, HD.new)
# Merging combined subsetted single/multi-observer/FD data set with subsetted HD data set

#########################################################
### CREATING/MODIFYING VARIABLES IN COMBINED DATA SET ###
#########################################################

Experience <- MOST.DATA %>% group_by(Observer) %>% summarise(FirstYear = min(SurveyYear))
MOST.DATA2 <- right_join(MOST.DATA, Experience)
MOST.DATA2 <- MOST.DATA2 %>% mutate(Experience=ifelse(SurveyYear==FirstYear, 1, 0))
MOST.DATA2$Experience <- ifelse(MOST.DATA2$Observer %in% c("Roth", "Shanahan", "Bockino"),
                                0, MOST.DATA2$Experience)
# Indicator variable for whether it's the first field season or not for an observer
# Didn't end up using but would need to fit observer-level covariate

MOST.DATA2$SiteID <- as.factor(MOST.DATA2$SiteID)
# Modifying SiteID to be a categorical variable instead of quantitative

Tree.Number <- MOST.DATA2 %>%
  group_by(SurveyYear, SiteID) %>%
  summarise(Trees=length(unique(TreeID)))
# Variable for number of trees on a transect

MOST.DATA2$jul.date <- as.Date(MOST.DATA2$SurveyDate2, "%m/%d/%y")
MOST.DATA2$jul.date <- julian(MOST.DATA2$jul.date,
                              origin = as.Date("2016-01-01"))-
                              365*(MOST.DATA2$SurveyYear-2016)
# Variable for julian date

##############################
### CREATING FULL DATA SET ###
##############################

FULL.DATA <- right_join(MOST.DATA2, Tree.Number)
# Joining number of trees information to combined
 # data set to create one full and complete data set

#############################################
### REPLACING DATA ENTRY ERRORS WITH NA'S ###
#############################################

FULL.DATA$DBH <- ifelse(FULL.DATA$LatestDBH==-999, NA, FULL.DATA$LatestDBH)

################################################
### CREATING FILE DIRECTORY OF FULL DATA SET ###
################################################

write.csv(FULL.DATA, file = "/Users/rachel.wyand/Downloads/clean_wbp_data.csv",
          row.names = F)
```

APPLICATION ANALYSIS CODE

```r
###############################
### READING IN CLEANED DATA ###
###############################

FULL.DATA <- read.csv("/Users/rachel.wyand/Downloads/clean_wbp_data.csv")

#########################
### PACKAGES REQUIRED ###
#########################

library(unmarked)
library(dplyr)

####################################
### CREATING DETECTION HISTORIES ###
####################################

FULL.DATA$Infection <- ifelse(FULL.DATA$InfectionPresent=="yes", 1, 0)
# Creating indicator variable for infection status

first.fun <- function(x){ifelse(length(x) > 0, x[1], NA)}
# First visit information, otherwise inputs NA
second.fun <- function(x){ifelse(length(x) > 1, x[2], NA)}
# Second visit information, otherwise inputs NA
third.fun <- function(x){ifelse(length(x) > 2, x[3], NA)}
# Third visit information, otherwise inputs NA

FULL.DATA$SiteID <- as.factor(FULL.DATA$SiteID)
# Modifying SiteID to be a categorical variable instead of quantitative

detection <- FULL.DATA %>% group_by(SurveyYear, SiteID, TreeID, MultiObserver) %>%
  summarise(Obs1_Inf = first.fun(Infection), Obs2_Inf = second.fun(Infection),
            Obs3_Inf = third.fun(Infection))
# Putting observed infection status for each visit to a tree

##################################
### OBTAINING DETECTION MATRIX ###
##################################

FULL.DET.DATA <- left_join(detection, FULL.DATA)
# Adding detection data to the full data set

det.matrix <- FULL.DET.DATA[, c(2, 3, 5, 6, 7)]
# Subsetting specific columns from full data set for detection matrix

idx <- which(duplicated(FULL.DET.DATA[, c(2, 3, 5, 6, 7)])=="TRUE")
# Identifying duplicate rows in detection matrix

no.duplicates <- det.matrix[-idx, ]
# Omitting duplicate rows from detection matrix
```

```
########################################
### RESPONSE AND COVARIATE MATRICES ###
########################################

y.matrix <- as.matrix(no.duplicates[, c("Obs1_Inf", "Obs2_Inf", "Obs3_Inf")])
# Creating detection response data matrix

siteCovs <- FULL.DET.DATA[, c("SiteID", "TreeID", "LatestDBH2", "elev_meters",
                              "topo", "Slope_Rad", "Trees", "jul.date", "Time")]
# Data set of site covariates

idx.site <- which(duplicated(siteCovs)=="TRUE")
# Identifying duplicate rows in siteCovs data set

siteCovs <- siteCovs[-idx.site, ]
# Omitting duplicate rows from siteCovs data set

siteCovs[ , c("LatestDBH2", "elev_meters", "topo",
              "Trees", "Time", "Slope_Rad", "jul.date")] <-
  scale(siteCovs[ , c("LatestDBH2", "elev_meters", "topo",
                      "Trees", "Time", "Slope_Rad", "jul.date")])
# Standardizing site covariates

siteCovs.scaled.DF <- data.frame(siteCovs)
# Creating site covariate data frame

################################################################
### MODEL FITTING WITH UNMARKED PACKAGE AND PLOTTING RESULTS ###
################################################################

umf <- unmarkedFrameOccu(y = y.matrix, siteCovs = siteCovs.scaled.DF)
# Creating unmarked data frame with detection response
 # data matrix and standardizedsite covariates

fm1 <- occu(~ LatestDBH2 + Time + poly(Trees, 2) + Slope_Rad + jul.date
            ~ LatestDBH2 + elev_meters + topo + Trees, umf)
# Fitting occupancy model with site-level covariates calling unmarked data frame

elev.data <- data.frame(LatestDBH2 = rep(0, 100),
                        elev_meters = seq(from = -2.5, to = 2.25, length = 100),
                        topo = rep(0, 100), Trees = rep(0, 100), jul.date = rep(0, 100))
# Generating data frame so that other covariates are at their
 # average values and elevation can range between -2.5 and 2.25

predictions <- predict(fm1, type = 'state', newdata = elev.data, appendData = TRUE)
# Predicts value of psi across range of elevation values

plot(Predicted~elev_meters, data = predictions, type = "l", xlab = "Elevation (m)",
     ylab = "Probability of Infection", main = "Probability of Infection vs. Elevation")
lines(predictions$lower ~ predictions$elev_meters, lty = 2)
lines(predictions$upper ~ predictions$elev_meters, lty = 2)
# Plot of psi versus elevation where other covariates
 # are at their average values with 95% confidence bands
```