CHAPTER

10 MODELING STATIC OCCURRENCE AND SPECIES DISTRIBUTIONS USING SITE-OCCUPANCY MODELS

CHAPTER OUTLINE

10.1 INTRODUCTION TO THE MODELING OF OCCURRENCE—INCLUDING SPECIES DISTRIBUTIONS

This chapter is about the joint modeling of occurrence and its ubiquitous false-negative measurement error. Occurrence means the presence or absence of some "thing" in some defined spatial and temporal unit. We have stressed many times that occurrence or presence/absence is a quantity that is directly derived

Applied Hierarchical Modeling in Ecology. http://dx.doi.org/10.1016/B978-0-12-801378-6.00010-2

from abundance, and that both abundance and occurrence are simple areal summaries of an underlying spatial point pattern. Thus, occurrence is exactly equivalent to the event that there is at least one "point" falling within a spatial unit or that the abundance of these "points" in a spatial unit is greater than zero. Despite being only a derived quantity, however, occurrence is hugely important in ecology and related sciences, such as wildlife management and conservation biology. Here, we usually deal with presence and absence of a species and "points" represent the individuals of that species. Reasons for the great importance of occurrence in ecology include the following (see also Chapter 2 in MacKenzie et al., 2006):

- Though only a reduced-information version of abundance, occurrence is typically positively related to abundance, and population changes typically are reflected by range changes (He and Gaston, 2000; Royle et al., 2005; see also Figure 1.2).
- Occurrence may be the only viable alternative for characterizing the state of a population if abundance cannot be reliably assessed for methodological or logistical reasons (MacKenzie et al., 2005), and the practical benefits relative to the measurement of the underlying spatial point pattern are much greater still.
- The parametric assumptions needed for modeling abundance (e.g., Poisson, negative binomial; see Chapter 6) may not be met in your data set. In contrast, the typical Bernoulli model for occurrence (see below) is likely pretty robust across a range of models for the underlying abundance distribution.
- Sometimes abundance may really not matter but occurrence is sufficient for the purpose at hand, e.g., for parasite infections where we may not worry about whether there are 10^5 or 10^6 parasites in a host (Lachish et al., 2012).
- Occurrence is identical to abundance when assessed at a spatial scale where a sample unit can be occupied by at most one individual, breeding pair, or family group. Examples include sites defined as territories of raptors or owls (MacKenzie et al., 2003; Martin et al., 2009). The number of occupied sites then corresponds to the number of breeding pairs, i.e., to the most widely used measure of the abundance of a population in avian ecology (Bibby et al., 2000).
- Occurrence is the basis for the most widely used biodiversity measure, species richness (see Chapter 11 in this book and Purvis and Hector, 2000). It is also the ingredient of an increasingly used index for species richness at top trophic levels computed from camera trap data, the Wildlife Picture Index (WPI; O'Brien et al., 2010).
- Occurrence is of great interest in the ecology of both invasive species (Rout et al., 2014) and diseases (McClintock et al., 2010b), both very popular fields of ecology.

Thus, occurrence is a very widely used state variable in ecology, and some of its subfields focus almost exclusively on it, such as metapopulation ecology (Hanski, 1998) or species distribution modeling (Elith and Leathwick, 2009). In addition, the occurrence of "things" in space is of interest in many scientific disciplines outside of ecology. Consequently, we believe that the potential scope of the models in this chapter may extend far beyond ecology, to the study of the occurrence of any kind of "thing" that is afflicted with false-negative measurement errors.

The basic approach in statistics to the modeling of occurrence is to treat presence and absence as a Bernoulli random variable governed by the "success probability" ψ, which in this context is known as *occupancy, or presence, probability.* Effects of covariates on ψ can be modeled on a link scale, in a logistic or related regression model, and many extensions are conceptually straightforward, including "wiggly" covariate relationships (GAMs, Hastie and Tibshirani, 1990; see Section 10.14) or the

modeling of spatial autocorrelation (Heikkinen and Högmander, 1994; Augustin et al., 1996; Wintle and Bardos, 2006; Bled et al., 2011a,b; Bardos et al., 2015; see also Chapters 21 and 22).

As for the underlying spatial point pattern and its areal summary of abundance, occurrence can rarely ever be assessed without error (Kéry, 2002; MacKenzie, 2005; Ferraz et al., 2007; Kéry and Schmidt, 2008; Kellner & Swihart, 2014; Lahoz-Monfort et al., 2014; Guillera-Arroita et al., 2014a, 2015). Instead, the *measurement of occurrence* often yields an observation $y = 0$ at an occupied site where $z = 1$, representing a false-negative error, or an observation of $y = 1$ where $z = 0$, representing a false-positive error. Both types of error usually lead to biased inferences about occupancy probability and its determinants, such as the strength of the relationships with environmental covariates. Occurrence measurement error has the potential to seriously mislead inferences from species distribution models (Kéry et al., 2010a,b, 2013; Kéry, 2011b; Dorazio, 2012; Guillera-Arroita et al., 2014a; Lahoz-Monfort et al., 2014). Thus, it would appear prudent to accommodate imperfect detection and false-positive errors in models of occurrence whenever possible. The dominant inference framework for the joint modeling of occurrence and its measurement error rate (especially for false-negatives) has the slightly odd name *site-occupancy model* (MacKenzie et al., 2002, 2006; Tyre et al., 2003). This chapter is the first in this book about this powerful class of models for presence/absence data; you will encounter more occupancy models in Chapter 11 (in this volume) and in many chapters in volume 2, especially 16−20. For now we only deal with false-negative measurement errors; see Chapter 19 for false-positive measurement errors in occupancy models.

In spite of its widespread use and regardless of the model used, the concept of presence/absence is widely misunderstood. First and foremost is the deterministic relationship between presence/absence and abundance: presence/absence (=occurrence) is simply a summary of local abundance, nothing more. When you have a good model of abundance you can explain both absences (=sites with abundance zero) and presences (=sites with any abundance greater than zero). Second is the importance to presence/absence of *your* definition of what constitutes a "presence". Whether you define an occurrence as the presence of a single individual, more than a single individual, a reproductive unit (e.g., pair, pack, etc.), or of a viable population will make a huge difference to the biological interpreation of "distribution" in your study. Third is the dependency of presence/absence on *your* choice of the size of the spatial and temporal scale, or grain, of the study. With increasing spatial grain and, if your species moves, also with increasing length of the observation period, occupancy probability increases monotonically, as has been rediscovered recently by Hayes and Monfils (2015). We can describe the spatial scale dependence under the assumption of a certain distribution of the underlying abundance, since occupancy probability is simply 1 minus the probability to get abundance zero under that distribution. Thus, for the simplest case of a Poisson abundance model, we can reconcile occupancy between different spatial scales by adopting a cloglog link for occupancy and treating the logarithm of quadrat size as an offset (see Section 3.3.6). Fourth, and finally, the issues around the unknown sampling area described in Section 6.10 in the context of an abundance model apply also to occupancy models (Efford and Dawson, 2012); see Chandler and Royle (2013), Chandler and Clark (2014) and Ramsey et al. (2015) for the modeling of a latent point process model based on observed counts or detection-nondetection data. Essentially, these models solve the problem of space as it relates to the definition of occupancy in occupancy models.

At least five types of data are used for the modeling of occurrence and species distributions: (1) point pattern data, (2) presence-only data, (3) "presence/absence" data, (4) "presence/absence" data replicated over time, and (5) count data, possibly replicated over time. The information content increases from 2 to 5. In theory, the information content of point pattern data (1) is greatest, but in practice inference based on this data type suffers from challenges for the proper modeling of measurement errors; see below. Here we briefly summarize these five data types, noting that the term "presence/absence" is misleading and actually wrong whenever there is occurrence measurement

error; "detection/nondetection" data is then a better term. Nevertheless, we sometimes use "presence/absence" because it is used so widely, recognizing however that an "absence" may in fact represent an erroneous measurement of presence (unless the false-negative error rate is zero) and a "presence" may represent an erroneous measurement of absence (unless the false-positive error rate is zero).

1. *Point pattern data*: This can be viewed as the "mother" of all distribution and abundance data in ecology and beyond. Point pattern data typically arise when you exhaustively search some clearly defined area within some time frame and record the location of each object present (Illian et al., 2008; Wiegand and Moloney, 2014). Spatial point pattern models (PPMs) treat both the number *and* the locations of the objects in an area as the outcome of a random process, governed by an underlying intensity field. This field and its dependence on spatially indexed covariates can be modeled akin to a Poisson generalized linear model (GLM). When some objects fail to be recorded, we say that the point pattern is *thinned.* Typical data sets for which PPMs are adopted are produced by complete area searches for immobile and easily detectable objects, such as ore deposits, earthquakes, or, in ecology, trees, plants, or gopher mounds. Thus, these powerful methods have largely been developed in fields where measurement error could be assumed to be minor or perhaps even absent. Only relatively rarely have data on moving objects been analyzed using PPMs (Illian et al., 2012), but this use of PPMs is on the increase.

   Recently, there has been a spate of publications on PPMs for species distributions (Warton and Shepherd, 2010; Aarts et al., 2012; Dorazio, 2012, 2014; Fithian and Hastie, 2013; Renner and Warton, 2013; Renner et al., 2015). These authors argue that PPMs are the proper way of modeling species distribution data, in part because the scale dependence of gridded-data-based modeling methods is lost. We share the feeling of excitement about the opportunities offered by the application of PPMs to spatial data on distribution and abundance. However, we make three cautionary comments about PPMs for inference about distribution and abundance in ecology:

   a. The measurement error process underlying most species distribution and abundance data in ecology is totally different from that in classical applications of PPMs. Most ecological data sets to which PPMs are now being applied (e.g., Renner et al., 2015) are *not* the result of a complete area search for some easily detectable and unmistakable object. Rather, they will be the result of some very complicated thinning process induced by spatial sampling bias and additional bias induced by false-negative and false-positive measurement errors. For most large-scale ecological data sets we do *not* know where the observers went to produce their species records (i.e., the observed points). Thus, we cannot usually directly model the spatial sampling bias. In addition, animals move and can be overlooked and misidentified and so can plants (except for the movement; see e.g., Chen et al., 2013). Consequently, there will *always* be false-negative and false-positive errors in real-world point pattern data in ecology. False-negatives correspond to another thinning mechanism while false-positives represent the addition of some "ghost point pattern" that may bias the inference about the target species. Hence, all that is usually modeled with PPMs in ecology is some poorly defined index to density, i.e., the combination of real density, of spatial sampling bias, and of the effects of false-negative and false-positive errors.

      Currently, no PPM methods appear to be available for species distribution modeling that enable you to disentangle the true state from the measurement errors, as we do with most hierarchical models in this book (but see (c) below). We view this as a severe disadvantage of

this otherwise elegant and powerful class of models. For this reason, there is an urgent need for more work along the lines of Dorazio (2012, 2014), who confronts the challenges for PPMs posed by the measurement error process in real-world ecological data sets. Such work is particularly urgent since the typical species distribution data sets to which PPMs are likely to be applied are "dirty data" from citizen-science schemes, where there is often hardly any control over the sampling protocol. Consequently, the problems of spatial sampling bias and measurement errors will be particularly severe in these data sets. Naive application of PPMs in these cases would therefore seem to be risky. We fear that the excitement over a fancy novel modeling framework such as PPMs may easily let people forget about fundamental limitations of their data, which cannot be mended by most current PPMs.

b. The discretization of space for grid-based methods such as most of those described in this book engenders a scale dependence of the inferences. For instance, the larger your pixel, the larger will be both expected abundance and occupancy probability. Some authors have claimed that this is a serious disadvantage (e.g., Fithian and Hastie, 2013, Renner et al., 2015). However, gridding is not necessarily a disadvantage because it can greatly simplify the analysis. First, models for distribution and abundance for grid-based data are conceptually *much* easier than are PPMs. Second, almost all spatial sampling is either explicitly grid based or can be easily characterized by a grid-based framework, and therefore spatial sampling bias can be dealt with formally as part of the modeling and inference. Third, with grid-based models it is easy to incorporate a measurement error component, which we argue is often crucial for valid inference with ecological data sets. And fourth, adding a temporal dimension to a model for gridded data is straightforward, as you will see in several chapters in volume 2. In contrast, in the PPM framework adding time leads to rather complicated models.

c. There is one class of spatial PPM that has an integrated measurement error model: spatial capture-recapture (SCR) models (Efford, 2004; Borchers and Efford, 2008; Royle and Young, 2008; Royle et al., 2013, 2014; Borchers et al., 2015) and some variants of distance sampling models (see Section 9.8). They usually require repeated measurements of the locations of known individuals to jointly estimate the point pattern and its measurement error. Such data are not usually available for large-scale distribution modeling (but see Dorazio, 2014).

2. *Presence-only data*: This data type is typically represented by museum or other collections of locations where a species was identified, collected, or otherwise recorded. Without auxiliary information, we cannot estimate a Bernoulli parameter characterizing presence *and* absence (Pearce and Boyce, 2006; Boyce, 2010). However, typically such data are augmented with data on the environmental conditions at some or all of the larger number of sites from which the sites with recorded presences are sampled, hence, such data may be called "presence/background data" (Lahoz-Monfort et al., 2014). There has been some debate about whether the Bernoulli parameter can be estimated from such data or not, and several authors have developed methods to estimate logistic regression parameters from such data, including Lele and Keim (2006) and Royle et al. (2012). These methods critically depend on assumptions about the correct model structure, including the exact link function, and slight violations of these parametric assumptions may lead to badly biased estimators (Fithian and Hastie, 2013; Phillips and Elith, 2013; see also Section 10.11.1).

Hence, the prevailing opinion now is that with presence-only data we can estimate relative occupancy probability only, i.e., a regression slope but not the intercept. That this is a serious constraint for this data type is shown in Figure 10.1, where you see 21 regression lines all with an

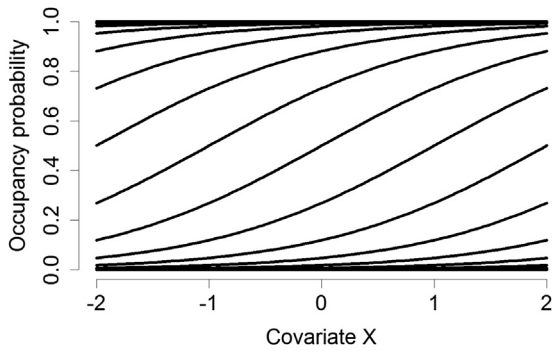

FIGURE 10.1

Twenty-one linear-logistic slopes equal to 1, or, what relative probability of presence means: in practice, it may mean hardly anything at all. All 21 linear-logistic regression lines have an identical slope of 1, but they differ in the intercept, which ranges from −10 to 10.

identical slope of 1 on the logit scale, but where the intercept ranges from −10 to +10. On the probability scale, depending on the intercept, a four-unit change in the covariate may correspond to a change of 0.0003 in occupancy probability, when the intercept is +10 or −10, or to a difference of 0.7616, when the intercept is zero. Hence, it appears to be difficult to obtain practically meaningful inferences from presence-only data.

```
alpha <- seq(-10, 10, by = 1)
curve(plogis(-10 + 1 * x), -2, 2, lwd = 3, ylim = c(0,1), xlab = "Covariate X",
ylab = "Occupancy prob.", frame = F, main = "", cex.lab = 1.5, cex.axis = 1.5)
for(i in 2:21){ curve(plogis(alpha[i] + 1 * x), -2, 2, lwd = 3, add = T) }
(min <- (plogis(-10 + 1 * 2) - plogis(-10 + 1 * -2)) )
(max <- (plogis(0 + 1 * 2) - plogis(0 + 1 * -2)) )
```

An alternative is to use PPMs for presence-only data; indeed, Renner and Warton (2013) have shown that the statistical model underlying the use of the popular `Maxent` software for presence-only data (Phillips and Dudik, 2008) is equivalent to a Poisson PPM under ideal conditions, in which sampling of space is uniform so that the thinning is random or else one has explicit covariates that describe the thinning. One advantage of the PPM approach is that the dependence on the spatial scale is lost (though presumably not the dependence on temporal scale). A disadvantage is that the power and the elegance of PPMs for such data make it easy to forget what is really modeled in most cases: relative density. This is some unknown convolution of the true intensity of the underlying pattern and a potentially very complicated thinning process represented by the sampling processes that produced the data at hand; see above. To properly account for the sampling processes underlying observed point pattern data is a fundamental area in the PPM field where much more research is needed.

3. *Presence/absence data (more properly called detection/nondetection data)*: This kind of data is the classical input for logistic regression, or binomial generalized models and their extensions, such as generalized additive models (GAMs) or generalized linear mixed models (GLMMs). This data type contains more information about the logistic regression of probability of occupancy and

lets one estimate the intercept robustly, but does not allow one to separately estimate detection probability, except under very strong parametric assumptions (Dorazio, 2012; Lele et al., 2012; Knape and Korner-Nievergelt, 2015).

4. *Presence/absence (detection/nondetection) data replicated over time*: This data type contains the most information among the types 2−4 and is the type of data that we primarily consider in this chapter. As we will see, under the closure assumption, this data type allows one to jointly model probability of occupancy and of detection using the powerful site-occupancy models (MacKenzie et al., 2002; Tyre et al., 2003).
5. *Count data*: We have seen that presence/absence data are simply a summary of count data. Hence, it is clear that count data can also be used to model species distributions. Unreplicated counts may be modeled as a Poisson GLM and the probability of occupancy obtained as the probability that a count is greater than zero. Count data that are not replicated over time do not enable one to model abundance or occurrence jointly with detection probability except again under very restrictive assumptions (Solymos et al., 2013; Knape and Korner-Nievergelt, 2015). In contrast, count data that are replicated over time over a short time span, so that the closure assumption is met, may be modeled using the N-mixture model (Chapter 6), which can naturally also serve as a species distribution model (Royle et al., 2005; Dorazio, 2007; see Section 6.9) and so may data collected under any other protocol that provides the extra information to estimate measurement error, such as capture-recapture data (Section 7.9) or distance sampling (see Chapters 8 and 9).

You will note strong structural similarities between this chapter and Chapter 6 on the binomial mixture model. This is no accident, rather, it should underline the strong conceptual similarity between occupancy models and N-mixture models. Indeed, both are simply examples of a general form of two-level hierarchical models for distribution and occurrence or abundance, respectively.

10.2 ANOTHER EXERCISE IN HIERARCHICAL MODELING: DERIVATION OF THE SITE-OCCUPANCY MODEL

As for the N-mixture model (Section 6.2), we now want to derive the basic static site-occupancy model from first principles by thinking about the processes that underlie the observed detection/nondetection data. We ask three questions, and their answers will naturally lead to the basic site-occupancy model.

Question 1: Assume that 100 sites were inhabited each by a certain number of individuals of a study species of your choice, i.e., each site has some value of abundance N, such as $N = \{0, 1, 3, 1, 4, 0\}$ for the first six sites. However, perhaps you cannot measure abundance reliably, and only a summary of N, presence ($N > 0$) or absence ($N = 0$) is recorded. Hence, the true state of interest is then the occurrence, or the presence/absence state, which we denote by z, which would be $z = \{0, 1, 1, 1, 1, 0\}$ for these sites. If we want to model presence/absence, we want to treat z as the realization of a random variable, i.e., as the output from a named stochastic process, or statistical distribution. The process should accommodate both the randomness in the observed data as well as patterns that hold on average only and that we can describe in a GLM manner by introducing covariates into the expectation. So the first question is: **What is the customary statistical description of such presence/absence states?**

Arguably, the natural answer is a Bernoulli distribution for presence/absence at site i. We would write $z_i \sim Bernoulli(\psi)$, where ψ is the expected proportion of sites that are occupied, or the occupancy probability.

Question 2: Every naturalist and also every ecologist (even every statistical ecologist and perhaps even a statistician sometimes) who has *ever* set his foot into the field must know the ugly truth of presence/absence studies—that a species may sometimes be missed where it occurs. This induces a specific type of error in our *presence/absence measurement* (y)—sometimes, we measure an absence ($y = 0$) at a presence site (with $z = 1$); this represents a false-negative error. For instance, if we went to some occupied sites twice, we have the following presence/absence measurements that are possible: {0, 0}, {1, 0}, {0, 1}, and {1, 1}. So the second question is: **What is a sensible statistical model for the measurement error process at an occupied site?**

We think that the natural answer would again be a Bernoulli distribution. That is, we could write $(y_i|z_i = 1) \sim Bernoulli(p)$, where p is detection probability, i.e., the complement of the false-negative measurement error.

Question 3: And what about an absence site, where $z = 0$; what are possible presence/absence measurements and what statistical model might we choose for this process? Basically, we could again measure either a presence ($y = 1$) or an absence ($y = 0$) and a sensible model would be another Bernoulli, $(y_i|z_i = 0) \sim Bernoulli(q)$, where q would represent the probability of a false-positive error. However, in virtually all situations false-positives are far scarcer than false-negatives. Therefore, for the moment we ignore them and assume $q = 0$, i.e., that the false-positive error probability is zero. Standard occupancy models make this assumption, but in Chapter 19 we will see how we can relax the assumption and encounter occupancy models that enable estimation of both false-negative and false-positive measurement errors with an observation model that looks like this: $y_i \sim Bernoulli(z_i p + (1 - z_i)q)$ (Royle and Link, 2006; Miller et al., 2011, 2013b; Sutherland et al., 2013; Chambert et al., 2015).

If we combine these answers, we obtain *exactly* the basic site-occupancy model that was independently developed by MacKenzie et al. (2002) and Tyre et al. (2003); see Section 4.3 in MacKenzie et al. (2006) for some history. Thus, we have just reinvented the site-occupancy model from first principles by thinking about the processes that plausibly underlie the observed presence/absence data. This ability, to sequentially incorporate into a statistical model multiple, linked processes underlying an observed outcome, is one of the principal benefits of hierarchical modeling (Royle and Dorazio, 2008). Related to this is the benefit that hierarchical modeling almost enforces on us a more mechanistic thinking about the multiple processes that produce an observed data set (Kéry and Schaub, 2012).

In summary, here is the simplest site-occupancy model written in algebra:

1. State process: $z_i \sim Bernoulli(\psi)$
2. Observation process: $y_{ij}|z_i \sim Bernoulli(z_i p)$

The latent variable z_i is the true state of occurrence at site i ($i = 1 \ldots M$) and the Bernoulli parameter ψ is the expected value of z, called the probability of occupancy or of presence. The observed variable y_{ij} is our *measurement* of occurrence at site i during survey j ($j = 1 \ldots J$) and is conditional on z_i, and p is the detection probability of the study species at site i during survey j, i.e., the complement of the presence/absence (false-negative) measurement error. Detection probability here refers to *all* individuals inhabiting a site together and not to each individual singly as in the N-mixture model (Conceptually, the two are related as $P^* = 1 - (1 - p)^N$, where P^* is the per-site detection probability, p is the per-individual detection probability and N is the number of individuals at the site; see Section 6.13.1.). The outcome of the observation process is conditional on the outcome of the state process, because the parameter of the second Bernoulli distribution is the product of z_i and p. Thus, at unoccupied sites, this product is zero and

only zero observations (absence measurements) can be made. It is here where our assumption about the absence of false-positives is manifest, i.e., that we assume that a species can be overlooked where it occurs but not erroneously recorded where it is absent.

Analogous to the *N*-mixture model, this hierarchical model can be described as consisting of two linked GLMs: a Bernoulli regression for the spatial variation in occurrence and another Bernoulli regression for the spatiotemporal variation of the observed detection/nondetection data at specific sites. The site-occupancy model is thus a hierarchical extension of a Bernoulli GLM or logistic regression. Logistic regression is the natural building block for models of occurrence (Royle and Dorazio, 2008) and is also the most widely used model for false-negative observation errors (i.e., imperfect detection; Kéry and Schaub, 2012). The site-occupancy model therefore combines the canonical model for species occurrence with the canonical model for imperfect detection. It is also a Bernoulli/Bernoulli mixture model. Recognizing the GLM character of the model, it becomes obvious that we can thus again start doing things that we do with GLMs, namely model structure in the parameters ψ and p, by first indexing them by site and site and time, respectively, and then expressing them as linear or other functions of covariates via some link function, or by the introduction of random effects to model hidden structure and correlations. We will see many examples of this in this chapter and throughout the book.

As for the *N*-mixture model (Chapter 6), we need repeated measurements of presence/absence for at least some sites; otherwise the parameters of the two parts of the model cannot be estimated separately. However, it is *not* required that we have the same number of repeated measurements at all sites, nor even that we have replicate observations for *all* sites! For instance, many site-occupancy models for species distributions will have replicate observations for only a minority of the sites; e.g., Kéry et al. (2010a,b), and Kéry (2011b). Nevertheless, the more replication the better (unless we risk violating the closure assumption; see below). If we do not have a balanced design with the same number of replicates at each site, it is best if the number of surveys per site is randomly allocated to a site. If it depends instead on some site characteristics, biased estimates may result. For instance, if multiple surveys are only undertaken at the "better" sites, where density and therefore detection probability (p) may be higher on average, the resulting estimate of p will be biased high with respect to all sites and therefore the occupancy estimator will be biased low.

Some authors have proposed variants of *N*-mixture (Solymos et al., 2012) and site-occupancy models for unreplicated surveys (Lele et al., 2012). Their models buy parameter identifiability by making very strong parametric assumptions about the covariates (Knape and Korner-Nievergelt, 2015). These assumptions are critical and they may well hold in some cases, but in others they may not, and it is not clear how they could be tested. Therefore, it appears risky to us to base inference about both ecological state and measurement error on unreplicated data alone. However, such data may of course be combined with data sets that *do* have replication in a form of integrated model (see Chapter 23).

You can fit a large array of occupancy models in a number of free software that use MLE. Most of all, program PRESENCE (Hines, 2006) has been developed specifically for occupancy models and allows you to fit a very large range of models for occurrence and also some for abundance. Then, MARK (White and Burnham, 1999) also contains a large number of occupancy models. Gimenez et al. (2014) have shown how E-SURGE (Choquet et al., 2009b), a powerful software for fitting hidden Markov models such as CJS and multistate models (see Chapter 15 in volume 2), can be tweaked to fit occupancy models as well. And of course, in this book we use `unmarked` and BUGS software.

The main assumptions of the basic site-occupancy model are the following. We will discuss them all in more detail later.

1. *Closure assumption*: We require that the presence/absence state z_i of site i does not change over the course of the study. This typically means that we will only use detection/nondetection data from a time period that is short relative to the dynamics of the modeled system. This may be hours or days if we model the occurrence of insects and years if we model the occurrence of trees. Certain violations of the closure assumption, corresponding to random temporary emigration, are usually not disastrous; they simply require one to interpret the occupancy parameter as *probability of use* sometime during the study period (i.e., of a site ever being occupied), rather than the probability of permanent occurrence. In addition, given the right kind of data (usually some form of temporal subsampling) one may estimate the probability of being temporarily absent formally in a multiscale occupancy model; see Section 10.10 for the occupancy case and Sections 6.14, 7.4, and 9.5 for related *N*-mixture models for abundance. Thus, if we have data collected under the so-called robust design (RD), closure is no problem. The RD denotes a sampling protocol with temporal replication at two scales, representing primary and nested secondary occasions (Williams et al., 2002). We assume that the system may change between primary occasions but is closed between secondary occasions within a primary occasion. With such data, we can simply fit the static model to each primary period separately or (and this results in identical parameter estimates) we fit a model to all data at once but fit separate parameters for every primary occasion. Alternatively, we can fit a dynamic model, where the change in occurrence is governed by parameters of persistence and colonization (see Chapter 16 in volume 2).
2. *No false positive errors*: This is an important assumption, since its violation can lead to strong bias in the occupancy estimator (Royle and Link, 2006; McClintock et al., 2010b; Miller et al., 2015). A common way to avoid false positives is to discard any observation with doubtful species identification or rather to treat it as a zero. This will lower the detection probability (if the record did in fact refer to the study species) but eliminate the deleterious false-positive errors (if it did not). There is important new work on joint estimation of both false-positive and false-negative measurement errors in occupancy models pioneered by Miller et al. (2011, 2013b) (see also Bailey et al., 2014, and Chambert et al., 2015). We review these models in Chapter 19 in volume 2.
3. *Independence of occurrence and independence of detection*: The former assumption means that occupancy probability at one site should be independent from the occupancy probability at another site, except insofar as we can explain such associations with covariates. The second assumption means that detection probability at a site should be independent across replicated visits. The most likely way in which the independence of occurrence assumption is violated is by mechanisms that lead to spatial autocorrelation and this can modeled; see Chapters 21 and 22. The most likely way in which the independence of detection assumption is violated is by "behavioral response" (Riddle et al., 2010), and this can also be modeled. A third case would be independence of the measurement error from the ecological state, i.e., the lack of density-dependent detection in a model for abundance (see Section 6.15), but this is not an issue in occupancy models, since for them detection is always conditional upon presence.
4. *Homogeneity of detection probability*: Unexplained heterogeneity in detection can greatly bias estimators (Miller et al., 2015). Specifically, unmodeled site-specific heterogeneity in detection

will lead to underestimates of occupancy (this is the second law of capture-recapture; Royle, 2006; Dorazio, 2007). We can try to eliminate such heterogeneity by modeling it via known covariates, by adopting the Royle-Nichols model (Section 6.13.1), using the *N*-mixture model if counts are available (Royle et al., 2005; Dorazio, 2007), or by modeling latent structure via finite or continuous mixture distributions (Royle, 2006). This can easily be achieved either in `unmarked` or else in BUGS.

5. *Parametric assumptions*: We assume that the two Bernoulli variables (typically with some covariates and potentially other structure in the mean) are a reasonable abstraction of reality in order to meet the objectives of the modeling. This and some of the other assumptions can be assessed with goodness-of-fit tests; see Section 10.8.

10.3 SIMULATION AND ANALYSIS OF THE SIMPLEST POSSIBLE SITE-OCCUPANCY MODEL

We illustrate and explain the simplest possible site-occupancy model using data simulation and analysis in `unmarked` and BUGS. We simulate data as described in 10.2, i.e., for a data set with constant occupancy (which we assume to be 0.8) and constant detection (assumed to be 0.5), collected at 100 sites with two presence/absence measurements each.

```
# Choose sample sizes and prepare observed data array y
set.seed(24)                              # So we all get same data set
M <- 100                                  # Number of sites
J <- 2                                    # Number of presence/absence measurements
y <- matrix(NA, nrow = M, ncol = J) # to contain the obs. data

# Parameter values
psi <- 0.8                                # Probability of occupancy or presence
p <- 0.5                                  # Probability of detection

# Generate presence/absence data (the truth)
z <- rbinom(n = M, size = 1, prob = psi)     # R has no Bernoulli

# Generate detection/nondetection data (i.e. presence/absence measurements)
for(j in 1:J){
   y[,j] <- rbinom(n = M, size = 1, prob = z*p)
}

# Look at data
sum(z)                                    # True number of occupied sites
[1] 86

sum(apply(y, 1, max))                     # Observed number of occupied sites
[1] 61
```

Thus, in our simulation the species occurs at 86 sites and is detected at 61. The overall measurement error for the apparent number of occupied sites is thus $(86 - 61)/86 = -29\%$. Under our binomial model we'd expect a combined detection probability (over J surveys) of $1 - (1 - p)^J = 75\%$,

i.e., a total measurement error of −25%. This difference between −29% and −25% is of course due to the sampling error inherent in the stochastic detection process. Now we inspect our data set:

```
head(cbind(z = z, y1 = y[,1], y2 = y[,2]))    # Truth and measurements for first 6 sites
      z y1 y2
[1,] 1  1  1
[2,] 1  1  1
[3,] 1  0  0
[4,] 1  0  1
[5,] 1  1  1
[6,] 0  0  0
```

Sites 1−5 are presence sites, while site 6 is unoccupied. Since we exclude false-positives, we will never observe the species at an absence site, but we may fail to detect it at a presence site. The first five sites illustrate three of the four possible detection histories at an occupied site: {1, 1} for sites 1, 2 and 5, {0, 1} at site 4, and {0,0} at site 3. You can look at the entire simulated, observed data *y* to see the fourth possible history, {1, 0}, first occurring at sites 25−27.

We now analyze the data with `unmarked` using function `occu`, where the linear model for detection is specified before that for occupancy.

```
library(unmarked)
umf <- unmarkedFrameOccu(y = y)    # Create unmarked data frame
summary(umf)                       # Summarize data frame
(fm1 <- occu(~1 ~1, data = umf))   # Fit model

Call:
occu(formula = ~1 ~ 1, data = umf)

Occupancy:
 Estimate    SE     z  P(>|z|)
     1.04 0.394 2.65 0.00807

Detection:
 Estimate   SE    z P(>|z|)
    0.329 0.26 1.26   0.207

AIC: 270.2257

backTransform(fm1, "state")      # Get estimates on probability scale
backTransform(fm1, "det")

Backtransformed linear combination(s) of Occupancy estimate(s)
 Estimate      SE LinComb (Intercept)
     0.74  0.0759    1.04           1

Backtransformed linear combination(s) of Detection estimate(s)
 Estimate     SE LinComb (Intercept)
    0.581 0.0634   0.329           1
```

We observed the species at 61% of the sites, but we estimate that it really occurs at 74%, because detection probability is estimated at 58% for a single survey. Next, we conduct a Bayesian analysis of the model with JAGS.

```
# Bundle data and summarize data bundle
str( win.data <- list(y = y, M = nrow(y), J = ncol(y)) )

# Specify model in BUGS langauge
sink("model.txt")
cat("
model {
# Priors
   psi ~ dunif(0, 1)
   p ~ dunif(0, 1)
# Likelihood
 for (i in 1:M) {                  # Loop over sites
    z[i] ~ dbern(psi)              # State model
    for (j in 1:J) {               # Loop over replicate surveys
       y[i,j] ~ dbern(z[i]*p)      # Observation model (only JAGS !)
#      y[i,j] ~ dbern(mu[i])       # For WinBUGS define 'straw man'
    }
#  mu[i] <- z[i]*p                # Only WinBUGS
  }
}
",fill = TRUE)
sink()

# Initial values
zst <- apply(y, 1, max)            # Avoid data/model/inits conflict
inits <- function(){list(z = zst)}

# Parameters monitored
params <- c("psi", "p")

# MCMC settings
ni <- 5000 ; nt <- 1 ; nb <- 1000 ; nc <- 3

# Call JAGS and summarize posteriors
library(jagsUI)
fm2 <- jags(win.data, inits, params, "model.txt", n.chains = nc,
   n.thin = nt, n.iter = ni, n.burnin = nb)
print(fm2, dig = 3)
        mean     sd   2.5%    50%  97.5% overlap0 f Rhat n.eff
psi    0.749  0.077  0.607  0.744  0.912    FALSE 1    1 12000
p      0.573  0.062  0.449  0.574  0.690    FALSE 1    1  4876
```

As usual, we get Bayesian estimates that are very similar to those using MLE (and we would get more similar ones still with a larger data set). As for the related *N*-mixture model (Section 6.3) we point out the striking similarity of a hierarchical model when written in algebra, in R when simulating the data, and in BUGS when fitting the model (Table 10.1). This illustrates well our frequent claims that once you know how to write a model in algebra, you're almost there at fitting it in BUGS, and that algebra, data simulation, and the BUGS language are similarly precise and useful ways of describing a hierarchical or, indeed, any model.

Table 10.1 Occupancy model descriptions in terms of algebra, data simulation code in R, and BUGS language. The latter is for JAGS only; for WinBUGS we have to define the success probability of the Bernoulli in the observation model outside as a "straw man" (see BUGS model code above).

	Algebraic Description	R Data Simulation	BUGS Model Statement
State model	$z_i \sim Bernoulli(\psi)$	`z <- rbinom(M,1,psi)`	`z[i] ~ dbern(psi)`
Obs. model	$y_{ij}\|z_i \sim Bernoulli(z_i p)$	`y[,j] <- rbinom(M,1,z*p)`	`y[i,j] ~ dbern(z[i]*p)`

Finally, when no patterns over time (i.e., across the J replicate surveys) are modeled, the observation model can be simplified by fitting the model to the aggregated site-specific data, where y_i now is the detection frequency, i.e., the number of times over J surveys a species was detected.

$$y_i|z_i \sim Binomial(J, z_i p)$$

This is exactly the same model and will lead to exactly the same estimates, but is computationally more efficient (and may be worthwhile especially for large data sets or when the state model is very complex). If the number of replicates J is variable across sites, it must be made a vector; see Exercise 1.

This completes our simulation-based introduction to the simplest possible occupancy model, that with only an intercept in both parts of the model. In the next section, we illustrate a slightly more realistic and interesting model with one covariate in each model component.

10.4 A SLIGHTLY MORE COMPLEX SITE-OCCUPANCY MODEL WITH COVARIATES

We will hardly ever use the null/null site-occupancy model from the previous section but will typically be interested in effects of covariates, e.g., to model environmental effects on occupancy. In this section, we show covariate modeling and predictions of occupancy and detection, discuss the difference between the estimate of occupancy probability versus that of the realized occurrence state, and do a bootstrap assessment of uncertainty. We work with simulated data once again (note that in Section 10.9 you will see a similar analysis of a real data set). We mostly work with `unmarked`, but fit one model with BUGS as well.

We simulate data under the following model:

$$z_i \sim Bernoulli(\psi_i), \text{ with } \text{logit}(\psi_i) = \beta_0 + \beta_1 * \text{vegHt}_i$$

$$y_{ij}|z_i \sim Bernoulli(z_i p_{ij}), \text{ with } \text{logit}(p_{ij}) = \alpha_0 + \alpha_1 * \text{wind}_{ij}$$

Occupancy is affected by a site covariate (vegetation height) and detection is affected by a sampling, or observational covariate (wind speed).

```
# Choose sample sizes and prepare obs. data array y
set.seed(1)                           # So we all get same data set
M <- 100                              # Number of sites
J <- 3                                # Number of presence/absence measurements
y <- matrix(NA, nrow = M, ncol = J)   # to contain the obs. data

# Create a covariate called vegHt
vegHt <- sort(runif(M, -1, 1))        # Sort for graphical convenience

# Choose parameter values for occupancy model and compute occupancy
beta0 <- 0                            # Logit-scale intercept
beta1 <- 3                            # Logit-scale slope for vegHt
psi <- plogis(beta0 + beta1 * vegHt) # Occupancy probability
# plot(vegHt, psi, ylim = c(0,1), type = "l", lwd = 3) # Plot psi relationship

# Now visit each site and observe presence/absence perfectly
z <- rbinom(M, 1, psi)                # True presence/absence

# Look at data so far
table(z)
z
 0  1
49 51

# Plot the true system state
par(mfrow = c(1, 3), mar = c(5,5,2,2), cex.axis = 1.5, cex.lab = 1.5)
plot(vegHt, z, xlab="Vegetation height", ylab="True presence/absence (z)", frame = F,
cex = 1.5)
plot(function(x) plogis(beta0 + beta1*x), -1, 1, add=T, lwd=3, col = "red")
```

Figure 10.1 (left) shows how the relationship between occupancy probability and `vegHt` translates into a pattern of presence/absence. Of course, this is hardly ever the whole story behind "presence/absence data." Rather, there will almost always be false-negative measurement errors. Occurrence z becomes a latent state then, i.e., it will be only partially observable. We simulate this next and imagine that detection probability p is related to the covariate `wind` via a logit-linear regression with intercept -2 and slope -3 and that we make $J = 3$ presence/absence measurements at each site (Figure 10.1 middle).

```
# Create a covariate called wind
wind <- array(runif(M * J, -1, 1), dim = c(M, J))

# Choose parameter values for measurement error model and compute detectability
alpha0 <- -2                          # Logit-scale intercept
alpha1 <- -3                          # Logit-scale slope for wind
p <- plogis(alpha0 + alpha1 * wind)   # Detection probability
# plot(p ~ wind, ylim = c(0,1))       # Look at relationship

# Take J = 3 presence/absence measurements at each site
for(j in 1:J) {
    y[,j] <- rbinom(M, z, p[,j])
}

sum(apply(y, 1, max))                 # Number of sites with observed presences
[1] 32
```

```
# Plot observed data and true effect of wind on detection probability
plot(wind, y, xlab="Wind", ylab="Observed det./nondetection data (y)", frame = F,
cex = 1.5)
plot(function(x) plogis(alpha0 + alpha1*x), -1, 1, add=T, lwd=3, col = "red")

# Look at the data: occupancy, true presence/absence (z), and measurements (y)
cbind(psi=round(psi,2), z=z, y1=y[,1], y2=y[,2], y3=y[,3])
       psi  z  y1  y2  y3
 [1,] 0.05 0   0   0   0
 [2,] 0.05 0   0   0   0
 [3,] 0.07 0   0   0   0
 [4,] 0.07 1   0   0   0
 [5,] 0.07 0   0   0   0
   [ output truncated]
[91,] 0.90 1   1   0   0
[92,] 0.91 1   1   0   0
[93,] 0.91 1   0   0   0
[94,] 0.92 0   0   0   0
[95,] 0.92 1   0   1   1
  [ output truncated]
```

We suggest that you look at this table to make sure you *really* understand the relationships among ψ (psi), z, and y. Next, we use the site-occupancy model to analyze these data using `unmarked` and BUGS. We start with `unmarked` and will also illustrate the fitting of two factors that are unrelated to the data (because the response was not generated with their effects "built in"): `time` will index the first through the third survey, while `hab` will contrast three imaginary habitat types.

```
# Create factors
time <- matrix(rep(as.character(1:J), M), ncol = J, byrow = TRUE)
hab <- c(rep("A", 33), rep("B", 33), rep("C", 34))  # Must have M = 100
```

To fit the model in `unmarked`, we package the data into an `unmarked` frame first. Note the difference between site covariates (indexed by site only) and sampling or observational covariates (indexed by site and survey). There is really a third possible type of covariate, for time, but in `unmarked`, this type has to be specified as an observational covariate, as we see for factor `time`, which codes for the first to the third survey.

```
# Load unmarked, format data and summarize
library(unmarked)
umf <- unmarkedFrameOccu(
   y = y,                                          # Pres/Abs measurements
   siteCovs = data.frame(vegHt = vegHt, hab = hab), # site-specific covs.
   obsCovs = list(wind = wind, time = time))       # obs-specific covs.
summary(umf)

unmarkedFrame Object

100 sites
Maximum number of observations per site: 3
Mean number of observations per site: 3
Sites with at least one detection: 32
```

```
Tabulation of y observations:
   0    1 <NA>
 255   45    0

Site-level covariates:
     vegHt          hab
 Min.   :-0.97322  A:33
 1st Qu.:-0.35384  B:33
 Median :-0.02438  C:34
 Mean   : 0.03569
 3rd Qu.: 0.53439
 Max.   : 0.98381

Observation-level covariates:
      wind           time
 Min.   :-0.99633   1:100
 1st Qu.:-0.54111   2:100
 Median :-0.10469   3:100
 Mean   :-0.03803
 3rd Qu.: 0.44824
 Max.   : 0.99215

# Fit model and extract estimates
# Detection covariates follow first tilde, then occupancy covariates
summary(fm1.occ <- occu(~wind ~vegHt, data=umf))

Call:
occu(formula = ~wind ~ vegHt, data = umf)

Occupancy (logit-scale):
             Estimate    SE      z  P(>|z|)
(Intercept)   -0.136 0.449 -0.303 0.76177
vegHt          2.432 0.839  2.897 0.00377

Detection (logit-scale):
             Estimate    SE     z       P(>|z|)
(Intercept)     -1.70 0.398 -4.27 0.000019825
wind            -3.07 0.594 -5.16 0.000000243

AIC: 183.4468
Number of sites: 100
optim convergence code: 0
optim iterations: 36
Bootstrap iterations: 0

# Predict occupancy and detection as function of covs (with 95% CIs)
# Add truth from data simulation (below for full code to produce fig. 10-2)
newdat <- data.frame(vegHt=seq(-1, 1, 0.01))
pred.occ <- predict(fm1.occ, type="state", newdata=newdat)
newdat <- data.frame(wind=seq(-1, 1, 0.1))
pred.det <- predict(fm1.occ, type="det", newdata=newdat)
```

```
# Predictions for specified values of vegHt, say 0.2 and 2.1
newdat <- data.frame(vegHt=c(0.2, 2.1))
predict(fm1.occ, type="state", newdata=newdat, append = T)
  Predicted          SE      lower     upper  vegHt
1 0.5866972 0.12345081 0.3435511 0.7938296    0.2
2 0.9931116 0.01301558 0.7759108 0.9998334    2.1

# ... for values of wind of -1 to 1
newdat <- data.frame(wind=seq(-1, 1, , 5))
predict(fm1.occ, type="det", newdata=newdat, append = T)
    Predicted           SE       lower     upper wind
1 0.797525949 0.081884645 0.593153464 0.9141025 -1.0
2 0.459433439 0.085800844 0.301588034 0.6258607 -0.5
3 0.154968761 0.052056383 0.077610026 0.2855637  0.0
4 0.038064095 0.022366542 0.011810009 0.1158403  0.5
5 0.008465936 0.007340634 0.001535816 0.0452499  1.0
```

We may summarize the analysis by plotting the observed data (the observed occurrence state of every site), the true data-generating values, and the estimated relationship between occupancy and covariate `vegHt` under the occupancy model (a "logistic regression" that does account for imperfect detection p) and under a simple logistic regression that does not account for p (Figure 10.2 right). We see that ignoring imperfect detection leads to (1) underestimation of the extent of occurrence and (2) to a bias toward zero (attenuation) of the regression coefficient of `vegHt`.

```
# Fit detection-naive GLM to observed occurrence and plot comparison
summary(fm.glm <- glm(apply(y, 1, max) ~ vegHt, family=binomial))
plot(vegHt, apply(y, 1, max), xlab="Vegetation height", ylab="Observed occurrence ('ever
observed ?')", frame = F, cex = 1.5)
plot(function(x) plogis(beta0 + beta1*x), -1, 1, add=T, lwd=3, col = "red")
lines(vegHt, predict(fm.glm,,"response"), type = "l", lwd = 3)
```

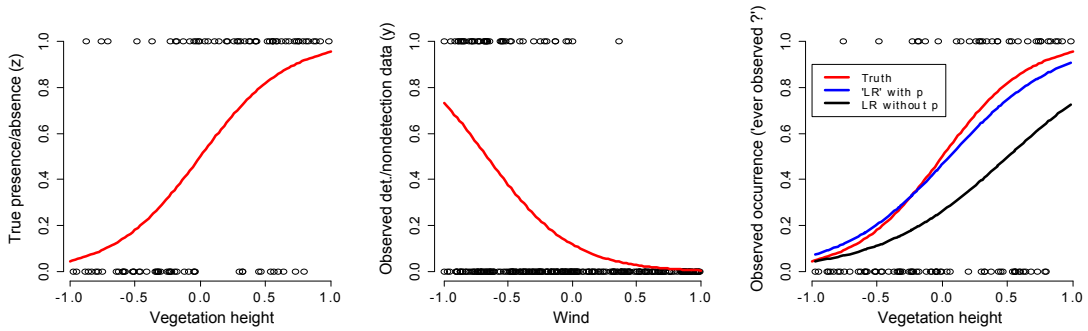

FIGURE 10.2

Left: The relationship between occurrence and vegetation height (red line); circles represent the true presence or absence (z) at each site. Middle: The relationship between detection probability and wind speed (red line); circles are the observed detection/nondetection (or "presence/absence") data. Right: Comparison between the true occupancy-covariate relationship (red) and its estimate under the occupancy model (blue) and a simple logistic regression (black); circles represent the observed occurrence status of each site.

```
lines(vegHt, predict(fm1.occ, type="state")[,1], col = "blue", lwd = 3)
legend(-1, 0.9, c("Truth", "'LR' with p", "LR without p"), col=c("red", "blue", "black"),
lty = 1, lwd=3, cex = 1.2)
```

Most of the data formatting in an `unmarked` frame, the model fitting, and the processing of the results in an occupancy analysis using function `occu` is very similar to what we did with the different N-mixture `unmarked` model-fitting functions in Chapters 6–9. Similarly, the binary random effects z_i can be estimated using the function `ranef`. These random effects have a very tangible meaning—they are the presence/absence state at each site, and their estimates represent our best guess of whether a particular site is occupied or not.

```
ranef(fm1.occ)
             Mean  Mode  2.5%  97.5%
 [1,] 0.039231087     0     0      1
 [2,] 0.005800029     0     0      0
 [3,] 0.067784966     0     0      1
 [4,] 0.044980764     0     0      1
 [5,] 0.032297869     0     0      1
   [ output truncated]
[91,] 1.000000000     1     1      1
[92,] 1.000000000     1     1      1
[93,] 0.776602771     1     0      1
[94,] 0.580214449     1     0      1
[95,] 1.000000000     1     1      1
      [ output truncated ]
```

These predictions of the random effects z are also called *conditional occupancy probability*, where conditional means "given the observed data at that site" (MacKenzie et al., 2006, pp. 97–98). When a species has been detected at least once at a site, under the usual assumption of no false-positives the site is occupied with certainty. This is why for sites 91, 92, and 95 in our example the conditional occupancy probability is equal to 1 with zero uncertainty. The case is more interesting for a site where a species was never detected during the J surveys, i.e., $\{y_i\} = 0$. The probability that site i is occupied then depends on three things: the expected occupancy probability for the site (ψ), detection probability for the site (p), and the number of surveys J:

$$\Pr(z_i = 1|\{y_i\} = 0) = \frac{\psi(1-p)^J}{(1-\psi) + \psi(1-p)^J}$$

This result follows directly from an application of Bayes' rule (Section 2.5.1) and makes sense intuitively—all else equal, given that the species was not observed at a site, we have higher confidence in its presence despite the negative survey results (1) when it is widespread overall (i.e., when occupancy probability ψ is high), (2) when it is elusive (i.e., when detection probability p is small), and (3) when the number of times we have looked for it (J) is small. When ψ is site- and p site- and survey-specific the equation changes to:

$$\Pr(z_i = 1|\{y_i\} = 0) = \frac{\psi_i \prod_{j=1}^{J}(1-p_{ij})}{(1-\psi_i) + \psi_i \prod_{j=1}^{J}(1-p_{ij})}$$

Let's double check this for site 1, where after three surveys the species was never detected. The probabilities of occupancy (1 value) and detection (1 value for each survey) for this site can be obtained from `unmarked` as follows:

```
(psi1 <- predict(fm1.occ, type="state")[1,1])
[1] 0.07565784
(p1 <- predict(fm1.occ, type="det")[c(1:3),1])
[1] 0.43290197 0.05942820 0.06472325
```

(Important note: The predictions of detection for the three surveys made at site 1 are in rows 1–3 and *not* in rows 1, 101, and 201, as you might perhaps think.) We can now calculate the conditional occupancy probability for site 1, given that all three surveys resulted in a negative result, as follows, and will find that it matches up the solution for site 1 obtained from the `ranef` function.

```
(z1 <- (psi1 * prod(1-p1)) / ((1 - psi1) + psi1 * prod(1-p1)))
[1] 0.03923109
```

One quantity that is frequently of interest is the finite-sample occupancy, i.e., the number of sites occupied in the sample of sites actually studied. In `unmarked`, we can obtain this quantity by summing over the estimates of the random effects z_i and for a confidence interval use a parametric bootstrap. For the latter, we first need to define a function that computes the finite-sample estimate of the number of sites occupied. Then, we use it for a large number of bootstrap samples to obtain uncertainty intervals around that estimate (Figure 10.3 left).

```
# Define function for finite-sample number and proportion of occupied sites
fs.fn <- function(fm){
   Nocc <- sum(ranef(fm)@post[,2,])
   psi.fs <- Nocc / nrow(fm@data@y)
   out <- c(Nocc = Nocc, psi.fs = psi.fs)
   return(out)
}
```

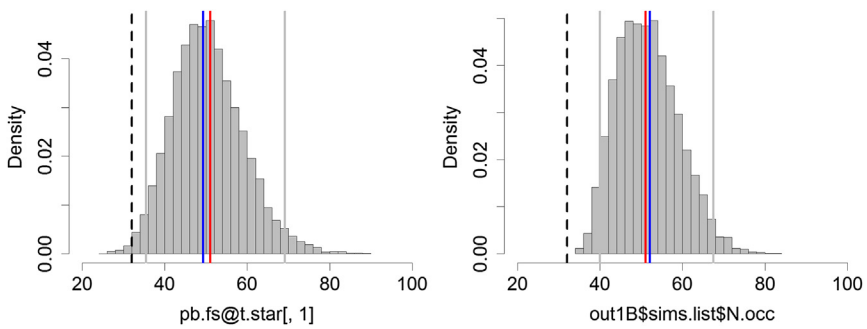

FIGURE 10.3

Bootstrap distribution (left) and posterior distribution (right) of the number of occupied sites in the actual sample of 100 surveyed sites (finite-sample occupancy). For the bootstrap, the point estimate (50.5) is the blue line and the 95% CI (35.4, 69.1) is shown in grey. The truth is 51 (red), and the species was observed at a total of 32 sites (dashed black line). For the Bayesian analysis, the point estimate is 52.0 and the 95% CRI is (40, 67.5).

```
# Bootstrap the function
fs.hat <- fs.fn(fm1.occ)           # Point estimate
pb.fs <- parboot(fm1.occ, fs.fn, nsim=10000, report=2) # Takes a while (33 min)
# system.time(pb.fs <- parboot(fm.occ1, fs.fn, nsim=100, report=10)) # quicker

# Summarize bootstrap distributions
summary(pb.fs@t.star)
      Nocc           psi.fs
 Min.   :24.08  Min.   :0.2408
 1st Qu.:44.41  1st Qu.:0.4441
 Median :49.93  Median :0.4993
 Mean   :50.51  Mean   :0.5051
 3rd Qu.:55.87  3rd Qu.:0.5587
 Max.   :89.11  Max.   :0.8911

# Get 95% bootstrapped confidence intervals
 (tmp1 <- quantile(pb.fs@t.star[,1], prob = c(0.025, 0.975)))
    2.5%    97.5%
35.44263 69.14740

(tmp2 <- quantile(pb.fs@t.star[,2], prob = c(0.025, 0.975)))
     2.5%     97.5%
0.3544263 0.6914740

# Plot bootstrap distribution of number of occupied sites (Fig. 10-3 left)
par(mfrow = c(1,2), mar = c(5,4,3,2))
hist(pb.fs@t.star[,1], col = "grey", breaks = 80, xlim = c(20, 100), main = "", freq = F)
abline(v = fs.hat[1], col = "blue", lwd = 3)          # add point estimate
abline(v = tmp1, col = "grey", lwd = 3)               # add 95% CI
abline(v = sum(apply(y, 1, max)), lty = 2, lwd = 3)   # observed #occ sites
abline(v = sum(z), col = "red", lwd = 3)              # true #occ sites
```

What is the difference between the estimates that you obtain with `predict` and those that you get from `ranef` ? To understand this, look at the state model:

$$z_i \sim Bernoulli(\psi_i)$$

In short, `predict` yields estimates of the population parameter ψ, i.e., the *expected* presence/absence status for a site *i* that is drawn at random from the same statistical population of sites as the 100 we studied and has the given covariate values. In contrast, `ranef` yields estimates for z_i, i.e., the realized presence/absence status exactly of site *i* in the studied sample of sites, taking into account both the values of the modeled covariates *and* the data y_i observed at that site.

Next, we quickly illustrate the fitting of factors by first fitting what could be called a main-effects ANCOVA linear model for both model parts, i.e., a model with additive effects of a discrete (`hab` and `time`) and of a continuous covariate (`vegHt` and `wind`) for occurrence and detection, respectively. All linear models in `unmarked` are specified in exactly the same way as in other R functions such as `lm` or `glm`. So let's fit a model in the "means parameterizations," where the parameters for the factor levels directly have the meaning of the intercepts for each level. We can write these linear models in algebra as:

$$\text{logit}(\psi_i) = \beta 0_{hab(i)} + \beta 1 * \text{vegHt}_i$$

$$\text{logit}(p_{ij}) = \alpha 0_{time(i)} + \alpha 1 * \text{wind}_{ij}$$

```
# Fit model p(time+wind), psi(hab+vegHt)
summary(fm2.occ <- occu(~time+wind-1 ~hab+vegHt-1, data=umf))
Call:
occu(formula = ~time + wind - 1 ~ hab + vegHt - 1, data = umf)

Occupancy (logit-scale):
     Estimate    SE      z  P(>|z|)
habA   -0.570 1.191 -0.479   0.632
habB    0.476 0.648  0.735   0.462
habC   -1.055 1.276 -0.827   0.408
vegHt   2.869 1.829  1.569   0.117

Detection (logit-scale):
      Estimate   SE     z   P(>|z|)
time1   -1.37 0.500 -2.75 6.01e-03
time2   -2.17 0.530 -4.10 4.05e-05
time3   -1.51 0.522 -2.89 3.88e-03
wind    -3.17 0.619 -5.11 3.16e-07

# Predict occupancy for habitat factor levels at average covariate values
newdat <- data.frame(vegHt=0, hab = c("A", "B", "C"))
predict(fm2.occ, type="state", newdata = newdat, appendData = TRUE)
  Predicted        SE      lower     upper vegHt hab
1 0.3611400 0.2747052 0.05195332 0.8536124     0   A
2 0.6168318 0.1531216 0.31138401 0.8514352     0   B
3 0.2582344 0.2445064 0.02773337 0.8094843     0   C

# Predict detection for time factor levels at average covariate values
newdat <- data.frame(wind=0, time = c("1", "2", "3"))
predict(fm2.occ, type="det", newdata=newdat, appendData = TRUE)
  Predicted         SE      lower     upper wind time
1 0.2020662 0.08060966 0.08680271 0.4028639    0    1
2 0.1020261 0.04854268 0.03866499 0.2429754    0    2
3 0.1813491 0.07748170 0.07377254 0.3812293    0    3
```

See Section 6.4 for how we form predictions for one specific level of a factor. For the sake of exercise, let's now also fit a model with interaction effects in both the occupancy and the detection model and then use a likelihood ratio test to decide which is better supported by the data. In algebra, that model can be written as this:

$$\text{logit}(\psi_i) = \beta 0_{hab(i)} + \beta 1_{hab(i)} * \text{vegHt}_i$$

$$\text{logit}(p_{ij}) = \alpha 0_{time(i)} + \alpha 1_{time(i)} * \text{wind}_{ij}$$

The difference in this model is that now the slope parameters of `vegHt` and `wind` ($\beta 1$ and $\alpha 1$, respectively) are no longer a single number (scalar), but they are indexed and hence vary over the levels of the two factors `hab` and `time`. Hence, they are now vectors of length 3, corresponding to the three levels of the factors `hab` and `time`.

```
# Fit model p(time*wind), psi(hab*vegHt)
summary(fm3.occ <- occu(~time*wind-1-wind ~hab*vegHt-1-vegHt, data=umf))
```

```
# Do likelihood ratio test
LRT(fm2.occ, fm3.occ)
     Chisq DF Pr(>Chisq)
1 6.233802  4   0.182355
```

The test says that interactive effects with the two continuous explanatory variables (i.e., model 3) are not preferred over a model with additive effects (model 2).

As a final part in this section, we illustrate the fitting of a simple occupancy model with covariates in BUGS and also show the forming of predictions (estimates of ψ), estimation of the realized presence/absence status of a site (estimates of z) and of the finite-sample occupancy, i.e., the number and proportion of occupied sites in the studied sample of 100 sites. As always in BUGS, knowing how to write a model in algebra gets us very close to the BUGS model description. We add three types of derived quantities: the number of occupied sites in the sample of 100 study sites (`N.occ`) and predictions of occupancy and of detection for a range of values of the covariates `vegHt` and `wind`, respectively. For the latter, we provide two sets of covariate values spaced evenly in the range over which predictions are desired (`XvegHt`, `Xwind`). We want to see the estimates of presence/absence at each site (`z`), so we add those in the list of parameters to be saved below.

```
# Bundle and summarize data set
str( win.data <- list(y = y, vegHt = vegHt, wind = wind, M = nrow(y), J = ncol(y), XvegHt =
seq(-1, 1, length.out=100), Xwind = seq(-1, 1, length.out=100)) )

# Specify model in BUGS language
sink("model.txt")
cat("
model {

# Priors
mean.p ~ dunif(0, 1)              # Detection intercept on prob. scale
alpha0 <- logit(mean.p)           # Detection intercept
alpha1 ~ dunif(-20, 20)           # Detection slope on wind
mean.psi ~ dunif(0, 1)            # Occupancy intercept on prob. scale
beta0 <- logit(mean.psi)          # Occupancy intercept
beta1 ~ dunif(-20, 20)            # Occupancy slope on vegHt

# Likelihood
for (i in 1:M) {
  # True state model for the partially observed true state
  z[i] ~ dbern(psi[i])            # True occupancy z at site i
  logit(psi[i]) <- beta0 + beta1 * vegHt[i]
  for (j in 1:J) {
     # Observation model for the actual observations
     y[i,j] ~ dbern(p.eff[i,j])   # Detection-nondetection at i and j
     p.eff[i,j] <- z[i] * p[i,j]  # 'straw man' for WinBUGS
     logit(p[i,j]) <- alpha0 + alpha1 * wind[i,j]
  }
}
```

```
# Derived quantities
N.occ <- sum(z[])     # Number of occupied sites among sample of M
psi.fs <- N.occ/M     # Proportion of occupied sites among sample of M
for(k in 1:100){
   logit(psi.pred[k]) <- beta0 + beta1 * XvegHt[k] # psi predictions
   logit(p.pred[k]) <- alpha0 + alpha1 * Xwind[k]  # p predictions
}
}
",fill = TRUE)
sink()

# Initial values: must give for same quantities as priors given !
zst <- apply(y, 1, max)          # Avoid data/model/inits conflict
inits <- function(){list(z = zst, mean.p = runif(1), alpha1 = runif(1), mean.psi
= runif(1), beta1 = runif(1))}

# Parameters monitored
params <- c("alpha0", "alpha1", "beta0", "beta1", "N.occ", "psi.fs", "psi.pred",
"p.pred", "z") # Also estimate z = "conditional occ. prob."

# MCMC settings
ni <- 25000  ;  nt <- 10  ;  nb <- 2000  ;  nc <- 3

# Call WinBUGS from R (ART 2 min) and summarize posteriors
out1B <- bugs(win.data, inits, params, "model.txt", n.chains = nc,
n.thin = nt, n.iter = ni, n.burnin = nb, debug = TRUE, bugs.directory = bugs.dir,
working.directory = getwd())
print(out1B, dig = 3)
          mean    sd   2.5%    25%    50%    75%  97.5%  Rhat n.eff
alpha0  -1.739 0.385 -2.510 -1.992 -1.738 -1.479 -0.991 1.001  4900
alpha1  -3.047 0.586 -4.283 -3.419 -3.018 -2.644 -1.971 1.001  6900
beta0    0.080 0.585 -0.836 -0.325  0.001  0.386  1.496 1.001  4900
beta1    3.074 1.294  1.239  2.183  2.818  3.672  6.323 1.001  6900
N.occ   52.050 7.461 40.000 46.000 51.000 57.000 67.523 1.001  6900
psi.fs   0.520 0.075  0.400  0.460  0.510  0.570  0.675 1.001  6900
[   Output truncated  ]
```

We compare the truth with MLEs from `unmarked` and posterior inference from BUGS (Figure 10.4).

```
# Compare truth with MLEs and bayesian posterior inference in table ...
truth <- c(alpha0, alpha1, beta0, beta1, sum(z), sum(z)/M)
tmp <- summary(fm1.occ)
MLEs <- rbind(tmp[[2]][1:2,1:2], tmp[[1]][1:2,1:2], sumZ = c(mean(pb.fs@t.star[,1]),
sd(pb.fs@t.star[,1])), psi.fs = c(mean(pb.fs@t.star[,2]), sd(pb.fs@t.star[,2])))
print(cbind(truth, MLEs, out1B$summary[1:6, 1:2]))
             truth   Estimate          SE        mean          sd
(Intercept)  -2.00 -1.6961500 0.39751824 -1.73942300 0.38527615
wind         -3.00 -3.0670527 0.59406301 -3.04706739 0.58621034
(Intercept)1  0.00 -0.1360580 0.44880554  0.07996883 0.58483924
vegHt         3.00  2.4319320 0.83948163  3.07435855 1.29439358
sumZ         51.00 50.5097710 8.62495498 52.04971014 7.46061747
psi.fs        0.51  0.5050977 0.08624955  0.52049710 0.07460617
```

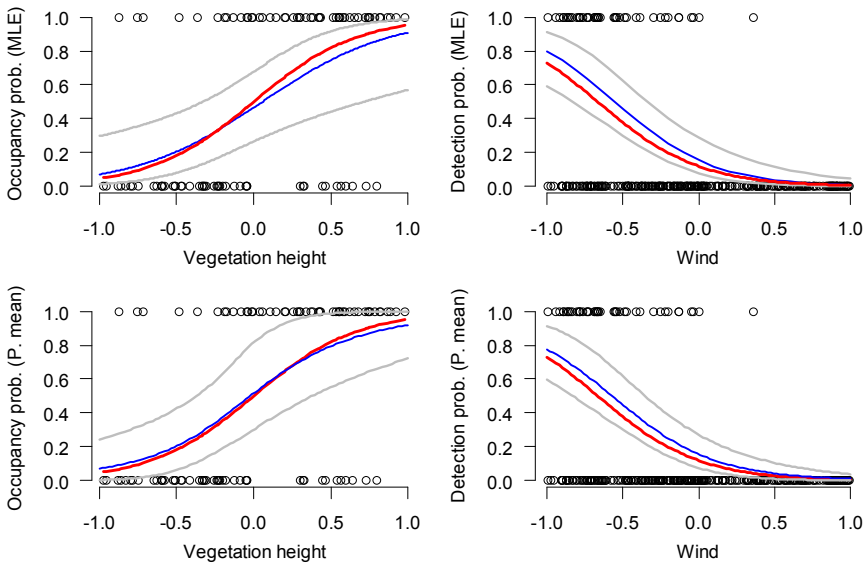

FIGURE 10.4

Estimated relationships (blue) between occurrence and vegetation height (left) and between detection probability and wind speed (right) from an occupancy model fit to the simulated data set. Top: maximum likelihood estimates (with 95% CIs in grey); bottom: Bayesian posterior means (with 95% CRIs in grey). Red lines represent the truth in the data simulation process. Circles show the true presence/absence (left) and the observed measurements of presence/absence (right).

```
# .... and in a graph (Fig. 10-4)
par(mfrow = c(2, 2), mar = c(4, 5, 2, 2), las = 1, cex.lab = 1, cex = 1.2)
plot(vegHt, z, xlab="Vegetation height", ylab="Occupancy prob. (MLE)", ylim = c(0, 1),
frame = F)                                  # True presence/absence
lines(seq(-1,1, 0.01), pred.occ[,1], col = "blue", lwd = 2)
matlines(seq(-1,1, 0.01), pred.occ[,3:4], col = "grey", lty = 1)
lines(vegHt, psi, lwd=3, col="red")    # True psi
plot(wind, y, xlab="Wind", ylab="Detection prob. (MLE)", ylim = c(0,1), frame=F)
lines(seq(-1, 1, 0.1), pred.det[,1], col = "blue", lwd = 2)
matlines(seq(-1, 1, 0.1), pred.det[,3:4], col = "grey", lty = 1)
plot(function(x) plogis(alpha0 + alpha1*x), -1, 1, add=T, lwd=3, col = "red")
plot(vegHt, z, xlab="Vegetation height", ylab="Occupancy prob. (P. mean)", las = 1,
frame = F)                                  # True presence/absence
lines(vegHt, psi, lwd=3, col="red")    # True psi
lines(win.data$XvegHt, out1B$summary[7:106,1], col="blue", lwd = 2)
matlines(win.data$XvegHt, out1B$summary[7:106,c(3,7)], col="grey", lty = 1)
plot(wind, y, xlab="Wind", ylab="Detection prob. (P. mean)", frame = F)
plot(function(x) plogis(alpha0 + alpha1*x), -1, 1, add=T, lwd=3, col = "red")
lines(win.data$Xwind, out1B$summary[107:206,1], col="blue", lwd = 2)
matlines(win.data$Xwind, out1B$summary[107:206,c(3,7)], col="grey", lty = 1)
```

The MLEs and Bayesian posterior means match fairly well in this realization of the simulated process (i.e., with a seed of 1 for the random number generator). However, during the development of this material we observed some cases where the posterior means did *not* match the MLEs so well, especially for the occupancy parameters. We believe that this was due to the relatively small sample size of only 100 sites. In this case, the priors have relatively more influence and the posterior will often be skewed. If we take the posterior mean as our point estimator, then by averaging over the whole posterior distribution, the Bayesian point estimate will differ from the MLE. This is presumably what McKann et al. (2013) called "small sample bias" in the case of the related dynamic occupancy model (see Chapter 16), where they observed that for truly extreme values of the probability parameters, the Bayesian estimates tended to be less extreme, i.e., pulled toward 0.5. Such a slight pulling in of extreme estimates may often be a good thing (Sauer and Link, 2002), for instance, it will completely prevent boundary estimates of 0 or 1 as they often occur in MLE (see Section 10.7).

Above, we discussed the finite-sample quantities, the number and the proportion of occupied sites among the sample of M studied sites. The estimate of the finite-sample occupancy is asymptotically equal to that of population occupancy. However, its uncertainty is smaller because one component of variation present in population occupancy is lacking: the binomial variance due to sampling M study sites from a hypothetical, infinite (statistical) population of sites. The only source of uncertainty in the variance estimate of finite-sample occupancy is due to imperfect detection and parameter uncertainty. These finite-sample quantities are frequently of more interest to practitioners than are the corresponding population quantities. Both are a function of the latent occurrence states z, which appear explicitly as latent variables in the model in BUGS, and you will obtain estimates of z simply by including them in the list of the estimated parameters.

Calculations on latent variables, such as z, in a Bayesian analysis is trivially easy and is conducted with a full propagation of all the involved uncertainties. In the Bayesian analysis, we directly estimate the quantities `N.occ` and `psi.fs`. Their estimates along with their uncertainties are quite comparable with their non-Bayesian counterparts when the variance is bootstrapped; see the table above and Figure 10.3 (right). However, the Bayesian posterior never extends to nonsensical values, i.e., the posterior does not extend to fewer occupied sites than were observed.

```
# Plot posterior distribution of number of occupied sites (see Fig. 10-3, right)
hist(out1B$sims.list$N.occ, col = "grey", breaks = 60, xlim = c(20, 100),
main = "", freq = F)
abline(v = out1B$mean$N.occ, col = "blue", lwd = 3)    # add point estimate
abline(v = out1B$summary[5,c(3,7)], col = "grey", lwd = 3)  # add 95% CRI
abline(v = sum(apply(y, 1, max)), lty = 2, lwd = 3)     # observed #occ sites
abline(v = sum(z), col = "red", lwd = 2)                # true #occ sites
```

Perhaps the most powerful aspect of hierarchical models is their invitation to build custom models that are exactly tailored to your system and your questions. However, in applied work with hierarchical models, much of the power of hierarchical modeling simply stems from your ability to specify linear models in a smart way. Hence, being able to fit complex linear models is important for all of your hierarchical modeling. For this reason we want to illustrate an occupancy model with a slightly more complex linear predictor. But first we introduce a data simulation function that, among other things, will provide us with a data set for such a more complex occupancy model in Section 10.6.

10.5 A GENERAL DATA SIMULATION FUNCTION FOR STATIC OCCUPANCY MODELS: simOcc

In our AHM package, you find an R function simOcc that permits simulation of data sets under a very wide variety of static occupancy models. The function is similar to those in Chapter 4 and in Section 6.5, but for occurrence rather than for count data. We provide this function in the hope that it may be directly useful for you in one of the many ways in which data simulation can be valuable (see Section 4.4) and that it may serve as a starting point for adapting it to your more specific needs. Later, we use simOcc to validate BUGS code (Section 10.6), to investigate estimator quality in the occupancy model when the information content of the data set is low and variable (Section 10.7) and to study goodness-of-fit assessments (Section 10.8). Using the function, you can simulate data under the following most general model, where sites are indexed i and repeated presence/absence measurements j and the main notation (e.g., z, ψ, y, p) is standard in this chapter.

Ecological model for presence/absence (z):

$$z_i \sim Bernoulli(\psi_i)$$

$$\text{logit}(\psi_i) = \beta_0 + \beta_1 * elev_i + \beta_2 * forest_i + \beta_3 * elev_i * forest_i$$

Observation/measurement error model for detection/nondetection data (y):

$$y_{ij} | z_i \sim Bernoulli(z_i * p_{ij})$$

$$\text{logit}(p_{ij}) = \alpha_0 + \gamma_j + \alpha_1 * elev_i + \alpha_2 * wind_{ij} + \alpha_3 * elev_i * wind_{ij} + \varepsilon_i + b * y_{i,j-1}$$

Effects of three continuous covariates can be built into the simulated data set: elevation, forest cover (two site covariates), and wind speed (an observational covariate), as well as the interactions between elevation and forest cover and between elevation and wind speed. Elevation can affect both occupancy and detection, and the elevation-wind speed interaction is set to zero by default (see below). In addition, we may add the following:

γ_j: time (j)-specific effects on baseline detection probability (*time effects*); these are expressed as deviations from the logistic-linear detection intercept α_0

$\varepsilon_i \sim Normal(0, \sigma)$: site-specific random effects assumed to be draws from a normal distribution with standard deviation (σ, called sd.lp below: *"heterogeneity" or site random effects*)

b: a "*behavioral response*" term, which increases or lowers detection probability depending on whether the species was detected at site i at the preceding occasion ($j - 1$) (see Riddle et al., 2010); this effect can only occur from the second occasion onward (i.e., for $j > 1$).

Models with a single one of the last three effects are called model M_t, M_h, and M_b in the capture-recapture literature (Otis et al., 1978; Royle and Dorazio, 2008; Kéry and Schaub, 2012). The function is called as follows, with its default arguments shown and explained below.

```
simOcc(M = 267, J = 3, mean.occupancy = 0.6, beta1 = -2, beta2 = 2, beta3 = 1, mean.detection =
0.3, time.effects = c(-1, 1), alpha1 = -1, alpha2 = -3, alpha3 = 0, sd.lp = 0.5, b = 2,
show.plot = TRUE)
# Function arguments:
M:          Number of spatial replicates (sites)
J:          Number of temporal replicates (occasions)
```

```
mean.occupancy: Occupancy probability at value 0 of occ. covariates
beta1:          Main effect of elevation on occurrence
beta2:          Main effect of forest cover on occurrence
beta3:          Interaction effect on occurrence of elevation and forest cover
mean.detection: Detection probability at value 0 of detection covariates
time.effects: bounds for uniform distribution from which time effects gamma
                  (on logit scale) will be drawn
alpha1:         Main effect of elevation on detection probability
alpha2:         Main effect of wind speed on detection probability
alpha3:         Interaction effect on detection of elevation and wind speed
sd.lp:          Standard deviation of random site effects (on logit scale)
b:              Constant value of 'behavioural response' leading to
                  'trap-happiness' (if b > 0) or 'trap shyness' (if b < 0)
show.plot:      if TRUE, plots of the data will be displayed;
                  should be set to FALSE if you are running simulations.
```

Executing the function produces two multipanel plots that visualize the simulated system (Figure 10.5 and 10.6). Here are some examples of the function's usage.

```
simOcc()                    # Execute function with default arguments
simOcc(show.plot = FALSE)   #     same, without plots
simOcc(M = 267, J = 3, mean.occupancy = 0.6, beta1 = -2, beta2 = 2, beta3 = 1, mean.detection =
0.3, time.effects = c(-1, 1), alpha1 = -1, alpha2 = -3, alpha3 = 0, sd.lp = 0.5, b = 2,
show.plot = TRUE)           # Explicit defaults
```

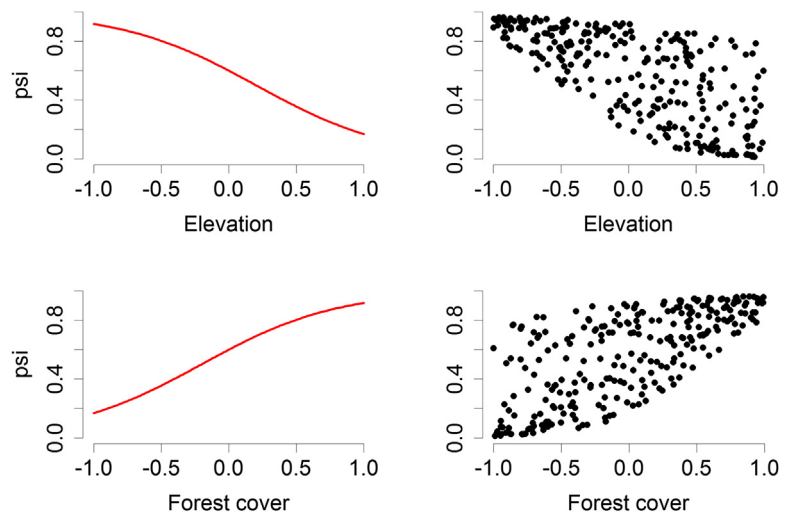

FIGURE 10.5

Visualization of the patterns in occupancy probability (psi) simulated by the function `simOcc`. Left panels show relationships between expected occupancy and one covariate when the other covariate is held constant, while the right panels show the same relationships at the observed values of the other covariate.

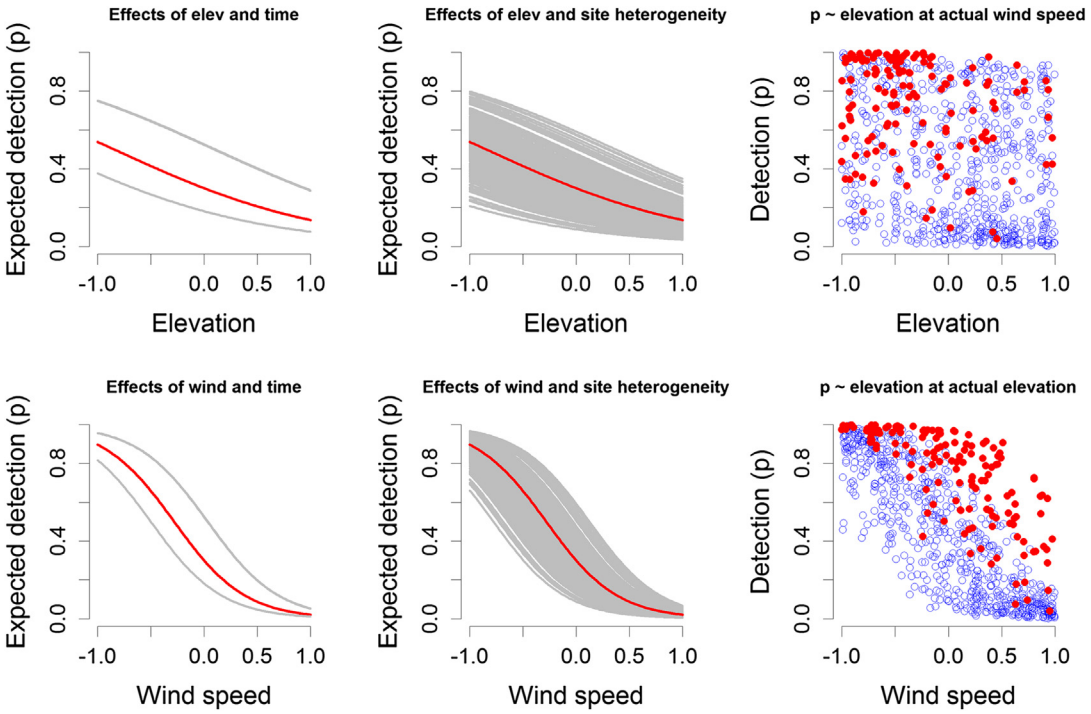

FIGURE 10.6

Visualization of the patterns in detection probability (p) simulated by the function `simOcc`. Left and middle panels show the relationships between the detection probability and one covariate when the other covariate is held constant (red line shows values with intercept). In the left panel, the effects of time, and in the middle panel, the effects of individual (site-specific) heterogeneity, are depicted in addition (these are the grey lines); each set of effects is also held constant in the other panel. In the right panel, these same relationships are shown at the observed values of the other covariate(s) and with red/blue color coding for the behavioral response effects (red: values of p when the species was detected at a site during the preceding survey; blue: same when the species was not detected during the preceding survey).

```
# Create a 'fix' data set and look at what we created
set.seed(24)
data <- simOcc()      # Assign results to an object called 'data'
str(data)
```

The output contains all argument settings employed plus all quantities created. Other than those that should be obvious from the preceding, we get occupancy probability (`psi`), presence/absence (`z`), the matrix of detection probability (`p`), the resulting data (`y`), the true number of occupied sites (`sumZ`), the number of sites at which at least one detection was made (`sumZ.obs`), the true proportion of occupied sites in the sample (`psi.fs.true`), and the observed proportion of occupied sites in the sample (`psi.fs.obs`). (The matrices `p0` and `p1` contain detection probability following a survey without detection (`p0`) or with detection (`p1`) and will not normally be of interest.).

You can use this function to generate data sets under various sampling designs (e.g., number of sites or surveys) and for a very large number of ecological and sampling "settings" (i.e., patterns in occurrence and detection). By setting to 0 or to 1 some arguments you can eliminate components from the simulation process, e.g., effects of covariates or patterns in detection probability (time, "heterogeneity" or "behavioral response") or the observation process altogether. It can be really useful for your understanding of occurrence data in general, and occupancy models specifically, to play around with this function with varying arguments to train your intuition about this important type of data and its sampling in the field under imperfect detection. Here are some illustrations of its use, with mostly only the relevant changes to the default arguments shown.

```
# Simplest possible occupancy model, with constant occupancy and detection
tmp <- simOcc(mean.occ=0.6, beta1=0, beta2=0, beta3=0, mean.det=0.3,
time.effects=c(0, 0), alpha1=0, alpha2=0, alpha3=0, sd.lp=0, b=0)
str(tmp)                    # give overview of results

# psi = 1 (i.e., species occurs at every site)
tmp <- simOcc(mean.occ=1)  ;  str(tmp)

# p = 1 (i.e., species is always detected when it occurs)
tmp <- simOcc(mean.det=1) ;  str(tmp)
```

Other potentially interesting settings include these:

```
simOcc(J = 2)               # Only 2 surveys
simOcc(M = 1, J = 100)      # No spatial replicates, but 100 measurements
simOcc(beta3 = 1)           # Including interaction elev-wind on p
simOcc(mean.occ = 0.96)     # A really common species
simOcc(mean.occ = 0.05)     # A really rare species
simOcc(mean.det = 0.96)     # A really easy species
simOcc(mean.det = 0.05)     # A really hard species
simOcc(mean.det = 0)        # The dreaded invisible species
simOcc(alpha1=-2, beta1=2)  # Opposing effects of elev on psi and p
simOcc(J = 10, time.effects = c(-5, 5)) # Huge time effects on p
simOcc(sd.lp = 10)          # Huge (random) site effects on p
simOcc(J = 10, b = 0)       # No behavioural response in p
simOcc(J = 10, b = 2)       # Trap happiness
simOcc(J = 10, b = -2)      # Trap shyness
```

You cannot simulate single-visit data (i.e., choose $J = 1$), but a simple workaround is to set J at any value greater than 1 and then discard everything except for one particular survey at each site (though you will have to compute "by hand" the observed number and proportion of occupied sites, `sumZ.obs` and `psi.fs.obs`). In addition, you cannot choose your own specific values for the time effects; rather, time effects on the logit scale are chosen randomly from a uniform distribution for which you specify the bounds.

We hope that this function is useful for you, either in its current version or as a template for modifications that you make to suit your needs. For instance, if you are interested in the question of how the quality of parameter estimates (e.g., bias, precision) varies as a function of sample size

(number of sites and temporal replicates) and the number of covariate effects estimated in the model, then you could adapt the function to contain 20 or so covariate effects in both the occupancy and detection parts of the data-generating model (but vectorizing and using matrix-vector multiplication to build up the linear predictor would then be a good idea, see Section 3.2.1). Or if you're interested in how different patterns of missing values affect your estimates, then you could adapt the function by incorporating a missing value-generating process, which could be governed by the same or a different set of covariates. Finally, with three exceptions (the time, site, and behavioral response effects in p) the function contains continuous covariates only and no factors. Obviously, if you need more factors you could incorporate them.

10.6 A MODEL WITH LOTS OF COVARIATES: USE OF R FUNCTION `model.matrix` WITH BUGS

Next, we show how a fairly complex linear model can be fit as part of an occupancy model in BUGS and how we can simplify our life by using the powerful R linear modeling function `model.matrix`. We start by generating one data set with the `simOcc` function where we eliminate the effects of all three factors by setting to zero the arguments controlling them.

```
set.seed(148)
data <- simOcc(time.effects = c(0,0), sd.lp = 0, b = 0)
str(data)                          # Look at data object
```

To illustrate a fairly complex linear model with covariates and factors we invent a further factor that is unrelated to the response: habitat (`hab`), which divides the default 267 sites into three imaginary habitat types.

```
# Create habitat factor
hab <- c(rep("A", 90), rep("B", 90), rep("C", 87)) # must have M = 267 sites

# Load library, format data and summarize unmarked data frame
library(unmarked)
umf <- unmarkedFrameOccu(
   y = data$y,
   siteCovs = data.frame(elev = data$elev, forest = data$forest, hab = hab),
   obsCovs = list(wind = data$wind))
summary(umf)
```

For illustration, let's now use `unmarked` to fit a model with additive effects of elevation and wind speed in detection and fully interactive effects of elevation, forest cover, and habitat in occupancy probability.

```
summary(fm <- occu(~elev+wind ~elev*forest*hab, data=umf))
```

Looking at the long list of parameter estimates in the output, you'll probably agree that this is a fairly complicated linear model. We now fit the same model in BUGS in two ways: first, by writing out all the linear model terms explicitly, and second, by defining a design matrix for the linear model and

fitting this matrix in BUGS (see Section 6.11.1 for another example of this in an N-mixture model). First, the more difficult solution:

```
# Bundle and summarize data set
HAB <- as.numeric(as.factor(hab)) # Get numeric habitat factor
str( win.data <- list(y = data$y, M = nrow(data$y), J = ncol(data$y), elev = data$elev,
forest = data$forest, wind = data$wind, HAB = HAB) )

# Specify model in BUGS language
sink("modelA.txt")
cat("
model {

# Priors
mean.p ~ dunif(0, 1)            # Detection intercept on prob. scale
alpha0 <- logit(mean.p)         #    same on logit scale
mean.psi ~ dunif(0, 1)          # Occupancy intercept on prob. scale
beta0 <- logit(mean.psi)        #    same on logit scale
for(k in 1:2){                  # 2 terms in detection model
   alpha[k] ~ dnorm(0, 0.1)     # Covariates on logit(detection)
}
for(k in 1:11){                 # 11 terms in occupancy model
   beta[k] ~ dnorm(0, 0.1)      # Covariates on logit(occupancy)
}
# Likelihood
for (i in 1:M) {                # Loop over sites
 z[i] ~ dbern(psi[i])
 logit(psi[i]) <- beta0 +                                  # occupancy (psi) intercept
  beta[1] * elev[i] +                                      # effect of elev
  beta[2] * forest[i] +                                    # effect of forest
  beta[3] * equals(HAB[i],2) +                             # effect of habitat 2 (= B)
  beta[4] * equals(HAB[i],3) +                             # effect of habitat 3 (= C)
  beta[5] * elev[i] * forest[i] +                                  # elev:forest
  beta[6] * elev[i] * equals(HAB[i],2) +                           # elev:habB
  beta[7] * elev[i] * equals(HAB[i],3) +                           # elev:habC
  beta[8] * forest[i] * equals(HAB[i],2) +                         # forest:habB
  beta[9] * forest[i] * equals(HAB[i],3) +                         # forest:habC
  beta[10] * elev[i] * forest[i] * equals(HAB[i],2) +              # elev:forest:habB
  beta[11] * elev[i] * forest[i] * equals(HAB[i],3)                # elev:forest:habC
 for (j in 1:J) {                          # Loop over replicates
    y[i,j] ~ dbern(z[i] * p[i,j])          # WinBUGS would need 'straw man' !
    logit(p[i,j]) <- alpha0 +              # detection (p) intercept
       alpha[1] * elev[i] +                # effect of elevation on p
       alpha[2] * wind[i,j]                # effect of wind on p
 }
}
}
",fill = TRUE)
sink()
```

```
# Inits
inits <- function(){list(z = apply(data$y, 1, max), mean.psi = runif(1), mean.p = runif(1),
alpha = rnorm(2), beta = rnorm(11))}

# Parameters monitored
params <- c("alpha0", "alpha", "beta0", "beta")

# MCMC settings
ni <- 50000  ;  nt <- 10  ;  nb <- 10000  ;  nc <- 3

# Run JAGS (ART 4 min), look at convergence and summarize posteriors
outA <- jags(win.data, inits, params, "modelA.txt", n.chains = nc, n.thin = nt,
n.iter = ni, n.burnin = nb, parallel = TRUE)
traceplot(outA)  ;  print(outA, 3)

# Compare MLEs and SEs with posterior means and sd's
tmp <- summary(fm)
cbind(rbind(tmp$state[1:2], tmp$det[1:2]), Post.mean = outA$summary[c(4:15, 1:3), 1],
Post.sd = outA$summary[c(4:15, 1:3), 2])
```

Second, the easy solution: in BUGS we simply fit a design matrix generated in R using `model.matrix`. We want to add the intercept in BUGS and hence create a design matrix without intercept and simply add the matrix to the data bundle. Then, in BUGS we define the linear predictor to be the matrix (or "inner") product of parameter vector and design matrix: `inprod(beta[], occDM[i,])`.

```
# Create design matrix for occupancy covariates and look at it
occDM <- model.matrix(~ data$elev * data$forest * hab)[,-1] # Drop first col.
head(occDM)              # Look at design matrix
str(occDM)

# Bundle and summarize data set
str( win.data <- list(y = data$y, M = nrow(data$y), J = ncol(data$y), elev = data$elev,
wind = data$wind, occDM = occDM) )

# Specify model in BUGS language
sink("modelB.txt")
cat("
model {

# Priors
mean.p ~ dunif(0, 1)          # Detection intercept on prob. scale
alpha0 <- logit(mean.p)       #    same on logit scale
mean.psi ~ dunif(0, 1)        # Occupancy intercept on prob. scale
beta0 <- logit(mean.psi)      #    same on logit scale
for(k in 1:2){                # 2 terms in detection model
   alpha[k] ~ dnorm(0, 0.1)   # Covariates on logit(detection)
}
for(k in 1:11){               # 11 terms in occupancy model
  beta[k] ~ dnorm(0, 0.1)     # Covariates on logit(occupancy)
}
```

```
# Likelihood
for (i in 1:M) {
 z[i] ~ dbern(psi[i])
 logit(psi[i]) <- beta0 + inprod(beta[], occDM[i,]) # slick !
  for (j in 1:J) {
    y[i,j] ~ dbern(z[i] * p[i,j])   # In WinBUGS need 'straw man'
    logit(p[i,j]) <- alpha0 +       # detection (p) intercept
      alpha[1] * elev[i] +          # effect of elevation on p
      alpha[2] * wind[i,j]          # effect of wind on p
  }
}
}
",fill = TRUE)
sink()
```

We can recycle all other parts of the code and directly launch JAGS.

```
# Call JAGS from R (ART 3.3 min) and summarize posteriors
outB <- jags(win.data, inits, params, "modelB.txt", n.chains = nc,
n.thin = nt, n.iter = ni, n.burnin = nb, parallel = TRUE)
traceplot(outB)  ;  print(outB, 3)
```

Though the chains mix less well, up to MC error model B yields estimates that are identical to those under model A. Thus, you can use the model definition language in R to create the design matrix of a linear model and then fit that model, i.e., that design matrix, in BUGS directly. That may be a great simplification, because it may be much easier to specify a linear model in R, import its design matrix into BUGS, and fit it there, rather than constructing the model for every column in the design matrix, as we did in the previous section (for model A).

10.7 STUDY DESIGN, AND BIAS AND PRECISION OF SITE-OCCUPANCY ESTIMATORS

We next present a small simulation study with a basic occupancy model. We do this to emphasize the power of simulation to answer questions about study design and about the estimator quality from a model. Questions frequently heard about HMs like occupancy models are: "How many sites do I need?" or "How many replicate surveys are enough?" or "Is it better to visit more sites fewer times or fewer sites more frequently?" or (now this one is mean) "How come I get an `NA` or `Inf` standard error with my occupancy model with four sites?" These are important questions about study design and estimator quality, and these must represent one of the most neglected topics in all of ecological statistics. There are several important papers and rules of thumb about the design of occupancy studies (MacKenzie et al., 2002; Tyre et al., 2003; MacKenzie and Royle, 2005; Bailey et al., 2007; Guillera-Arroita et al., 2010, 2014b; Guillera-Arroita and Lahoz-Monfort, 2012; Ellis et al., 2015) and also software specifically designed for occupancy study design (`GENPRES`; Bailey et al., 2007; `SODA`; Guillera-Arroita et al., 2010, `rSPACE`; Ellis et al., 2015). However, by far the most powerful and most flexible way of answering such questions is by running your own custom simulations. For instance, the above questions may be tackled by running a factorial design where you vary both the number of sites

and the number of surveys, simulate and analyze 100−1000 data sets for each combination of this simulation design using the expected occupancy and detection probability of the species of your interest, and see which combination gives you the "best" estimates, where "best" would typically include statistical considerations such as the bias or the precision of the estimates as well as economical/logistical ones (e.g., how much do additional sites or additional surveys cost?; see MacKenzie and Royle, 2005). A simulation like the following can help you find the best trade-off between number of sites and number of visits specifically for *your* study.

Moreover, it is important to note that in principle, with enough data, all parameters of the site-occupancy model may be estimated. However, the quality of the estimates (i.e., whether there is bias and the magnitude of the precision of the estimates) will depend strongly on the sample size (number of sites, number of surveys) and on the magnitude of the parameters (ψ and p). For small sample sizes, the widely proclaimed unbiasedness of MLEs is typically lost (Le Cam, 1990) and solutions may become unstable (Welsh et al., 2013; Guillera-Arroita et al., 2015). Our little simulation draws attention to this basic fact of statistical inference and shows how the quality of estimators for particular scenarios (sample sizes, parameter values) can very easily be ascertained by simulation. In our example we do this for the null/null model without covariates, but the simulation could easily be extended to more complex models.

We study estimator quality in a design that varies the number of sites ($M = 20$, 120, or 250) and of surveys ($J = 2$, 5, or 10) and the magnitude of detection probability (covering almost the entire range between 0 and 1) for a species with occupancy equal to 0.5. For each combination of the two factors (sites, surveys) we repeat the following 1000 times: (1) randomly pick a value for detection probability from a *Uniform*(0.01, 0.99), (2) use `simOcc` to generate one data set, and (3) estimate parameters using MLE with `unmarked`. We don't need the SEs so we don't compute them, to avoid the simulation from breaking whenever a Hessian becomes singular.

```
# Do simulation with 1000 reps
simreps <- 1000
library(unmarked)

# Define arrays to hold the results
p <- array(dim = c(simreps,3,3))
estimates <- array(dim = c(2,simreps,3,3))

# Choose number and levels of simulation factors
nsites <- c(20, 120, 250)      # Number of sites
nsurveys <- c(2, 5, 10)        # Number of repeat surveys

# Start simulation
system.time(                   # Time whole thing
for(j in 1:3) {                # Loop j over site factor
  for(k in 1:3) {              # Loop k over survey factor
    for(i in 1:simreps){       # Loop i over simreps
      # Counter
      cat("** nsites", j, "nsurveys", k, "simrep", i, "***\n")
      # Generate a data set: pick p and use p in simOcc()
      det.prob <- runif(1, 0.01, 0.99)
      data <- simOcc(M = nsites[j], J = nsurveys[k], mean.occupancy = 0.5,
          beta1 = 0, beta2 = 0, beta3 = 0, mean.detection = det.prob,
```

```
          time.effects = c(0, 0), alpha1 = 0, alpha2 = 0, alpha3 = 0,
          sd.lp = 0, b = 0, show.plot = F)
        # Fit model
        umf <- unmarkedFrameOccu(y = data$y)
        tmp <- occu(~1 ~1, umf, se = FALSE)      # Only get MLEs, not SEs
        # Save results (p and MLEs)
        p[i,j,k] <- data$mean.det
        estimates[,i,j,k] <- coef(tmp)
      }
    }
  }
)

# Plot results
par(mfrow = c(3,3), mar = c(4,5,3,1), cex.main = 1.2)
for(j in 1:3){
  for(k in 1:3){
    lab <- paste(nsites[j],"sites,", nsurveys[k],"surveys")
    plot(p[,j,k], plogis(estimates[1,,j,k]), xlab = "Detection prob.",
       ylab = "Occupancy prob.", main = lab, ylim = c(0,1))
       abline(h = 0.5, col = "red", lwd = 2)
       lines(smooth.spline(plogis(estimates[1,,j,k])~ p[,j,k], df = 5),
       col = "blue", lwd = 2)
  }
}
```

Generation and analysis of 9,000 data sets takes less than four minutes on a moderate laptop! Figure 10.7 summarizes the results from the simulation for the nine scenarios that combine three levels each of the site and the survey factor and for the whole range of values of p between 1% and 99%. It shows that there can be substantial variation in the quality of the estimates under the site-occupancy model. When sample size (number of sites M or visits J) is small, the quality of the estimates can be fairly bad. Moreover, according to the first law of capture-recapture, the quality of the estimators becomes bad when p is low for any combination of M and J. Depending on the particular combination of number of sites and visits, the occupancy estimator is biased high when p is less than about 0.1 or 0.2, something that had already been noted by MacKenzie et al. (2002) and Guillera-Arroita et al. (2010).

We then repeated this simulation with WinBUGS, using the model in Section 10.3 with vague *Uniform*(0, 1) priors, and after waiting for about 15 hours, obtained the results in Figure 10.8. Using as the usual Bayesian point estimator the posterior mean rather than the posterior mode (which with vague priors would correspond to the MLE) always pulls estimates away from the boundary of the parameter space at 0 and 1 (McKann et al., 2013). Thus, the Bayesian estimates were much better than the MLEs because use of the posterior mean avoids the instabilities of the MLEs in situations with little information (small M, J, or p). Table 10.2 compares the Bayesian posterior means with the MLEs and shows that in every scenario the average estimation error (RMSE) was about halved in the Bayesian analysis with vague priors. Boundary estimates can be a serious problem with ML when

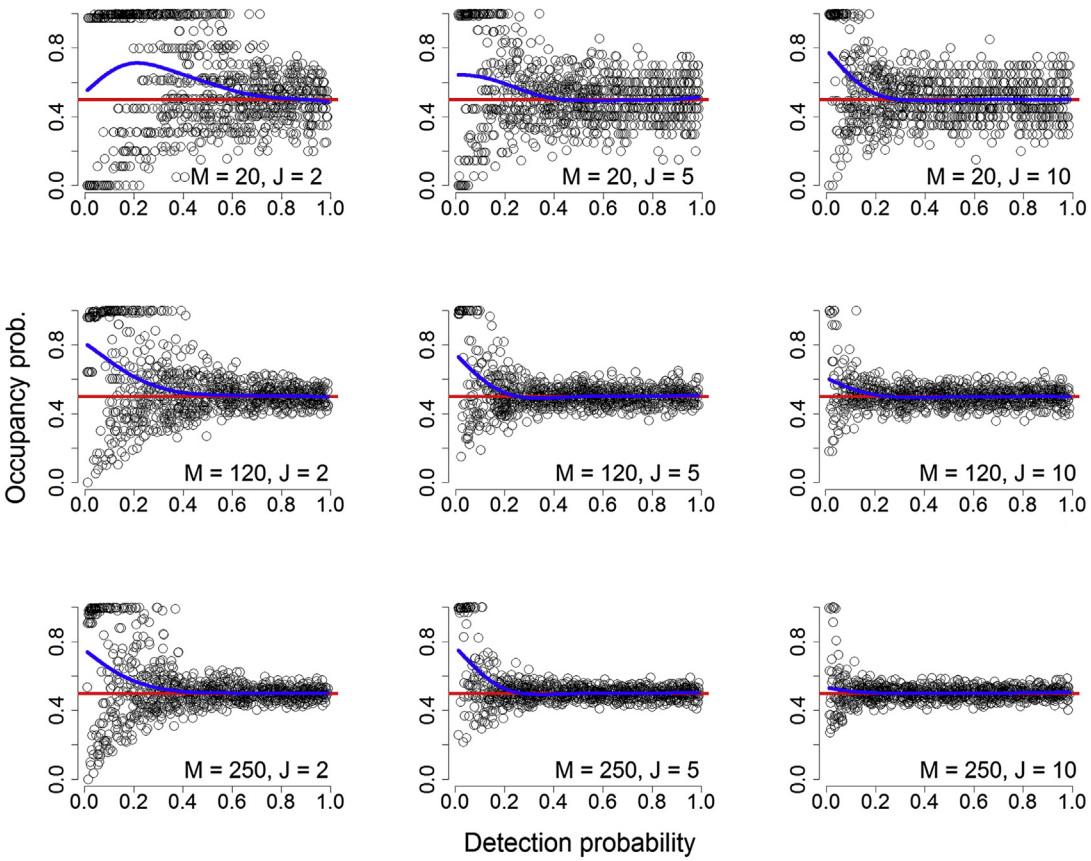

FIGURE 10.7

Results of a simulation study about the quality of MLEs of the site-occupancy estimator for occupancy probability in a grid spanned by three levels each of the two factors "number of sites" (M) and "number of repeat surveys" (J) and for a continuous range of values for detection probability ($0.01 \leq p \leq 0.99$). The red line shows the true occupancy probability (0.5) and the blue line is a spline smoother to show the average behavior of the estimator for a given value of p. The number of simulation replicates is 1000 for each plot. Look at Figure 10.8 for the Bayesian results of the same simulation exercise.

there is little information in the data. Using penalized likelihood can then stabilize the estimators. For static occupancy models, the function `occuPEN` has been incorporated in `unmarked` (Hutchinson et al., 2015).

The R code in this section can easily serve as a template for many other simulation studies, for instance, it is straightforward to gauge the effects of assumption violations, e.g., due to individual (site-specific) detection heterogeneity or behavioral response (this is Exercise 4).

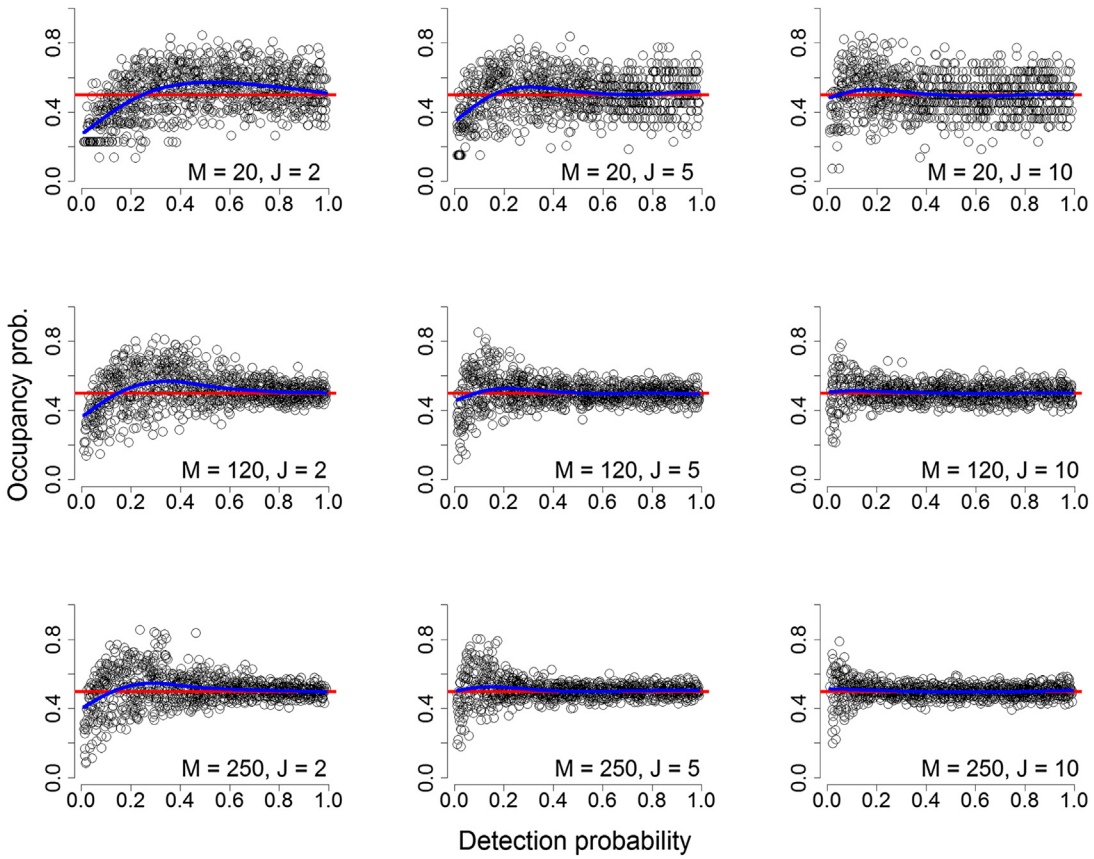

FIGURE 10.8

Results from the Bayesian version of the simulation. Figure 10.7 shows MLE results of the same simulation exercise; see it for explanations.

Table 10.2 Average estimation error (RMSE) for MLEs and for Bayesian posterior means under vague *Uniform*(0,1) priors for the nine scenarios (X / Y denotes root mean square error, in percent of the mean, for MLE / Bayes).

	2 Surveys	5 Surveys	10 Surveys
20 sites	59%/27%	42%/24%	35%/22%
150 sites	40%/20%	24%/15%	16%/12%
250 sites	36%/18%	22%/12%	12%/9%

10.8 GOODNESS-OF-FIT

Goodness-of-fit (GoF) implies a comparison of the observed data with the data expected under the model using some fit statistic, or discrepancy measure, such as residuals, Chi-square or deviance. With occupancy models, the data are binary unless aggregated to binomial counts (Section 10.3 and 11.6.1). Standard fit statistics are then a simple deterministic function of sample size and hence uninformative about model fit; see Section 4.4.5 in McCullagh and Nelder (1989) and Section 8.4.1.1 in Royle et al. (2014). In order to test GoF in binary response models such as the occupancy model, we always have to aggregate the binary response in some way. Clearly, many such aggregations are possible and it is not *a priori* clear which one is best in order to indicate a particular assumption violation nor how sensitive a test based on any such aggregation is. One particular way in which we can aggregate the binary detection/nondetection data is by unique detection history; we can then compare the observed with the expected number of sites that exhibit a certain detection history. This is what the GoF test by MacKenzie and Bailey (2004) does, which is implemented as function `mb.gof.test` in the R package `AICcmodavg` (Mazerolle, 2015). Another possible aggregation is to compute a fit statistic on row or column sums of the detection history matrix, i.e., aggregating over occasions or over sites (see also Section 8.4.2 in Royle et al., 2014).

To illustrate, we conducted a little simulation study and used function `simOcc` to simulate 20 data sets that each contained some effect that was not present in the data-analyzing model and that therefore represented an assumption violation of the analysis model. The basic function settings were the following, representing a data-generating model with forest cover effects in occupancy and elevation and wind speed effects in detection:

```
simOcc(M = 267, J = 3, mean.occupancy = 0.6, beta1 = 0, beta2 = 1, beta3 = 0, mean.detection =
0.3, time.effects = c(0, 0), alpha1 = 1, alpha2 = -1, alpha3 = 0, sd.lp = 0, b = 0)
```

We then either added an effect in the data simulation or dropped one in the analysis model, leading to a mismatch between data generation and data analysis, which we might hope to pick up with a GoF test. For each model fit we computed a Chi-square GoF test either directly on the cells of the detection history matrix or on the detection frequencies obtained by aggregating over columns or over rows. In addition, we calculated the GoF test by MacKenzie and Bailey (2004). To obtain a *p*-value for each, we bootstrapped 500 times (for Chi2) and 100 times (for the M&B test). Table 10.3 shows the median and the range of the *p*-values over the 20 data sets for each scenario.

We see that tests computed directly on the observed binary data (column "Chi2 on cells") are totally uninformative about lack of fit, but that tests on aggregated data have some power to detect to lack of fit, with aggregation by capture history (i.e., the MacKenzie-Bailey test) being the best among those tested. However, neither test was powerful in every case, and surprisingly, missing covariates remained undetected by all types of aggregation. We also see that test performance was highly variable among samples, with detection of lack of fit in some data sets and not in others.

Hence, diagnosing lack of fit in an occupancy model remains difficult and should perhaps not be relegated to the calculation of one single number such as a *p*-value from some GoF test. We could also plot residuals (which again must be computed on aggregated data) against modeled and unmodeled covariates or spatial coordinates to detect any systematic pattern, which could then be taken account of in an improved version of the model (see Section 6.9). Finally, there's the question of what to do if despite all our best efforts we don't succeed in identifying a model that passes our GoF test. The

Table 10.3 Median and range of *p*-values of Goodness-of-fit tests (for 20 simulated data sets for each type of assumption violation) from disaggregated data (Chi2 on cells) and from detection histories aggregated over columns (Chi2 on rows), over rows (Chi2 on columns) and per detection history type (MB test, MacKenzie and Bailey 2004) for a selection of seven scenarios of assumption violations.

Assumption Violation Type	Chi2 on Cells	Chi2 on Rows	Chi2 on Columns	MB Test
Strong behavioral response ($b = 2$)	0.78 (0.30−0.98)	0.18 (0.02−0.81)	0.00 (0.00−0.03)	0.00 (0.00−0.01)
Weak behavioral response ($b = 1$)	0.63 (0.11−0.89)	0.30 (0.01−0.69)	0.09 (0.00−0.99)	0.03 (0.00−0.31)
Detection heterogeneity (sd.lp = 1)	0.52 (0.22−0.95)	0.29 (0.00−0.65)	0.67 (0.02−0.99)	0.28 (0.01−0.86)
Missing site covariate in psi (forest)	0.61 (0.20−0.90)	0.44 (0.09−0.81)	0.35 (0.02−0.96)	0.45 (0.00−0.99)
Missing site covariate in *p* (elevation)	0.43 (0.06−0.90)	0.33 (0.09−0.91)	0.55 (0.10−0.93)	0.41 (0.07−0.90)
Missing observational covariate in *p* (wind)	0.52 (0.20−0.80)	0.55 (0.18−0.89)	0.63 (0.00−0.96)	0.44 (0.01−0.97)
Missing time effects in p (time.effects = c(−1,1))	0.57 (0.11−0.91)	0.51 (0.14−0.97)	0.00 (0.00−0.10)	0.00 (0.00−0.21)

discussions in Sections 6.9 and 8.4.3 are of course relevant here as well. That is, we could in theory throw away a data set as unanalyzable, but realistically this will rarely be done. More typically, we may simply stick to our analysis and acknowledge that we have more uncertainty about the inferences than what we formally account for in the SEs or CIs. Better still, we could inflate SEs and CIs by an estimate of the overdispersion parameter (c-hat) for the model at hand by dividing the observed Chi-square statistic by the mean of the statistics obtained from a bootstrap simulation. Seeing how little is changed in the SEs or CIs of predictions from the model may perhaps make us more comfortable in keeping an ill-fitting model.

Clearly all the same comments apply for a Bayesian model fit. That is, we must conduct posterior predictive checks on a response that is aggregated by summing over sites, occasions, or individual capture history. Computing it directly on the binary responses, as we erroneously did in Chapter 20 in Kéry (2010), will fail to indicate an ill-fitting model.

10.9 DISTRIBUTION MODELING AND MAPPING OF SWISS RED SQUIRRELS

At various places in this book we have emphasized that any model for abundance or occurrence with spatially indexed covariates can be used to produce a map of species abundance or occurrence, that is, a species distribution map. In particular, there is a sense in which site-occupancy models represent the most genuine species distribution model because they model true occupancy probability separately from false-negative detection error (Kéry et al., 2010a,b; 2013). This is different from any

other species distribution modeling framework, which only model apparent occurrence, i.e., the product of occupancy and detection probability (Kéry, 2011b; Lahoz-Monfort et al., 2014; Guillera-Arroita et al., 2015). To emphasize the species distribution modeling role of site-occupancy models, and to finally show some real-data analysis with occupancy models, we next use `unmarked` to model the distribution of the European red squirrel (*Sciurus vulgaris*, Figure 10.9) in Switzerland. We base our analysis on data from the Swiss breeding bird survey MHB, where red squirrels are recorded as some sort of honorary avian species. Survey methods for the species are essentially identical to those for birds; see Section 6.9. The data set `SwissSquirrels.txt` contains detection/nondetection data for the red squirrel in 265 1 km^2 survey quadrats in Switzerland for 2007, along with some covariates. The goals of our analysis are threefold and exactly analogous to those for an analysis of great tit abundance in Section 6.9:

1. Identify environmental factors that affect the Swiss squirrel distribution
2. Produce a distribution map of the species
3. Estimate the Swiss range size of the species

We show a complete analysis that includes model selection, inference, GoF assessment, and prediction/mapping. We use two site (elevation, forest cover) and two observational covariates (survey

FIGURE 10.9

European red squirrel (*Sciurus vulgaris*), Cairngorms, Scotland, 2009 (*Photo by Aender Brepsom*).

date and duration) but do not use transect length now; see Sections 6.9 and 7.9.5 for how we could include route length into the analysis to accommodate coverage bias.

```
# Read in data set, select squirrels and harvest data
data <- read.table("SwissSquirrels.txt", header = TRUE)
str(data)
y <- as.matrix(data[,7:9])       # Grab 2007 squirrel det/nondet data
elev.orig <- data[,"ele"]        # Unstandardised, original values of covariates
forest.orig <- data[,"forest"]
time <- matrix(as.character(1:3), nrow=265, ncol = 3, byrow = T)
date.orig <- as.matrix(data[,10:12])
dur.orig <- as.matrix(data[,13:15])

# Overview of covariates
covs <- cbind(elev.orig, forest.orig, date.orig, dur.orig)
par(mfrow = c(3,3))
    for(i in 1:8){
    hist(covs[,i], breaks = 50, col = "grey", main = colnames(covs)[i])
}
pairs(cbind(elev.orig, forest.orig, date.orig, dur.orig))

# Standardise covariates and mean-impute date and duration
# Compute means and standard deviations
(means <- c(apply(cbind(elev.orig, forest.orig), 2, mean), date.orig =
mean(c(date.orig), na.rm = TRUE), dur.orig=mean(c(dur.orig), na.rm = TRUE)))
(sds <- c(apply(cbind(elev.orig, forest.orig), 2, sd), date.orig = sd(c(date.orig),
na.rm = TRUE), dur.orig=sd(c(dur.orig), na.rm = TRUE)))

# Scale covariates
elev <- (elev.orig - means[1]) / sds[1]
forest <- (forest.orig - means[2]) / sds[2]
date <- (date.orig - means[3]) / sds[3]
date[is.na(date)] <- 0
dur <- (dur.orig - means[4]) / sds[4]
dur[is.na(dur)] <- 0

# Load unmarked, format data and summarize
library(unmarked)
umf <- unmarkedFrameOccu(y = y, siteCovs = data.frame(elev = elev, forest = forest), obsCovs
= list(time = time, date = date, dur = dur))
summary(umf)
```

We want to identify a model that is useful for inference, specifically for prediction of squirrel distribution to the whole of Switzerland. We do some stepwise model selection first on the detection part, then on the occupancy part, while keeping the detection part as identified in the first step.

```
# Fit a series of models for detection first and do model selection
summary(fm1 <- occu(~1 ~1, data=umf))
summary(fm2 <- occu(~date ~1, data=umf))
summary(fm3 <- occu(~date+I(date^2) ~1, data=umf))
```

```
summary(fm4 <- occu(~date+I(date^2)+I(date^3) ~1, data=umf))
summary(fm5 <- occu(~dur ~1, data=umf))
summary(fm6 <- occu(~date+dur ~1, data=umf))
summary(fm7 <- occu(~date+I(date^2)+dur ~1, data=umf))
summary(fm8 <- occu(~date+I(date^2)+I(date^3)+dur ~1, data=umf))
summary(fm9 <- occu(~dur+I(dur^2) ~1, data=umf))
summary(fm10 <- occu(~date+dur+I(dur^2) ~1, data=umf))
summary(fm11 <- occu(~date+I(date^2)+dur+I(dur^2) ~1, data=umf))
summary(fm12 <- occu(~date+I(date^2)+I(date^3)+dur+I(dur^2) ~1, data=umf))

# Put the fitted models in a "fitList" and rank them by AIC
fms <- fitList("p(.)psi(.)"                    = fm1,
      "p(date)psi(.)"                          = fm2,
      "p(date+date2)psi(.)"                    = fm3,
      "p(date+date2+date3)psi(.)"              = fm4,
      "p(dur)psi(.)"                           = fm5,
      "p(date+dur)psi(.)"                      = fm6,
      "p(date+date2+dur)psi(.)"                = fm7,
      "p(date+date2+date3+dur)psi(.)"          = fm8,
      "p(dur+dur2)psi(.)"                      = fm9,
      "p(date+dur+dur2)psi(.)"                 = fm10,
      "p(date+date2+dur+dur2)psi(.)"           = fm11,
      "p(date+date2+date3+dur+dur2)psi(.)"     = fm12)

(ms <- modSel(fms))
                                  nPars     AIC  delta   AICwt  cumltvWt
p(date+dur+dur2)psi(.)                5  789.09   0.00  0.4612      0.46
p(date+date2+dur+dur2)psi(.)          6  790.94   1.85  0.1825      0.64
p(date+date2+date3+dur+dur2)psi(.)    7  791.59   2.50  0.1321      0.78
p(date+dur)psi(.)                     4  791.97   2.88  0.1091      0.88
p(date+date2+dur)psi(.)               5  793.74   4.65  0.0451      0.93
p(date+date2+date3+dur)psi(.)         6  794.11   5.03  0.0373      0.97
p(date)psi(.)                         3  795.98   6.89  0.0147      0.98
p(date+date2+date3)psi(.)             5  797.62   8.54  0.0065      0.99
p(date+date2)psi(.)                   4  797.67   8.58  0.0063      0.99
p(dur+dur2)psi(.)                     4  798.52   9.43  0.0041      1.00
p(dur)psi(.)                          3  801.29  12.20  0.0010      1.00
p(.)psi(.)                            2  805.90  16.82  0.0001      1.00

# Continue with model fitting for occupancy, guided by AIC as we go
# Check effects of elevation
summary(fm13 <- occu(~date+dur+I(dur^2) ~elev, data=umf))
summary(fm14 <- occu(~date+dur+I(dur^2) ~elev+I(elev^2), data=umf))
summary(fm15 <- occu(~date+dur+I(dur^2) ~elev+I(elev^2)+ I(elev^3), data=umf))
cbind(fm13@AIC, fm14@AIC, fm15@AIC) # model 14 with elev2 best

# Check effects of forest and interactions
summary(fm16 <- occu(~date+dur+I(dur^2) ~elev+I(elev^2)+forest, data=umf))
summary(fm 17<- occu(~date+dur+I(dur^2) ~elev+I(elev^2)+forest+I(forest^2), data=umf))
```

```
summary(fm18 <- occu(~date+dur+I(dur^2) ~elev+I(elev^2)+forest+I(forest^2)+elev:
forest, data=umf))
summary(fm19 <- occu(~date+dur+I(dur^2) ~elev+I(elev^2)+forest+I(forest^2)+elev:
forest+elev:I(forest^2), data=umf))
summary(fm20 <- occu(~date+dur+I(dur^2) ~elev+I(elev^2)+forest+I(forest^2)+elev:
forest+elev:I(forest^2)+I(elev^2):forest, data=umf))
summary(fm21 <- occu(~date+dur+I(dur^2) ~elev+I(elev^2)+forest+I(forest^2)+elev:
forest+elev:I(forest^2)+I(elev^2):forest+ I(elev^2):I(forest^2), data=umf))
cbind(fm16@AIC, fm17@AIC, fm18@AIC, fm19@AIC, fm20@AIC) # fm20 is best

# Check for some additional effects in detection
summary(fm22 <- occu(~date+dur+I(dur^2)+elev ~elev+I(elev^2)+forest+I(forest^2)+
elev:forest+elev:I(forest^2)+I(elev^2):forest, data=umf))
summary(fm23 <- occu(~dur+I(dur^2)+date*(elev+I(elev^2)) ~elev+I(elev^2)+
forest+I(forest^2)+elev:forest+elev:I(forest^2)+I(elev^2):forest, data=umf))
summary(fm24 <- occu(~dur+I(dur^2)+date*(elev+I(elev^2))+forest ~elev+I(elev^2)+
forest+I(forest^2)+elev:forest+elev:I(forest^2)+I(elev^2):forest, data=umf))
cbind(fm22@AIC, fm23@AIC, fm24@AIC) # None better, hence, stay with model 20
```

We do a bootstrapped GoF test on detection history frequencies (MacKenzie and Bailey, 2004), note that only the *observed* detection histories are shown in the table below.

```
library(AICcmodavg)
system.time(gof.boot <- mb.gof.test(fm20, nsim = 1000))
gof.boot

MacKenzie and Bailey goodness-of-fit for single-season occupancy model

Pearson chi-square table:

     Cohort  Observed  Expected  Chi-square
000       0       102    103.26        0.02
001       0        14     13.66        0.01
010       0        18     16.55        0.13
011       0        10     12.85        0.63
100       0        22     20.30        0.14
101       0        17     15.65        0.12
110       0        17     19.07        0.22
111       0        17     15.66        0.11
00NA      1        47     46.57        0.00
01NA      1         1      0.43        0.74

Chi-square statistic = 3.1134
Number of bootstrap samples = 1000
P-value = 0.878

Quantiles of bootstrapped statistics:
   0%    25%    50%    75%   100%
 0.67   4.16   6.12   8.60  25.64

Estimate of c-hat = 0.45
```

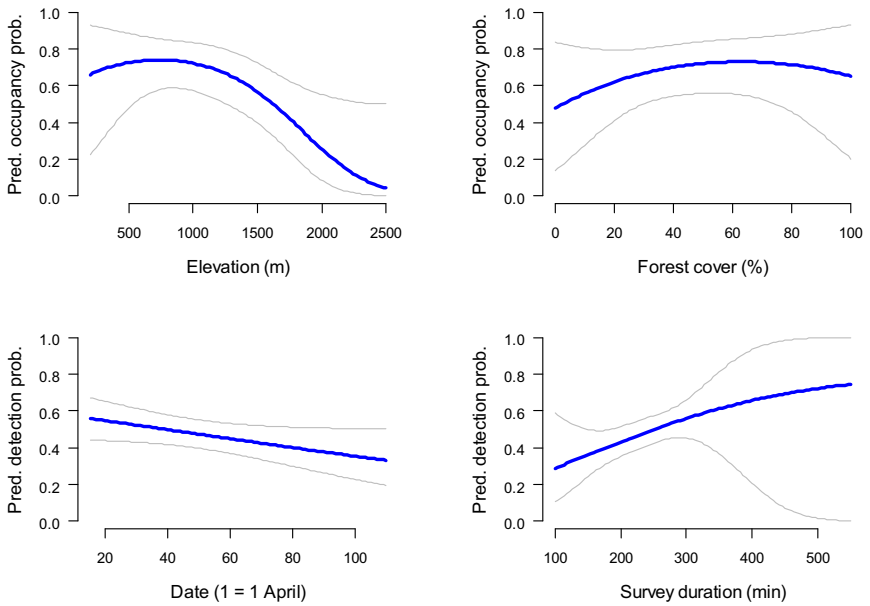

FIGURE 10.10

One-dimensional prediction: Estimated covariate relationships in the site-occupancy model for Swiss red squirrels in 2007. Grey lines show the 95% CIs.

Hence, the observed frequency of the squirrel site-level detection histories agrees reasonably well with that expected under the AIC-best model `fm20`. We conclude that this model is suitable to use for inference and to inspect covariate relationships and project them onto Swiss geographic space. First, we plot some one-dimensional covariate relationships. We use the `predict` function for the `unmarked` fitted object, which uses the delta rule to compute SEs and 95% CIs; the latter we plot as well (Figure 10.10). If our GoF analysis had detected some lack of fit, we could use it to compute an overdispersion factor c-hat and inflate the prediction variances by it, exactly analogous to what we did in Section 6.9.

```
# Create new covariates for prediction ('prediction covs')
orig.elev <- seq(200, 2500,,100)        # New covs for prediction
orig.forest <- seq(0, 100,,100)
orig.date <- seq(15, 110,,100)
orig.duration <- seq(100, 550,,100)
ep <- (orig.elev - means[1]) / sds[1]     # Standardize them like actual covs
fp <- (orig.forest - means[2]) / sds[2]
dp <- (orig.date - means[3]) / sds[3]
durp <- (orig.duration - means[4]) / sds[4]

# Obtain predictions
newData <- data.frame(elev=ep, forest=0)
pred.occ.elev <- predict(fm20, type="state", newdata=newData, appendData=TRUE)
```

```
newData <- data.frame(elev=0, forest=fp)
pred.occ.forest <- predict(fm20, type="state", newdata=newData, appendData=TRUE)
newData <- data.frame(date=dp, dur=0)
pred.det.date <- predict(fm20, type="det", newdata=newData, appendData=TRUE)
newData <- data.frame(date=0, dur=durp)
pred.det.dur <- predict(fm20, type="det", newdata=newData, appendData=TRUE)

# Plot predictions against unstandardized 'prediction covs'
par(mfrow = c(2,2), mar = c(5,5,2,3), cex.lab = 1.2)
plot(pred.occ.elev[[1]] ~ orig.elev, type = "l", lwd = 3, col = "blue", ylim = c(0,1),
las = 1, ylab = "Pred. occupancy prob.", xlab = "Elevation (m)", frame = F)
matlines(orig.elev, pred.occ.elev[,3:4], lty = 1, lwd = 1, col = "grey")
plot(pred.occ.forest[[1]] ~ orig.forest, type = "l", lwd = 3, col = "blue", ylim = c(0,1),
las = 1, ylab = "Pred. occupancy prob.", xlab = "Forest cover (%)", frame = F)
matlines(orig.forest, pred.occ.forest[,3:4], lty = 1, lwd = 1, col = "grey")
plot(pred.det.date[[1]] ~ orig.date, type = "l", lwd = 3, col = "blue", ylim = c(0,1),
las = 1, ylab = "Pred. detection prob.", xlab = "Date (1 = 1 April)", frame = F)
matlines(orig.date, pred.det.date[,3:4], lty = 1, lwd = 1, col = "grey")
plot(pred.det.dur[[1]] ~ orig.duration, type = "l", lwd = 3, col = "blue", ylim = c(0,1), las
= 1, ylab = "Pred. detection prob.", xlab = "Survey duration (min)", frame = F)
matlines(orig.duration, pred.det.dur[,3:4], lty = 1, lwd = 1, col = "grey")
```

We like to form predictions for two covariates simultaneously and therefore predict on a grid that combines each in a suitable range of values (Figure 10.11; see section 6.9.4 for similar R code to do this for predictions of expected abundance in an *N*-mixture model).

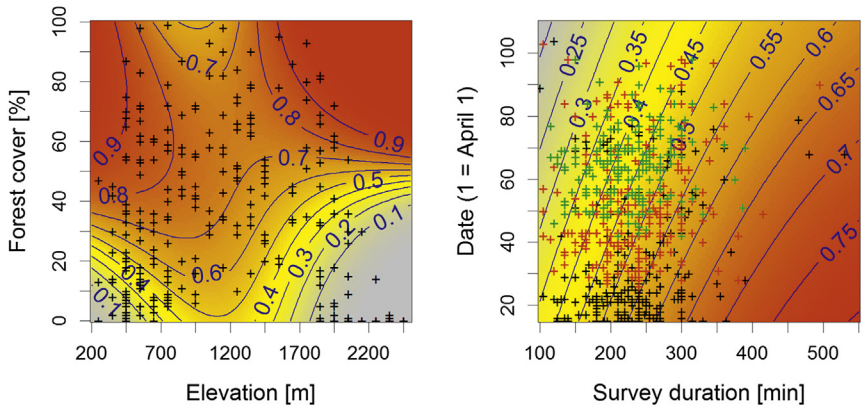

FIGURE 10.11

Two-dimensional predictions of the joint relationships of occupancy and detection probability, respectively, with two covariates under the site-occupancy model for Swiss red squirrels in 2007. Left: occupancy probability, right: detection probability. Plus signs denote the observed covariate values in the data set; in the right plot, their colors denote the first (black), second (red), and third survey (green).

```
# Predict abundance and detection jointly along two separate covariate gradients
# abundance ~ (forest, elevation) and detection ~ (survey duration, date)
pred.matrix1 <- pred.matrix2 <- array(NA, dim = c(100, 100)) # Define arrays
for(i in 1:100){
   for(j in 1:100){
       newData1 <- data.frame(elev=ep[i], forest=fp[j])       # For abundance
       pred <- predict(fm20, type="state", newdata=newData1)
       pred.matrix1[i, j] <- pred$Predicted
       newData2 <- data.frame(dur=durp[i], date=dp[j])        # For detection
       pred <- predict(fm20, type="det", newdata=newData2)
       pred.matrix2[i, j] <- pred$Predicted
   }
}

par(mfrow = c(1,2), cex.lab = 1.2)
mapPalette <- colorRampPalette(c("grey", "yellow", "orange", "red"))
image(x=orig.elev, y=orig.forest, z=pred.matrix1, col = mapPalette(100), axes = FALSE,
xlab = "Elevation [m]", ylab = "Forest cover [%]")
contour(x=orig.elev, y=orig.forest, z=pred.matrix1, add = TRUE, lwd = 1.5, col = "blue",
labcex = 1.3)
axis(1, at = seq(min(orig.elev), max(orig.elev), by = 250))
axis(2, at = seq(0, 100, by = 10))
box()
title(main = "Expected squirrel occurrence prob.", font.main = 1)
points(data$ele, data$forest, pch="+", cex=1)

image(x=orig.duration, y=orig.date, z=pred.matrix2, col = mapPalette(100), axes = FALSE,
xlab = "Survey duration [min]", ylab = "Date (1 = April 1)")
contour(x=orig.duration, y=orig.date, z=pred.matrix2, add = TRUE, lwd = 1.5, col = "blue",
labcex = 1.3)
axis(1, at = seq(min(orig.duration), max(orig.duration), by = 50))
axis(2, at = seq(0, 100, by = 10))
box()
title(main = "Expected squirrel detection prob.", font.main = 1)
matpoints(as.matrix(data[, 13:15]), as.matrix(data[, 10:12]), pch="+", cex=1)
```

Next, we produce a Swiss distribution map for the red squirrel in 2007, along with a map of the uncertainty in these predictions at each 1-km^2 quadrat (Figure 10.12). As always, producing a map is simple if we have effects of spatially indexed covariates: we simply predict the response (occupancy or detection probability) for each quadrat in the area for which we want to produce the map and then we plot this. (If your computer has problems predicting at all $\sim$42,000 km pixels at once, do it in batches of $\sim$10,000 and then stack them afterwards.)

It is important to be able to gauge the uncertainty (SE, CI, etc.) in an estimate. For instance, we would never be able to publish the results of an analysis of variance (say, a histogram of group means) without indicating SEs or posterior standard deviations. However, in the species distribution modeling world, presenting estimates (i.e., maps) without showing the associated uncertainty is currently still the rule. Clearly, this is a state that can be improved. With a regression model as the site-occupancy model, it is easy to obtain uncertainty assessments for every estimate such as, here, a prediction of occupancy

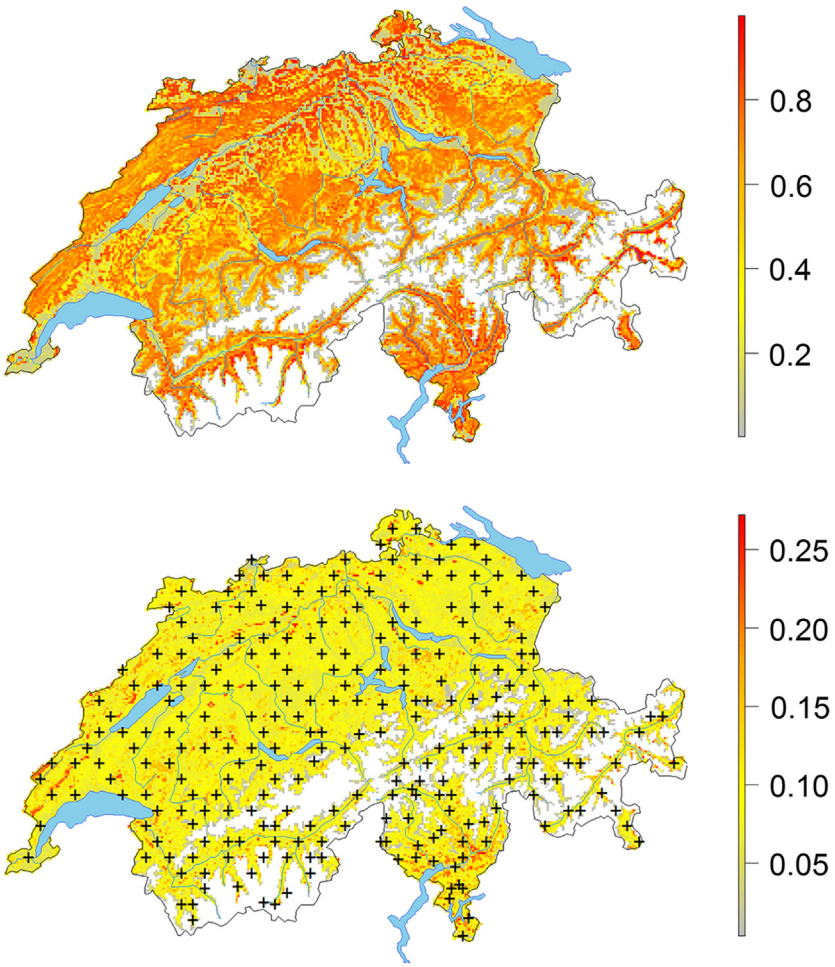

FIGURE 10.12

Species distribution map for red squirrels in Switzerland in 2007 under the best-fitting site-occupancy model (`fm20`) for data modeled at the 1-km^2 scale. The map on the top shows the expected probability of occupancy and that on the bottom the standard errors of those predictions, along with the locations of the sample locations (shown as plus signs). Areas with median elevation greater than 2250 m a.s.l. are masked (white).

or detection. So we next produce a map of the uncertainty in the preceding species distribution map, by also plotting the SEs of these predictions. (Why don't we also produce a map of the detection probability of Swiss red squirrels?)

```
# Load the Swiss landscape data from unmarked
data(Switzerland)       # Load Swiss landscape data in unmarked
CH <- Switzerland
```

```
# Get predictions of occupancy prob for each 1km2 quadrat of Switzerland
newData <- data.frame(elev = (CH$elevation - means[1])/sds[1], forest = (CH$forest - means
[2])/sds[2])
predCH <- predict(fm20, type="state", newdata=newData)

# Prepare Swiss coordinates and produce map
library(raster)

# Define new data frame with coordinates and outcome to be plotted
PARAM <- data.frame(x = CH$x, y = CH$y, z = predCH$Predicted)
r1 <- rasterFromXYZ(PARAM)     # convert into raster object

# Mask quadrats with elevation greater than 2250
elev <- rasterFromXYZ(cbind(CH$x, CH$y, CH$elevation))
elev[elev > 2250] <- NA
r1 <- mask(r1, elev)

# Plot species distribution map (Fig. 10-14 left)
par(mfrow = c(1,2), mar = c(1,2,2,5))
mapPalette <- colorRampPalette(c("grey", "yellow", "orange", "red"))
plot(r1, col = mapPalette(100), axes = F, box = F, main = "Red squirrel distribution in 2007")

# Plot SE of the species distribution map (Fig. 10-14 right)
r2 <- rasterFromXYZ(data.frame(x = CH$x, y = CH$y, z = predCH$SE))
r2 <- mask(r2, elev)
plot(r2, col = mapPalette(100), axes = F, box = F, main = "Uncertainty map 2007")
```

Finally, we estimate the area of occurrence of red squirrels in Switzerland in 2007. For this, we make the assumption that each MHB route samples exactly 1 km^2, and, hence, we simply add up the occupancy probability for each quadrat. We do this both with and without the mask cutting out areas at elevation greater than 2250 m and see that our model predicts hardly any squirrels occurring at these very high elevations.

```
# Get extent of squirrel occurrence in 2007
sum(predCH$Predicted)                          # All quadrats
[1] 17354.57
sum(predCH$Predicted[CH$elevation < 2250])  # Only at elevations < 2250 m
[1] 17350.34
```

We also want to assess the uncertainty around this estimate via the bootstrap. We do the prediction "by hand" rather than using the `predict` function because we don't need the SEs and `predict` would take way too much time when used in a bootstrapped function.

```
# Standardise prediction covariate identical to those in analysis
pelev <- (CH$elevation - means[1]) / sds[1]
pforest <- (CH$forest - means[2]) / sds[2]

# Define function that predicts occupancy under model 20
Eocc <- function(fm) {
  betavec <- coef(fm)[1:8]        # Extract coefficients in psi
```

```
  DM <- cbind(rep(1,length(pelev)), pelev, pelev^2, pforest, pforest^2, pelev*pforest,
pelev*pforest^2, pelev^2*pforest)  # design matrix
  pred <- plogis(DM%*%(betavec)) # Prediction = DM * param. vector
  Eocc <- sum(pred)              # Sum over all Swiss quadrats (no mask)
  Eocc
}

(estimate.of.occurrence <- Eocc(fm20))  # Same as before, without mask
system.time(Eocc.boot <- parboot(fm20, Eocc, nsim=1000, report=10)) # 100 sec
plot(Eocc.boot)         # Plot bootstrap distribution of extent of occurrence
quantile(Eocc.boot@t.star, c(0.025, 0.975))
     2.5%    97.5%
15131.14 20185.21

# Convert these estimates to a proportion of the country
(c(point = 17354.57, quantile(Eocc.boot@t.star, c(0.025, 0.975))) / 42275
    point      2.5%     97.5%
0.4104161 0.3579217 0.4774740
```

We conclude that in 2007 red squirrels occurred in Switzerland in about 41% of the 1-km^2 quadrats (95% CI 36−48%).

Importantly, whether we can interpret the estimated range size in terms of "permanent presence of at least one squirrel" or as "use by a squirrel sometimes during the study period" (see Section 10.2) depends on the assumption that the effective sampling area associated with an MHB sampling quadrat is exactly 1 km^2; see the discussion in Section 6.10 and also Efford and Dawson (2012). If the effective sampling area is smaller, then our estimate will be an underestimate with respect to the area of "permanent presence", and if it is bigger we will overestimate the range when we want to interpret the latter as the area of permanent presence. We think that to account for coverage bias and "sunflower effects" we could employ similar *ad hoc* methods as we did in Sections 6.9 and 6.10. To deal with the scaling of occupancy on area, we could perhaps model the underlying abundance rather than occupancy directly; see Section 3.3.6. Alternatively, a more formal treatment might be one along the lines of the seminal paper by Chandler and Clark (2014), who specify a model for binary detection/nondetection (i.e., occupancy) data that is formulated in terms of an underlying, i.e., latent point process model that accommodates movement of individuals within their home ranges. In an important paper, Ramsey et al. (2015) have reformulated the spatially explicit *N*-mixture model of Chandler and Royle (2013) for detection/nondetection data. That is, they estimate the parameters of a latent point process from the spatial correlation in the detection/nondetection data from of replicated surveys of adjacent sites (This means that their model could not be applied to MHB data as is, since no survey quadrat has any direct neighbor. Perhaps progress could be made by subdividing a 1 km^2 quadrat into four or 16 subquadrats and then modeling occurrence at that finer scale.). Making occupancy models spatially explicit is an important avenue for research.

10.10 MULTISCALE OCCUPANCY MODELS

The classical occupancy model has a single spatial scale (that for the chosen grid size) and two levels: one level for the latent occurrence state in a grid cell and another level for the repeated measurements. This two-level hierarchy can be extended to more than two levels in a straightforward fashion. For instance, we may repeatedly survey multiple spatial subunits that are each nested in multiple main units (representing

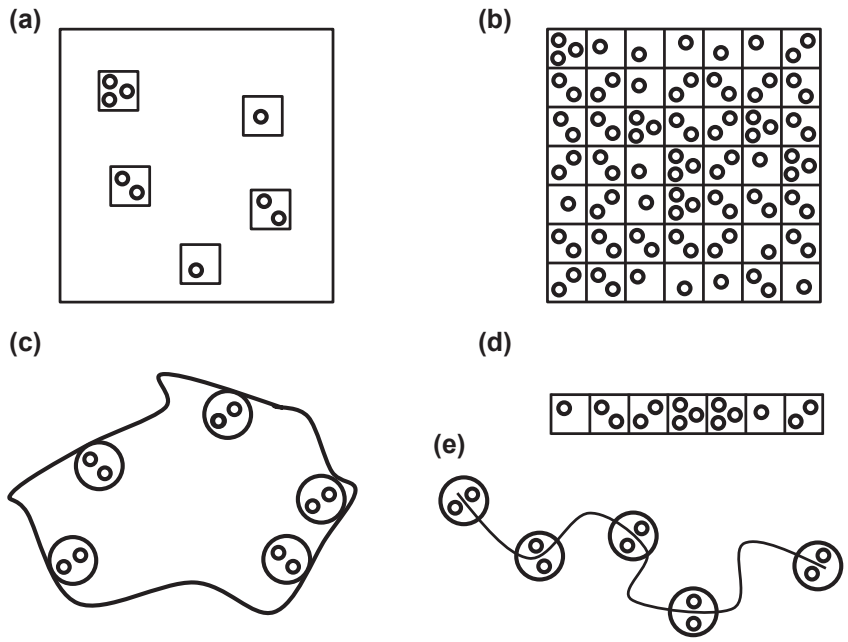

FIGURE 10.13

Five examples of multiscale designs, where smaller subunits are nested within a larger unit. A study will always comprise multiple such units, which represent the main spatial replicates that are required in occupancy modeling. Small circles represent the presence/absence measurements taken within a subunit, either at different locations or at different times at the same location. Often, we have imbalanced data, i.e., the number of measurements is not identical in every subunit; see (a), (b), and (d).

spatial subsampling); we may sample multiple units over several days, with each day subdivided into, say, three shorter time segments (representing temporal subsampling); or we may survey multiple spatial units using multiple detection methods over multiple occasions each. In disease surveillance, there may be a multilayered, nested sampling process, such as replicate PCRs run for each of a number of ducks examined at multiple ponds, which are nested within a collection of refuges (McClintock et al., 2010b). Hence, scales are defined spatially, temporally, spatiotemporally, or by different measurement methods, and, importantly, their definition will determine the precise meaning of the associated model parameters; see below. In reality, such multiscale occupancy designs are very common, but we believe that this has not yet been recognized widely enough. Formal analysis of multiscale occupancy designs started only fairly recently with Nichols et al. (2008), Aing et al. (2011) and Mordecai et al. (2011). In this section we focus on the simplest multiscale occupancy model, that with two scales and therefore three levels.

In a multiscale design, smaller units are nested within larger units (Figure 10.13). This scheme comes in a large number of variants and shapes: the larger sample units may have artificial (A, B, D) or "natural" shapes (e.g., C may be a pond and E may represent a river); only a fraction of the area (A, C, D) or the entire area of the larger unit may be sampled (B, D), and the nesting may be two-dimensional (A, B, C) or along a linear structure, such as a river or a transect; in the latter we typically have directionality in the sampling (D, E). Usually, the actual detection/nondetection data are collected in the subunits. However, there are cases where data may in addition be available at the unit level, for instance,

we may have information about occurrence of a species *somewhere* within a unit, but the particular subunit in which the species was observed may be unavailable. Cases B and D represent a special case where all subunits in a main unit are sampled, i.e., each main unit is exhaustively sampled. In a sense, the relationship between occurrence at the two levels then becomes deterministic; see below. Combining presence/absence information at the unit level and covariate information at the subunit level in a three-level model forms the basis for the interesting work by Keil et al. (2014a,b) to downscale occupancy to form finer-scale maps from rougher-scale occurrence data; see also Dupuis et al. (2011).

Several issues arise in the analysis of multiscale designs. Basically, subunits provide replicate measurements for the main units and their proximity induces a similarity, or in other words, there is a dependency among subunits within the same unit that needs to be addressed in a model. At the same time, exactly this dependency can be used to make inferences about the units based on the subunits. To develop an intuition for a simple three-level occupancy model, let's first look at a nested sampling scheme in the context of experimental design outside of distribution modeling. Let's assume that we have measured plant mass (y_{ijk}) of multiple stems (k) in a sample of plants (j) collected in a number of populations (i). The basic *block structure* of the study is given by "population/plant/stem," that is, plants are nested within populations and stems within plants, and the resulting dependencies ought to be accounted for in a model for these data. In addition, we may have covariate measurements or experimental treatments applied at any of the three levels, representing a *treatment structure*. This experimental design is called a split-plot (Nelder, 1965a,b), leading to a nested analysis of variance (ANOVA). The basic block structure for this study can be described by the following hierarchical model for the individual plant mass measurements y_{ijk}.

Unit-level with population means γ_i: $\gamma_i \sim Normal(\mu, \sigma^2_{pop})$
Subunit-level with plant means a_{ij}: $a_{ij}|\gamma_i \sim Normal(\gamma_i, \sigma^2_{plant})$
Individual stem measurement y_{ijk}: $y_{ijk}|a_{ij} \sim Normal(a_{ij}, \sigma^2_{stem})$

Population means γ_i (note this is a Greek gamma, not y) cluster around the grand mean μ with variance σ^2_{pop}, plant means a_{ij} cluster around population mean γ_i with variance σ^2_{plant}, and the measurements y_{ijk} cluster around plant mean a_{ij} with residual variance σ^2_{stem}. Such nested ANOVA or split-plot designs can be analyzed in many software packages, including BUGS (Qian and Shen, 2007; Hector et al., 2011).

Now let's see how we can apply the same concepts to the case where instead of a continuous variable we measure binary "presence/absence" at three hierarchical levels and where that measurement may be affected by false-negative errors. Imagine that we had taken up to three presence/absence measurements (k) at each of five subunits (j) in a number of main units (i), as in Figure 10.13(a) . One possible analysis inspired by the nested ANOVA treats the units as contributing a random block effect in the logit-linear model for occupancy ψ_{ij} of the subunits. That is, $\text{logit}(\psi_{ij}) = \gamma_i + \varepsilon_{ij}$, hence, the occupancy probability in a subunit (on the logit scale) is simply the sum of a contribution from unit (γ_i) and another one from subunit j in that unit (ε_{ij})

Unit-level random effect γ_i: $\gamma_i \sim Normal(\mu, \sigma^2_{pop})$
Subunit-level level presence/absence a_{ij}: $a_{ij}|\gamma_i \sim Bernoulli(\text{logit}^{-1}(\gamma_i + \varepsilon_{ij}))$
Replicated pres/abs measurement y_{ijk}: $y_{ijk}|a_{ij} \sim Bernoulli(a_{ij} * p)$

In this "bottom-up" analysis (G. Guillera-Arroita, pers. comm.), the focus is on the subunit, and the nonindependence of subunits within the same unit is accounted for by a random unit effect (γ_i) in the

logit of the occupancy probability at the subunit level. This random unit effect is the contribution of unit i to the occupancy probability at the smaller scale, making it a little higher or lower, on average, for subunits that are in the same unit. That is, γ_i is a typical random effect invoked to account for the correlation of grouped measurements, and it characterizes the unit. These random effects lack any specific biological meaning.

In an alternative model the unit-specific random effects *do* have a clear biological meaning—they are the usual "presence/absence" state z of the units. This leads to the following three-level occupancy model, which was nicely described by Aing et al. (2011) and Mordecai et al. (2011).

Unit-level presence/absence z_i: $z_i \sim Bernoulli(\psi)$
Subunit-level presence/absence a_{ij}: $a_{ij}|z_i \sim Bernoulli(z_i * \theta)$
Replicated pres/abs measurement y_{ijk}: $y_{ijk}|a_{ij} \sim Bernoulli(a_{ij} * p)$

In this "top-down" analysis (G. Guillera-Arroita, pers. comm.), we define a specific dependency in the presence/absence states between the unit and the subunit levels—a subunit can only be occupied ($a_{ij} = 1$) when the unit it belongs to is also occupied ($z_i = 1$). Random variables z_i and a_{ij} represent presence/absence in unit i and subunit ij, respectively, and y_{ijk} is the detection/nondetection datum in replicate k, subunit j and unit i. The parameters governing the three random variables are the unit-level (large-scale) occupancy probability ψ, the subunit-level (small-scale) occupancy probability θ, and the detection probability p associated with the actual replicated presence/absence measurement y taken at a subunit. These parameters can be indexed by i, ij, and ijk, respectively, though in practice we need to impose constraints, i.e., add covariates or random effects, to make the model identifiable.

The middle level has an important interpretation as the probability of availability (i.e., 1 minus temporal emigration) or small-scale, or temporary, occupancy, whereas the top level is the probability of permanent, or asymptotic, presence or large-scale occupancy probability (Efford and Dawson, 2012; Pavlacky et al., 2012). The middle level can either be viewed as a component of the state model (when we focus on different scales of occupancy) or as a component of the observation model (when temporary emigration is treated as a nuisance to be addressed to better estimate occupancy in the main units). Hence, if we are worried about the meaning of the occupancy parameter in a two-level occupancy model when there is temporary emigration (random in/out type of closure violation), we can simply collect data at an additional level and then estimate both probabilities of use (corresponding to ψ) and probability of temporary presence (corresponding to availability, θ). That is, to estimate parameters at each level, we need extra information, i.e., nested replicates at every level. Alternatively, we need extra information, e.g., an estimate of, say, p, from another study, which can be introduced as an informative prior in a Bayesian analysis, or perhaps make assumptions that some covariate contains information about variability in one of the parameters. There are two main ways in which we can get back to the usual two-level model: either by setting $\theta = 1$ or by setting $p = 1$. In the former, we have perfect availability, and whenever a unit is occupied, each single subunit is so, too. In the latter, whenever a subunit is occupied, we only observe detections, never nondetections.

In both the ANOVA-type and the triple-Bernoulli, three-level occupancy models just described, the similarity within the same unit is assumed to be identical for all subunits, and we assume there is no further spatial autocorrelation among subunits within the same unit. It is possible to account for spatial autocorrelation among subunits within a unit using methods presented in Chapters 21 and 22 (in volume 2), and analogous occupancy models have been developed for linear designs (such as D and E, representing sampling along trails or rivers; Hines et al., 2010, 2014; Aing et al., 2011). Another

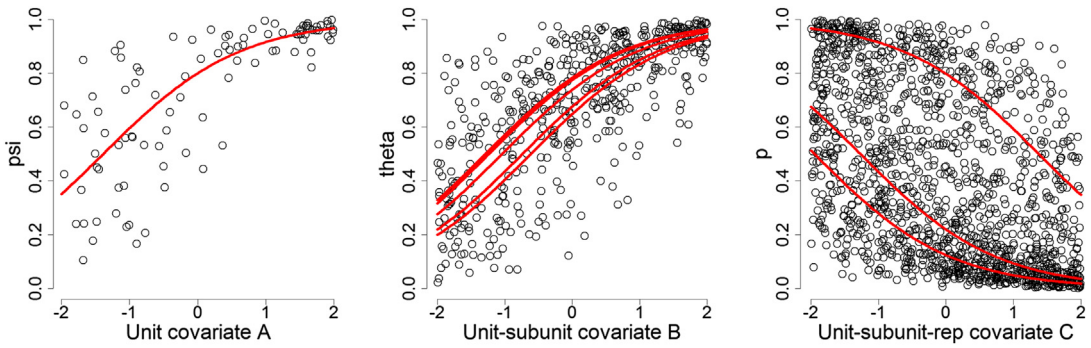

FIGURE 10.14

Example of the graphical output from the data simulation function `sim3Occ` (from model 4 in the text). Left: large-scale occupancy probability at 100 units (psi, red is expected value), middle: small-scale occupancy or availability probability for 500 subunits (theta, red shows expected value for five subunits which are best imagined to represent temporal variability; see Schmidt et al. (2013)), right: detection probability for 1500 individual measurements (p, red shows expected value for three replicates).

possibility to deal with temporal autocorrelation for multiscale data with temporal subsampling is to fit a dynamic occupancy model (MacKenzie et al., 2003; see Chapter 16 in volume 2); this is done by Rota et al. (2009). We present some simpler occupancy models for data collected along linear structures in Section 10.12.

Next, we illustrate a three-level occupancy model using simulated data and terminology inspired by a disease surveillance study reported in Schmidt et al. (2013). The function `sim3Occ` lets us generate three-level occupancy data sets with covariates, time effects, and unstructured random variability at every level possible (Figure 10.14). The function defaults are:

```
sim3Occ(nunit = 100, nsubunit = 5, nrep = 3, mean.psi = 0.8, beta.Xpsi = 1, sd.logit.psi = 0,
mean.theta = 0.6, theta.time.range = c(-1, 1), beta.Xtheta = 1, sd.logit.theta = 0, mean.p =
0.4, p.time.range = c(-2,2), beta.Xp = -1, sd.logit.p = 0)
```

This has us simulate data collected at 100 units, with five subunits in each and three presence/absence measurements on each subunit. The three main parameters of the model (large-scale occupancy ψ, small-scale occupancy θ, and detection p) are determined by their intercepts on the probability scale, `mean.psi`, `mean.theta`, and `mean.p`, respectively, and all three may depend on one unit, unit-subunit, and unit-subunit-replicate-specific, continuous covariate, with coefficients `beta.Xpsi`, `beta.Xtheta`, and `beta.Xp`, respectively. In addition, we can specify differences between the subunits (identical for all units) by the arguments `theta.time.range` and `p.time.range` and random noise/heterogeneity in all three parameters by setting to nonzero the arguments `sd.logit.psi`, `sd.logit.theta`, and `sd.logit.p = 0`. Executing the function also creates plots of how the three main parameters vary as a function of the covariates, time, and heterogeneity (Figure 10.14). We give a partially edited summary of the function output with some added comments.

```
data <- sim3Occ()          # Execute function with default args
str(data)                  # Summary of output with some comments added
```

```
List of 28
 [ ... output truncated ... ]
 $ p                 : num [1:100, 1:5, 1:3] 0.699     # p for each datum
 $ z                 : int [1:100(1d)] 0 1 1 0 1 0 1   # presence at unit
 $ a                 : int [1:100, 1:5] 0 0 1 0 0      # presence at subunit
 $ y                 : int [1:100, 1:5, 1:3] 0 0 1 0   # detection/nondetection
 $ sum.z             : int 65          # True number of units with presence
 $ obs.sum.z         : int 63          # Observed number of units with presence
 $ sum.z.a           : int 63          # see below
[ ... output truncated ... ]
```

Output `sum.z.a` is the true number of units with presence within the subunits actually sampled. Note that this will not always be identical to the number of occupied units, since a species may occur in a unit but happen to be absent in the finite number of the particular subunits surveyed. This will happen more often with small values of `nsubunit` and `mean.theta`. See Section 11.2 for the analogous problem of sampling a metacommunity of species and Adams et al. (2010) for that of estimating occurrence of a disease at a site (=unit) when individual amphibians are treated as subunits.

As always, we encourage you to play with this function with changed arguments to train your intuition about three-level occupancy designs, by looking at a summary of the output (as above) and the plots produced. Here is a sample of four possible settings. The simplest possible model has constant parameters and no other sources of variation.

```
# 'Null' model (model 1)
str(data <- sim3Occ(nunit = 100, nsubunit = 5, nrep = 3, mean.psi = 0.8, beta.Xpsi = 0,
sd.logit.psi = 0, mean.theta = 0.6, theta.time.range = c(0, 0), beta.Xtheta = 0,
sd.logit.theta = 0, mean.p = 0.4, p.time.range = c(0,0), beta.Xp = 0, sd.logit.p = 0))
```

We can let `theta` and `p` vary by "time," to make `theta` different among subunits and to make `p` different for each replicate. Such a design might be sensible if subunits denote different samples that are taken in time, as in the example described below (from Schmidt et al., 2013).

```
# No covariate effects, no random variability (model 2)
str(data <- sim3Occ(nunit = 100, nsubunit = 5, nrep = 3, mean.psi = 0.8, beta.Xpsi = 0,
sd.logit.psi = 0, mean.theta = 0.6, theta.time.range = c(-1, 1), beta.Xtheta = 0,
sd.logit.theta = 0, mean.p = 0.4, p.time.range = c(-2,2), beta.Xp = 0, sd.logit.p = 0))
```

We can let `psi`, `theta`, and `p` be affected by linear-logistic effects of three separate covariates.

```
# All covariate effects, but no random variability (model 3)
str(data <- sim3Occ(nunit = 100, nsubunit = 5, nrep = 3, mean.psi = 0.8, beta.Xpsi = 1,
sd.logit.psi = 0, mean.theta = 0.6, theta.time.range = c(-1, 1), beta.Xtheta = 1,
sd.logit.theta = 0, mean.p = 0.4, p.time.range = c(-2,2), beta.Xp = -1, sd.logit.p = 0))
```

And we can also add random unstructured noise at every level (Figure 10.14).

```
# Most complex model with all effects allowed for by sim function (model 4)
str(data <- sim3Occ(nunit = 100, nsubunit = 5, nrep = 3, mean.psi = 0.8, beta.Xpsi = 1,
sd.logit.psi = 1, mean.theta = 0.6, theta.time.range = c(-1, 1), beta.Xtheta = 1,
sd.logit.theta = 1, mean.p = 0.4, p.time.range = c(-2,2), beta.Xp = -1, sd.logit.p = 1))
```

Multiscale occupancy models cannot be fit in `unmarked`, but some can in PRESENCE and MARK. We here illustrate the fitting of model 3 in BUGS. We generate a data set and imagine a story using the scenario of disease sampling by environmental DNA (eDNA) described in Schmidt et al. (2013): five water samples were taken in each of 100 ponds and then analyzed twice with PCR for the presence or absence of a fungus that kills amphibians and that has an unpronouncable name that starts with letter B. Thus, we have the results of two PCR samples nested within five water samples (=subunits) nested within each of 100 ponds (=units).

```
set.seed(1)
str(data <- sim3Occ(nunit = 100, nsubunit = 5, nrep = 3, mean.psi = 0.8, beta.Xpsi = 1,
mean.theta = 0.3, theta.time.range = c(-1, 1), beta.Xtheta = 1, mean.p = 0.2, p.time.range =
c(-1,1), beta.Xp = -1))
```

```
Occupied units:                                81
Units with >=1 occupied, surveyed subunit:     65
Observed number of occupied units:             57
```

In this data set, among 100 ponds, 81 turned out to be occupied, but due to small-scale heterogeneity of occurrence, within the five water samples the B fungus actually occurred in only 65 of them. That is, it must have occurred elsewhere in 16 among the 81 occupied ponds. Due to imperfect detection of the PCR methods in the lab, the B fungus was only detected in 57 ponds among the 65 in which it did occur among the five water samples taken.

```
# Look at data
str(data$z)      # True quadrat (pond) occurrence state
str(data$a)      # True subquadrat (water sample) occurrence state
str(data$y)      # Observed data
cbind("pond"=data$z, "sample 1"= data$a[,1], "sample 2"= data$a[,2], "sample 3"=
data$a[,3], "sample 4"= data$a[,4], "sample 5"= data$a[,5])
which(data$z-apply(data$a, 1, max)==1) # Fungus present in pond, but not in examined
samples
```

We fit the data-generating model in BUGS.

```
# Bundle and summarize data set
y <- data$y
str( win.data <- list(y = y, n.pond = dim(y)[1], n.samples = dim(y)[2], n.pcr = dim(y)[3],
covA = data$covA, covB = data$covB, covC = data$covC) )

# Define model in BUGS language
sink("model.txt")
cat("
model {

# Priors and model for params
int.psi ~ dunif(0,1)          # Intercept of occupancy probability
for(t in 1:n.samples){
   int.theta[t] ~ dunif(0,1) # Intercepts of availability probability
}
```

```
for(t in 1:n.pcr){
   int.p[t] ~ dunif(0,1)      # Intercepts of detection probability (1-PCR error)
}
beta.lpsi ~ dnorm(0, 0.1)     # Slopes of three covariates
beta.ltheta ~ dnorm(0, 0.1)
beta.lp ~ dnorm(0, 0.1)

# "Likelihood" (or basic model structure)
for (i in 1:n.pond){
   # Occurrence in pond i
   z[i] ~ dbern(psi[i])
   logit(psi[i]) <- logit(int.psi) + beta.lpsi * covA[i]
   for (j in 1:n.samples){
      # Occurrence in sample j
      a[i,j] ~ dbern(mu.a[i,j])
      mu.a[i,j] <- z[i] * theta[i,j]
      logit(theta[i,j]) <- logit(int.theta[j]) + beta.ltheta * covB[i,j]
      for (k in 1:n.pcr){
         # PCR detection error process in sample k
         y[i,j,k] ~ dbern(mu.y[i,j,k])
         mu.y[i,j,k] <- a[i,j] * p[i,j,k]
         logit(p[i,j,k]) <- logit(int.p[k]) + beta.lp * covC[i,j,k]
      }
   }
  tmp[i] <- step(sum(a[i,])-0.1)
}

# Derived quantities
sum.z <- sum(z[])     # Total # of occupied ponds in sample
sum.a <- sum(tmp[])  # Total # of ponds with presence in >=1 of the 5 samples
} # end model
",fill=TRUE)
sink()

# Initial values
zst <- apply(y, 1, max)         # inits for presence (z)
ast <- apply(y, c(1,2), max)   # inits for availability (a)
inits <- function() list(z = zst, a = ast, int.psi = 0.5, beta.lpsi = 0)

# Parameters monitored
params <- c("int.psi", "int.theta", "int.p", "beta.lpsi", "beta.ltheta", "beta.lp",
"sum.z", "sum.a")

# MCMC setting
ni <- 5000  ;  nt <- 2  ;  nb <- 1000  ;  nc <- 3

# Call WinBUGS and summarize posterior
out <- bugs(win.data, inits, params, "model.txt", n.chains = nc, n.thin = nt, n.iter = ni,
n.burnin = nb, debug = TRUE, bugs.dir = bd) # bd="c:/WinBUGS14/"
print(out, 3)
data$sum.z
```

Thus, in BUGS, it is straightforward to extend the basic two-level model to a three-level model and it would be equally easy to add more levels still (though JAR notes: "but with a concomitant decrease in our ability to understand the model"). Multiscale occupancy models are very powerful and occur surprisingly often in ecology and management, although only a mere handful of papers has used them so far. For related multiscale models of abundance, see Sections 6.14 and 9.5.

10.11 SPACE-FOR-TIME SUBSTITUTION

Typically, the information to estimate the detection parameters separately from the parameters in the occupancy submodel comes from temporal replicates, or else, from simultaneous deployment of several observers or detection devices. However, it has been argued that we can use *spatial* replicates as a surrogate, for instance, by dividing up a larger unit such as a 40-km route in the North American breeding bird survey into smaller, nested subunits of five 8-km segments or as in any of the cases depicted in Figure 10.13. Such a space-for-time substitution was used in some of the earlier capture-recapture literature on the estimation of species richness (Boulinier et al., 1998; Nichols et al., 1998a,b; Cam et al., 2002b,c; Doherty et al., 2003) and is sometimes also used in occupancy modeling (Royle and Kéry, 2007; Guillera-Arroita, 2011; Sadoti et al., 2013). Being able to use space for time in this way is very advantageous since it enables occupancy and detection probability to be estimated separately in an otherwise unreplicated design. This space-for-time substitution must be one of the least well understood topics in occupancy modeling. We include this section in part to motivate others to shed more light on this topic, but we would like to warn you that what we say here is quite tentative only.

One way to view the space-for-time design is as a restricted three-level occupancy model, where there is no replication at the bottom level. Therefore, when fitting the traditional two-level occupancy model, two parameters in the data-generation mechanism are lumped and only their product can be estimated. That is, we collapse our three-level model to two levels.

Large-scale occupancy at unit level (i): $z_i \sim Bernoulli(\psi)$
Detection model for unreplicated subunit surveys (ik): $y_{ik}|z_i \sim Bernoulli(z_i \times \theta \times p)$

As before, ψ is the (large-scale) occupancy probability, but what we estimate as "detection probability" in the two-level model now is the product of small-scale occupancy probability (θ) and detection probability proper (p). We have seen in the last section that with sufficient replication (i.e., at least some units have more than one subunit sampled and at least some subunits have more than a single survey), we can estimate all three parameters (ψ, θ, p) or functions thereof, such as regression coefficients for covariates. In contrast, when we only have a single observation per subunit (i.e., no replication at the lowest level), what we estimate when we feed these data into a traditional two-level occupancy model is ψ as "occupancy" and the product θp as "detection." An exception is the unlikely case in which we have measured covariates that exactly explain variability in θ and p. In this case, and under the strong assumption that the covariate model is known exactly, we can estimate all parameters in a restricted three-level design even without replication at the bottom level. Arguably, this special case is analogous to the restrictive conditions under which we can estimate parameters of a logistic regression from presence-only data (Lele and Keim, 2006; Royle et al., 2012) or those of a two-level occupancy model from unreplicated data (Lele et al., 2012; Knape and Korner-Nievergelt, 2015).

We think that one implicit assumption of space-for-time is that both unit- and subunit-level occupancy must be distinct random variables. This means that the subunits must not exhaustively sample the area represented by the unit because otherwise unit-level occupancy (z) is no longer a separate random variable but simply a deterministic function of the occurrence states (a) of the subunits comprising that unit. With exhaustive sampling, $z = \max(a)$ and ψ is

$$\psi_i = 1 - \prod_{j=1}^{J}(1 - \theta_{ij})$$

for the subunits in unit i, there is only one random process involved in the middle and the top level of the design. A different way of looking at this, but which leads to the same result, has to do with the movement of individuals among subunits. When the movement of individuals on and off a subunit lets a subunit be occupied or not in some random fashion, then unit-scale occupancy can be estimated in a space-for-time design, even when a unit is exhaustively sampled by subunits in the study. When subunits cover the entire unit (i.e., we sample exhaustively), we must then assume that individuals may be able to temporarily move off the unit, making the occurrence of the study species at the precise time at which surveys take place a random variable. In contrast, when individuals cannot temporarily move outside of a subunit, then we don't think that the third model in Section 10.10 makes sense and that instead the second model with an ANOVA-type of unit random effect ought to be used for inference. However, in this latter case, subunits don't contain information about detection probability in the absence of (temporal) replication at the bottom level.

If this argument makes sense, then in the case of the North American BBS (e.g., Royle and Kéry, 2007), we can adopt space-for-time *if* we assume that each route samples the species occurrence in some larger area, because the points sample species occurrence in some vaguely defined area around that route. A species may occur in the larger area without necessarily occurring at the sample points. In contrast, if we have regular quadrats (units) that are, say, divided up into four quadrants (subunits) each, then we are not sure whether space-for-time makes sense. Presumably it does, if we can assume that the units (and therefore also the subunits) sample some larger area around. Then, at any given time when surveyed, the subunit and the unit may be occupied by a species or not in some random fashion, because each individual present in the larger area happens to use a part of its home range that lies within the (sub)unit or it does not and this happens according to a Bernoulli process.

We illustrate space-for-time with two data sets generated with the `sim3Occ` function. First, we show the perhaps unlikely case where there is no replication at the bottom level, but where we have a covariate that explains detection at a known scale (here, the logistic). Only in this case, or if we have knowledge about one of the parameters from elsewhere, can we estimate all parameters of the three levels in the absence of replication at one of the levels. After that, we illustrate a more typical case where no such covariate is available.

10.11.1 A MAGICAL COVARIATE

In the first case, we assume that we have a magical covariate that explains variation in detection probability or in availability probability at a known scale. Here we illustrate the former and simulate additional random noise in all three parameters and set `nrep = 1`, thus, we have no replication at the bottom level.

```
set.seed(1)
data <- sim3Occ(nunit = 500, nsubunit = 5, nrep = 1, mean.psi = 0.8, beta.Xpsi = 1,
sd.logit.psi = 0.4, mean.theta = 0.6, theta.time.range = c(0, 0), beta.Xtheta = 1,
sd.logit.theta = 0.6, mean.p = 0.4, p.time.range = c(0,0), beta.Xp = -1, sd.logit.p = 0.8)
```

```
Occupied units:                              367
Units with >=1 occupied, surveyed subunit: 361
Observed number of occupied units:           285
```

We use JAGS to fit the full three-level model with covariates, without estimating the magnitude of the random noise in the parameters.

```
# Bundle and summarize data set
y <- data$y
str( win.data <- list(y = y, nunit = dim(y)[1], nsubunit = dim(y)[2], nrep = dim(y)[3],
covA = data$covA, covB = data$covB, covC = data$covC) )

# Define model in BUGS langauge
sink("model.txt")
cat("
model {

# Priors
int.psi ~ dunif(0,1)    # Occupancy probability
int.theta ~ dunif(0,1) # Availability probability
int.p ~ dunif(0,1)      # Detection probability
beta.lpsi ~ dnorm(0, 0.01)
beta.ltheta ~ dnorm(0, 0.01)
beta.lp ~ dnorm(0, 0.01)

# Likelihood
for (i in 1:nunit){
   # Occupancy model for quad i
   z[i] ~ dbern(psi[i])
   logit(psi[i]) <- logit(int.psi) + beta.lpsi * covA[i]
   for (j in 1:nsubunit){
      # Availability in subquad j
      a[i,j] ~ dbern(mu.a[i,j])
      mu.a[i,j] <- z[i] * theta[i,j]
      logit(theta[i,j]) <- logit(int.theta) + beta.ltheta * covB[i,j]
      for (k in 1:nrep){
         # PCR detection error process in replicate k
         y[i,j,k] ~ dbern(mu.y[i,j,k])
         mu.y[i,j,k] <- a[i,j] * p[i,j]
         logit(p[i,j]) <- logit(int.p) + beta.lp * covC[i,j,1]
      }
   }
   tmp[i] <- step(sum(a[i,])-0.1)
}
```

```
# Derived quantities
sum.z <- sum(z[])       # Total number of occupied quadrats
sum.a <- sum(tmp[])     # Total number of quads with presence in samples
p.theta <- int.p * int.theta  # What a 2-level model estimates as 'p'
}
",fill=TRUE)
sink()

# Initial values
inits <- function() list(z = array(1, dim = data$nunit), a = array(1, dim =c(data$nunit,
data$nsubunit)) )      # Set all to 1 to avoid conflict

# Parameters monitored
params <- c("int.psi", "int.theta", "int.p", "beta.lpsi", "beta.ltheta",
"beta.lp","p.theta", "sum.z", "sum.a")

# MCMC settings
ni <- 25000  ;  nt <- 2  ;  nb <- 2000  ;  nc <- 3

# Call JAGS (ART 15 min) and summarize posterior
out <- jags(win.data, inits, params, "model.txt", n.chains = nc, n.thin = nt, n.iter = ni,
n.burnin = nb, parallel = T)
traceplot(out)  ;  print(out, 3)

# Compare truth and estimate in table
tmp <- cbind(rbind(mean.psi = data$mean.psi, mean.theta = data$mean.theta,
mean.p = data$mean.p, beta.lpsi = data$beta.Xpsi, beta.ltheta = data$beta.Xtheta, beta.lp
= data$beta.Xp, product.theta.p = data$mean.theta* data$mean.p, sum.z = data$sum.z),
out$summary[c(1:8),c(1, 3, 7)])
colnames(tmp) <- c("Truth", "Post.mean", "LCRL", "UCRL")
print(tmp, 3)
                  Truth  Post.mean    LCRL     UCRL
mean.psi           0.80      0.783   0.708    0.863
mean.theta         0.60      0.615   0.466    0.776
mean.p             0.40      0.421   0.326    0.552
beta.lpsi          1.00      0.879   0.532    1.298
beta.ltheta        1.00      0.845   0.557    1.243
beta.lp           -1.00     -0.904  -1.168   -0.706
product.theta.p    0.24      0.255   0.226    0.287
sum.z            367.00    370.350 348.000  394.000
```

Note that the species was observed at 285 quadrats.

10.11.2 NO MAGICAL COVARIATE KNOWN: θ AND p ARE CONFOUNDED

By far the more typical case is when we *don't* have a magical covariate to explain variation at the unreplicated level and where θ and p are confounded; this is the typical application of the space-for-time design. Here, we demonstrate via simulation that a traditional (two-level) occupancy model fitted to space-for-time data appears to produce unbiased estimates of ψ and a covariate affecting ψ

and yields an estimate of the product of θ and p as its "probability of detection." Notably, we add unexplained noise in all three levels of the model—"time variation" in θ and random noise in all three levels. This represents the typical case where there is unexplained variation in θ and p as we would expect in the real world. When discussing the space-for-time occupancy design with colleagues, we had quite often sensed an uneasiness about a possible hidden assumption that availability probability (θ) must be constant within a unit. In our simulation, availability probability is *not* constant at all and neither is ψ and p. So let's see whether we can still estimate features of the model for ψ. We define a function to fit the model in `unmarked`, so that we can use the R function `try` to prevent a crash of the simulation whenever estimation fails in function `occu`.

```
# Load unmarked and define a function to fit the model with unmarked
library(unmarked)
occUM.fn <- function(data = data, inits = c(1, 1, -1)){
   umf <- unmarkedFrameOccu(y = data$y[,,1], siteCovs =
      data.frame(covA = data$covA))
   tmp1 <- summary(fm <- occu(~1 ~covA, data=umf, starts=inits))
   tmp <- matrix(unlist(c(tmp1$state[,1], tmp1$det[,1], tmp1$state[,2],
      tmp1$det[,2])), ncol = 2, byrow = F)
   dimnames(tmp) <- list(c("Occ_Int", "Occ_A", "Det_Int"), c("MLE", "SE"))
   return(MLE = tmp)
}

# Choose number of simulations and create structures to hold results
simreps <- 10000                  # takes about 30 min
obs.stats <- array(NA, dim = c(simreps, 3))
dimnames(obs.stats) <- list(NULL, c("sum.z", "obs.sum.z", "sum.z.x"))
MLEs <- array(NA, dim = c(3,2,simreps))
dimnames(MLEs) <- list(c("Occ_Int", "Occ_A", "Det_Int"), c("MLE", "SE"), NULL)

# Set timer and launch simulation
system.time(
for(i in 1:simreps){
   cat("\n\n*** Simrep Number:", i, "***\n\n")
   # Generate data set
   data <- sim3Occ(nunit = 500, nsubunit = 5, nrep = 1, mean.psi = 0.8,
      beta.Xpsi = 1, sd.logit.psi = 0.4, mean.theta = 0.6,
      theta.time.range = c(-1, 1), beta.Xtheta = 0, sd.logit.theta = 0.6,
      mean.p = 0.4, p.time.range = c(0,0), beta.Xp = 0, sd.logit.p = 0.8)
   # Save stats
   obs.stats[i,] <- unlist(data[23:25])
   # Get MLEs of occupancy model and save them
   UMmle <- try(occUM.fn(data = data, inits = c(1,1,-1)))
   if (class(UMmle) == "try-error") {v<-1} else {
    MLEs[,,i] <- UMmle
   }
   rm(data, UMmle)
   }
)
```

```
# Visualize results
par(mfrow = c(1,3), mar = c(5,5,3,2), cex.lab = 1.5, cex.axis = 1.5, cex.main = 1.5)
# Estimate of occupancy (psi)
hist(plogis(MLEs[1,1,]), breaks = 40, col = "grey", main = "Quadrat occupancy (psi)")
abline(v = mean(plogis(MLEs[1,1,]), na.rm = T), col = "blue", lwd = 3)
abline(v = 0.8, col = "red", lwd = 3)
# Estimate of occupancy covariate (A)
hist(MLEs[2,1,], breaks = 40, col = "grey", main = "Quadrat occupancy covariate (covA)")
abline(v = mean(MLEs[2,1,], na.rm = T), col = "blue", lwd = 3)
abline(v = data$beta.Xpsi, col = "red", lwd = 3)
# Estimate of "detection": product of theta and p
hist(plogis(MLEs[3,1,]), breaks = 40, col = "grey", main = "'Detection
probability' = \n theta * p")
abline(v = mean(plogis(MLEs[3,1,]), na.rm = T), col = "blue", lwd = 3)
abline(v = data$mean.theta * data$mean.p, col = "red", lwd = 3, lty = 2)
```

The means of all estimates in Figure 10.15 agree very well with the values used to simulate the data. In particular, what the two-level occupancy models calls "detection probability" (right) matches well the product of the probabilities of availability (θ) and detection (p) in the three-level data-generating occupancy model. Hence, it seems that valid inferences about occupancy, including covariate relationships, at the unit scale can be obtained with a space-for-time substitution design, even when there is plenty of unexplained variation, including nonconstant availability probability.

In spite of this, we think that a much better understanding of the space-for-time design is needed to apply it with confidence. Simulation studies that vary different parameters and that include an explicit description of the movements of individuals in their home range appear to be important in this endeavor.

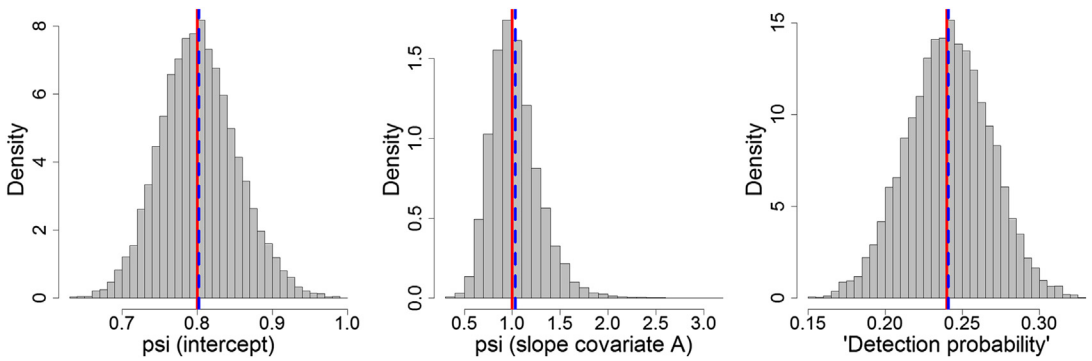

FIGURE 10.15

Simulation study with a space-for-time occupancy design. Left and middle: Estimates of intercept (on probability scale) and slope of a site covariate in the model for ψ. Right: Estimate of constant "detection probability"; the red line right shows the product of the probabilities of availability (θ) and detection (p) in the three-level data-generating occupancy model. Blue lines show the mean of 10,000 simulation replicates, and red lines show the truth in data generation.

10.12 MODELS FOR DATA ALONG TRANSECTS: POISSON, EXPONENTIAL, WEIBULL, AND REMOVAL OBSERVATION MODELS

Frequently, presence/absence data are collected along linear structures, which may be natural (e.g., surveys along rivers or coastlines) or artificial, such as any line transect in continuous habitat (e.g., in the North American BBS or in the Swiss MHB; see Figure 6.8). Issues that arise include the following: whether to discretize space, model serial autocorrelation, aggregate multiple detections, or use only part of the data to avoid dependency issues. Interestingly, the same issues arise when modeling occupancy data collected over time (e.g., with camera traps). In this sense, space and time can be thought of as more or less exchangeable in this section.

If you discretize occupancy data collected along a transect and analyze using a standard occupancy model, you lose some information whenever more than a single detection per occasion is subsumed into a single "1" in the resulting detection history. Hence, it might seem that it would be best to discretize so finely that you never have more than a single detection per occasion. On the other hand, this may lead to very large data sets (lots of occasions), and especially, there will likely be serial dependence, i.e., adjacent occasions may not be independent samples from the underlying process. You would have to deal with this problem by adopting more complex occupancy models, e.g., which assume Markovian dependence among neighboring segments (Hines et al., 2010, 2014; Aing et al., 2011). Such models have been implemented in PRESENCE and MARK, but not in `unmarked`.

Some might argue that discretization of measurements made on a continuous process is always a bad idea and that you should directly model a continuous detection process, in space or in time. This typically leads to Poisson process models for occupancy designs (Guillera-Arroita et al., 2011, 2012). The simplest such models assume independence in space or time—individual detections can be aggregated into a detection frequency per transect (or camera) and some total time interval, say C_i for the number of detections at trap or transect i of length L_i. We can then specify the following variant of an occupancy model, which simply has a Poisson instead of a binomial observation process, but is otherwise identical to the standard model.

$$z_i \sim Bernoulli(\psi)$$

$$C_i|z_i \sim Poisson(z_i * L_i * \lambda_i)$$

Here, L_i is the length of the spatial or temporal "observation window," λ_i is the detection rate per unit time or transect length, i.e., the expected number of detections per unit-length transect, given that a transect is occupied ($z_i = 1$). Patterns in occupancy can be modeled in the usual manner and those in the observation process via a log link, exactly as in a Poisson GLM. This model can easily be fitted in BUGS. Note that a single replicate (i.e., visit) is sufficient since there is no index j for replicates (though there would be if you have also temporal replicates, and more visits typically means more information and more precise estimates).

If independence can be assumed, this is a useful model. However, with dependence, e.g., aggregation of occurrence in time or space, we need to account for it, and one approach in continuous time/space is then to model this serial autocorrelation at the level of the individual detections. That is, we imagine two latent processes underlying the measured detections, between which there are switches with some probability that can be estimated. Guillera-Arroita et al. (2012) developed a two-state Markov-modulated point process model for this, and Murray Efford (pers. comm.) has written R code to fit the model, which is available from him on request.

The need for a more complex model for detections in continuous space or time in the case of serial dependence is one motivation for discretization and modeling of the aggregated data. In that way one may get rid of the serial dependence and be able to use a simpler model. Another way to avoid autocorrelation is to throw away some information and only model the first detection along every transect segment. In continuous space/time, this leads to an occupancy model that has an observation process like a continuous-time survival model (Garrard et al., 2008), where the time to first detection is modeled as a random variable with an exponential or Weibull distribution. For discretized data, we obtain a model with removal design (MacKenzie and Royle, 2005, p. 102 in MacKenzie et al., 2006; Rota et al., 2009; Guillera-Arroita and Lahoz-Monfort, in press). We cover all three in the remainder of this section.

10.12.1 OCCUPANCY MODELS WITH "SURVIVAL MODEL" OBSERVATION PROCESS: EXPONENTIAL TIME-TO-DETECTION MODEL WITH SIMULATED DATA

In this occupancy model with a continuous-time observation process, known as time-to-detection (TTD) protocol, "survival analysis" (in the medical or engineering sense) or time-to-event analysis, we obtain separate information about occurrence and detection probability from a single visit at each site (Garrard et al., 2008, 2013, 2015; also see McCarthy et al., 2013, and Bornand et al., 2014). It appears a very natural modeling approach for the types of search behavior that includes long and not necessarily very standardized surveys. The response variable in this model is the time (or possibly the transect length) until the first individual of the target species is detected (or the first cue, e.g., if you are searching for distinctive feces). A natural place to start modeling is the adoption of an exponential distribution, which describes the time between events in a Poisson process, i.e., where events occur independently with a constant rate in continuous time. This includes the time until the first event is observed. For a Poisson rate parameter λ, the expected (i.e., mean) time between events is $1/\lambda$.

One complication is that whenever the species is not detected after the maximum search time at a site, $Tmax_i$, there is uncertainty about the state of site, therefore, we must model time to detection (y_i) as a censored exponential random variable. By $d(Tmax_i)$ we indicate the response at a site to be censored at the value $Tmax_i$, i.e., when the species is not detected after $Tmax_i$, $d = 1$ and we treat the response y_i as missing. The model can be written as follows:

$$\text{Model for presence/absence: } z_i \sim Bernoulli(\psi)$$

$$\text{Model for censoring of data: } d(T\max_i) = z_i * I(y_i > T\max_i) + (1 - z_i)$$

$$\text{Model for the observed data: } y_i|z_i \sim Exponential(\lambda_i) \quad \text{if } d_i = 0$$

$$y_i = NA \quad \text{if } d_i = 1$$

Hence, response y_i is censored and will be set at NA either if time to detection happens to be greater than the maximum search time (but the site is occupied) or if the site is unoccupied. Detection probability (p) until time t is a function of the detection rate λ and the search time t: $p = 1 - \exp(-\lambda_i t)$ (Garrard et al., 2008). We next use a function (`simOccttd`) to simulate data under this model, where we may specify effects of one continuous covariate in each of occupancy and the Poisson rate parameter,

specified via the usual logit and log links, respectively. The function arguments with their defaults are the following.

```
simOccttd(M = 250, mean.psi = 0.4, mean.lambda = 0.3, beta1 = 1, alpha1 = -1, Tmax = 10)
# Function arguments:
# M:              Number of sites
# mean.psi:       Intercept of occupancy probability
# mean.lambda:    Intercept of Poisson rate parameter
# beta1:          Slope of continuous covariate B on logit(psi)
# alpha1:         Slope of continuous covariate A on log(lambda)
# Tmax:           Maximum search time (in same units as response)
#                      (response will be censored at Tmax)
```

We execute the function once with default arguments to generate one data set.

```
set.seed(1)
data <- simOccttd()
str(data)
   Number of occupied sites ( among 250 ): 102
   Number of sites at which detected: 81
   Number of times censored: 169
```

We plot the response and then fit the data-generating exponential model in BUGS.

```
# Plot response (not shown)
hist(data$ttd, breaks = 50, col = "grey", main = "Observed distribution of time
to detection", xlim = c(0, data$Tmax), xlab = "Measured time to detection")
abline(v = data$Tmax, col = "grey", lwd = 3)

# Bundle data
str( win.data <- list(ttd = data$ttd, d = data$d, covA = data$covA,
   covB = data$covB, nobs = data$M, Tmax = data$Tmax) )

# Define occupancy model with exponential observation process
cat(file = "model1.txt", "
model {

# Priors
int.psi ~ dunif(0, 1)                 # Intercept occupancy on prob. scale
beta1 ~ dnorm(0, 0.001)               # Slope coefficient in logit(occupancy)
int.lambda ~ dgamma(0.0001, 0.0001) # Poisson rate parameter
alpha1 ~ dnorm(0, 0.001)              # Slope coefficient in log(rate)

# Likelihood
for (i in 1:nobs){
# Model for occurrence
   z[i] ~ dbern(psi[i])
   logit(psi[i]) <- logit(int.psi) + beta1 * covB[i]

   # Observation model
   # Exponential model for time to detection ignoring censoring
   ttd[i] ~ dexp(lambda[i])
   log(lambda[i]) <- log(int.lambda) + alpha1 * covA[i]
```

```
   # Model for censoring due to species absence and ttd>=Tmax
   d[i] ~ dbern(theta[i])
   theta[i] <- z[i] * step(ttd[i] - Tmax) + (1 - z[i])
}
# Derived quantities
n.occ <- sum(z[])     # Number of occupied sites among M
}
")

# Inits function for some params
# Initialize with z = 1 throughout and
#   all NA's due to censoring, rather than non-occurrence
zst <- rep(1, length(win.data$ttd))
ttdst <-rep(win.data$Tmax+1, data$M)
ttdst[win.data$d == 0] <- NA
inits <- function(){list(z =zst, ttd = ttdst, int.psi = runif(1), int.lambda = runif(1))}

# Parameters to estimate
params <- c("int.psi", "beta1", "int.lambda", "alpha1", "n.occ")

# MCMC settings
ni <- 12000  ;  nt <- 2  ;  nb <- 2000  ;  nc <- 3

# Call WinBUGS from R (ART 1.3 min) and summarize posteriors
out1 <- bugs(win.data, inits, params, "model1.txt", n.chains=nc, n.iter=ni, n.burn = nb,
n.thin=nt, debug = TRUE, bugs.directory = bd)
print(out1, dig = 3)
               mean    sd    2.5%     25%      50%      75%   97.5%  Rhat n.eff
int.psi       0.369 0.046   0.284   0.337    0.367    0.398   0.463 1.002  2200
beta1         1.408 0.243   0.963   1.240    1.396    1.564   1.910 1.001 15000
int.lambda    0.249 0.040   0.177   0.221    0.248    0.275   0.333 1.002  1600
alpha1       -1.248 0.157  -1.555  -1.355   -1.247   -1.141  -0.943 1.002  1900
n.occ       102.300 5.679  92.000  98.000  102.000  106.000 114.000 1.002  1700
```

The estimates seem to agree with the truth in the data simulation. You could run a simulation to check whether any discrepancy between the truth in the data simulation and the estimates is simply sampling variance or represents bias. As often with occupancy models, the number of occupied sites is estimated with staggering accuracy.

10.12.2 TIME-TO-DETECTION ANALYSIS WITH REAL DATA: WEIBULL OCCUPANCY MODEL FOR THE PEREGRINE SPRING SURVEY

To refine our understanding of a TTD occupancy model, we now analyze a small data set produced by a peregrine survey in the French Jura mountains during March 7−9, 2015, where *L'Equipe de choc helveto-britannique* (D. Parish, M. Kéry) visited 38 breeding cliffs for a total of 45 times and saw 57 peregrines (30 females, 27 males). Observation duration ranged from 3−95 mins and time to first detection from 0.1−48 mins. We saw birds in a total of 28 territories; hence, the observed proportion occupied was 0.74. We recorded time to detection separately for every bird present in a territory

(typically, a male and a female, but sometimes including a visiting immature bird as well). The aims of our analyses are:

1. to estimate the proportion of occupied territories among the 38 visited sites,
2. to study effects on detection probability of time of day, sex, and duration of observation, and
3. to distinguish between a constant or a time-varying hazard (i.e., choose between an exponential or a Weibull distribution).

The Weibull distribution is a generalization of the exponential. In the exponential, the hazard rate (i.e., the instantaneous event rate) is constant over time, whereas the Weibull allows the hazard to vary over time, leading to an "accelerated failure time" survival model if the hazard increases over time. Conversely, in our case it could be that we lost faith with increasing time of not seeing a bird. The Weibull allows us to test and therefore adjust for this by way of an additional parameter, the shape, which governs the change of the hazard. We have an exponential with constant hazard when shape $= 1$, while $0 <$ shape < 1 means that the hazard declines over time and so does detection probability, whereas with shape > 1 the hazard and detection probability increase over time.

```
# Read in data
data <- read.table("ttdPeregrine.txt", header = T, sep = "\t")
str(data)  # Will be part of the AHM package later

# Manage data and standardize time of day
nobs <- length(data$SiteNumber)    # Number of observations
d <- as.numeric(is.na(data$ttd))   # Censoring indicator
mean.tod <- mean(data$MinOfDay)
sd.tod <- sd(data$MinOfDay)
tod <- (data$MinOfDay -mean.tod) / sd.tod

# Bundle and summarize data set
str( win.data <- list(M = max(data$SiteNumber), site = data$SiteNumber,
   tod = tod, male = as.numeric(data$sex)-1, ttd = data$ttd, d = d, nobs = nobs,
   Tmax = data$Tmax) )
```

Compared with Section 10.12.1, this analysis contains the following new elements: it assumes a Weibull instead of an exponential response, survey duration is not constant but a vector, there are multiple observations per site, and we have an observational-level covariate, the sex of each bird. We parameterize `sex` as an indicator for males. Its value is not known when no bird is observed at a site, therefore, we have to specify a model for `sex` to estimate it for the missed birds. We do this by simply specifying a Bernoulli model with success probability being the sex ratio (specified as the proportion of males). This is a case of a missing individual covariate that is exactly analogous to the case of a size covariate affecting detection probability in a closed model to estimate population size (see Section 6.4 in Kéry and Schaub 2012 and also Chapter 9 in this book, where we model group size in distance sampling).

```
# Define model
cat(file = "model2.txt", "
model {

# Priors
psi ~ dunif(0, 1)                       # Occupancy intercept
lambda.int[1] ~ dgamma(0.001, 0.001)    # Poisson rate parameter for females
lambda.int[2] ~ dgamma(0.001, 0.001)    # Poisson rate parameter for males
```

```
alpha1 ~ dnorm(0, 0.001)          # Coefficient of time of day (linear)
alpha2 ~ dnorm(0, 0.001)          # Coefficient of time of day (squared)
shape ~ dgamma(0.001,0.001)       # Weibull shape
sexratio ~ dunif(0,1)             # Sex ratio (proportion males)

# Likelihood
for (i in 1:M){                   # Model for occurrence at site level
   z[i] ~ dbern(psi)
}

for (i in 1:nobs){                # Observation model at observation level
  # Weibull model for time to detection ignoring censoring
  ttd[i] ~ dweib(shape, lambda[i])
 log(lambda[i]) <- (1-male[i])*log(lambda.int[1]) + male[i]*log(lambda.int[2])+ alpha1
* tod[i] + alpha2 * pow(tod[i],2)
  # Model for censoring due to species absence and ttd>=Tmax
  d[i] ~ dbern(theta[i])
  theta[i] <- z[site[i]] * step(ttd[i] - Tmax[i]) + (1 - z[site[i]])
  # Model for sex of unobserved individuals
  male[i] ~ dbern(sexratio)    # Will impute sex for unobserved individuals
}
# Derived quantities
n.occ <- sum(z[])    # Number of occupied sites among M
}
")

# Inits function
zst <- rep(1, win.data$M)
ttdst <-rep(win.data$Tmax+1)
ttdst[win.data$d == 0] <- NA
inits <- function(){list(z =zst, ttd = ttdst, psi = runif(1), lambda.int = runif(2),
alpha1 = rnorm(1), alpha2 = rnorm(1), shape = runif(1))}

# Parameters to estimate
params <- c("psi", "lambda.int", "alpha1", "alpha2", "n.occ", "z", "sexratio", "shape")

# MCMC settings
ni <- 15000  ;  nt <- 2  ;  nb <- 2000  ;  nc <- 3

# Call JAGS from R (ART 0.6 min) and summarize posteriors
out2 <- bugs(win.data, inits, params, "model2.txt", n.chains=nc, n.iter=ni, n.burn = nb,
n.thin=nt, debug = T, bugs.directory = bd)
print(out2, dig = 3)
                 mean    sd   2.5%    25%    50%    75%  97.5%  Rhat n.eff
psi             0.923 0.062  0.771  0.889  0.938  0.972  0.998 1.002  2200
lambda.int[1]   0.129 0.045  0.061  0.098  0.123  0.154  0.236 1.002  2000
lambda.int[2]   0.109 0.042  0.048  0.079  0.102  0.132  0.208 1.001  5000
alpha1         -0.115 0.141 -0.392 -0.211 -0.115 -0.020  0.162 1.002  2800
alpha2          0.382 0.155  0.076  0.278  0.382  0.487  0.683 1.001  6900
n.occ          35.929 1.871 32.000 35.000 36.000 37.000 38.000 1.002  1300
z[1]            1.000 0.000  1.000  1.000  1.000  1.000  1.000 1.000     1
z[2]            1.000 0.000  1.000  1.000  1.000  1.000  1.000 1.000     1
z[3]            1.000 0.000  1.000  1.000  1.000  1.000  1.000 1.000     1
```

```
[ output truncated ]
z[36]          1.000 0.000  1.000  1.000   1.000   1.000  1.000 1.000     1
z[37]          0.632 0.482  0.000  0.000   1.000   1.000  1.000 1.002  1500
z[38]          1.000  0.000 1.000  1.000   1.000   1.000  1.000 1.000     1
sexratio       0.483 0.066  0.356  0.438   0.482   0.528  0.612 1.001 15000
shape          0.734 0.079  0.585  0.681   0.731   0.786  0.895 1.003  1100
```

We see that the sexes don't differ in their Weibull base rate, since the lambda intercepts for females (`lambda.int[1]`) and for males (`lambda.int[2]`) are very similar. The effect of time of day is not immediately clear, so we will form predictions below. We estimate that about 36 territories were occupied and, as always with a Bayesian fit of an occupancy model, we get the estimated presence/absence state of every site for free (these are the z's). Based on the observed birds, the sex ratio is estimated to be roughly even. Finally, interestingly, the Weibull shape parameter is estimated at a value clearly less than 1, indicating that the hazard for peregrine detection *declines* with increasing observation time. Perhaps we became increasingly pessimistic about the possible presence of a peregrine with increasing time without seeing it and then actually became more likely to miss a peregrine when it did appear? An alternative explanation, that the birds are disturbed and behave less conspicuous in our presence, does not apply here, because the distances between observer and the birds are simply too great.

We summarize the results from the analysis with Figure 10.16. To visualize the effects of time of day and duration on the detection probability, we average over the (very small) sex effect, and conduct

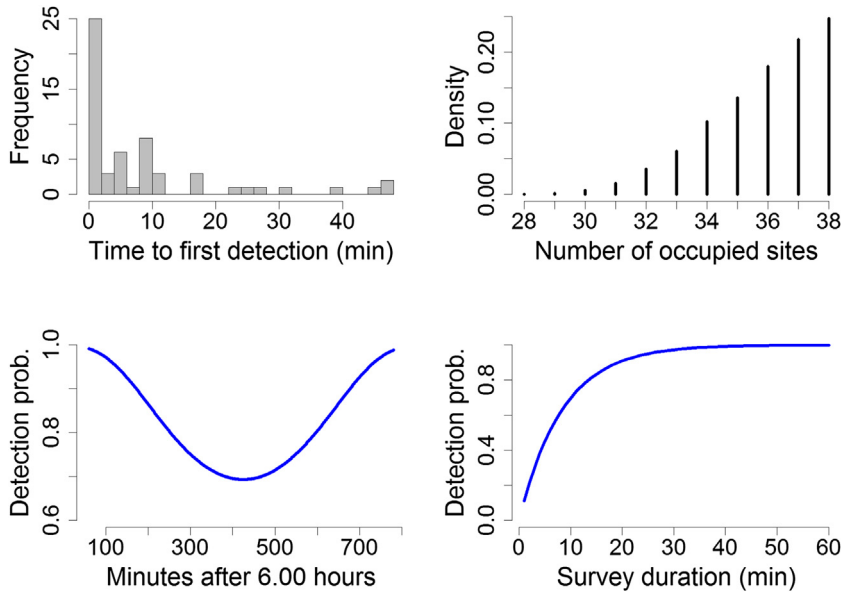

FIGURE 10.16

Results from fitting a Weibull time-to-detection occupancy model to part of the data from the peregrine spring survey 2015. Top left: the observed response: time to detection of the first bird for an individual visit; top right: posterior distribution of the number of occupied sites among the 38 sites visited; bottom: relationships between detection probability and time of day (left) and survey duration (right). Time of day is expressed in minutes after 6.00 h, so the x-axis extends from 7.00 to 19.00 hours.

the necessary calculations first. Time of day varies from 86−757 mins after 6.00 h, so we will predict for 60−780 mins and the x-axis extends from 7.00 to 19.00 hours.

```
# Predict detection over time of day: prediction cov. runs from 7h to 19h
minutes <- 60:780
pred.tod <- (minutes - mean.tod) / sd.tod  # Standardize as real data

# Predict p over time of day, averaging over sex, and for duration of 10 min
sex.mean <- apply(out2$sims.list$lambda.int, 1, mean)
p.pred1 <- 1 - exp(-exp(log(mean(sex.mean)) + out2$mean$alpha1 * pred.tod +
out2$mean$alpha2 * pred.tod^2) * 10)

# Predict p for durations of 1-60 min, averaging over time of day and sex
duration <- 1:60
p.pred2 <- 1 - exp(-exp(log(mean(sex.mean))) * duration)

# Visualize analysis
par(mfrow = c(2,2), mar = c(5,5,3,2), cex.lab = 1.5, cex.axis = 1.5)
hist(data$ttd, breaks = 40, col = "grey", xlab = "Time to first detection
(min)", main = "")
plot(table(out2$sims.list$n.occ)/length(out2$sims.list$n.occ), xlab = "Number of
occupied sites", ylab = "Density", frame = F)
plot(minutes, p.pred1, xlab = "Minutes after 6.00 hours (i.e., 7.00 - 19.00h)",
ylab = "Detection prob.", ylim = c(0.6, 1), type = "l", col = "blue", lwd = 3,
frame = F)
plot(duration, p.pred2, xlab = "Survey duration (min)", ylab = "Detection
prob.", ylim = c(0, 1), type = "l", col = "blue", lwd = 3, frame = F)
```

10.12.3 OCCUPANCY MODELS WITH REMOVAL DESIGN OBSERVATION PROCESS

A removal design is the discrete-time counterpart to what the time-to-detection design is in continuous time—we only model the events until the first detection, and a site needs only be surveyed until the species is first detected (or the data from later occasions is discarded). Although not always the most efficient occupancy design (MacKenzie and Royle, 2005; Bailey et al., 2007; Guillera-Arroita and Lahoz-Monfort, in press), the removal design tends to be popular among fieldworkers, who may not be motivated to revisit a site where a species' presence has already been ascertained. In individual capture-recapture (see Chapter 7), a removal design is coded using a categorical distribution for the type of observed detection history, and we can do the analogous thing also for an occupancy model with removal design. However, it is much easier if we simply turn all observations (if there are any) after the first detection at a site into NAs and fit the standard Bernoulli-Bernoulli occupancy model. This yields identical estimates to those obtained with the categorical parameterization and enables you to fit a removal design not only in BUGS but also in software such as `unmarked`, PRESENCE, MARK, or E-SURGE.

10.13 OCCUPANCY MODELING OF A COMMUNITY OF SPECIES

The occupancy framework can also be used to model a community, where species take the place of sites and the total number of "sites" is the list of species that can reasonably be assumed to occur in some area (MacKenzie et al., 2006). Such modeling makes sense especially in well-studied areas where the identity of all likely species in the regional pool is fairly well known, e.g., parts of

Europe and North America. The occupancy parameter might be interpreted as "relative species richness" or community integrity (Karr, 1990; Cam et al., 2002b,c), corrected for imperfect detection, i.e., species richness relative to some baseline list of species that may be thought to represent a regional pool of species present (Kéry, 2011a). Detection probability among animal or plant species in a community differs tremendously, hence, in this use of occupancy modeling for a community, adoption of a "heterogeneity model" (Royle, 2006) is imperative, that is we need a model that lets individual species differ in their detection probability. Otherwise, severe underestimation of the number of species, or the occupancy parameter, would result by unmodeled detection heterogeneity among species (this is again the second law of capture-recapture). Occupancy models with site heterogeneity in detection probability (beyond what can be explained by measured covariates) can only be fitted with PRESENCE and MARK and of course with BUGS. Typical specifications for such species-specific heterogeneity that cannot be explained by covariates include the logit-normal (see Section 6.11.2), beta-binomial, and finite mixture distributions (Dorazio and Royle, 2003; Royle, 2006). All of them can easily be specified in BUGS.

10.14 MODELING WIGGLY COVARIATE RELATIONSHIPS: PENALIZED SPLINES IN HIERARCHICAL MODELS

We end this chapter with a topic that is not specific to occupancy models and illustrate the modeling of highly irregular, "wiggly" covariate relationships using penalized splines (Ruppert et al., 2003; Crainiceanu et al., 2005; Gimenez et al. 2006a,b). Splines can be very useful in an exploratory analysis when nothing is known about the functional form of a covariate relationship; they may help suggesting a new parametric model. Penalized splines can be implemented as a form of GLMM, where the main features of the covariate relationship are described by fixed effects (e.g., linear and quadratic terms) and the wiggly part is accommodated by the random effects. We use function `wigglyOcc` to simulate static occupancy data with a really wild covariate relationship in both the occupancy and the detection part of the model (sample size is 240 sites and 3 replicate surveys). Executing the function draws two figures (Figure 10.17) and produces the simulated data and some summary output.

```
# Execute the function and inspect file produced
data <- wigglyOcc(seed = 1)
   True number of occupied sites: 141
   Observed number of occupied sites: 97
   Proportional underestimation of distribution: 0.31
str(data)
```

We use BUGS code due to Crainiceanu et al. (2005) to fit a model with penalized splines with truncated polynomial basis for each covariate in both parts of the model (i.e., for both occupancy and for detection). We also compare with a simpler model with two quadratic polynomials of the covariates. To spline-smooth a covariate in the detection model, we have to vectorize the analysis (see Chapter 21 in Kéry, 2010 and Chapter 16 in volume 2).

```
# Convert matrix data into vectors and prepare stuff
y <- c(data$y)                    # Detection/nondetection data (response)
Xsite <- data$Xsite               # Fine as is
Xsurvey <- c(data$Xsurvey)        # Survey covariate
site <- rep(1:data$M, data$J)     # Site index
```

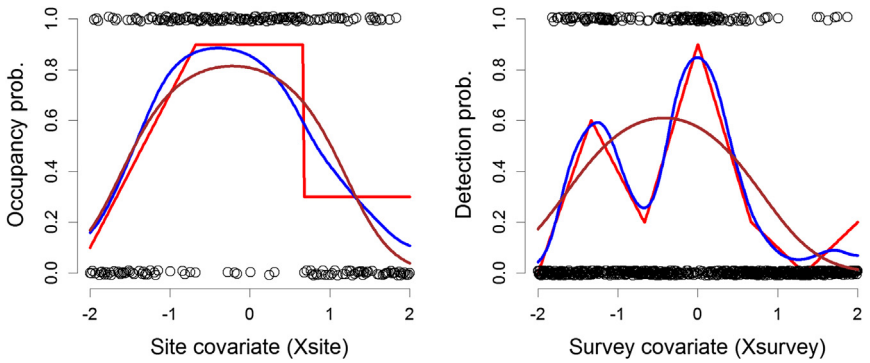

FIGURE 10.17

The crazy truth in our data set simulated using function `wiggly0cc`: red lines and circles show true occupancy probability and presence/absence z (left) and detection probability and the observed data y (right). Predictions under the spline occupancy model are blue and those under a simple model with linear and quadratic polynomial terms are brown. For both submodels in the spline model, 35 knots were evenly dispersed within the range of the covariates.

We then use our function `spline.prep` to create the two sets of fixed- and random-effects design matrices, based on code by Crainiceanu et al. (2005) and Zuur et al. (2012). The function call requires the name of a covariate and either the desired number of knots or else "NA," in which the latter is chosen using the rule of Ruppert (2002).

```
# tmp1 <-spline.prep(Xsite, 20)  # This would choose 20 knots for Xsite
tmp1 <-spline.prep(Xsite, NA)    # Choose variable default number of knots
tmp2 <-spline.prep(Xsurvey, NA)
Xocc <- tmp1$X          # Fixed-effects part of covariate Xsite in occ
Zocc <- tmp1$Z          # Random-effects part of covariate Xsite in occ
Xdet <- tmp2$X          # Fixed-effects part of covariate Xsite in det
Zdet <- tmp2$Z          # Random-effects part of covariate Xsite in det
nk.occ <- length(tmp1$knots)     # Number of knots in occupancy spline
nk.det <- length(tmp2$knots)     # Number of knots in detection spline
```

To compare estimates under the spline model with a simpler, parametric model with quadratic effects of both covariates, we fit both models inside of the same "hypermodel," by supplying a duplicate of the response data y.

```
# Bundle and summarize data set
win.data <- list(y1 = y, site = site, M = data$M, Xocc = Xocc, Zocc = Zocc, nk.occ = nk.occ,
Xdet = Xdet, Zdet = Zdet, nk.det = nk.det, nobs = data$M*data$J, y2 = y, onesSite = rep(1,
240), onesSurvey = rep(1, 720), Xsite = Xsite, Xsite2 = Xsite^2,  Xsurvey = Xsurvey,
Xsurvey2 = Xsurvey^2)
str(win.data)  # onesSite and onesSurvey are for occ and det intercepts
```

```
# Specify two models in one in BUGS language
cat(file = "hypermodel.txt",
"model {

# *** Spline model for the data***
# --------------------------------
# Priors
for(k in 1:3){                  # Regression coefficients
   alpha1[k] ~ dnorm(0, 0.1)    # Detection model
   beta1[k] ~ dnorm(0, 0.1)     # Occupancy model
}
for(k in 1:nk.occ){ # Random effects at specified knots (occupancy)
   b.occ[k] ~ dnorm(0, tau.b.occ)
}
for(k in 1:nk.det){ # Random effects at specified knots (detection)
   b.det[k] ~ dnorm(0, tau.b.det)
}
tau.b.occ ~ dgamma(0.01, 0.01)
tau.b.det ~ dgamma(0.01, 0.01)

# Likelihood
# Model for latent occupancy state
for (i in 1:M) {
   z1[i] ~ dbern(psi1[i])
   logit(psi1[i]) <- fix.terms.occ[i] + smooth.terms.occ[i]
   fix.terms.occ[i] <- beta1[1]*Xocc[i,1] + beta1[2]*Xocc[i,2] + beta1[3]*Xocc[i,2]
   smooth.terms.occ[i] <- inprod(b.occ[], Zocc[i,])
}
# Model for observations
for(i in 1:nobs){
   y1[i] ~ dbern(mu.y1[i])
   mu.y1[i] <- z1[site[i]] * p1[i]
   logit(p1[i]) <- fix.terms.det[i] + smooth.terms.det[i]
   fix.terms.det[i] <- alpha1[1]*Xdet[i,1] + alpha1[2]*Xdet[i,2] +
      alpha1[3]*Xdet[i,2]
   smooth.terms.det[i] <- inprod(b.det[], Zdet[i,])
}

# Derived quantities
sum.z1 <- sum(z1[])             # Number of occupied sites in sample
sd.b.occ <- sqrt(1/tau.b.occ) # SD of spline random effects variance Occ.
sd.b.det <- sqrt(1/tau.b.det) # SD of spline random effects variance Det.

# *** Polynomial model for same data ***
# --------------------------------------
# Priors
for(k in 1:3){                  # Regression coefficients
  alpha2[k] ~ dnorm(0, 0.1)     # Detection model
  beta2[k] ~ dnorm(0, 0.1)      # Occupancy model
}
```

```
# Likelihood
# Model for latent occupancy state
for (i in 1:M) {
    z2[i] ~ dbern(psi2[i])
    logit(psi2[i]) <- beta2[1]*onesSite[i] + beta2[2]*Xsite[i] + beta2[3]*Xsite2[i]
}
# Model for observations
for(i in 1:nobs){
   y2[i] ~ dbern(mu.y2[i])
   mu.y2[i] <- z2[site[i]] * p2[i]
    logit(p2[i]) <- alpha2[1]*onesSurvey[i] + alpha2[2]*Xsurvey[i] +
       alpha2[3] * Xsurvey2[i]
}

# Derived quantities
sum.z2 <- sum(z2[])             # Number of occupied sites in sample
}
")

# Initial values
zst <- apply(data$y, 1, max)
inits <- function(){list(z1=zst, alpha1=rnorm(3), beta1=rnorm(3), b.occ =
rnorm(nk.occ), b.det = rnorm(nk.det), tau.b.occ = runif(1), tau.b.det =
runif(1), z2=zst, alpha2=rnorm(3), beta2=rnorm(3))}

# Parameters monitored
params <- c("alpha1", "beta1", "psi1", "p1", "fix.terms.occ", "smooth.terms.occ",
"b.occ", "fix.terms.det", "smooth.terms.det", "b.det", "sum.z1", "sd.b.occ", "sd.b.det",
"alpha2", "beta2", "psi2", "p2", "sum.z2")

# MCMC settings
ni <- 100000  ;  nb <- 10000  ;  nt <- 90  ;  nc <- 3
```

We run JAGS in parallel, since this model takes a while.

```
# Call JAGS from R (ART 95 min) and summarize posteriors
system.time(fhm <- jags(win.data, inits, params, "hypermodel.txt", n.chains = nc, n.thin =
nt, n.iter = ni, n.burnin = nb, parallel = TRUE))
traceplot(fhm)  ;  print(fhm, 3)
print(fhm$summary[c(1:6, 2957:2965, 3926),], 3)  # Compare some key estimands

# Plot prediction of psi and p (Fig. 10-17)
par(mfrow = c(1,2), mar = c(5,4,3,2), cex.lab = 1.5, cex.axis = 1.5)
plot(Xsite, data$psi, main = "Occupancy probability",type = "l", ylim = c(-0.1, 1.1),
col = "red", xlab = "Site covariate (Xsite)", ylab = "", lwd = 3)
points(Xsite, jitter(data$z, amount = 0.02))
lines(Xsite, fhm$mean$psi1, col = "blue", lty = 1, lwd = 3)
lines(Xsite, fhm$mean$psi2, col = "brown", lty = 1, lwd = 3)

plot(Xsurvey[order(data$x.index)], data$p.ordered, main = "Detection probability
",type = "l", ylim = c(-0.1, 1.1), col = "red", xlab = "Survey covariate (Xsurvey)",
ylab = "", lwd = 3)
```

```
points(Xsurvey, jitter(y, amount = 0.02))
lines(Xsurvey[order(data$x.index)], fhm$mean$p1[order(data$x.index)], col = "blue",
lwd = 3)
lines(Xsurvey[order(data$x.index)], fhm$mean$p2[order(data$x.index)], col = "brown",
lwd = 3)
```

Thus, even with really wiggly covariate relationships in both model parts, we have a quite impressive ability to estimate these patterns (Figure 10.17). The inference was better about the detection model than for occupancy, where the inferred pattern with splines was not much different from that by a simple quadratic polynomial. This is generally the case for such HMs—features of the latent process are harder to estimate than those of the observed process. In spite of the better covariate fit of the spline model, there was only a slight improvement in the estimate of a key quantity, the number of occupied sites in the sample. The species occurred at 141 sites and was observed at 97. The simpler model with quadratic covariate effects produced an estimate of 130 occupied sites (95% CRI 115−148) and the spline model produced an estimate of 135 (95% CRI 120−152).

In general, when your interest focuses on the precise functional form of a covariate relationship (Strebel et al., 2014), you can add splines inside of your hierarchical model. While this offers great flexibility and power, it should probably also be applied with care since we can imagine that it may be easy to run into identifiability problems. Finally, as we have seen, the substantive conclusions may not always be much different from those under a much simpler, parametric model.

10.15 SUMMARY AND OUTLOOK

We have given an overview of a very powerful and flexible class of hierarchical models: occupancy models (MacKenzie et al., 2002, 2006; Tyre et al., 2003). The hierarchical definition of the model makes extremely transparent what these models are—they have one logistic regression for the incompletely observed presence/absence state, which is governed by probability of occupancy or presence, and then another, conditional model for the measurement of presence/absence, that accommodates false-negative observations. This measurement error model will typically be a binomial or Bernoulli distribution, but depending on the precise data collection protocol may be Poisson, exponential, or Weibull, or multinomial, or even another distribution, and we have given examples for most of those. As always, the occupancy model may be fit using likelihood or Bayesian methods. The occupancy model is applicable to any binary response with a binary measurement error (in the case of classical Bernoulli observation model). Therefore, depending on how you choose to define the event that is classified as a "presence," the same model may be applied to a vast number of different situations. In particular, it is the canonical species distribution model with explicit accommodation of the ubiquitous false-negative detection error (Kéry, 2011b; Lahoz-Monfort et al., 2014; Guillera-Arroita et al., 2015).

Nevertheless, we would like to remind you of a couple of caveats. First, there has been some debate recently about the usefulness of this model in practice (Welsh et al., 2013; also see McKann et al., 2013, and Hayes and Monfils, 2015). Most doubts expressed in the Welsh et al. paper were convincingly responded to by Guillera-Arroita et al. (2014a), and for McKann et al., see our comments in Section 10.7. (Hayes and Monfils, 2015, worry about the effects of temporary emigration (TE) on the meaning of the occupancy parameter. This interpretation can range from 'probability of permannent presence of the species' when there is no TE to 'probability of use sometimes during the study period' when there

is TE; see Section 10.2. Most people don't see this as a big problem. Moreover, we would claim that inferences under ANY model for occurrence may be affected by TE. So their main point really seems to be not specific to occupancy models at all.) But don't forget that to estimate parameters, you first need *data*. So, with extreme paucity of data, you cannot expect to obtain very good parameter estimates for any model (hierarchical or not) and with any inference method (e.g., Bayesian or not). Many such doubts could be avoided by conducting customized simulations, using a template like the one given in Section 10.7. With R, it is really easy to ascertain how good estimates under a model are likely to be for exactly *your* sample sizes, data collection protocol and model, and for hypothesized values of the parameters in your study. The second caveat is that the meaning of your occupancy parameters depends on three things: the definition of what constitutes a "presence," and the spatial and temporal scale of your sampling. Part of your job as an ecologist is to make a good choice here to ensure that you obtain parameters that are meaningful in the context of the biology of your species. And the third caveat is the relationship of occupancy with abundance—you cannot "escape" abundance (McCarthy et al., 2013), for instance, spatial variation in abundance will create spatial heterogeneity in detection probability when you study presence/absence, and this may bias your occupancy estimators unless modeled. And somewhat related to the last point, never forget that presence/absence is "the poor man's abundance." If you have counts you should always try to model those directly and only throw out information by modeling presence/absence instead if there are very good reasons for doing so (for instance, if you can't find a fitting abundance model).

A vast number of extensions are possible for the basic occupancy model. Many correspond to the relaxation of a specific assumption in the basic model (see Section 10.2). Lack of closure can be addressed by "multiseason" models (MacKenzie et al., 2003), especially dynamic occupancy models (see Chapter 16), and a specific lack of closure in the basic model has been addressed by Kendall et al. (2013). False-positives can be accommodated in addition to false-negatives; see Royle and Link (2006), Miller et al. (2011, 2013b), Chambert et al. (2015), and Chapter 19 (in volume 2). The assumption that sites are independent can be relaxed by modeling spatial dependence (autocorrelation); see Chapters 21 and 22. Finally, the assumption of homogeneous detection among sites can be relaxed by adopting a mixture model for detection probability (Royle, 2006), such as the "Royle-Nichols model" (Royle and Nichols, 2003; see Section 6.13) or, if replicated counts are available, an *N*-mixture model (see Chapter 6 in Dorazio, 2007). The Royle-Nichols model can be fit using `unmarked` with function `occuRN`. Other ways to address site-specific detection heterogeneity, including finite mixtures or the logit-normal model, are easy to implement in BUGS; for the latter, see the code for the random-effects models in Section 6.11.2.

Other extensions include multiple scales of occupancy, as we have shown in Section 10.10, and extensions to more than two scales (corresponding to four or more levels in the hierarchical model) are straightforward (McClintock et al., 2010b). We have also discussed the space-for-time substitution and suggested that it may be understood as a restricted multiscale occupancy model that lacks replication at the bottom level. Multiscale occupancy models are important because many occupancy sampling designs are in fact nested, leading to such models, but we think that this is not yet sufficiently widely recognized. We have also emphasized that the space-for-time substitution is perhaps one of the least understood topics in occupancy modeling that really warrants more research to help decide when it can and when it cannot be applied to model occupancy and detection probability with spatially instead of temporally replicated measurements.

We have also shown a variety of data collection protocols, which lead to different observation models (Section 10.12), as well as the move from a basic GLM-type occupancy model to a GAM-type occupancy model (Section 10.13). Fitting GAM-type models in BUGS is something that you can do with any flat or hierarchical model. Two-dimensional splines are one computationally efficient way of addressing spatial autocorrelation in N-mixture or occupancy models (Collier et al., 2012; Guélat and Kéry, in review). But there is more. Instead of two states (here, presence and absence), you may have multiple states, for instance absence, and presence with and without reproduction, leading to multistate occupancy models (Chapter 18), or instead of a single species, you may want to model multiple species jointly, with or without interactions (see Chapters 11, 17, and 20). Finally, Roth and Amrhein (2009) developed an occupancy model for territorial birds that yields estimates of what they call "local survival rates," somewhat bridging the gap between a site-level model for occurrence and an individual-level model.

A recurrent theme in ecological statistics is that of the integration of disparate information about the same state or process, so-called "integrated models" or IMs (Schaub and Abadi, 2011). Such models are very powerful and topical and, essentially, they are only available in practice to ecologists because of the accessible BUGS language. Integrated models typically link different types of data sets via a description of the most detailed, or "bottom," level; or we may also say that they simply link the modeling of different data sets that bear on the same process by *shared parameters* which simultaneously occur in two or more data sets. For the models in this chapter, in their most trivial form, an IM would fuse multiple data sets for the ecological state of occurrence that differ only in their observation process. For instance, it is very easy to combine in a single model occupancy data with observation models for capture-recapture (that is the standard model) and removal and time-to-detection designs. More complex IMs link data sets where the fundamental state about which the data are informative differs. For instance, Chandler and Clark (2014) integrate occupancy and telemetry data via a description of an underlying latent spatial point process for the individuals, and the occupancy data are simply an aggregation (or different observation process) of that "bottom" system description. The fusion of count and presence/absence data is straightforward conceptually (for instance, for a Royle-Nichols model for the latter) and should be done much more often. In an important paper, Conroy et al. (2008) show how occupancy and individual capture-recapture data can be integrated, again via a Royle-Nichols variant for the former. Dorazio (2014) demonstrated the fusion of spatial point pattern data and replicated counts, and a similar thing conceptually should be possible with detection/non-detection data. Such combinations lead to more precise estimates and in some cases to the compensation of the weaknesses of one data type by the information in the other data type. In Chapter 23 we illustrate some variants of such integrated models. They represent an extremely powerful and as yet mostly untapped framework for modeling multisite data on distribution and abundance.

EXERCISES

1. Referring to Section 10.1 and our classification of data used to model species distribution, we mention how occupancy probability scales with the area of the sampling unit. Write a little simulation that allows you to observe this relationship (hint: you may distribute individuals in a plane according to a Poisson process, overlay with grids of different size, and observe how the proportion of cells occupied changes with grid size).

2. In Section 10.3, extend the simulation such that false-positives are allowed with a probability of 0.01. Check how the traditional occupancy estimator in this section is biased when we conduct 2, 4, 6…20 surveys.
3. Also in Section 10.3, we claimed that the Bernoulli model illustrated was equivalent to a binomial model for the detection frequencies when no time-specific patterns in detection probability are present or modeled. (a) Prove this to yourself by fitting the binomial model to an example where you increase the number of sites 100-fold and the number of surveys 10-fold. How much faster in BUGS is the binomial model compared to the Bernoulli model? (b) Turn some of the final surveys into NAs and fit the binomial model when the binomial sample size varies and J has to be supplied in the analysis as a vector.
4. In Section 10.5, play around with function `simOcc` for at least half an hour to train your intuition about occurrence data, imperfect detection, and occupancy models.
5. Also in Section 10.5, use function `simOcc` to devise a simulation so see how much the true and the observed number of occupied sites vary by chance when using the default function arguments.
6. In Section 10.7, adapt the code to study the effects on the occupancy estimator of the standard model of assumption violations due to individual (site) detection heterogeneity and behavioral response.
7. In Section 10.9, produce a histogram that shows the elevation gradient of the distribution of the red squirrel in Switzerland.
8. In Section 10.14, fit the spline model with no fixed effect of the covariates at all (other than an intercept), so see how well the random parts of the spline model can recover the functional form of the two covariates and whether the resulting model can still estimate the number of occupied sites decently.