CHAPTER

Fully Spatial Capture-Recapture Models

5

In the previous chapter, we discussed models that could be viewed as primitive spatial capture-recapture models. We looked at a basic distance sampling model, and we also considered classical individual covariate models, in which we defined a covariate to be the distance from the (estimated) home range center to the center of the trap array. The individual covariate model that we conjured up was "spatial" in the sense that it included some characterization of where individuals live but, on the other hand, only a primitive or no characterization of trap location. That said, there is only a small step from that model to spatial capture-recapture models, which we consider in this chapter, and which fully recognize the spatial attribution of both individual animals *and* the locations of encounter devices.

Capture-recapture models must accommodate the spatial organization of individuals and the encounter devices, because the encounter process occurs at the level of individual traps. Failure to consider the trap-specific data is one of the key deficiencies with classical ad hoc approaches that aggregate encounter information to the resolution of the entire trap array. We have previously addressed some problems that this causes, including induced heterogeneity in encounter probability, imprecise notation of "sample area", and not being able to accommodate trap-specific effects or trap-specific missing values. In this chapter, we resolve these issues by developing our first fully spatial capture-recapture model. This model is not too different from that considered in Section 4.5 but, instead of defining the individual covariate to be distance to the centroid of the array, we define J individual covariates—the distance to *each* trap. And, instead of using estimates of individual locations $\mathbf{s}$, we consider a fully hierarchical model in which we regard $\mathbf{s}$ as a latent variable and impose a prior distribution on it.

In this chapter, we investigate the basic spatial capture-recapture model, which we refer to as "model SCR0", and address some important considerations related to its analysis in **BUGS**. We demonstrate how to summarize posterior output for the purposes of producing density maps or spatial predictions of density. The key aspect of the SCR models considered in this chapter is the formulation of a model for encounter probability that is a function of distance between individual home range center and trap locations. We also discuss how encounter probability models are related to explicit models of space usage or "home range area". Understanding this allows us to compute, for example, the area used by an individual during some prescribed time. While it is intuitive that SCR models should be related to some model of space usage, this has not been discussed much in the literature.

Spatial Capture-Recapture. http://dx.doi.org/10.1016/B978-0-12-405939-9.00005-0

5.1 Sampling design and data structure

In our development here, we will assume a standard sampling design in which an array of J traps is operated for K sample occasions (say, nights), producing encounters of n individuals. Because sampling occurs by traps and over time, the most general data structure yields temporally *and* spatially indexed encounter histories for *each individual*. Thus, a typical data set will include an encounter history *matrix* for each individual, indicating which trap the individual was captured in, during each sample occasion. For example, suppose we sample at 4 traps over 3 nights. A plausible data set for a single individual captured one time in trap 1 on the first night and one time in trap 3 on the third night is:

```
       night1  night2  night3
trap1     1       0       0
trap2     0       0       0
trap3     0       0       1
trap4     0       0       0
```

This data structure would be obtained for *each* of the $i = 1, 2, \ldots, n$ captured individuals.

We develop models in this chapter for passive detection devices such as "hair snares" or other DNA sampling methods (Kéry et al., 2010; Gardner et al., 2010b) and related types of sampling devices in which (i) devices ("traps") may capture any number of individuals (i.e., they don't fill up); (ii) an individual may be captured in more than one trap during each occasion, but (iii) individuals can be encountered at most once by each trap during any occasion. Hair snares for sampling DNA from bears and other species function according to these rules. An individual bear wandering about its territory might come into contact with >1 devices; a device may encounter multiple bears; however, in practice, it will often not be possible to attribute multiple visits of the same individual to the same snare during a single occasion (e.g., night) to distinct encounter events. Thus, an individual may be captured at most 1 time in each trap during any occasion. While this model, which we refer to as SCR0, is most directly relevant to hair snares and other DNA sampling methods for which multiple detections of an individual are not distinguishable, we will also make use of the model for data that arise from camera trapping studies. In practice, with camera trapping, individuals might be photographed several times in a night, but it is common to distill such data into a single binary encounter event for reasons discussed later in Chapter 9.

The statistical assumptions we make to build a model for these data are that individual encounters within and among traps are independent, and this allows us to regard individual- and trap-specific encounters as *independent* Bernoulli trials (see next section).

5.2 The binomial observation model

We begin by considering the simple model in which there are no time-varying or trap-specific covariates that influence encounter, there are no explicit individual-specific

Table 5.1 Hypothetical spatial capture-recapture data set showing 6 individuals captured in 4 traps. Each entry is the number of captures out of $K = 3$ nights of sampling.

Individual	Trap 1	Trap 2	Trap 3	Trap 4
1	1	0	0	0
2	0	2	0	0
3	0	0	0	1
4	0	1	0	0
5	0	0	1	1
6	1	0	1	0

covariates, and there are no covariates that influence density. In this case, we can aggregate the binary encounters over the K sample occasions and record the total number of encounters out of K. We will denote these individual- and trap-specific encounter frequencies by y_{ij} for $i = 1, 2, \ldots, n$ captured individuals and $j = 1, 2, \ldots, J$ traps. For example, suppose we observe 6 individuals in sampling at 4 traps over 3 nights of sampling, then a plausible data set is the 6×4 matrix of encounters (out of 3 sampling occasions) shown in Table 5.1. We assume that y_{ij} are mutually independent outcomes of a binomial random variable, which we express as:

$$y_{ij} \sim \text{Binomial}(K, p_{ij}). \tag{5.2.1}$$

This is the basic model underlying standard closed population models (Chapter 4) except that, in the present case, the encounter frequencies are individual- *and* trap-specific, and encounter probability p_{ij} depends on both individual *and* trap.

As we did in Section 4.5, we will make explicit the notion that p_{ij} is defined conditional on *where* individual i lives. Naturally, we think about defining an individual home range and then relating p_{ij} explicitly to a summary of its location relative to each trap. In what follows, we define $\mathbf{s}_i$, a two-dimensional spatial coordinate, to be the home range or activity center of individual i (Efford, 2004; Borchers and Efford, 2008; Royle and Young, 2008). Then, the SCR model postulates that encounter probability, p_{ij}, is a decreasing function of distance between $\mathbf{s}_i$ and the location of trap j, $\mathbf{x}_j$ (also a two-dimensional spatial coordinate). A standard approach for modeling binomial counts is the logistic regression, where we express the dependence of p_{ij} on distance according to:

$$\text{logit}(p_{ij}) = \alpha_0 + \alpha_1 \|\mathbf{x}_j - \mathbf{s}_i\|, \tag{5.2.2}$$

where $\|\mathbf{x}_j - \mathbf{s}_i\|$ is the distance between $\mathbf{s}_i$ and $\mathbf{x}_j$. We sometimes write $\|\mathbf{x}_j - \mathbf{s}_i\| = \text{dist}(\mathbf{x}_j, \mathbf{s}_i) = d_{ij}$. Alternatively, a popular model is

$$p_{ij} = p_0 \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x}_j - \mathbf{s}_i\|^2\right), \tag{5.2.3}$$

which is similar to the "half-normal" model in distance sampling, except with an intercept $p_0 \leq 1$, which can be estimated in SCR studies. Because it is the kernel of

a bivariate normal, or Gaussian, probability density function for the random variable "individual location", we will refer to it as the "(bivariate) normal" or "Gaussian" model although the distance sampling term "half-normal" is widely used. In the context of two-dimensional space, the model is clearly interpretable as a primitive model of movement outcomes or space usage (we discuss this in Section 5.4).

There are a large number of standard detection models commonly used (see Chapter 7). All other standard models that relate encounter probability to $\mathbf{s}$ will also have a parameter that multiplies distance in some non-linear function. To be consistent with parameter naming across models, we will sometimes parameterize any encounter probability model so that the coefficient on distance (or distance squared) is α_1. So, for the Gaussian model, $\alpha_1 = 1/(2\sigma^2)$. A characteristic of the common parametric forms is that they are monotone decreasing with distance, but vary in their behavior as they approach distance = 0. We show the standard Gaussian, Gaussian hazard, negative exponential, and logistic models in Figure 5.1. The negative exponential model has $p_{ij} = p_0 \exp(-\alpha_1 \|\mathbf{x}_j - \mathbf{s}_i\|)$ and the Gaussian hazard model has $p_{ij} = 1 - \exp(-\lambda_0 k(\mathbf{x}_j, \mathbf{s}_i))$ where $k(\mathbf{x}_j, \mathbf{s}_i)$ is the Gaussian kernel and $\lambda_0 > 0$ is the baseline encounter rate. Whatever model we choose for encounter probability, we should always keep in mind that the activity center for individual i, $\mathbf{s}_i$, is an unobserved random variable. To be precise about this in the model, we should express the observation model as

$$y_{ij}|\mathbf{s}_i \sim \text{Binomial}(K, p(\mathbf{s}_i; \boldsymbol{\alpha}))$$

where $\boldsymbol{\alpha}$ represents all parameters of the model. Sometimes, for notational simplicity, we abbreviate this by omitting some or all of the arguments to p.

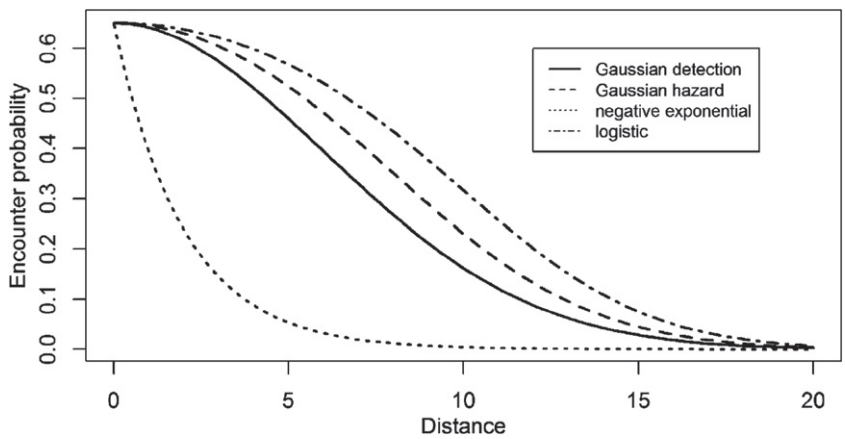

FIGURE 5.1

Some common encounter probability models showing the characteristic monotone decrease of encounter probability with distance between activity center and trap location.

5.2.1 Definition of home range center

We define an individual's home range as *the area used by an organism during some time period* which has a clear meaning for most species regardless of their biology. We therefore define the home range center (or activity center) to be the center of the space that individual occupied (or used) during the period in which traps were active. Thinking about it in that way, the activity center could even be observable (almost) as the centroid of a very large number of radio fixes over the course of a survey period or a season. Thus, this practical version of a home range center in terms of space usage is a well-defined construct regardless of whether one thinks that home range itself is a meaningful concept. We use the terms *home range center* and *activity center* interchangeably, and we recognize that this is a transient quantity that applies only to a well-defined period of study.

5.2.2 Distance as a latent variable

If we knew precisely every $\mathbf{s}_i$ in the population (and population size N), then the model specified by Eqs. (5.2.1) and (5.2.2) would be just an ordinary logistic regression type of a model (with covariate d_{ij}) which we learned how to fit using **WinBUGS** previously (Chapter 3). However, the activity centers are unobservable even in the best possible circumstances. In that case, d_{ij} is an unobserved variable, analogous to the situation in classical random effects models. Therefore we need to extend the model to accommodate these random variables with an additional model component—the random effects distribution. The customary assumption is the so-called "uniformity assumption," which states that the $\mathbf{s}_i$ are uniformly distributed over space (the obvious next question: "which space?" is addressed below). This uniformity assumption amounts to a uniform prior distribution on $\mathbf{s}_i$, i.e., the pdf of $\mathbf{s}_i$ is constant, which we may express

$$\Pr(\mathbf{s}_i) \propto \text{constant}. \tag{5.2.4}$$

As it turns out, this assumption is usually not precise enough to fit SCR models in practice for reasons we discuss shortly. We will give another way to represent this prior distribution that is more concrete, but depends on specifying the "state-space" of the random variable $\mathbf{s}_i$. The term state-space is a technical way of saying "the space of all possible outcomes" of the random variable.

5.3 The binomial point process model

In the SCR model, the individual activity centers are unobserved and thus we treat them as random effects. Specifically, the collection of individual activity centers $\mathbf{s}_1, \ldots, \mathbf{s}_N$ represents a realization of a *binomial point process* (Illian et al., 2008, p. 61). The binomial point process (BPP) is analogous to a Poisson point process in the sense that it represents a "random scatter" of points in space—except that the total number of points is *fixed*, whereas, in a Poisson point process, it is random (having a Poisson distribution). As an example, we show in Figure 5.2 locations of 20 individual

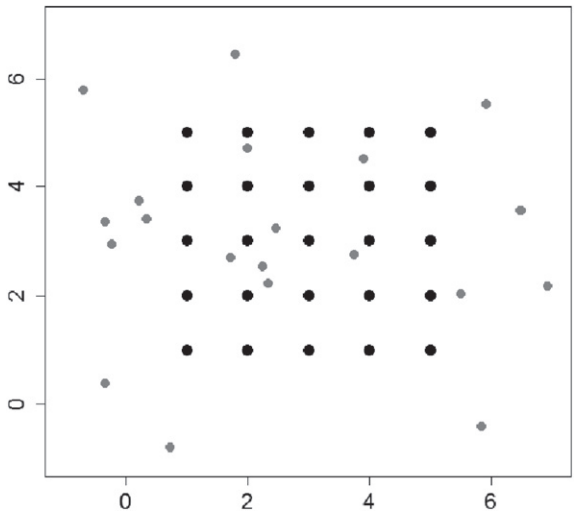

FIGURE 5.2

Realization of a binomial point process with $N = 20$ (small dots). The large dots represent trap locations.

activity centers (black dots) in relation to a grid of 25 traps. For a Poisson point process the number of such points in the prescribed state-space is random whereas we will often be interested in simulating a fixed numbers of points, e.g., for evaluating the performance of procedures, e.g., how well does our estimator perform when $N = 50$?

It is natural to consider a binomial point process in the context of capture-recapture models because it preserves N in the model and thus preserves the direct linkage directly with closed population models. In fact, under the binomial point process model, model M_0 and other closed models are simple limiting cases of SCR models, i.e., they arise as the coefficient on distance (α_1 above) tends to 0.

One consequence of having a fixed N in the BPP model is that the model is not strictly a model of "complete spatial randomness." This is because, if one forms counts $n(A_1), \ldots, n(A_k)$ in any set of disjoint regions of the state-space, say $A_1, \ldots, A_k$, then these counts are *not* independent. In fact, they have a multinomial distribution (see Illian et al., 2008, p. 61). Thus, the BPP model introduces a slight bit of dependence in the distribution of points. However, in most situations this will have no practical effect on any inference or analysis and, as a practical matter, we will usually regard the BPP model as one of spatial independence among individual activity centers, because each activity center is distributed independently of each other activity center. Despite this independence, we see in Figure 5.2 that *realizations* of randomly distributed points will typically exhibit distinct non-uniformity. Thus, independent, uniformly distributed points will almost never appear regularly, uniformly, or systematically distributed. For this reason, the basic binomial (or Poisson) point process models are enormously useful in practical settings, since they allow for a range of distribution

patterns without violating the assumption of spatial randomness. This is because, for SCR models, we actually have a little bit of data on a sample of individuals and thus the resulting posterior point pattern can deviate strongly from uniformity, a point we come back to repeatedly in this book. The uniformity hypothesis is only a *prior* distribution which is directly affected by the quantity and quality of the observed data and may appear distinctly non-uniform. In addition, we can build more flexible models for the point process a topic we take up in Chapter 11.

5.3.1 The state-space of the point process

Shortly we will focus on Bayesian analysis of model SCR0 with N known so that we can gain some basic experience with important elements of the model and its analysis. To do this, we note that the individual activity centers $\mathbf{s}_i, \ldots, \mathbf{s}_N$ are unknown quantities and we will need to be able to simulate each $\mathbf{s}_i$ in the population from the posterior distribution. In order to simulate the $\mathbf{s}_i$, it is necessary to describe precisely the region over which they are distributed. This is the quantity referred to above as the state-space, which is sometimes called the *observation window* in the point process literature. We denote the state-space henceforth (throughout this book) by $\mathcal{S}$ representing a region or a set of points comprising the potential values (the support) of the random variable $\mathbf{s}$. Thus, an equivalent explicit statement of the "uniformity assumption" is

$$\mathbf{s}_i \sim \text{Uniform}(\mathcal{S}),$$

where $\mathcal{S}$ is a precisely defined region, e.g., in Figure 5.2, $\mathcal{S}$ is the square defined by $[-1, 7] \times [-1, 7]$. Thus, each of the $N = 20$ points was generated by randomly selecting each coordinate on the line $[-1, 7]$. When points are distributed uniformly over some region, the point process is usually called a *homogeneous point process*.

5.3.1.1 *Prescribing the state-space*

Evidently, to define the model, we need to define the state-space, $\mathcal{S}$. How can we possibly do this objectively? Prescribing any particular $\mathcal{S}$ seems like the equivalent of specifying a "buffer" which we have criticized as being ad hoc. How is it, then, that the choice of a state-space is *not* ad hoc? As we observed in Section 4.5, it is true that N increases with $\mathcal{S}$, but only at the same rate as the area of $\mathcal{S}$ increases under the prior assumption of constant density. As a result, we say that density is invariant to $\mathcal{S}$ *as long as* $\mathcal{S}$ *is sufficiently large* to include all animals with non-negligible probability of encounter (see next sub-section). Thus, while choice of $\mathcal{S}$ is (or can be) essentially arbitrary, once $\mathcal{S}$ is chosen, it defines the population being exposed to sampling, which scales appropriately with the size of the state-space.

For our simulated system developed previously in this chapter, we defined the state-space to be a square within which our trap array was centered. For many practical situations this might be an acceptable approach to defining the state-space, i.e., just a rectangle around the trap array. Although defining the state-space to be a regular polygon has computational advantages (e.g., we can implement this more efficiently in **BUGS** and cannot for irregular polygons), a regular polygon induces an apparent

problem of admitting into the state-space regions that are distinctly non-habitat (e.g., oceans, large lakes, ice fields, etc.). It is difficult to describe complex regions in mathematical terms that can be used in **BUGS**. As an alternative, we can provide a representation of the state-space as a discrete set of points, which the **R** package `secr` (Efford, 2011a) permits (`secr` uses the term "mask" for what we call the state-space). Defining the state-space by a discrete set of points is handy because it allows specific points to be deleted or not, depending on whether they represent available or suitable habitat (see Section 5.10). We can also define the state-space as an arbitrary collection of polygons stored as a GIS shapefile which can be analyzed easily by MCMC in **R** (see Section 17.7), but not so easily in the **BUGS** engines. In Section 5.10, we provide an analysis of the wolverine camera trapping data, in which we define the state-space to be a regular continuous polygon (a rectangle).

5.3.1.2 *Invariance to the state-space*

We will assert for all models we consider in this book that density is invariant to the size and extent of $\mathcal{S}$, if $\mathcal{S}$ is sufficiently large, and as long as our model relating p_{ij} to $\mathbf{s}_i$ is a decreasing function of distance. We can demonstrate this easily by drawing an analogy with a 1-d case involving distance sampling on a transect. Let y_j be the number of individuals captured in some interval $[d_{j-1}, d_j)$, and define the maximum observation distance $d_J = B$ for some large value of B. The observations from a survey are $y_1, \ldots, y_J$, and the likelihood is a multinomial likelihood, so the log-likelihood is of the form

$$\log L(y_1, \ldots, y_J) = \sum_{j=1}^{J} y_j \log(\pi_j),$$

where π_j is the probability of detecting an individual in distance class j, which depends on parameters of the detection function (the manner of which is not relevant for the present discussion). Choosing B sufficiently large guarantees that $\mathbb{E}(y_J) = 0$ and therefore the observed frequency in the "last cell" contributes nothing to the likelihood, in regular situations in which the detection function decays monotonically with distance and prior density is constant. We can think of B as being related to the state-space in an SCR model, as the width of a rectangular state-space with area $B \times L$, L being the length of the transect. Thus, if we choose B large enough, then we ensure that the expected trap frequencies beyond B will be 0, and thus contribute nothing to the likelihood.

Sometimes our estimate of density can be affected by choosing an $\mathcal{S}$ that is small relative to animal movement or home range size. However, this might be sensible if $\mathcal{S}$ is naturally well defined. As we discussed in Chapter 1, $\mathcal{S}$ *is part of the model*, and thus it is sensible that estimates of density might be sensitive to its definition in problems where it is natural to restrict $\mathcal{S}$. One could imagine, however, in specific cases (e.g., a small population with well-defined habitat preferences), that a problem could arise because changing the state-space based on differing opinions and GIS layers might have substantial affects on the density estimate. But this is a real biological problem, and a natural consequence of the spatial formalization of capture-recapture

models—a feature, not a bug or some statistical artifact—and it should be resolved with better information, research, and thinking. For situations where there is not a natural choice of $\mathcal{S}$, we should default to choosing $\mathcal{S}$ to be very large relative to typical home range size of the species being studied in order to achieve invariance or, otherwise, evaluate sensitivity of density estimates by trying a couple of different choices of $\mathcal{S}$. This is a standard "sensitivity to prior" argument that Bayesians always have to be conscious of. We demonstrate this in our analysis of Section 5.9 below. As an additional practical consideration, we note that the area of the state-space $\mathcal{S}$ affects data augmentation. If you increase the size of $\mathcal{S}$, then there are more individuals to account for and therefore the size of the augmented data set M must increase, which has computational implications.

5.3.2 Connection to model M_h and distance sampling

SCR models are closely related to "model M_h" and distance sampling. In SCR models, heterogeneity in encounter probability is induced by both the effect of distance in the model for detection probability and from specification of the state-space. To understand this, suppose activity centers have the uniform distribution:

$$\mathbf{s} \sim \text{Uniform}(\mathcal{S})$$

and encounter probability is a function of $\mathbf{s}$, denoted by $p(\mathbf{s}) = p(y = 1|\mathbf{s})$. For example, under Eq. (5.2.2) we have that

$$p(\mathbf{s}) = \text{logit}^{-1}(\alpha_0 - \alpha_1 \|\mathbf{x}_j - \mathbf{s}_i\|)$$

and we can work out, either analytically or empirically, what is the implied distribution of p for a population of individuals. Figure 5.3 shows a histogram of p for a

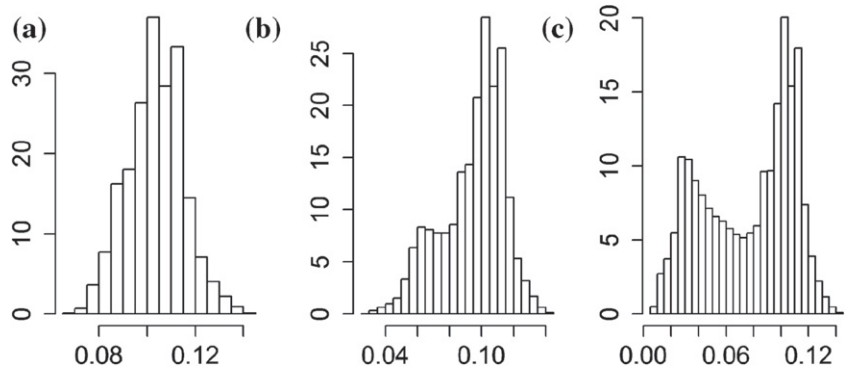

FIGURE 5.3

Implied distribution of p_i for a population of individuals as a function of the size of the state-space buffer around the trap array. The state-space buffer is 0.2, 0.5, and 1.0 for panels (a), (b), (c), respectively. In each case, the trap array is fixed and centered within a square state-space.

hypothetical population of 100,000 individuals on a state-space enclosing our 5×5 trap array above, under the logistic model for distance given by Eq. (5.2.2), with buffers of 0.2, 0.5, and 1.0. We see the mass shifts to the left as the buffer increases, implying more individuals with lower encounter probabilities, as their home range centers increase in distance from the trap array.

Another way to understand this is by representing $\mathcal{S}$ as a set of discrete points on a grid. In the coarsest possible case where $\mathcal{S}$ is a single arbitrary point, then every individual has exactly the same p. As we increase the number of points in $\mathcal{S}$, more distinct values of p are possible. Indeed, when $\mathcal{S}$ is characterized by discrete points, then SCR models are precisely a type of finite-mixture model (Norris and Pollock, 1996; Pledger, 2004), except that, in the case of SCR models, we have some information about which group an individual belongs to (i.e., where their activity centers are), as a result of which traps it is captured in.

It is also worth reemphasizing that the basic SCR encounter model is a binomial encounter model in which distance is a covariate. As such, it is similar to classical distance sampling models (Buckland et al., 2001). Both have distance as a covariate but, in classical distance sampling problems, the focus is on the distance between the observer and the animal at an instant in time, not the distance between a trap and an animal's home range center. As a practical matter, in distance sampling, "distance" is *observed* for those individuals that appear in the sample. Conversely, in SCR problems, it is only imperfectly observed (we have partial information in the form of trap observations). Clearly, it is preferable to observe distance if possible, but distance sampling requires field methods that are not practical in many situations, e.g., when studying carnivores such as bears or large cats. Furthermore, SCR models allow us to relax many of the assumptions made in classical distance sampling, such as perfect detection at distance zero, and SCR models allow for estimates of quantities other than density, such as home range size, and space usage (Chapters 12 and 13).

5.4 The implied model of space usage

We developed the basic SCR model in terms of a latent variable, $\mathbf{s}$, the home range center or activity center. Surely, the encounter probability model, which relates encounter of individuals in specific traps to $\mathbf{s}$, must somehow imply a certain model for home range geometry and size. Here, we explore the nature of that relationship, and we argue that any given detection model implies a model of space usage—i.e., the amount and extent of area used some prescribed percentage of the time. So we might say, for example, 95% of animal movements are within some distance from an individual's activity center.

Indeed, it is natural to interpret the detection model as the composite of two processes: movement of an individual about its home range, i.e., how it uses space within its home range ("space usage"), and detection *conditional on use* in the vicinity of a trapping device. It is natural to decompose encounter probability according to:

$$\Pr(\text{encounter at } \mathbf{x}|\mathbf{s}) = \Pr(\text{encounter}|\text{usage of } \mathbf{x}, \mathbf{s}) \Pr(\text{usage of } \mathbf{x}|\mathbf{s}).$$

As before, $\mathbf{s}$ is the activity center and, here, $\mathbf{x}$ is any potential trap location on the landscape. We will think of $\mathbf{x}$ as a pixel of some small size. In practice, it might make sense to think about the first component, i.e., Pr(encounter|usage of $\mathbf{x}$, $\mathbf{s}$) as being a constant (e.g., if $\mathbf{x}$ is the area in close proximity to a trap). In that case, the encounter probability model is directly proportional to this model for individual movements about their home range center which determines the use frequency of each location $\mathbf{x}$. This is a sensible heuristic model for what ecologists would call a central place forager, although, as we have stated previously, it may be meaningful as a description of transient space usage as well (that is, the space usage during the period of sampling).

To motivate a specific model for space usage, imagine the area we are interested in consists of some large number of small pixels (i.e., we're looking at a discrete representation of space), and that we have some kind of perfect observation device (e.g., continuous telemetry) so that we observe every time an individual moves into a pixel. After a long period of time, we observe an enormous sample size of values of $\mathbf{x}$. We tally those up into each pixel, producing the frequency $m(\mathbf{x}, \mathbf{s})$, which is the "true" usage of pixel $\mathbf{x}$ by individual with activity center $\mathbf{s}$. So, then, the usage model should be regarded as a probability mass function for these counts and, naturally, we regard the counts $m(\mathbf{x}, \mathbf{s})$ as a multinomial observation with probabilities $\pi(\mathbf{x}|\mathbf{s})$, and prescribe a suitable model for $\pi(\mathbf{x}|\mathbf{s})$ that describes how use events should accumulate in space. A natural null model for $\pi(\mathbf{x}|\mathbf{s})$ has a decreasing probability of use as $\mathbf{x}$ gets far away from $\mathbf{s}$, i.e., animals spend more time close to their activity centers than far away. We can regard points used by the individual with activity center $\mathbf{s}$ as the realization of a point process with conditional intensity:

$$\pi(\mathbf{x}|\mathbf{s}) = \frac{g(\mathbf{x}, \mathbf{s})}{\sum_x g(\mathbf{x}, \mathbf{s})}, \tag{5.4.1}$$

where $g(\mathbf{x}, \mathbf{s})$ is any positive function. In continuous space, the equivalent representation is:

$$\pi(\mathbf{x}|\mathbf{s}) = \frac{g(\mathbf{x}, \mathbf{s})}{\int g(\mathbf{x}, \mathbf{s}) dx}.$$

If we use a negative exponential function, then this produces a standard resource selection function (RSF) model (e.g., Manly et al., 2002, Chapter 8). But, we often use a Gaussian kernel, i.e.,

$$g(\mathbf{x}, \mathbf{s}) = \exp(-d(\mathbf{x}, \mathbf{s})^2/(2\sigma^2))$$

so that contours of the probability of space usage resemble a bivariate normal or Gaussian probability distribution function.

To apply this model of space usage to SCR problems we allow for imperfect detection by introducing a "thinning" mechanism applied to the true counts $m(\mathbf{x}, \mathbf{s})$. This yields, precisely, our Gaussian encounter probability model where the thinning rate is our baseline encounter probability p_0 for each pixel where we place a trap, and $p = 0$ in each pixel where we don't place a trap.

The main take-away point here is that underlying most SCR models is some kind of model of space usage, implied by the specific choice of $g(\mathbf{x}, \mathbf{s})$. Whether or not

we have perfect sampling devices, the function we use in the encounter probability model equates to some conditional distribution of points, a utilization distribution, as in Eq. (5.4.1), from which we can compute effective home range area, i.e., the area that contains some percent of the mass of a probability distribution proportional to $g(\mathbf{x}, \mathbf{s})$; e.g., 95% of all space used by an individual with activity center $\mathbf{s}$.

5.4.1 Bivariate normal case

One encounter model that allows direct analytic computation of home range area is the Gaussian encounter probability model

$$p(\mathbf{x}, \mathbf{s}) = p_0 \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x} - \mathbf{s}\|^2\right).$$

For this model, encounter probability is proportional to the kernel of a bivariate normal (Gaussian) pdf and so the natural interpretation is that in which movement outcomes (or successive locations of an individual) are draws from a bivariate normal distribution with standard deviation σ. We say that use of this model implies a bivariate normal model of space usage. Under this model we can compute precisely the effective home range area. In particular, if use outcomes are bivariate normal, then $\|\mathbf{x} - \mathbf{s}\|^2$ has a chi-square distribution with 2 df. The quantity $B(\alpha)$ that encloses $(1 - \alpha)\%$ of all realized distances, i.e., $\Pr(d \leq B(\alpha)) = 1 - \alpha$, is $B(\alpha) = \sigma\sqrt{q(\alpha, 2)}$ where $q(\alpha, 2)$ is the α chi-square critical value on 2 df. For example, to compute $q(.05, 2)$ in **R** we execute the command `qchisq(.95,2)`, which is $q(2, \alpha) = 5.99$. Then, for $\sigma = 1$, $B(\alpha) = 1 \times \sqrt{5.99} = 2.447$. Therefore, 95% of the points used will be within 2.447 (standard deviation) units of the home range center. So, in practice, we can estimate σ by fitting the bivariate normal encounter probability model to some SCR data, and then use the estimated σ to compute the "95% radius," say $r_{.95} = \sigma\sqrt{5.99}$, and convert this to the *95% use area*—the area around $\mathbf{s}$ which contains 95% of the movement outcomes—according to $A_{.95} = \pi r_{.95}^2$.

An alternative bivariate normal model is the bivariate normal hazard rate model:

$$p(\mathbf{x}, \mathbf{s}) = 1 - \exp\left(-\lambda_0 \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x} - \mathbf{s}\|^2\right)\right). \tag{5.4.2}$$

We use λ_0 here because this parameter, the baseline encounter *rate*, can be >1. This arises by assuming the latent "use frequency" $m(\mathbf{x}, \mathbf{s})$ is a Poisson random variable with intensity $\lambda_0 g(\mathbf{x}, \mathbf{s})$. The model is distinct from our Gaussian encounter model $p(\mathbf{x}, \mathbf{s}) = p_0 g(\mathbf{x}, \mathbf{s})$ used previously, although we find that they produce similar results in terms of estimates of density or 95% use area, as long as baseline encounter probability is low. We discuss these two formulations of the bivariate normal model further in Chapter 9.

5.4.2 Calculating space usage

For any encounter model we can compute space usage quantiles empirically by taking a fine grid of points and either simulating movement outcomes with probabilities

proportional to $p(\mathbf{x}, \mathbf{s})$ and accumulating area around $\mathbf{s}$, or else we can do this precisely by varying $B(\alpha)$ to find that value within which 95% of all movements are concentrated, i.e., the set of all $\mathbf{x}$ such that $\|\mathbf{x} - \mathbf{s}\| \leq B(\alpha)$. Under any detection model, movement outcomes will occur in proportion to $p(\mathbf{x}, \mathbf{s})$, as long as the probability of encounter is constant, *conditional on use*, and so we can define our space usage distribution according to:

$$\pi(\mathbf{x}|\mathbf{s}) = \frac{p(\mathbf{x}, \mathbf{s})}{\sum_x p(\mathbf{x}, \mathbf{s})}.$$

Given the probabilities $\pi(\mathbf{x}, \mathbf{s})$ for all $\mathbf{x}$, we can find the value of $B(\alpha)$, for any α, such that

$$\left(\sum_{\mathbf{x} \ni \|\mathbf{x}-\mathbf{s}\| \leq B(\alpha)} \pi(\mathbf{x}, \mathbf{s}) \right) \leq 1 - \alpha$$

(here, we use $\ni$ to mean "such that"). We have a function called `hra` in the `scrbook` package that computes the home range area for any encounter model and prescribed parameter values. The help file for `hra` has an example of simulating some data. The following commands illustrate this calculation for two different bivariate normal models of space usage:

```
##
## Define encounter probability model as R function
##
> pGauss2 <- function(parms,Dmat){
    a0 <- parms[1]
    sigma <- parms[2]
    lp <- parms[1] -(1/(2*parms[2]*parms[2]))*Dmat*Dmat
    p <- 1-exp(-exp(lp))
    p
}

> pGauss1 <- function(parms,Dmat){
    a0 <- parms[1]
    sigma <- parms[2]
    p <- plogis(parms[1])*exp(-(1/(2*parms[2]*parms[2]))*Dmat*Dmat)
    p
}

##
## Execute hra with sigma = .3993
##
> hra(pGauss1,parms=c(-2,.3993),plot=FALSE,xlim=c(0,6),ylim=c(0,6),
      ng=500,tol=.0005)

[1] 0.9784019
radius to achieve 95% of area: 0.9784019
home range area: 3.007353
[1] 3.007353
```

```
## Analytic solution:
##     true sigma that produces area of 3
> sqrt(3/pi)/sqrt(5.99)
[1] 0.3992751
```

What this means is that $B(\alpha) = 0.978$ is the radius that encloses about 95% of all movements under the standard bivariate normal encounter model. Therefore, the area is about $\pi\ 0.978^2 = 3.007$ spatial units. You can change the intercept of the model and find that it has no effect. The true (analytic) value of σ that produces a home range area of 3.0 is 0.3993 which is the value we input to the `hra` function. We can improve on the numerical approximation to home range area (get it closer to 3.0) by increasing the resolution of our spatial grid (increase the `ng` argument) along with the `tol` argument which determines how close to the target value is close enough.

We can also reverse this process, and find, for any detection model, the parameter values that produce a certain $(1 - \alpha)\%$ home range area, which we imagine would be useful for doing simulation studies. The function `hra` will compute the value of the scale parameter that achieves a certain target $(1 - \alpha)\%$ home range area, by simply providing a non-null value of the variable `target.area`. Here we use `target.area = 3.00735` (from above) to obtain a close approximation to the value σ we started with (the `parms` argument is meaningless here):

```
> hra(pGauss1,parms=c(-2,.3993),plot=FALSE,xlim,ylim,ng=500,
      target.area=3.00735,tol=.0005)

Value of parm[2] to achieve 95% home range area of 3.00735: 0.3993674
```

5.4.3 Relevance of understanding space usage

One important reason that we need to be able to deduce "home range area" from a detection model is so that we can compare different models with respect to a common biological currency. Many encounter probability models have some scale parameter, which we might call σ no matter the model, but this relates to 95% area in a different manner under each model. Therefore, we want to be able to convert different models to the same currency. Another reason to understand the relationship between models of encounter probability and space usage is that it opens the door to combining traditional resource selection data from telemetry with spatial capture-recapture data. In Chapter 13 we consider this problem, for the case in which a sample of individuals produces encounter history data suitable for SCR models and, in addition, we have telemetry locations on a sample of individuals. This is achieved by regarding the two sources of data as resulting from the same underlying process of space usage, but telemetry data produce "perfect" observations, like always-on camera traps blanketing a landscape.

5.4.4 Contamination due to behavioral response

Interpretation of encounter probability models as models of animal home range and space usage can be complicated by a number of factors, including whether traps are

baited or not. In the case of baited traps, this might lead to a behavioral response (Section 7.2.3), which could affect animal space usage. For example, if traps attract animals from a long distance, it could make typical home ranges appear larger than normal. More likely, in our view, it wouldn't change the typical size of a range but would change how individuals use their range, e.g., by moving from baited trap to baited trap, so that observed movement distances of individuals are typically larger than normal.

In other cases, the reliance on Euclidean distance in models for encounter probability might be unrealistic and can lead to biased estimates of density (Royle et al., 2013). For example, animals might concentrate their movements along trails, roads, or other landscape features. In this case, models that accommodate other distance metrics can be considered. We present models based on least-cost path in Chapter 12.

5.5 Simulating SCR data

It is always useful to simulate data, because it allows you to understand the system that you're modeling and calibrate your understanding with specific values of the model parameters. That is, you can simulate data using different parameter values until you obtain data that "look right" based on your knowledge of the specific situation that you're interested in. Here we provide a simple script to illustrate how to simulate spatial encounter history data. In this exercise, we simulate data for 100 individuals and a 25-trap array laid out in a 5×5 grid of unit spacing. The specific encounter model is the Gaussian model given above and we used the code below to simulate data used in subsequent analyses. The 100 activity centers were simulated on a state-space defined by an 8×8 square within which the trap array was centered (thus, the trap array is buffered by 2 units). Therefore, the density of individuals in this system is fixed at 100/64:

```
> set.seed(2013)
# Create 5 x 5 grid of trap locations with unit spacing
> traplocs <- cbind(sort(rep(1:5,5)),rep(1:5,5))
> ntraps <- nrow(traplocs)
# Compute distance matrix:
> Dmat <- e2dist(traplocs,traplocs)

# Define state-space of point process. (i.e., where animals live).
# "buffer" just adds a fixed buffer to the outer extent of the traps.
#
> buffer <- 2
> xlim <- c(min(traplocs[,1] - buffer),max(traplocs[,1] + buffer))
> ylim <- c(min(traplocs[,2] - buffer),max(traplocs[,2] + buffer))

> N <- 100     # population size
> K <-  20     # number nights of effort

> sx <- runif(N,xlim[1],xlim[2])     # simulate activity centers
> sy <- runif(N,ylim[1],ylim[2])
> S <- cbind(sx,sy)
```

```
# Compute distance matrix:
> D <- e2dist(S,traplocs)      # distance of each individual from each trap

> alpha0 <- -2.5       # define parameters of encounter probability
> sigma <- 0.5         # scale parameter of half-normal
> alpha1 <- 1/(2*sigma*sigma) # convert to coefficient on distance

# Compute Probability of encounter:
#
> probcap <- plogis(-2.5)*exp(- alpha1*D*D)

# Generate the encounters of every individual in every trap
> Y <- matrix(NA,nrow=N,ncol=ntraps)
> for(i in 1:nrow(Y)){
    Y[i,] <- rbinom(ntraps,K,probcap[i,])
  }
```

The data matrix produced above has N rows (individuals) and J columns (traps), with each element being the frequency of encounter (out of K) of individuals in traps. We remind the reader that, in presenting **R** or other code snippets throughout the book, we will deviate from our standard variable expressions for some quantities. In particular, we sometimes substitute words for integer variable designations: `nind` (for n), `ntraps` (for J), and `nocc` (for K). In our opinion this leaves less to be inferred by the reader in trying to understand code snippets.

Subsequently, we will generate data using this code packaged in an **R** function called `simSCR0` in the package `scrbook`, which takes a number of arguments including `discard0`, which, if `TRUE`, will return only the encounter histories for captured individuals. A second argument is `array3d`, which, if `TRUE`, returns the three-dimensional encounter history array instead of the aggregated individual and trap-specific encounter frequencies (see below). Finally, we provide a random number seed, `rnd = 2013`, to ensure repeatability of the analysis. We obtain a data set as above using the following command:

```
> data <- simSCR0(discard0=TRUE, array3d=FALSE, rnd=2013)
```

The **R** object `data` is a list, so let's take a look at what's in the list and then harvest some of its elements for further analysis below:

```
> names(data)
[1] "Y"        "traplocs" "xlim"     "ylim"     "N"      "alpha0"     "beta"
[8] "sigma"      "K"

## Grab encounter histories from simulated data list
> Y <- data$Y
## Grab the trap locations
> traplocs <- data$traplocs
```

5.5.1 Formatting and manipulating data sets

Conventional capture-recapture data are easily stored and manipulated as a two-dimensional array, an `nind` × `K` (individuals by sample occasions) matrix, which

is maximally informative for any conventional capture-recapture model, but not for spatial capture-recapture models. For SCR models we must preserve the spatial information (trap locations of capture) in the encounter history information. We will routinely analyze data from three standard formats:

(1) The basic two-dimensional data format, which is an $n \times J$ encounter frequency matrix such as that simulated previously. These are the total number of encounters in each trap, summed over the K sample occasions.
(2) The maximally informative three-dimensional array, for which we establish here the convention that it has dimensions $n \times J \times K$.
(3) We use a compact format—the "encounter data file" which we describe in Section 5.9.

To simulate data in the most informative format—the "3-d array"—we can use the **R** commands given previously but replace the last four lines with the following:

```
> Y <- array(NA,dim=c(N,ntraps,K))

> for(i in 1:nrow(Y)){
    for(j in 1:ntraps){
       Y[i,j,1:K] <- rbinom(K,1,probcap[i,j])
    }
  }
```

We see that a collection of K binary encounter events are generated for *each* individual and for *each* trap. The probabilities of those Bernoulli trials are computed based on the distance from each individual's home range center and the trap (see calculation above), and those are housed in the matrix `probcap`. Our data simulator function `simSRC0` will return the full 3-d array if `array3d=TRUE` is specified in the function call. To recover the 2-d matrix from the 3-d array, and subset the 3-d array to individuals that were captured, we do this:

```
# Sum over the "sample occasions" dimension (3rd margin of the array)
> Y2d <- apply(Y,c(1,2),sum)

# Compute how many times each individual was captured
> ncaps <- apply(Y2d,1,sum)

# Keep those individuals that were captured
> Y <- Y[ncaps>0,,]
```

5.6 Fitting model SCRO in BUGS

If we somehow knew the value of N then we could fit this model directly because, in that case, it is a special kind of logistic regression model, one with a random effect (**s**) that enters into the model in a peculiar fashion. Our aim here is to analyze the known-N problem, using our simulated data, as an incremental step in our progress toward fitting more generally useful models. To begin, we use our simulator

to grab a data set and then harvest the elements of the resulting object for further analysis:

```
> data <- simSCR0(discard0=FALSE,rnd=2013)
> y <- data$Y
> traplocs <- data$traplocs

# In this case nind=N because we're doing the known-N problem
#
> nind <- nrow(y)
> X <- data$traplocs
> J <- nrow(X)     # number of traps
> K <- data$K
> xlim <- data$xlim
> ylim <- data$ylim
```

Note that we specify `discard0 = FALSE` so that we have a "complete" data set, i.e., one with the all-zero encounter histories corresponding to uncaptured individuals. Now, within an **R** session, we can create the **BUGS** model file and fit the model using the following commands:

```
cat("
 model{
 alpha0 ~ dnorm(0,.1)
 logit(p0) <- alpha0
 alpha1 ~ dnorm(0,.1)
 sigma <- sqrt(1/(2*alpha1))
 for(i in 1:N){    # note N here -- N is KNOWN in this example
  s[i,1] ~ dunif(xlim[1],xlim[2])
  s[i,2] ~ dunif(ylim[1],ylim[2])
  for(j in 1:J){
   d[i,j] <- pow(pow(s[i,1]-X[j,1],2) + pow(s[i,2]-X[j,2],2),0.5)
   y[i,j] ~ dbin(p[i,j],K)
   p[i,j] <- p0*exp(- alpha1*d[i,j]*d[i,j])
   }
  }
}
",file = "SCR0a.txt")
```

This model describes the Gaussian encounter probability model, but it would be trivial to modify that to various others including the logistic described above. We have to constrain the encounter probability to be in [0, 1], which we do here by defining `alpha0` to be the logit of the intercept parameter `p0`. Note that the distance covariate is computed within the **BUGS** model specification given the matrix of trap locations, `X`, which is provided to **WinBUGS** as data.

Next, we do a number of organizational activities including bundling the data for **WinBUGS**, defining some initial values, the parameters to monitor, and some basic MCMC settings. We choose initial values for the activity centers **s** by generating uniform random numbers in the state-space but, for the observed individuals, we replace those values by each individual's mean trap coordinate of its encounters:

```
### Starting values for activity centers, s
> sst <- cbind(runif(nind,xlim[1],xlim[2]),runif(nind,ylim[1],ylim[2]))
> for(i in 1:nind){
      if(sum(y[i,])==0) next
      sst[i,1] <- mean( X[y[i,]>0,1] )
      sst[i,2] <- mean( X[y[i,]>0,2] )
  }

> data <- list (y=y, X=X, K=K, N=nind, J=J, xlim=xlim, ylim=ylim)
> inits <- function(){
      list (alpha0=rnorm(1,-4,.4), alpha1=runif(1,1,2), s=sst)
  }

> library(R2WinBUGS)
> parameters <- c("alpha0","alpha1","sigma")
> out <- bugs (data, inits, parameters, "SCR0a.txt", n.thin=1, n.chains=3,
               n.burnin=1000,n.iter=2000,debug=TRUE,working.dir=getwd( ))
```

There is little to say about the preceding operations other than to suggest that you might explore the output and investigate additional analyses by running the `simSCR0` script provided in the **R** package `scrbook`.

For purposes here, we ran 1,000 burn-in and 1,000 post-burn-in iterations, and 3 chains, to obtain 3,000 posterior samples. Because we know N for this particular data set we only have two parameters of the detection model to summarize (`alpha0` and `alpha1`), along with the derived parameter σ, the scale parameter of the Gaussian kernel, i.e., $\sigma = \sqrt{1/(2\alpha_1)}$. When the object `out` is produced we print a summary of the results as follows:

```
 > print(out,digits=2)
 Inference for Bugs model at "SCR0a.txt", fit using WinBUGS,
  3 chains, each with 2000 iterations (first 1000 discarded)
  n.sims = 3000 iterations saved
            mean     sd    2.5%     25%     50%     75%   97.5%  Rhat  n.eff
alpha0     -2.50   0.22   -2.95   -2.65   -2.48   -2.34   -2.09  1.01    190
alpha1      2.44   0.42    1.64    2.15    2.44    2.72    3.30  1.00    530
sigma       0.46   0.04    0.39    0.43    0.45    0.48    0.55  1.00    530
deviance  292.80  21.16  255.60  277.50  291.90  306.00  339.30  1.01    380

 [...some output deleted...]
```

The data were generated with `alpha0` $= -2.5$ and `alpha1` $= 2$. The estimates look reasonably close to those data-generating values and we probably feel pretty good about the performance of the Bayesian analysis and MCMC algorithm that **WinBUGS** cooked up based on our sample size of 1 data set. It is worth noting that the `Rhat` statistics indicate reasonable convergence.

5.7 Unknown *N*

In all real applications N is unknown. We handled this important issue in Chapter 4 using the method of data augmentation (DA), which we apply here to achieve a realistic analysis of model SCR0. As with the basic closed population models

considered previously, we formulate the problem by augmenting our observed data set with a number of "all-zero" encounter histories—what we referred to in Chapter 4 as potential individuals. If n is the number of observed individuals, then let $M - n$ be the number of potential individuals in the data set. For the two-dimensional y_{ij} data structure (n individual $\times$ J traps encounter frequencies) we simply add additional rows of all-zero observations to that data set. Because such "individuals" are unobserved, they therefore necessarily have $y_{ij} = 0$ for all j. A data set with 4 traps and 6 individuals, augmented with 4 potential individuals therefore might look like this:

```
       trap1  trap2  trap3  trap4
 [1,]      1      0      0      0
 [2,]      0      2      0      0
 [3,]      0      0      0      1
 [4,]      0      1      0      0
 [5,]      0      0      1      1
 [6,]      1      0      1      0
 [7,]      0      0      0      0
 [8,]      0      0      0      0
 [9,]      0      0      0      0
[10,]      0      0      0      0
```

We typically have more than 4 traps and, if we're fortunate, many more individuals in our data set.

For the augmented data set, we introduce a set of binary latent variables (the data augmentation variables), z_i, and the model is extended to describe $\Pr(z_i = 1)$, which, in the context of this problem, is the probability that an individual in the augmented data set is a member of the population of size N that was exposed to sampling. In other words, if $z_i = 1$ for one of the all-zero encounter histories, this is implied to be a sampling zero whereas observations for which $z_i = 0$ are "structural zeros." Under DA, we also express the binomial observation model *conditional on* z_i as follows:

$$y_{ij}|z_i \sim \text{Binomial}(K, z_i\, p_{ij}),$$

where we see that the binomial probability evaluates to 0 if $z_i = 0$ (so y_{ij} is a fixed 0 in that case) and evaluates to p_{ij} if $z_i = 1$.

How big does the augmented data set have to be? We discussed this issue in Chapter 4 where we noted that the size of the data set is equivalent to the upper limit of a uniform prior distribution on N. Practically speaking, it should be sufficiently large so that the posterior distribution for N is not truncated. On the other hand, if it is too large then unnecessary calculations are being done. An approach to choosing M by trial-and-error is indicated. Do a short MCMC run and then consider whether you need to increase M. See Chapter 17 for an example of this. Kéry and Schaub (2012, Chapter 6) provide an assessment of choosing M in closed population models. Using data augmentation, N is a derived parameter, computed by $N = \sum_{i=1}^{M} z_i$. Similarly, *density*, D, is also a derived parameter, computed as $D = N/\text{area}(\mathcal{S})$.

5.7.1 Analysis using data augmentation in WinBUGS

We provide a complete **R** script for simulating and organizing a data set, and analyzing it in **WinBUGS**. As before, we begin by obtaining a data set using our `simSCR0` function and then harvesting the required data objects from the resulting data list. Note that we use the `discard0=TRUE` option this time so that we get a "real looking" data set with no all-zero encounter histories:

```
## Simulate the data and extract the required objects
##
> data <- simSCR0(discard0=TRUE,rnd=2013)
> y <- data$Y
> nind <- nrow(y)
> X <- data$traplocs
> K <- data$K
> J <- nrow(X)
> xlim <- data$xlim
> ylim <- data$ylim
```

After harvesting the data we augment the data matrix `y` with $M - n$ all-zero encounter histories, and create starting values for the variables z_i and the activity centers $\mathbf{s}_i$ of which, for each, we require M values. One thing to take care of in using the **BUGS** engines is the starting values for the activity centers. It is usually helpful to start the $\mathbf{s}_i$ for each observed individual at or near the trap(s) it was captured. All of this happens as follows:

```
## Data augmentation
> M <- 200
> y <- rbind(y,matrix(0,nrow=M-nind,ncol=ncol(y)))
> z <- c(rep(1,nind),rep(0,M-nind))

## Starting values for s
> sst <- cbind(runif(M,xlim[1],xlim[2]),runif(M,ylim[1],ylim[2]))
> for(i in 1:nind){
    sst[i,1] <- mean( X[y[i,]>0,1] )
    sst[i,2] <- mean( X[y[i,]>0,2] )
  }
```

Next, we write out the **BUGS** model specification and save it to an external file called `SCR0b.txt`. The model specification now includes M encounter histories including the augmented potential individuals, the data augmentation parameters z_i, and the data augmentation parameter ψ:

```
> cat("
model{
  alpha0 ~ dnorm(0,.1)
  logit(p0) <- alpha0
  alpha1 ~ dnorm(0,.1)
  sigma <- sqrt(1/(2*alpha1))
  psi ~ dunif(0,1)
```

```
for(i in 1:M){
  z[i] ~ dbern(psi)
  s[i,1] ~ dunif(xlim[1],xlim[2])
  s[i,2] ~ dunif(ylim[1],ylim[2])
  for(j in 1:J){
    d[i,j] <- pow(pow(s[i,1]-X[j,1],2) + pow(s[i,2]-X[j,2],2),0.5)
    y[i,j] ~ dbin(p[i,j],K)
    p[i,j] <- z[i]*p0*exp(- alpha1*d[i,j]*d[i,j])
  }
}
N <- sum(z[])
D <- N/64
}
",file = "SCR0b.txt")
```

The remainder of the code for bundling the data, creating initial values and executing **WinBUGS**, looks much the same as before except with more or differently named arguments:

```
> data <- list (y=y, X=X, K=K, M=M, J=J, xlim=xlim, ylim=ylim)
> inits <- function(){
    list (alpha0=rnorm(1,-4,.4), alpha1=runif(1,1,2), s=sst, z=z)
    }

> library(R2WinBUGS)
> parameters <- c("alpha0","alpha1","sigma","N","D")
> out <- bugs (data, inits, parameters, "SCR0b.txt", n.thin=1,
        n.chains=3, n.burnin=1000,n.iter=2000,debug=TRUE,working.dir=getwd())
```

Note the differences in this new **WinBUGS** model with that appearing in the known-N version—there are not many! The loop over individuals goes up to M now, and there is a model component for the DA variables z. The input data has changed slightly too, as the augmented data set has more rows to include excess all-zero encounter histories. This analysis can be run directly using the `SCR0bayes` function once the `scrbook` package is loaded, by issuing the following commands:

```
> library(scrbook)
> data <- simSCR0(discard0=TRUE,rnd=2013)
> out1 <- SCR0bayes(data,M=200,engine="winbugs",ni=2000,nb=1000)
```

Summarizing the output from **WinBUGS** produces:

```
> print(out1,digits=2)
Inference for Bugs model at "SCR0b.txt", fit using Win BUGS,
   3 chains, each with 2000 iterations (first 1000 discarded)
   n.sims = 3000 iterations saved
             mean     sd    2.5%     25%     50%     75%   97.5%  Rhat  n.eff
alpha0     -2.57   0.23   -3.04   -2.72   -2.56   -2.41   -2.15  1.01    320
alpha1      2.46   0.42    1.63    2.16    2.46    2.73    3.33  1.02    120
sigma       0.46   0.04    0.39    0.43    0.45    0.48    0.55  1.02    120
N         113.62  15.73   86.00  102.00  113.00  124.00  147.00  1.01    260
D           1.78   0.25    1.34    1.59    1.77    1.94    2.30  1.01    260
deviance  302.60  23.67  261.19  285.47  301.50  317.90  354.91  1.00   1400

[...some output deleted...]
```

The `Rhat` statistic (discussed in Sections 3.5.2 and 17.6.4) for this analysis indicates satisfactory convergence. We see that the estimated parameters (α_0 and α_1) are comparable to the previous results obtained for the known-N case, and also not too different from the data-generating values. The posterior of N overlaps the data-generating value substantially.

5.7.1.1 *Use of other BUGS engines:* JAGS

There are two other popular **BUGS** engines in widespread use: **OpenBUGS** (Thomas et al., 2006) and **JAGS** (Plummer, 2003). Both of these are easily called from **R**. **OpenBUGS** can be used instead of **WinBUGS** by changing the package option in the `bugs` call to `package="OpenBUGS"`. **JAGS** can be called using the function `jags` in package `R2jags` which has nearly the same arguments as `bugs`. Or, it can be executed from the **R** package `rjags` (Plummer, 2011) which has a slightly different implementation that we demonstrate here as we reanalyze the simulated data set in the previous section (note: the same **R** commands are used to generate the data and package the data, inits and parameters to monitor). The software and **R** packages mentioned here, and elsewhere in the book, are listed in Appendix 1. The function `jags.model` is used to initialize the model and run the MCMC algorithm for an adaptive period during which tuning of the MCMC algorithm might take place. These samples cannot be used for inference. Then the Markov chains are updated using `coda.samples` to obtain posterior samples for analysis, as follows:

```
> jinit <- jags.model("SCR0b.txt", data=data, inits=inits,
                      n.chains=3, n.adapt=1000)
> jout <- coda.samples(jinit, parameters, n.iter=1000, thin=1)
```

These commands can be executed using the function `SCR0bayes` provided with the **R** package `scrbook` by setting the 'engine' argument to "jags".

5.7.2 Implied home range area

Here to assess implied space usage differences among encounter probability models, we apply the method described in Section 5.4, and use the function `hra` to compute the effective home range area under different encounter probability models fit to simulated data. We simulated a data set from the Gaussian kernel model as in Section 5.7 and then we fitted four models to it: (1) the true data-generating Gaussian encounter probability model; (2) the "hazard" or complementary log-log link model (Eq. (5.4.2)); (3) the negative exponential model; and (4) the logit model (Eq. (5.2.2)). We modified the function `SCR0bayes` for this purpose, which you should be able to do with little difficulty. We fit each model to the same simulated data set using **WinBUGS**, based only on 1,000 post-burn-in samples and 3 chains, which produced the posterior summaries given in Table 5.2. The main thing we see is that, while the implied home range area can vary substantially, there are smaller differences in the estimated N and hence D.

Table 5.2 Posterior mean of model parameters for four different encounter probability models fitted to a single simulated data set, and the effective home range area under each detection model computed using the function `hra`.

Parameter	Gaussian	Cloglog	Exponential	Logit
N	113.62	114.16	119.69	118.29
D	1.78	1.78	1.87	1.85
α_0	−2.57	−2.60	−1.51	−0.47
α_1	2.46	2.56	3.59	3.86
`hra`	3.85	3.78	5.51	2.64

5.7.3 Realized and expected population size

In Bayesian analysis of the SCR model, we estimate a parameter N, which is the size of the population for the prescribed state-space (presumably the state-space is defined so as to be relevant to where our traps were located, so N can be thought of as the size of the sampled population). In the context of Efford and Fewster (2012) this is the *realized* population size. Conversely, sometimes we see estimates of *expected* population size reported, which are estimates of $\mathbb{E}(N)$, the expected size of some hypothetical, unspecified population. Usually, the distinction between realized and expected population size is not made in SCR models, because almost everyone only cares about actual populations—and their realized population size.

If you do likelihood analysis of SCR models, then the distinction between realized and expected is often discussed by whether the estimator is "conditional-on-N" (realized) or not (expected). The naming arises because in obtaining the MLE of N, its properties are evaluated *conditional* on N—in particular, if the estimator is unbiased then $\mathbb{E}(\hat{N}|N) = N$ and $\text{Var}(\hat{N}|N) = \tilde{\sigma}^2_{\hat{N}}$ is the sampling variance. This does not conform to any concept or quantity that is relevant to Bayesian inference. If we care about N for the population that we sampled it is understood to be a realization of a random variable, but the relevance of "conditional-on-N" is hard to see. Bayesian analysis will provide a prediction of N that is based on the posterior $[N|y, \theta]$—which is certainly *not* conditional on N.

There is a third type of inference objective that is relevant in practice and that is prediction of N for a population that was not sampled—i.e., a "new" population. To elaborate on this, consider a situation in which we are concerned about the tiger populations in two distinct reserves in India. We do a camera trapping study on one of the reserves to estimate N_1 and we think the reserves are similar and homogeneous so we're willing to apply a density estimate based on N_1 to the second reserve. For the second reserve, do we want a prediction of the realized population size, N_2, or do we want an estimate of its expected value? We believe the former is the proper quantity for inference about the population size in the second reserve. An estimate of N_2 should include the uncertainty with which the mean is estimated (from reserve 1) and it should also include "process variation" for making the prediction of the latent variable N_2.

As a practical matter, to do a Bayesian analysis of this you could just define the state-space to be the union of the two state-spaces, increase M so that the posterior of the total population size is not truncated, and then have MCMC generate a posterior sample of individuals on the joint state-space. You can tally up the ones that are on $\mathcal{S}_2$ as an estimate of N_2. Alternatively, we can define $\mu = \psi M/A_1$ and then simulate posterior samples of $N_2 \sim \text{Binomial}(M, \mu A_2/M)$ for the new state-space area, A_2.

To carry out a classical likelihood analysis of this second type of problem, what should we do? The argument for making a prediction of a new value of N would go something like this: If you obtain an MLE of N, say $\hat{N}$, then the inference procedure tells us the variance of this *conditional* on N, i.e., $Var(\hat{N}|N)$. This is fine, if we care about the specific value of N that generated our data set. However, if we don't care about the specific value in question then we want to "uncondition" on N to introduce a new variance component. Law of total variance says:

$$\text{Var}(\hat{N}) = \mathbb{E}[\text{Var}(\hat{N}|N)] + \text{Var}[\mathbb{E}(\hat{N}|N)].$$

If $\hat{N}$ is unbiased then we say the unconditional variance is

$$\text{Var}(\hat{N}) = \sigma^2_{\hat{N}} + \text{Var}(N).$$

The first variance component is estimation error and the second component is the "process variance." If you do Bayesian analysis, then you don't have to worry too much about how to compute variances properly. You decide if you care about N, or its expected value, or predictions of some "new" N, and you tabulate the correct posterior distribution from your MCMC output.

The considerations for estimating density are the same. Realized density is N/A, where N is the realized population, unless we put an expectation operator around the N, $\mathbb{E}(N)/A$. The formula for obtaining "expected density" is slightly different depending on whether we assume N has a Poisson distribution, or whether we assume a binomial distribution (under data augmentation). In the latter case ψ is related to the point process intensity (see Chapter 11) in the sense that, under the binomial prior:

$$\mathbb{E}(N) = M \times \psi$$

so, what we think of as "density," D, is a derived parameter $D = M\psi/A$. Under the Poisson point process model density is the canonical parameter and the expected population size is:

$$\mathbb{E}(N) = D \times A.$$

In summary, there are three basic inference problems that relate to estimating population size (or density):

(1) What is the value of N for some population that was sampled. This is what Efford and Fewster call "realized N." In general, we want the uncertainty to reflect having to estimate n_0, the part of the population not seen.

(2) We need to estimate N for some population that we didn't sample but it is "similar" to the population that we have information on. In this case, we have to account for both variation in having to estimate parameters of the distribution of N, and we have to account for process variation in N (i.e., due to the stochastic model of N).

(3) In some extremely limited cases we might care about estimating the expected value of N, $\mathbb{E}(N)$.

5.8 The core SCR assumptions

It's always a good idea to sit down and reflect on the meaning of any particular model, its various assumptions, and what they mean in a specific context. From the statistician's point of view, the basic assumption, the omnibus assumption, as in all of statistics, and for every statistical model, is that "the model is correctly specified." Naturally, that precludes everything that isn't explicitly addressed by the model. To point this out to someone seems to cause a lot of anxiety, so we enumerate here what we think are the most important statistical assumptions of the basic SCR0 model:

- *Demographic closure.* The model does not allow for demographic processes. There is no recruitment or entry into the sampled population. There is no mortality or exit from the sampled population.
- *Geographic closure.* We assume no permanent emigration or immigration from the state-space. However, we allow for "temporary" movements around the state-space and variable exposure to encounter as a result. The whole point of SCR models is to accommodate this dynamic. In ordinary capture-recapture models we have to assume geographic closure to interpret N in a meaningful way.
- *Activity centers are randomly distributed.* That is, uniformity and independence of the underlying point process $\mathbf{s}_1, \ldots, \mathbf{s}_N$ (see Section 5.3).
- *Detection is a function of distance.* A detection model that describes how encounter probability declines as a function of distance from an individual's home range center.
- *Independence of encounters* among individuals. Encounter of any individual is independent of encounter of any other individual.
- *Independence of encounters* of the same individual. Encounter of an individual in any trap is independent of its encounter in any other trap, and subsequent sample occasion.

It's easy to get worried and question the whole SCR enterprise just on the grounds that these assumptions combine to form such a simplistic model, one that surely can't describe the complexity of real populations. On this sentiment, a few points are worth making. First, you don't have inherently fewer assumptions by using an ordinary capture-recapture model but, rather, the SCR model relaxes a number of important assumptions compared to the non-spatial counterpart. For one, here, we're not assuming that p is constant for all individuals but rather that p varies substantially as a matter of the spatial juxtaposition of individuals with traps. So maybe the manner in which p varies isn't quite right, but that's not an argument that supports doing less

modeling. Fundamentally, a distance-based model for p has some basic biological justification in virtually every capture-recapture study. Secondly, for some of these core assumptions such as uniformity, and independence of individuals and of encounters, we expect a fair amount of robustness to departures. They function primarily to allow us to build a model and an estimation scheme and we don't usually think they represent real populations (of course, no model does!). Third, we can extend these assumptions in many different ways and we do that to varying extents in this book, and more work remains to be done in this regard. Fourth, we can also evaluate the reasonableness of the assumptions formally in some cases using standard methods of assessing model fit (Chapter 8).

Sometimes you see in capture-recapture literature statements like "we assume no marks are lost," "marks are correctly identified," and similar things. We will typically neglect such statements because, in our view, we should separate statistical assumptions about model parameters or aspects of the probability model, from what are essentially logistical or operational assumptions about how we interpret our data, or our ability to conduct the study.

5.9 Wolverine camera trapping study

We provide an illustration of some of the concepts we've introduced previously in this chapter by analyzing camera trapping study of wolverines *Gulo gulo* (Magoun et al., 2011; Royle et al., 2011b). In this study, wolverines were individually identified from their ventral pattern (Fig. 1.1), thus allowing for the application of SCR models. The study took place in SE Alaska (Figure 5.4) where 37 cameras were operational for variable periods of time (min = 5 days, max = 108 days, median = 45 days). A consequence of this is that the number of sampling occasions, K, is variable for each camera. Thus, we must provide a vector of sample sizes as data to **BUGS** and modify the model specification in Section 5.7 accordingly.

5.9.1 Practical data organization

To carry out an analysis of these data, we require the matrix of trap coordinates and the encounter history data. We usually store data in two distinct data files which contain all the information needed for an analysis. These files are:

- The encounter data file (EDF), containing a record of at which traps and when each individual encounter occurred.
- The trap deployment file (TDF), which contains the coordinates of each trap, along with information indicating which sample occasions each trap was operating.

Encounter Data File (EDF)—We store the encounter data in an efficient "flat" file format which is easily manipulated in **R** and easy to create in Excel and other spreadsheets that are widely used for data management. The file structure is a simple matrix with four columns (1) **session ID**, the trap *session* which usually corresponds to a year or a primary period in the context of a Robust Design situation, but it could

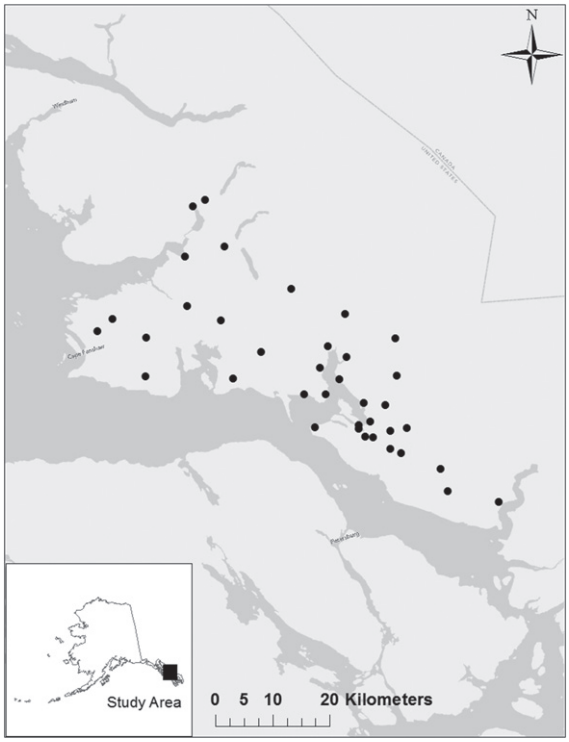

FIGURE 5.4

Wolverine camera trap locations (black dots) from a study that took place in SE Alaska. See Magoun et al. (2011) for details.

also correspond to a distinct spatial unit (see Section 6.5.4 and Chapter 14). For a single-year study (as considered here) this should be an integer that is the same for all records; (2) **individual ID**, the individual identity, being an integer from 1 to n (repeated for multiple captures of the same individual), indicating which individual the record (row) of the matrix belongs to; (3) **occasion ID**, the integer sample occasion which generated the record; and (4) **trap ID**, the trap identity, an integer from 1 to J, the number of traps. The structure of the EDF is the same as used in the `secr` package (Efford, 2011a) and similar to that used in the **SPACECAP** (Gopalaswamy et al., 2012a), and `SCRbayes` (Russell et al., 2012) packages, both of which have a 3-column format (`trapID`, `indID`, `sampID`). We note that the naming of the columns is irrelevant as far as anything we do in this book, although `secr` and other software may have requirements on variable naming.

To illustrate this format, the wolverine data are available in the package `scrbook` by typing:

```
> data(wolverine)
```

which contains a list with elements `wcaps` (the EDF) and `wtraps` (the TDF). We see that `wcaps` has 115 rows, each representing a unique encounter event including the trap identity, the individual identity, and the sample occasion index (`sample`). The first five rows of `wcaps` are:

```
> wolverine$wcaps[1:5,]
      year  individual  day  trap
[1,]     1           2  127     1
[2,]     1           2  128     1
[3,]     1           2  129     1
[4,]     1          18  130     1
[5,]     1           3  106     2
```

The first column here, labeled `year`, is in integer indicating the year or session of the encounter. All these data come from a single year (2008) and so `year` is set to 1. The variable `trap` will have to correspond to the row of a matrix containing the trap coordinates—in this case the TDF file `wtraps`, which we describe further below.

Note that the information provided in this encounter data file `wcaps` does not represent a completely informative summary of the data. For example, if no individuals were captured in a certain trap or during a certain sample occasion, then this compact data format will have no record of this occasion or trap. Thus, we need to know J, the number of traps, *and* K, the number of sample occasions when reformatting this SCR data format into a 2-d encounter frequency matrix or 3-d array.

For our purposes, we need to convert the `wcaps` file into the $n \times J$ array of binomial encounter frequencies, although more general models, such as those containing a behavioral response, might require an encounter-history formulation of the model which requires a full 3-d array. To obtain our encounter frequency matrix, we do this the hard way by first converting the encounter data file into a 3-d array and then summarizing to trap totals. We have a handy function, `SCR23darray`, which takes the compact encounter data file, and converts it to a 3-d array, and then we use the **R** function `apply` to summarize over the sample occasion dimension (by convention here, this is the 2nd dimension). To apply this to the wolverine data we do this:

```
> y3d <- SCR23darray(wolverine$wcaps,wolverine$wtraps)
> y <- apply(y3d,c(1,2),sum)
```

See the help file for more information on `SCR23darray`.

Trap Deployment File (TDF)—The other important information needed to fit SCR models is the "trap deployment file" (TDF) which provides additional information not contained in the encounter data file. The traps file has $K + 3$ columns. The first column is assumed to be a trap identifier, columns 2 and 3 are the easting and northing coordinates (assumed to be in a Euclidean coordinate system), and columns 4 to $K + 3$ are binary indicators of whether each trap was operational during each sample occasion. The first 10 rows (out of 37) and 10 columns (out of 167) of the trap deployment file for the wolverine data are shown as follows:

```
> wolverine$wtraps[1:10,1:10]

   Easting  Northing  1  2  3  4  5  6  7  8
1   632538   6316012  0  0  0  0  0  0  0  0
2   634822   6316568  1  1  1  1  1  1  1  1
3   638455   6309781  0  0  0  0  0  0  0  0
4   634649   6320016  0  0  0  0  0  0  0  0
5   637738   6313994  0  0  0  0  0  0  0  0
6   625278   6318386  0  0  0  0  0  0  0  0
7   631690   6325157  0  0  0  0  0  0  0  0
8   632631   6316609  0  0  0  0  0  0  0  0
9   631374   6331273  0  0  0  0  0  0  0  0
10  634068   6328575  0  0  0  0  0  0  0  0
```

This tells us that trap 2 was operated during occasions (days) 1–7, but the other traps were not operational during those periods. To compute the vector of sample sizes K, and extract the trap locations, we do this:

```
> traps <- wolverine$wtraps
> traplocs <- traps[,1:2]
> K <- apply(traps[,3:ncol(traps)],1,sum)
```

This results in a matrix `traplocs`, which contains the coordinates of each trap, and a vector K containing the number of days that each trap was operational. We now have all the information required to fit a basic SCR model in **BUGS**.

Summarizing the data for the wolverine study, we see that 21 unique individuals were captured a total of 115 times. Most individuals were captured 1–6 times, with 4, 1, 4, 3, 1, and 2 individuals captured 1–6 times, respectively. In addition, 1 individual was captured 8 and one captured 14 times, and 2 individuals each were captured 10 and 13 times. The number of unique traps that captured a particular individual ranged from 1 to 6, with 5, 10, 3, 1, 1, and 1 individual captured in each of 1–6 different traps, respectively, for a total of 50 unique wolverine-trap encounters. These numbers might be hard to get your mind around whereas some tabular summary is often more convenient. For that it seems natural to tabulate individuals by trap and total encounter frequencies. For the wolverine data, we reproduce Table 5.1 from Royle et al. (2011b) as Table 5.3. Generally, effective estimation in SCR studies requires a sufficient sample size of spatial recaptures, which are captures of the same individual in multiple traps. We address this in the context of sampling design in Chapter 10. Therefore, it is informative to understand how many unique traps each individual was captured in, and the total number of encounters.

5.9.2 Fitting the model in WinBUGS

Here, we fit the simplest SCR model with the Gaussian encounter probability model, although we revisit these data and fit additional models in later chapters. Model SCR0 is summarized by the following four elements:

(1) $y_{ij}|\mathbf{s}_i \sim \text{Binomial}(K, z_i\, p_{ij})$;

(2) $p_{ij} = p_0 \exp(-\alpha_1 \|\mathbf{x}_j - \mathbf{s}_i\|^2)$;

Table 5.3 Individual frequencies of capture for wolverines encountered by camera traps in SE Alaska in 2008. Rows index unique traps of capture for each individual and columns represent total number of captures (e.g., we captured 4 individuals 1 time, necessarily in only 1 trap; we captured 3 individuals 3 times but in 2 different traps).

	Number of Captures									
Number of Traps	**1**	**2**	**3**	**4**	**5**	**6**	**8**	**10**	**13**	**14**
1	4	1	0	0	0	0	0	0	0	0
2	0	0	3	2	0	2	1	2	0	0
3	0	0	1	1	0	0	0	0	0	1
4	0	0	0	0	0	0	0	0	1	0
5	0	0	0	0	1	0	0	0	0	0
6	0	0	0	0	0	0	0	0	1	0

(3) $\mathbf{s}_i \sim \text{Uniform}(\mathcal{S})$;
(4) $z_i \sim \text{Bernoulli}(\psi)$.

We assume customary flat priors on the structural (hyper-) parameters of the model, $\alpha_0 = \text{logit}(p_0), \alpha_1$ and ψ.

It remains to define the state-space $\mathcal{S}$. For this, we nested the trap array (Figure 5.4) in a rectangular state-space extending 20 km beyond the traps in each cardinal direction. We scaled the coordinate system so that a unit distance was equal to 10 km, producing a rectangular state-space of dimension 9.88×10.5 units ($area = 10,374$ km^2) within which the trap array was nested. As a general rule, we recommend scaling the state-space so that it is defined near the origin $(x, y) = (0, 0)$. While the scaling of the coordinate system is theoretically irrelevant, a poorly scaled coordinate system can produce Markov chains that mix poorly. The buffer of the state-space should be large enough so that individuals with activity centers beyond the state-space boundary are not likely to be encountered (Section 5.3.1). To evaluate this, we fit models for various choices of a rectangular state-space based on buffers from 1.0 to 5.0 units (10–50 km). In the **R** package `scrbook` we provide a function `wolvSCR0` which will fit model SCR0. For example, to fit the model in **WinBUGS** using data augmentation with $M = 300$ potential individuals, using three Markov chains each of 12,000 total iterations, discarding the first 2,000 as burn-in, we execute the following **R** commands:

```
> library(scrbook)
> data(wolverine)
> traps <- wolverine$wtraps
> y3d <- SCR23darray(wolverine$wcaps,wolverine$wtraps)
> wolv <- wolvSCR0(y3d,traps,nb=2000,ni=12000,buffer=1,M=300)
```

The argument `buffer` determines the buffer size of the state-space in the scaled units (i.e., 10 km). Note that this analysis takes between 1 and 2 h on many machines (in 2013) so we recommend testing it with lower values of M and fewer iterations. The posterior summaries are shown in Table 5.4.

Table 5.4 Posterior summaries of SCR model parameters for the wolverine camera trapping data from SE Alaska, using state-space buffers from 10 up to 50 km. Each analysis was based on 3 chains, 12,000 iterations, 2,000 burn-in, for a total of 30,000 posterior samples.

	σ			N			D		
Buffer	**Mean**	**SD**	**n.eff**	**Mean**	**SD**	**n.eff**	**Mean**	**SD**	**n.eff**
10	0.65	0.06	1,800	39.63	6.70	7,100	5.97	1.00	7,100
15	0.64	0.06	510	48.77	9.19	3,300	5.78	1.09	3,300
20	0.64	0.06	1,200	59.84	11.89	20,000	5.77	1.15	20,000
25	0.64	0.05	3,600	72.40	14.72	2,700	5.79	1.18	2,700
30	0.63	0.05	5,600	86.42	17.98	3,900	5.82	1.21	3,900
35	0.63	0.05	4,500	101.79	21.54	30,000	5.85	1.24	30,000
40	0.64	0.05	410	118.05	26.17	410	5.87	1.30	450
45	0.64	0.05	10,000	134.43	28.68	3,300	5.83	1.24	3,300
50	0.63	0.05	4,700	151.61	31.65	3,400	5.79	1.21	3,400

5.9.3 Summary of the wolverine analysis

We see that the estimated density is roughly consistent as we increase the state-space buffer from 15 to 55 km. We do note that the data augmentation parameter ψ (and, correspondingly, N) increases with the size of the state-space in accordance with the relationship $E(N) = D \times A$. However, density is more or less constant as we increase the size of the state-space beyond a certain point. For the 10 km state-space buffer, we see a slight effect on the posterior distribution of D because the state-space is not sufficiently large. The full results from the analysis based on a 20 km state-space buffer are given in Table 5.5.

Our point estimate of wolverine density from this study, using the posterior mean from the state-space based on the 20 km buffer, is approximately 5.77 individuals/ 1,000 km^2 with a 95% posterior interval of [3.86, 8.29]. Density is estimated imprecisely, which might not be surprising given the low sample size ($n = 21$ individuals!). This seems to be a basic feature of carnivore studies although it should not (in

Table 5.5 Posterior summaries of SCR model parameters for the wolverine camera trapping data from SE Alaska. The model was run with the trap array centered in a state-space with a 20 km rectangular buffer.

Parameter	**Mean**	**SD**	**2.5%**	**25%**	**50%**	**75%**	**97.5%**	**Rhat**
N	59.84	11.89	40.00	51.00	59.00	67.00	86.00	1
D	5.77	1.15	3.86	4.92	5.69	6.46	8.29	1
α_1	1.26	0.21	0.87	1.11	1.25	1.40	1.71	1
p_0	0.06	0.01	0.04	0.05	0.06	0.06	0.08	1
σ	0.64	0.06	0.54	0.60	0.63	0.67	0.76	1
ψ	0.20	0.05	0.12	0.17	0.20	0.23	0.30	1

our view) preclude the study of their populations by capture-recapture nor attempts to estimate density or vital rates.

It is worth thinking about this model, and these estimates, computed under a rectangular state-space, roughly centered over the trapping array (Figure 5.4). Does it make sense to define the state-space to include, for example, ocean? What are the possible consequences of this? What can we do about it? There's no reason at all that the state-space has to be a regular polygon—we defined it as such here strictly for convenience and for ease of implementation in **WinBUGS**, where it enables us to specify the prior for the activity centers as uniform priors for each coordinate. While it would be possible to define a more realistic state-space using some general polygon GIS coverage, it might take some effort to implement that in the **BUGS** language although it is not difficult to devise custom MCMC algorithms to do that (see Chapter 17). Alternatively, we recommend using a discrete representation of the state-space—i.e., approximate $\mathcal{S}$ by a grid of G points. We discuss this in Section 5.10.

5.9.4 Wolverine space usage

The parameter α_1 is related to the home range radius (Section 5.4). For the Gaussian model we interpret the scale parameter σ, related to α_1 by $\alpha_1 = 1/(2\sigma^2)$, as the radius of a bivariate normal model of space usage. In this case $\sigma = 0.64$ standardized units (10 km), which corresponds to $0.64 \times 10 = 6.4$ km. It can be argued then that 95% of space used by an individual is within $6.4 \times \sqrt{5.99} = 15.66$ km of the home range center. The effective "home range area" is then the area of this circle, which is $\pi \times 15.66^2 = 770.4$ km^2. Using our handy function `hra` we do this:

```
hra(pGauss1,parms=c(-2,1/(2*.64*.64)),xlim=c(-1,7),ylim=c(-1,7))

[1] 7.731408
```

which is in units of 100 km^2, or 773.1 km^2. The difference in this case is due to numerical approximation of our all-purpose tool `hra`. This home range size is relatively huge for measured home ranges of wolverines, which range between 100 and 535 km^2 (Whitman et al., 1986).

Royle et al. (2011b) reported estimates for σ in the range 6.3–9.8 km depending on the model, which isn't too different from what we report here.[1] However, these estimates are larger than the typical home range sizes suggested in the literature. One possible explanation is that if a wolverine is using traps as a way to get yummy chicken, so it's moving from trap to trap instead of adhering to "normal" space usage patterns, then the implied home range size might not be biologically meaningful. Thus, interpretation of detection models in terms of home range area depends on

[1] Royle et al. (2011b) expressed the model as $\text{cloglog}(p_{ij}) = \alpha_0 - (1/\sigma^2) * d_{ij}^2$, but the estimates of σ reported in their Table 5.2 are actually based on the model according to $\text{cloglog}(p_{ij}) = \alpha_0 - \frac{1}{2\sigma^2} * d_{ij}^2$, and so the estimates of σ they report in units of km are consistent with what we report here except based on the complementary log-log (Gaussian hazard) model, instead of the Gaussian encounter probability model.

some additional context or assumptions, such as that traps don't effect individual space usage patterns. As such, we caution against direct biological interpretations of home range area based on σ, although SCR models can be extended to handle more general, non-Euclidean, patterns of space usage (Chapters 12 and 13).

We can calibrate the size of the state-space by looking at the estimated home range radius of the species. We should target a buffer of width 2 to 3 $\times$ σ in order that the probability of encountering an individual is very close to 0 beyond the prescribed state-space. Essentially, by specifying a state-space, we're setting $p = 0$ for individuals beyond the prescribed state-space. For the wolverine data, with σ in the range of 6–9 km, a state-space buffer of 20 km is sufficiently large.

5.10 Using a discrete habitat mask

The SCR model developed previously in this chapter assumes that individual activity centers are distributed uniformly over the prescribed state-space. Clearly, this will not always be a reasonable assumption. In Chapter 11, we develop models that allow explicitly for non-uniformity of the activity centers by modeling covariate effects on density. A simplistic method of affecting the distribution of activity centers, which we address here, is to modify the shape and organization of the state-space explicitly. For example, we might be able to classify the state-space into distinct blocks of habitat and non-habitat. In that case we can remove the non-habitat from the state-space and assume uniformity of the activity centers over the remaining portions judged to be suitable habitat. There are several ways to approach this: We can use a grid of points to represent the state-space, i.e., the set of coordinates $\mathbf{x}_1, \ldots, \mathbf{x}_G$, and assign equal probabilities to each possible value. Alternatively, we can retain the continuous formulation of the state-space but attempt to describe constraints analytically, or we can use polygon clipping methods to enforce constraints on the state-space in the MCMC analysis. We focus here on the formulation of the basic SCR model in terms of a discrete state-space but, in Chapter 17, we demonstrate the latter approach based on using polygon operations to define an irregular state-space. Use of a discrete state-space can be computationally expensive in **WinBUGS**. That said, it isn't too difficult to perform the MCMC calculations in **R** (discussed in Chapter 17). The **R** package `SPACECAP` (Gopalaswamy et al., 2012a) was motivated as a discrete-space implementation of an MCMC algorithm for basic SCR models.

While clipping out non-habitat seems like a good idea, we think investigators should go about this very cautiously. We might prefer to do it when non-habitat represents a clear-cut restriction on the state-space such as a reserve boundary or a lake, ocean, or river. But, having the capability to do this also causes people to start defining "habitat" vs. "non-habitat" based on their understanding of the system whereas it is usually unknown whether the animal being studied has the same understanding. Moreover, differentiating the landscape by habitat or habitat quality must affect the geometry and morphology of home ranges (see Chapter 13) much more so than the plausible locations of activity centers. That is, a home range centroid could, in actual fact, occur in a shopping mall parking lot if there is pretty good habitat around the shopping mall,

so there is probably no sense precluding it as the location for an activity center. It would generally be better to include some definition of habitat quality in the model for the detection probability (Royle et al., 2013), which we address in Chapters 12 and 13.

5.10.1 Evaluation of coarseness of habitat mask

The coarseness of the state-space should not really have much of an effect on estimates if the grain is sufficiently fine relative to typical animal home range sizes. Why is this? We have two analogies that can help us understand this concept. First is the relationship to model M_h. As noted in Section 5.3.2 above, we can think about SCR models as a type of finite mixture (Norris and Pollock, 1996; Pledger, 2004) where we are fortunate to be able to obtain direct information about which group individuals belong to (group being location of activity center). In the standard finite-mixture models we typically find that a small number of groups (e.g., 2 or 3 at the most) can explain high levels of heterogeneity and are adequate for most data sets of small to moderate sample sizes. We therefore expect a similar effect in SCR models when we discretize the state-space. We can also think about discretizing the state-space as being related to numerical integration where we find (see Chapter 6) that we don't need a very fine grid of support points to evaluate the integral to a reasonable level of accuracy. We demonstrate this here by reanalyzing simulated data using a state-space defined by a different number of support points. We provide an **R** script called `SCR0bayesDss` in the **R** package `scrbook`. For this comparison we generated the actual activity centers as a continuous random variable and thus the discrete state-space is, strictly speaking, an approximation to truth. That said, we regard all state-space specifications as approximations to truth in the sense that they represent a component of the SCR model.

As with our **R** function `SCR0bayes`, the modification `SCR0bayesDss` will use either **WinBUGS** or **JAGS**. In addition, it requires a grid resolution argument (`ng`) which is the dimension of 1 side of a square state-space. To execute this function we do, for example:

```
> library(scrbook)
> data <- simSCR0(discard0=TRUE,rnd=2013)   # Generate data set

# Run with JAGS
> out1 <- SCR0bayesDss(data,ng=8,M=200,engine="jags",ni=2000,nb=1000)

# Run with WinBUGS
> out2 <- SCR0bayesDss(data,ng=8,M=200,engine="winbugs",ni=2000,nb=1000)
```

We fit this model to the same simulated data set for 6×6, 9×9, 12×12, and 15×15 state-space grids. For **WinBUGS**, we used 3 chains of 5,000 total length 1,000 burn-in, which yields 12,000 total posterior samples. Summary results are shown in Table 5.6. The results are broadly consistent except for the 6×6 case. We see that the runtime increases with the size of the state-space grid (not unexpected), such that we imagine it would be impractical to run models with more than a few hundred state-space grid points. We found (not shown here) that the runtime of **JAGS** is much faster and, furthermore, relatively *constant* as we increase the grid size. We

Table 5.6 Comparison of the effect of state-space grid coarseness on estimates of N for a simulated data set with true $N = 100$. Posterior summaries and runtime are given. Results obtained using **WinBUGS** run from `R2WinBUGS`.

Grid Size	Mean	SD	NaiveSE	Time-seriesSE	Runtime (s)
6×6	111.670	16.614	0.152	0.682	2,274
9×9	114.230	17.991	0.164	0.833	4,300
12×12	115.981	17.384	0.159	0.763	7,100
15×15	115.379	17.937	0.164	0.832	13,010

suspect that **WinBUGS** is evaluating the full conditional for each activity center at all possible values of the discrete grid whereas it may be that **JAGS** is evaluating the full conditional only at a subset of values or perhaps using previous calculations more effectively. While this might suggest that one should always use **JAGS** for this analysis, we found in our analysis of the wolverine data (next section) that **JAGS** could be extremely sensitive to starting values, producing MCMC algorithms that often simply do not work for some problems, so be careful when using **JAGS**. The performance of either should improve if we compute the full distance matrix outside of **BUGS** and pass it as data, although we haven't fully evaluated this approach.

5.10.2 Analysis of the wolverine camera trapping data

We reanalyzed the wolverine data using discrete state-space grids with points spaced by 2, 4, and 8 km (see Figure 5.5). These were constructed from a 40 km buffered

FIGURE 5.5

Three habitat mask (state-space) grids used in the comparison of the effect of pixel size on the estimated density surface of wolverines. The three cases are 2-km (left), 4-km (center), and 8-km (right) spacing of state-space points, extending 40 km from the vicinity of the trap array. Larger dots are the camera trap locations.

state-space, deleting the points over water (see Royle et al., 2011b). Our interest in doing this was to evaluate the relative influence of grid resolution on estimated density. The coarser grids will be more efficient from a computational standpoint and so we would prefer to use them, but only if there is no strong influence on estimated density. The posterior summaries for the three habitat grids are given in Table 5.7. We see that the density estimates are quite a bit larger than obtained in our analysis (Table 5.4) based on a rectangular, continuous state-space. We also see that there are slight differences depending on the resolution of the state-space grid. Interestingly, the effectiveness of the MCMC algorithms, as measured by effective sample size (`n.eff`), is pretty remarkably different. Furthermore, the finest grid resolution (2 km spacing) took about 6 days to run and thus, it would not be practical for large problems or with many models.

Table 5.7 Posterior summaries for the wolverine camera trapping data, using model SCR0, with a Gaussian hazard encounter probability model, and a discrete habitat mask of three different resolutions: 2, 4, and 8 km. Parameters are λ_0 = baseline encounter rate, $p_0 = 1 - \exp(-\lambda_0)$, σ is the scale parameter of the Gaussian kernel, ψ is the data augmentation parameter, N and D are population size and density, respectively. Models fitted using **WinBUGS**, 3 chains, each with 11,000 iterations (first 1,000 discarded) producing 30,000 posterior samples.

	Mean	SD	2.5%	25%	50%	75%	97.5%	Rhat	n.eff
2 km spacing									
N	86.56	16.94	57.00	75.00	85.00	97.00	124.00	1.00	510
D	8.78	1.72	5.78	7.60	8.62	9.83	12.57	1.00	510
λ_0	0.05	0.01	0.04	0.04	0.05	0.06	0.07	1.01	320
p_0	0.05	0.01	0.03	0.04	0.05	0.05	0.06	1.01	320
σ	0.62	0.05	0.54	0.59	0.62	0.65	0.73	1.01	160
ψ	0.43	0.09	0.27	0.37	0.43	0.49	0.63	1.00	560
4 km spacing									
N	89.25	17.44	59.00	77.00	88.00	100.00	127.00	1.00	1100
D	9.01	1.76	5.96	7.77	8.88	10.10	12.82	1.00	1100
λ_0	0.05	0.01	0.04	0.05	0.05	0.06	0.07	1.00	2500
p_0	0.05	0.01	0.03	0.04	0.05	0.05	0.07	1.00	2500
σ	0.61	0.04	0.53	0.58	0.61	0.64	0.71	1.00	1600
ψ	0.45	0.09	0.28	0.38	0.44	0.50	0.64	1.00	1300
8 km spacing									
N	83.18	16.14	56.00	72.00	82.00	93.00	119.00	1.00	700
D	8.28	1.61	5.57	7.17	8.16	9.26	11.84	1.00	700
λ_0	0.05	0.01	0.03	0.04	0.05	0.05	0.06	1.00	560
p_0	0.05	0.01	0.03	0.04	0.04	0.05	0.06	1.00	560
σ	0.68	0.05	0.59	0.64	0.67	0.71	0.77	1.01	220
ψ	0.42	0.09	0.26	0.36	0.41	0.47	0.61	1.00	940

5.11 Summarizing density and activity center locations

One of the most useful aspects of SCR models is that they are parameterized in terms of individual locations—i.e., *where* each individual lives—and, thus, we can compute many useful and interesting summaries of the activity centers using output from an MCMC simulation, including maps of density (the number of activity centers per unit area), estimates of N for any well-defined polygon, or estimates of where the activity centers for specific individuals reside. In Bayesian analysis by MCMC, obtaining such summaries entails no added calculations, because we need only post-process the output for the individual activity centers to obtain the desired summaries (Section 3.5.4). We demonstrate that in this section. Note that you have to be sure to retain the MCMC history for the **s** variables and the data augmentation variables z in order to do the following analyses.

5.11.1 Constructing density maps

Because SCR models are spatially explicit, it is natural to want to summarize the results of fitting a model by producing a map of density. Using Bayesian analysis by MCMC, it is most easy to make a map of *realized* density. We can do this by tallying up the number of activity centers $\mathbf{s}_i$ in pixels of arbitrary size and then producing a nice multi-color spatial plot of the result. Specifically, let $B(\mathbf{x})$ indicate a pixel centered at $\mathbf{x}$, then

$$N(\mathbf{x}) = \sum_{i=1}^{M} I(\mathbf{s}_i \in B(\mathbf{x}))$$

(here, $I(arg)$ is the indicator function which evaluates to 1 if arg is true, and 0 otherwise) is the population size of pixel $B(\mathbf{x})$, and $D(\mathbf{x}) = N(\mathbf{x})/\|B(\mathbf{x})\|$ is the local density. Note that these $N(\mathbf{x})$ parameters are just "derived parameters" as we normally obtain from posterior output using the appropriate Monte Carlo average (see Chapter 3).

One thing to be careful about, in the context of models in which N is unknown, is that, for each MCMC iteration m, we only tabulate those activity centers which correspond to individuals in the sampled population, i.e., for which the data augmentation variable $z_i = 1$. In this case, we take all of the output for MCMC iterations $m = 1, 2, \ldots,$ `niter` and compute this summary:

$$N(\mathbf{x}, m) = \sum_{i:z_{i,m}=1} I(\mathbf{s}_{i,m} \in B(\mathbf{x})).$$

Thus, $N(\mathbf{x}, 1)$, $N(\mathbf{x}, 2)$, . . ., is the Markov chain for parameter $N(\mathbf{x})$. In what follows we will provide a set of **R** commands for doing this calculation and making a basic image plot from the MCMC output:

Step 1: Define the center points of each pixel $B(\mathbf{x})$, or point at which local density will be estimated:

```
> xg <- seq(xlim[1],xlim[2], ,50)
> yg <- seq(ylim[1],ylim[2], ,50)
```

Step 2: Extract the MCMC histories for the activity centers and the data augmentation variables. Note that these are each $N \times$ `niter` matrices. Here we do this assuming that **WinBUGS** was run producing the **R** object named `out`:

```
> Sxout <- out$sims.list$s[, ,1]
> Syout <- out$sims.list$s[, ,2]
> z <- out$sims.list$z
```

Step 3: We associate each coordinate with the proper pixel using the **R** command `cut`. Note that we keep only the activity centers for which $z = 1$ (i.e., individuals that belong to the population of size N):

```
> Sxout <- cut(Sxout[z==1], breaks=xg, include.lowest=TRUE)
> Syout <- cut(Syout[z==1], breaks=yg, include.lowest=TRUE)
```

Step 4: Use the `table` command to tally up how many activity centers are in each $B(\mathbf{x})$:

```
> Dn <- table(Sxout, Syout)
```

Step 5: Use the `image` command to display the resulting matrix:

```
> image(xg, yg, Dn/nrow(z), col=terrain.colors(10))
```

We'll apply this analysis to create a density map from the wolverine MCMC output shortly. It is worth emphasizing here that density maps will not usually appear uniform despite that we have assumed that activity centers are uniformly distributed. This is because the observed encounters of individuals provide direct information about the location of the $i = 1, 2, \ldots, n$ activity centers and thus their "estimated" locations will be affected by the observations. In a limiting sense, were we to sample space intensely enough, every individual would be captured a number of times and we would have considerable information about all N point locations. Consequently, the uniform prior would have almost no influence at all on the estimated density surface in this limiting situation. Thus, in practice, the influence of the uniformity assumption decreases as the fraction of the population encountered, and the total number of encounters per individual, increases.

On the non-intuitiveness of `image`—the **R** function `image`, invoked for a matrix M by `image(M)`, might not be very intuitive to some—it plots $M[1, 1]$ in the lower left corner. If you want $M[\,]$ to be plotted "as you look at it" then $M[1, 1]$ should be in the upper left corner. We have a function, `rot` which does that. If you do `image(rot(M))` then it puts it on the monitor as if it was a map you were looking at. You can always specify the x and y labels explicitly as we did above.

Spatial dot plots—A cruder version of the density map can be made using our "spatial dot map" function `spatial.plot` (in `scrbook`). This function requires,

as input, point locations and the value to be displayed. A simplified version of this function is as follows:

```
> spatial.plot <- function(x,y){
    nc <- as.numeric(cut(y,20))
    plot(x,pch=" ")
    points(x,pch=20,col=topo.colors(20)[nc],cex=2)
    image.scale(y,col=topo.colors(20))
  }
#
# To execute the function do this:
#
> spatial.plot(cbind(xg,yg), Dn/nrow(z))
```

5.11.2 Wolverine density map

We return to the wolverine study from SE Alaska (Figure 5.4) and we produce a density map of wolverines from that analysis. We include the function `SCRdensity`, which requires a specific data structure as shown below. In particular, we have to package up the MCMC history for the activity centers and the data augmentation variables z into a list. This also requires that we add those variables to the parameters-to-be-monitored list when we pass things to **BUGS**.

We used the posterior output from the wolverine model fitted previously to compute a relatively coarse version of a density map, using 100 pixels in a 10×10 grid (Figure 5.6 top panel) and using 900 pixels arranged in a 30×30 grid (Figure 5.6 lower panel) for a fine-scale map. The **R** commands for producing such a plot (for a short MCMC run) are as follows:

```
> library(scrbook)
> data(wolverine)
> traps <- wolverine$wtraps
> y3d <- SCR23darray(wolverine$wcaps,wolverine$wtraps)

# This takes 341 seconds on a standard CPU circa 2011
> out <- wolvSCR0(y3d,traps,nb=1000,ni=2000,buffer=1,M=100,keepz=TRUE)

> Sx <- out$sims.list$s[, ,1]
> Sy <- out$sims.list$s[, ,2]
> z <- out$sims.list$z
> obj <- list(Sx=Sx,Sy=Sy,z=z)
> tmp <- SCRdensity(obj,nx=10,ny=10,scalein=100,scaleout=100)
```

In these figures density is expressed in units of individuals per 100 km^2, while the area of the pixels is about 103.7 km^2 and 11.5 km^2, respectively. That calculation is based on:

```
> total.area <- (ylim[2]-ylim[1])*(xlim[2]-xlim[1])*100
> total.area/(10*10)
[1] 103.7427
> total.area/(30*30)
[1] 11.52697
```

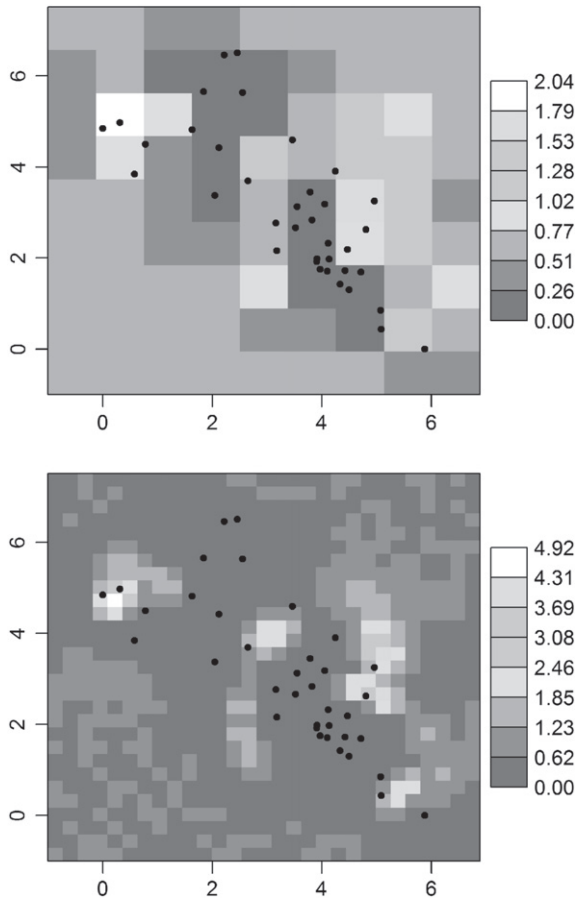

FIGURE 5.6

Density of wolverines (individuals per 100 km^2) in SE Alaska in 2007 based on model SCR0. Map grid cells are about 103.7 km^2 (top panel) and 11.5 km^2 (bottom panel) in area. Dots are the trap locations.

A couple of things are worth noting: First, as we move away from "where the data live" (away from the trap array) we see that the density approaches the mean density. This is a property of the estimator as long as the detection function decreases sufficiently rapidly as a function of distance. Relatedly, it is also a property of statistical smoothers such as splines, kernel smoothers, and regression smoothers—predictions tend toward the global mean as the influence of data diminishes. Another way to think of it is that it is a consequence of the prior, which imposes uniformity, and as you get far away from the data, the predictions tend to the expected constant density under the prior. Another thing to note about this map is that density is not 0 over water (although the coastline is not shown). This might be perplexing to some who might be fairly

certain that wolverines do not like water. However, there is nothing about this model that recognizes water from non-water and so the model predicts over water *as if* it were habitat similar to that within which the array is nested. But, all of this is OK as far as estimating density goes and, furthermore, we can compute valid estimates of N over any well-defined region, which presumably wouldn't include water if we so wished. Alternatively, areas covered by water could be masked out, which we discussed in Section 5.10.2.

5.11.3 Predicting where an individual lives

The density maps in the previous section show the expected number of individuals per unit area. A closely related problem is that of producing a map of the probable location of a specific individual's activity center. For any observed encounter history, we can easily generate a posterior distribution of $\mathbf{s}_i$ for individual i. In addition, for an individual that is *not* captured, we can use the MCMC output to produce a corresponding plot of where such an individual might live, say $\mathbf{s}_{n+1}$. Obviously, all such uncaptured individuals (for $i = n + 1, \ldots, N$) should have the same posterior distribution. To illustrate, we show the posterior distribution of $\mathbf{s}_1$, the activity center for the individual labeled 1 in the data set, in Figure 5.7. This individual was captured a single time at trap 30, which is circled in Figure 5.7. We see that the posterior

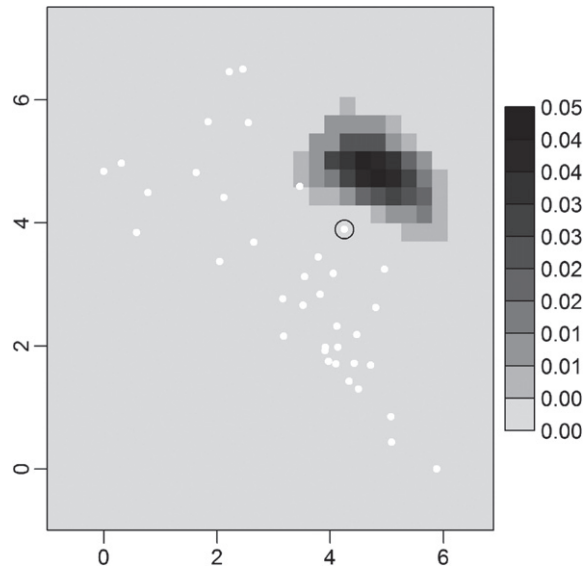

FIGURE 5.7

Posterior probability distribution of $\mathbf{s}_1$, the activity center for individual 1 in the wolverine data set. This individual was captured a single time in one trap (trap 30) which is circled. White dots are trap locations.

distribution is affected by traps of capture *and* traps of non-capture in fairly intuitive ways. In particular, because there are other traps in close proximity to trap 30 in which individual 1 was *not* captured, the model pushes its activity center away from the trap array. The help file for `SCRdensity` shows how to calculate Figure 5.7.

5.12 Effective sample area

One of the key issues in using ordinary capture recapture models that we've brought up over and over again is the issue that the area sampled by a trapping array is unknown—in other words, the N that is estimated by capture-recapture models does not have an explicit region of space associated with it. Classically, this has been addressed in the ad hoc way of prescribing an area that contains the trap array, usually by adding a buffer of some width, which is not estimated as part of the capture-recapture model. In SCR models we avoid the problem of not having an explicit linkage between N and "area," by prescribing explicitly the area within which the underlying point process is defined—the state-space of the point process.

However, it is possible to provide a characterization of effective sampled (or sample) area (ESA) under any SCR model. This is directly analogous to the calculation of "effective strip width" in distance sampling (Buckland et al., 2001; Borchers et al., 2002). The conceptual definition of ESA follows from equating density to "apparent density"—ESA is the magic number that satisfies that equivalence:

$$D = N/A = n/ESA.$$

In other words, the ratio of N to the area of the state-space should be equal to the ratio of the observed sample size n to this number ESA. Both of these should equal density. So, to compute ESA for a model, we substitute $\mathbb{E}(n)$ for n into the above equation, and solve for ESA, to get:

$$ESA = \mathbb{E}(n)/D.$$

Our following development assumes that D is constant, but these calculations can be generalized to allow for D to vary spatially. Imagine our habitat mask for the wolverine data, or the bins we just used to produce a density map, then we can write $\mathbb{E}(n)$ according to

$$\mathbb{E}(n) = \sum_s \Pr(\text{encounter}|\mathbf{s})\mathbb{E}(N(\mathbf{s})),$$

where if we prefer to think of this more conceptually, we could replace the summation with an integration (which, in practice, we would just replace with a summation, and so we just begin there). In this expression, note that $\mathbb{E}(N(\mathbf{s}))$ is the expected population size at pixel $\mathbf{s}$ which is the density times the area of the pixel, i.e., $\mathbb{E}(N(\mathbf{s})) = D \times a$. Therefore,

$$\mathbb{E}(n) = D \times a \times \sum_s \Pr(\text{encounter}|\mathbf{s}),$$

and (plugging this into the expression above for ESA)

$$ESA = \frac{D \times a \times \sum_s \Pr(\text{encounter}|\mathbf{s})}{D}.$$

We see that D cancels and we have ESA $= a \times \sum_s \Pr(\text{encounter}|\mathbf{s})$. So what you have to do here is substitute in Pr(encounter|**s**) and just sum them up over all pixels. For the Bernoulli model SCR0

$$\Pr(\text{encounter}|\mathbf{s}) = 1 - (1 - p(\mathbf{s}))^K$$

with slight modifications when encounter probability depends on covariates. Thus,

$$ESA = a \sum_s (1 - (1 - p(\mathbf{s}))^K). \tag{5.12.1}$$

Clearly, the calculation of ESA is affected by the use of a habitat mask, because the summation in Eq. (5.12.1) only occurs over pixels that define the state-space.

For the wolverine camera trapping data, we used the 2×2 km habitat mask and the posterior means of p_0 and σ (see Section 5.10.2) to compute the probability of encounter for each **s** of the mask points. The result is shown graphically in Figure 5.8. The ESA is the sum of the values plotted in that figure multiplied by 4, the area of each

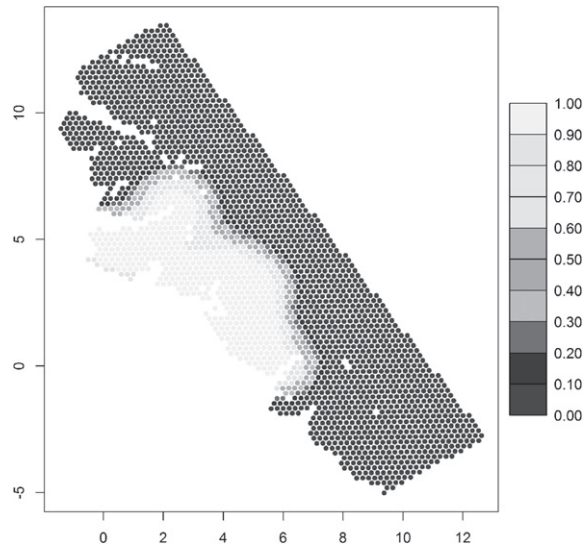

FIGURE 5.8

Probability of encounter used in computing effective sampled area for the wolverine camera trapping array, using the parameter estimates (posterior means) for the 2×2 km habitat mask.

pixel. For the wolverine study, the result is 2507.152 km^2. We note that the probability of encounter declines rapidly to 0 as we move away from the periphery of the camera trap array, indicating the state-space constructed from a 40 km buffered trap array was indeed sufficient for the analysis of these data. And, grid cells contribute less to ESA as their distance to the array increases. An **R** script for producing this figure is in the `wolvESA` function of the `scrbook` package.

5.13 Summary and outlook

In this chapter, we introduced the simplest SCR model—"model SCR0"—which is an ordinary capture-recapture model like model M_0, but augmented with a set of latent individual effects, $\mathbf{s}_i$, which relate encounter probability to some sense of individual location using a covariate, "distance," from $\mathbf{s}_i$ to each trap location. Thus, individuals in close proximity to a trap will have a higher probability of encounter, and *vice versa*. The explicit modeling of individual locations and distance in this fashion resolves classical problems related to estimating density using non-spatial models: unknown sample area for density estimation, and heterogeneous encounter probability due to variable exposure to traps.

SCR models are closely related to classical individual covariate models ("model M_x," as introduced in Chapter 4), but with imperfect information about the individual covariate. Therefore, they are also not too dissimilar from standard GLMMs used throughout statistics and, as a result, we find that they are easy to analyze using standard MCMC methods encased in black boxes such as **WinBUGS** or **JAGS**. We will also see that they are easy to analyze using likelihood methods, which we address in Chapter 6.

Formal consideration of the collection of individual locations ($\mathbf{s}_1, \ldots, \mathbf{s}_N$) is fundamental to all models considered in this book. In statistical terminology, we think of this collection of points as a realization of a point process. Because SCR models formally link individual encounter history data to an underlying point process, we can obtain formal inferences about the point process. For example, we showed how to produce a density map (Figure 5.6), or even a probability map for an individual's home range center (Figure 5.7). We can also use SCR models as the basis for doing more traditional point process analyses, such as testing for "complete spatial randomness" (CSR) (see Chapter 8), and computing other point process summaries (Illian et al., 2008).

Part of the promise and ongoing challenge of SCR models is to develop models that reflect interesting biological processes, for example interactions among points or temporal dynamics in point locations. In this chapter we considered the simplest possible point process model in which points are independent and uniformly ("randomly") distributed over space. Despite the simplicity of this model, it should suffice in many applications of SCR models, although we do address generalizations in later chapters. Moreover, even though the *prior* distribution on the point locations is uniform, the realized pattern may deviate markedly from uniformity as the observed encounter

data provide information to impart deviations from uniformity. Thus, estimated density maps will typically appear distinctly non-uniform (as we saw in the wolverine example). In applications of the basic SCR model, we find that this simple *a priori* model can effectively reflect or adapt to complex realizations of the underlying point process. For example, if individuals are highly territorial then the data should indicate this in the form of individuals not being encountered in the same trap, and the resulting posterior distribution of point locations should therefore reflect non-independence. Obviously, the complexity of posterior estimates of the point pattern will depend on the quantity of data, both number of individuals and captures per individual. Because the point process is such an integral component of SCR models, the state-space of the point process plays an important role in developing SCR models. As we emphasized in this chapter, the state-space is part of the model. And, under certain circumstances, it can have an influence on parameter estimates and other inferences, such as model selection (see Chapter 8).

One concept we introduced in this chapter, which has not been discussed much in the literature on SCR models, is the manner in which the encounter probability model relates to a model of space usage by individuals. The standard SCR models of encounter probability can all be motivated as simplistic models of space usage and movement, in which individuals make random use decisions from a probability distribution proportional to the encounter probability model. This simple formulation suggests a direct extension to produce more realistic models, which we discuss in Chapter 13. We consider a number of other important extensions of the basic SCR model in later chapters. For example, we consider models that include covariates that vary by individual, trap, or over time (Chapter 7), spatial covariates on density (Chapter 11), and open populations (Chapter 16), and methods for model assessment and selection (Chapter 8) among other topics. We also consider technical details of maximum likelihood (Chapter 6) and Bayesian (Chapter 17) estimation, so that the interested reader can develop or extend methods to suit their own needs.

Non Print Items

Abstract: In this chapter we investigate the basic spatial capture-recapture model, which we refer to as "model SCR0." The model is a hierarchical model composed of two conditionally-related components: (1) a spatial point process describing the number and location of animal activity centers and (2) an observation model specifying capture or encounter probability as a function of the distance between individual activity centers and traps. As with all models, it includes several assumptions that must be understood and critically evaluated. We list the basic assumptions of SCR models and we provide some basic tools for simulating and analyzing spatial capture-recapture data in R, so that the assumptions and properties of the model can be fully appreciated. The chapter also focuses on practical issues such as how to format data and how to choose certain model settings. For inference, we focus on Bayesian methods using Markov chain Monte Carlo simulation and data augmentation. An example analysis is presented using wolverine data collected in southern Alaska, and this example demonstrates how to summarize posterior output for the purposes of producing posterior density maps and computing derived quantities such as the effective sample area.

Keywords: Activity center, Bayesian inference, Data format, Density map, Effective sample area, Habitat mask, Observation model, Sampling design, Spatial point process