CHAPTER

9

Estimation of Survival and Movement from Capture–Recapture Data Using Multistate Models

OUTLINE

Bayesian Population Analysis using WinBUGS
DOI: 10.1016/B978-0-12-387020-9.00009-2

9.1 INTRODUCTION

When an individual is encountered, it is often possible to assign it to a certain *state*. A state may be described as a categorical individual covariate that can change over time. A state may be a geographical location, a class of reproductive success, or a disease status, to name just a few. Stratification of individuals according to their state then allows estimation of state-dependent survival and recapture probabilities, as well as state-transition probabilities. In this chapter, we extend the seminal CJS model from Chapter 7 to more than a single state. Perhaps not surprisingly, the resulting model is called a multistate (or historically also multistrata) model (Lebreton et al., 2009).

Depending on how the states are defined, a vast array of research questions can be addressed using this broad class of capture–recapture models. For example, if states represent geographic locations, we can estimate movement probabilities among locations, that is, dispersal. We may test whether movement is symmetrical or is related to habitat quality of a location. If states represent "not infected by a disease" and "infected by a disease", we may study whether survival is related to disease status or identify conditions that are positively correlated with the infection probability, that is, the transition probability from "not infected" to "infected". State definitions may also be less obvious. States could be age classes, which allows estimation of age-dependent survival probabilities, they could be "dead" and "alive" enabling modeling of mark-recovery data or the joint analysis of capture–recapture and mark-recovery data, they could be "present" and "absent" allowing estimation of temporary emigration, or they could be "captured" and "not captured", which enables fitting of models with immediate trap response. In fact, multistate capture–recapture models are extremely useful and should be used for many interesting ecological

questions. Multistate models represent a unification of many different kinds of capture–recapture model (Lebreton et al., 1999, 2009).

There are many applications of multistate models in the frequentist framework. As an example, see the seminal papers by Arnason (1972, 1973), Hestbeck et al. (1991), Schwarz et al. (1993), Lebreton and Pradel (2002) and the review by Lebreton et al. (2009). Much fewer authors have adopted the Bayesian framework so far (e.g., Dupuis, 1995; King and Brooks, 2002; Gimenez et al., 2003; King and Brooks, 2004; Clark et al., 2005; Dupuis and Schwarz, 2007; Calvert et al., 2009; Schofield et al., 2009; King et al., 2010; Schaub et al., 2010).

As the single-state capture–recapture and mark-recovery models in Chapters 7 and 8, multistate capture–recapture models can be analyzed either in a state-space modeling framework or using a multinomial likelihood. Frequentist analyses typically use the multinomial likelihood applied to the multistate m-array (Williams et al., 2002). This approach requires definition of all cell probabilities in the multistate m-array, which is straightforward when matrix multiplication is available in a software. Unfortunately, matrix multiplication is not available in WinBUGS (though it is in JAGS); hence, we will only use the state-space formulation here. It has the advantage that exactly the same concepts (modeling along the time and individual axis, with fixed and random effects) as introduced in Chapters 6 and 7 can be applied providing great flexibility in modeling. We will not repeat these concepts in this chapter anymore, but refer to Chapters 6 and 7 and to the exercises. The state equation of the state-space formulation describes the true development of the states, that is, it describes the state of an individual at time $t + 1$, given its state at time t. The observation equation maps the true state at time t on to the observed state. Since there are more than two possible true and observed states, the likelihood is not based on the Bernoulli distribution, as is the case with single-state capture–recapture models, but on the categorical distribution. The categorical distribution is the extension of a Bernoulli distribution to more than two categories. It is a special case of the multinomial distribution with trial size equal to 1.

For a description of the state process in a system with two geographic locations, let us assume that a marked individual is at location 1 (referred to as state 1) at time t. At the next sampling occasion, $t + 1$, this individual may still be alive, and at location 1 with probability $\Phi_{11,t}$, it may still be alive but has moved to location 2 (state 2) with probability $\Phi_{12,t}$, or it may be dead with probability $1 - \Phi_{11,t} - \Phi_{12,t}$. If the individual is still alive at time $t + 1$, it may again survive and move among locations. If the individual is at location 2 at time $t + 1$, then the corresponding probability to survive and move to location 1 is $\Phi_{21,t+1}$, the probability to survive and remain at location 2 is $\Phi_{22,t+1}$, and the probability to die is the complement

of the two, that is, $1 - \Phi_{21,t+1} - \Phi_{22,t+1}$. This is continued until the individual is either dead or a study ends. Once an individual is dead, its fate is no longer stochastic; the individual will remain dead with probability 1 until the end of the study. These probabilities define the state process, and we would like to know the parameters governing this process. Yet, inevitably, we cannot observe the state process without errors. Instead, we augment our system description with a component that describes the observation process. A marked individual that is alive at time t in state 1 may be encountered in state 1 with probability $p_{1,t}$, whereas an individual that is alive at time t and in state 2 may be encountered in state 2 with probability $p_{2,t}$. We can imagine that for each individual and occasion, a coin is tossed determining whether the individual is encountered (with probability $p_{1,t}$ or $p_{2,t}$) or not (with probability $1 - p_{1,t}$ or $1 - p_{2,t}$). We here assume the absence of state assignment errors; thus, an individual in state s can only be encountered in state s (or be not encountered), but not encountered in another state. In other words, we allow only false-negative, but no false-positive errors. Once dead, an individual can no longer be encountered. This completes our mostly verbal description of the observation process, which is conditional on the state process, and thus the model is conceptually hierarchical. The model is conditional on first capture, that is, we do not model and estimate the initial capture probability. In Fig. 9.1, the two processes are shown graphically for an example.

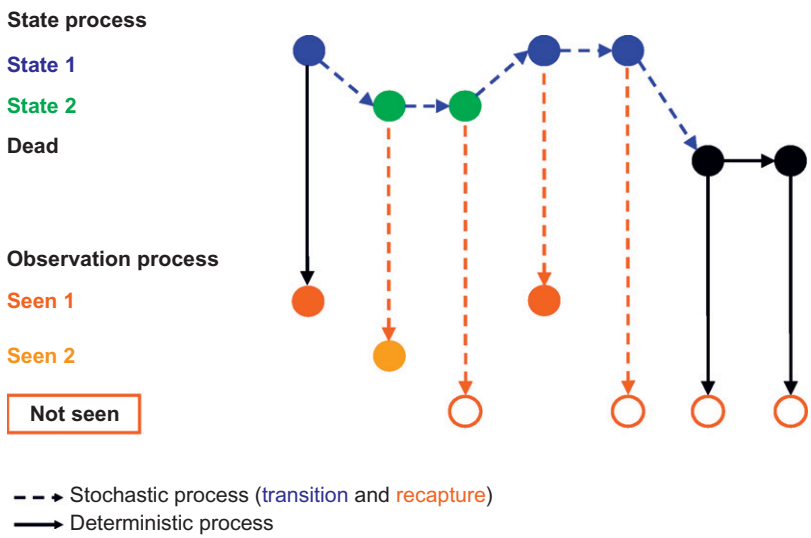

FIGURE 9.1 Example of the state and observation process of a marked individual over time for the multistate model. The sequence of true states in this individual is z = [1, 2, 2, 1, 1, 3, 3], and the observed capture-history is y = [1, 2, 0, 1, 0, 0, 0].

For an algebraic description of our two-location model, assume matrix $\mathbf{z}$ with element $z_{i,t}$, which indicates the true state of individual i at time t. States are numbered from 1 to S; S is the number of true states. Moreover, let us assume that vector fs_i denotes the true state of individual i at its first encounter. As we assume there are no state assignment errors, fs_i is identical to the observed state at first encounter. Furthermore, we define the four-dimensional state-transition matrix (Ω). The first and second dimension of Ω denote the states of departure and of arrival, respectively, the third dimension the individual (i), and the fourth dimension time (t). Element $\omega_{n,m,i,t}$ of Ω is the probability that individual i, which is in state n at time t, is in state m at $t+1$. The observation matrix (Θ) also has four dimensions, the first denotes the true state of an individual, the second the observed state, and the remaining two dimensions are for individual and time. The element $\varphi_{n,m,i,t}$ of Θ is the probability that individual i, which is in state n at time t, is observed in state m at time t. The first two dimensions of the state-transition matrix Ω are identical ($S \times S$), but this need not be the case for the observation matrix (Θ). The first dimension of Θ must also be S (the true number of states), but the second dimension is O (the number of observed states) which can be smaller or larger than S. As usual, the state-space model then consists of two model parts. The set of state equations is

$$z_{i,f_i} = fs_i$$

$$z_{i,t+1} \,|\, z_{i,t} \sim \text{categorical}(\Omega_{z_{i,t},1\ldots S,i,t}).$$

The first equation describes the state at the first encounter that is assumed to be assigned without error. Note that the argument of the categorical distribution is a vector of length S, that is, it is the complete row of the matrix Ω for given values of the first ($z_{i,t}$), third (i), and fourth (t) dimension. The second equation describes the development of the state membership over time. The observation equation links the true state with the observed state:

$$y_{i,t} \,|\, z_{i,t} \sim \text{categorical}(\Theta_{z_{i,t},1\ldots O,i,t})$$

where $\mathbf{y}$ is the observed multistate capture–recapture data. Similarly as before, the argument of the categorical distribution is a vector of length O, that is, it is the complete row of the matrix Θ for given values of the first ($z_{i,t}$), third (i), and fourth (t) dimension. The state and the observation process are both defined for $t \geq f_i$.

The multistate capture–recapture model makes similar assumptions as the CJS model (Chapter 7): the transition and observation probabilities must be the same for all individuals at a given occasion and state and individuals must be independent of each other. Individuals and states are recorded without error, and no marks must be lost. Of course, it is possible

to relax some of these assumptions by adopting a more general model. For example, by incorporating age or group effects, the assumption of equal transition and observation probabilities is no longer required for all individuals, but only for the individuals within an age class or group. Some of these assumptions can be tested with appropriate goodness-of-fit tests (Pradel et al., 2003; Choquet et al., 2009).

Because multistate capture–recapture models are so extremely flexible, there is a host of possible models that we might present. We have selected a few models that we think represent frequently encountered problems. In our presentation, we first give a description of the model by showing the first two dimensions of the state-transition (Ω) and observation (Θ) matrices. The formulation of these matrices requires the definition of exhaustive sets of true and observed states. Typically, state-transition probabilities may be composed of several parameters that we want to estimate separately, for example, state-specific survival and movement probabilities. These matrices form the heart of multistate modeling. They are essential for your understanding of this model class in principle and for the way in which these models are written in the BUGS language in particular. After presentation of the matrices, we either simulate data or use a real-data example and fit the model. Throughout, we denote state-transition matrices with rows representing states at occasion t and column states at occasion $t+1$. Sometimes states at time t (rows) are also called "states of departure", whereas states at time $t+1$ (columns) are named "states of arrival". For the observation matrix, true states are in rows and the observed states are in columns. In the list of true states, we always include the state "dead" and in the list of observed states the state "not seen/encountered". This is because both matrices need to be row stochastic, that is, the probabilities in a row must sum to 1. In Appendix 2, we show two additional useful multistate models along with the BUGS code for their analysis.

9.2 ESTIMATION OF MOVEMENT BETWEEN TWO SITES

9.2.1 Model Description

This is one of the simplest multistate models. Consider individuals that are marked and recaptured at two sites (A and B) and that there is movement between sites. Recapture is not perfect, and we want to estimate site-specific survival, as well as movement probabilities between sites. This is a common problem, and for this very situation, multistate capture–recapture models were originally developed (Arnason, 1972; Hestbeck et al., 1991; Brownie et al., 1993).

We start by defining the three true states: "alive at site A", "alive at site B", and "dead". Here is the state-transition matrix:

		True State at Time $t+1$		
		Site A	**Site B**	**Dead**
True State at Time t	**Site A**	$\phi_A(1-\psi_{AB})$	$\phi_A\psi_{AB}$	$1-\phi_A$
	Site B	$\phi_B\psi_{BA}$	$\phi_B(1-\psi_{BA})$	$1-\phi_B$
	Dead	0	0	1

or simply

$$\begin{array}{c} \\ \text{site A} \\ \text{site B} \\ \text{dead} \end{array} \begin{array}{c} \begin{array}{ccc} \text{site A} & \text{site B} & \text{dead} \end{array} \\ \begin{bmatrix} \phi_A(1-\psi_{AB}) & \phi_A\psi_{AB} & 1-\phi_A \\ \phi_B\psi_{BA} & \phi_B(1-\psi_{BA}) & 1-\phi_B \\ 0 & 0 & 1 \end{bmatrix} \end{array}$$

States are ordered from top to bottom and from left to right in the order as given above ("site A", "site B", and "dead"). For example, the probability of being in state "site B" at time $t+1$, given presence in state "site A" at time t, is $\phi_A\psi_{AB}$. The parameters in this matrix are the site-specific survival probabilities (ϕ_A, ϕ_B) and the movement probabilities (ψ_{AB}, ψ_{BA}).

When written in this way, we make a crucial assumption: survival from t to $t+1$ refers to the state in which the individual is at time t, and there is no mortality during movement. Thus, ψ_{AB} is the probability that an individual, which survived at site A from time t to $t+1$, moves to site B shortly before $t+1$. This is the typical way how the survival and movement parameters are defined. If we wish, we could also define that movement comes first and then the individual survives with the site-specific survival probability of the new site, in which case the transition matrix is this:

$$\begin{array}{c} \\ \text{site A} \\ \text{site B} \\ \text{dead} \end{array} \begin{array}{c} \begin{array}{ccc} \text{site A} & \text{site B} & \text{dead} \end{array} \\ \begin{bmatrix} (1-\psi_{AB})\phi_A & \psi_{AB}\phi_B & 1-\sum row_1 \\ \psi_{BA}\phi_A & (1-\psi_{BA})\phi_B & 1-\sum row_2 \\ 0 & 0 & 1 \end{bmatrix} \end{array}.$$

Joe and Pollock (2002) developed a general model in which the time of movement is a random variable with a known distribution.

Regardless of how the state matrix is defined, the list of observed states includes "seen in A", "seen in B", and "not seen". The observation matrix links the true states with the observed states:

		Observation at Time t		
		Seen in A	**Seen in B**	**Not seen**
True State at Time t	**Site A**	p_A	0	$1 - p_A$
	Site B	0	p_B	$1 - p_B$
	Dead	0	0	1

or simply

$$\begin{array}{c} \\ \text{site A} \\ \text{site B} \\ \text{dead} \end{array} \begin{array}{c} \begin{array}{ccc} \text{seen in A} & \text{seen in B} & \text{not seen} \end{array} \\ \begin{bmatrix} p_A & 0 & 1-p_A \\ 0 & p_B & 1-p_B \\ 0 & 0 & 1 \end{bmatrix} \end{array}$$

The parameters in this matrix are the site-specific recapture probabilities (p_A, p_B). The true states at time t correspond to the rows of the observation matrix in the order "site A", "site B", and "dead" from top to bottom, and the observed states are in columns in the order "seen in A", "seen in B", and "not seen" from left to right. For example, the probability that an individual in true state "site A" at time t is in the observed state "not seen" at time t is $1 - p_A$.

For ease of presentation, we have dropped several indices in these matrices. In reality, however, both matrices could have an index t for time and i for individual.

9.2.2 Generation of Simulated Data

Imagine that we sampled capture–recapture data of little ringed plovers (Fig. 9.2) at two sites (A and B). Habitat quality at site A is higher than at site B, and thus annual survival in A is higher (0.8) than that in B (0.7), and movements from A to B are less likely ($\psi_{AB} = 0.3$) than movements from B to A ($\psi_{BA} = 0.5$). We assume that capture effort was higher at A than at B. In the data sets, captures of an individual at site A are labeled "1"and captures at site B "2".

To generate multistate capture–recapture data, we first need to define the probabilities in both the state-transition and the observation matrix. For individual i released at time t in state m, we simulate its state at time $t + 1$ using a categorical distribution. The probabilities of the categorical distribution are taken from row m of the state-transition matrix

FIGURE 9.2 Little ringed plover (*Charadrius dubius*) (Photograph by D. Occhiato).

(PSI.STATE) of individual i at time t. We then simulate the observation for individual i at time $t+1$. Assume that individual i is in state n at time $t+1$. We then simulate the observation using a categorical distribution again. The probabilities of the categorical distribution are taken this time from row n of the observation matrix (PSI.OBS) for individual i at time $t+1$. These steps are first repeated for individual i until the end of the study and then for all released individuals. In program R, we specify the categorical distribution as a multinomial distribution with trial size equal to 1. In the following code, we first define the simulation parameters, the state-transition and observation matrix and then apply the function simul.ms, which simulates multistate capture–recapture data as described earlier.

```
# Define mean survival, transitions, recapture, as well as number of
  occasions, states, observations and released individuals
phiA <- 0.8
phiB <- 0.7
psiAB <- 0.3
psiBA <- 0.5
pA <- 0.7
pB <- 0.4
n.occasions <- 6
n.states <- 3
```

```
n.obs <- 3
marked <- matrix(NA, ncol = n.states, nrow = n.occasions)
marked[,1] <- rep(100, n.occasions)
marked[,2] <- rep(60, n.occasions)
marked[,3] <- rep(0, n.occasions)

# Define matrices with survival, transition and recapture probabilities
# These are 4-dimensional matrices, with
   # Dimension 1: state of departure
   # Dimension 2: state of arrival
   # Dimension 3: individual
   # Dimension 4: time

# 1. State process matrix
totrel <- sum(marked)*(n.occasions-1)
PSI.STATE <- array(NA, dim=c(n.states, n.states, totrel, n.occasions-1))
for (i in 1:totrel){
   for (t in 1:(n.occasions-1)){
      PSI.STATE[,,i,t] <- matrix(c(
      phiA*(1-psiAB), phiA*psiAB,     1-phiA,
      phiB*psiBA,     phiB*(1-psiBA), 1-phiB,
      0,              0,              1       ), nrow = n.states,
         byrow = TRUE)
      } #t
   } #i

# 2.Observation process matrix
PSI.OBS <- array(NA, dim=c(n.states, n.obs, totrel, n.occasions-1))
for (i in 1:totrel){
   for (t in 1:(n.occasions-1)){
      PSI.OBS[,,i,t] <- matrix(c(
      pA, 0,  1-pA,
      0,  pB, 1-pB,
      0,  0,  1       ), nrow = n.states, byrow = TRUE)
      } #t
   } #i

# Define function to simulate multistate capture-recapture data
simul.ms <- function(PSI.STATE, PSI.OBS, marked, unobservable = NA){
   # Unobservable: number of state that is unobservable
    n.occasions <- dim(PSI.STATE)[4] + 1
    CH <- CH.TRUE <- matrix(NA, ncol = n.occasions, nrow = sum(marked))
    # Define a vector with the occasion of marking
    mark.occ <- matrix(0, ncol = dim(PSI.STATE)[1], nrow = sum(marked))
    g <- colSums(marked)
    for (s in 1:dim(PSI.STATE)[1]){
    if (g[s]==0) next # To avoid error message if nothing to replace
       mark.occ[(cumsum(g[1:s])-g[s]+1)[s]:cumsum(g[1:s])[s],s] <-
       rep(1:n.occasions, marked[1:n.occasions,s])
       }#s
    for (i in 1:sum(marked)){
       for (s in 1:dim(PSI.STATE)[1]){
          if (mark.occ[i,s]==0) next
          first <- mark.occ[i,s]
```

```
      CH[i,first] <- s
      CH.TRUE[i,first] <- s
      } #s
    for (t in (first+1):n.occasions){
      # Multinomial trials for state transitions
      if (first==n.occasions) next
      state <- which(rmultinom(1, 1, PSI.STATE[CH.TRUE[i,t-1],,
        i,t-1])==1)
      CH.TRUE[i,t] <- state
      # Multinomial trials for observation process
      event <- which(rmultinom(1, 1, PSI.OBS[CH.TRUE[i,t],,i,
        t-1])==1)
      CH[i,t] <- event
      } #t
    } #i
  # Replace the NA and the highest state number (dead) in the file by 0
  CH[is.na(CH)] <- 0
  CH[CH==dim(PSI.STATE)[1]] <- 0
  CH[CH==unobservable] <- 0
  id <- numeric(0)
  for (i in 1:dim(CH)[1]){
    z <- min(which(CH[i,]!=0))
    ifelse(z==dim(CH)[2], id <- c(id,i), id <- c(id))
    }
  return(list(CH=CH[-id,], CH.TRUE=CH.TRUE[-id,]))
  # CH: capture-histories to be used
  # CH.TRUE: capture-histories with perfect observation
  }

# Execute function
sim <- simul.ms(PSI.STATE, PSI.OBS, marked)
CH <- sim$CH
```

The multistate capture–recapture data simulated may look like this (first five rows shown):

```
[1,]   1   2   0   0   0   0
[2,]   1   0   0   0   0   0
[3,]   1   0   1   1   0   0
[4,]   1   0   2   0   0   0
[5,]   1   0   2   0   0   0
```

A "1" denotes capture at site A, a "2" capture at site B, and a "0" denotes noncapture. To fit a multistate model in the state-space formulation, we need to compute a vector indicating the occasion of marking for each individual. Furthermore, we must replace the "0" in the data with a "3" to match the observed states, which are numbered "1" (seen at site A), "2" (seen at site B), and "3" (not seen).

```
# Compute vector with occasion of first capture
get.first <- function(x) min(which(x!=0))
f <- apply(CH, 1, get.first)
```

```
# Recode CH matrix: note, a 0 is not allowed in WinBUGS!
# 1 = seen alive in A, 2 = seen alive in B, 3 = not seen
rCH <- CH # Recoded CH
rCH[rCH==0] <- 3
```

The final multistate capture–recapture data ready to be analyzed in BUGS is named rCH and may look like this:

```
[1,]   1   2   3   3   3   3
[2,]   1   3   3   3   3   3
[3,]   1   3   1   1   3   3
[4,]   1   3   2   3   3   3
[5,]   1   3   2   3   3   3
```

9.2.3 Analysis of the Model

As so often, the BUGS code to analyze the multistate capture–recapture data closely resembles the way in which we simulate the data. In addition, we define priors for each parameter and can impose constraints on them. Then, we define the state-transition and the observation matrices using the model parameters. They are named `ps` and `po`, respectively, in the BUGS code, and correspond to Ω and Θ of Section 9.1. These matrices have four dimensions because they not only indicate states of departure and arrival but also time and individual. Finally, the state-space likelihood section of code is similar to that introduced for CJS models. The state-transition process describes the probability of the state at time $t+1$ (stored in matrix Z) conditional on the state at time t, and the observation process describes the mapping of the state at time $t+1$ on the observed data. The only difference with the CJS model is that in the likelihood we now use a categorical instead of a Bernoulli distribution. Note that the categorical distribution in WinBUGS needs the empty index in the last position; hence, the matrix dimensions are ordered differently from the formulas in Section 9.1.

```
# Specify model in BUGS language
sink("ms.bug")
cat("
model {

# ------------------------------------------------
# Parameters:
# phiA: survival probability at site A
# phiB: survival probability at site B
# psiAB: movement probability from site A to site B
# psiBA: movement probability from site B to site A
# pA: recapture probability at site A
# pB: recapture probability at site B
# ------------------------------------------------
# States (S):
# 1 alive at A
```

```
# 2 alive at B
# 3 dead
# Observations (O):
# 1 seen at A
# 2 seen at B
# 3 not seen
# ------------------------------------------------

# Priors and constraints
for (t in 1:(n.occasions-1)){
    phiA[t] <- mean.phi[1]
    phiB[t] <- mean.phi[2]
    psiAB[t] <- mean.psi[1]
    psiBA[t] <- mean.psi[2]
    pA[t] <- mean.p[1]
    pB[t] <- mean.p[2]
    }
for (u in 1:2){
    mean.phi[u] ~ dunif(0, 1)    # Priors for mean state-spec. survival
    mean.psi[u] ~ dunif(0, 1)    # Priors for mean transitions
    mean.p[u] ~ dunif(0, 1)      # Priors for mean state-spec. recapture
    }

# Define state-transition and observation matrices
for (i in 1:nind){
    # Define probabilities of state S(t+1) given S(t)
    for (t in f[i]:(n.occasions-1)){
        ps[1,i,t,1] <- phiA[t] * (1-psiAB[t])
        ps[1,i,t,2] <- phiA[t] * psiAB[t]
        ps[1,i,t,3] <- 1-phiA[t]
        ps[2,i,t,1] <- phiB[t] * psiBA[t]
        ps[2,i,t,2] <- phiB[t] * (1-psiBA[t])
        ps[2,i,t,3] <- 1-phiB[t]
        ps[3,i,t,1] <- 0
        ps[3,i,t,2] <- 0
        ps[3,i,t,3] <- 1

        # Define probabilities of O(t) given S(t)
        po[1,i,t,1] <- pA[t]
        po[1,i,t,2] <- 0
        po[1,i,t,3] <- 1-pA[t]
        po[2,i,t,1] <- 0
        po[2,i,t,2] <- pB[t]
        po[2,i,t,3] <- 1-pB[t]
        po[3,i,t,1] <- 0
        po[3,i,t,2] <- 0
        po[3,i,t,3] <- 1
        } #t
    } #i

# Likelihood
for (i in 1:nind){
   # Define latent state at first capture
   z[i,f[i]] <- Y[i,f[i]]
   for (t in (f[i]+1):n.occasions){
```

```
      # State process: draw S(t) given S(t-1)
      z[i,t] ~ dcat(ps[z[i,t-1], i, t-1,])
      # Observation process: draw O(t) given S(t)
      y[i,t] ~ dcat(po[z[i,t], i, t-1,])
      } #t
   } #i
}
",fill = TRUE)
sink()
```

Before we load the data, we may again compute a matrix with known latent states z. This will reduce computation time and improve convergence; see Section 7.3.1. The latent state z now contains information not only about survival as in the CJS or the mark-recovery model but also about a location (or "state"). Therefore, the latent state is always unknown during occasions when an individual is not encountered. Care must also be taken when computing the known-state matrix when the names of the true and the observed states differ. The following function creates the required matrix for the current example.

```
# Function to create known latent states z
known.state.ms <- function(ms, notseen){
    # notseen: label for 'not seen'
    state <- ms
    state[state==notseen] <- NA
    for (i in 1:dim(ms)[1]){
        m <- min(which(!is.na(state[i,])))
        state[i,m] <- NA
        }
    return(state)
    }
```

Initial values need to be given only for the unknown z; the following function creates these values.

```
# Function to create initial values for unknown z
ms.init.z <- function(ch, f){
   for (i in 1:dim(ch)[1]){ch[i,1:f[i]] <- NA}
   states <- max(ch, na.rm = TRUE)
   known.states <- 1:(states-1)
   v <- which(ch==states)
   ch[-v] <- NA
   ch[v] <- sample(known.states, length(v), replace = TRUE)
   return(ch)
   }
```

Now, we are ready to fit the model.

```
# Bundle data
bugs.data <- list(y = rCH, f = f, n.occasions = dim(rCH)[2], nind =
   dim(rCH)[1], z = known.state.ms(rCH, 3))
```

```
# Initial values
inits <- function(){list(mean.phi = runif(2, 0, 1), mean.psi = runif(2,
   0, 1), mean.p = runif(2, 0, 1), z = ms.init.z(rCH, f))}

# Parameters monitored
parameters <- c("mean.phi", "mean.psi", "mean.p")

# MCMC settings
ni <- 10000
nt <- 6
nb <- 2000
nc <- 3

# Call WinBUGS from R (BRT 8 min)
ms <- bugs(bugs.data, inits, parameters, "ms.bug", n.chains = nc,
   n.thin = nt, n.iter = ni, n.burnin = nb, debug = TRUE, bugs.directory =
   bugs.dir, working.directory = getwd())
```

This model converges relatively quickly.

```
print(ms, digits = 3)
```

```
              mean    sd  2.5%   25%   50%   75% 97.5%  Rhat n.eff
mean.phi[1] 0.808 0.030 0.753 0.786 0.806 0.826 0.872 1.005   550
mean.phi[2] 0.690 0.031 0.627 0.670 0.691 0.711 0.748 1.001  4000
mean.psi[1] 0.422 0.079 0.271 0.363 0.423 0.485 0.559 1.007   480
mean.psi[2] 0.419 0.069 0.311 0.367 0.412 0.460 0.578 1.005   660
mean.p[1]   0.803 0.102 0.626 0.723 0.795 0.888 0.985 1.009   310
mean.p[2]   0.288 0.062 0.201 0.242 0.277 0.322 0.430 1.008   590
```

The posterior distributions of the six model parameters look reasonably good (Fig. 9.3). However, the posteriors for transitions and recapture parameters are wide, indicating that these parameters are not estimated precisely. This is probably because the study duration was short in our example, so the data contained little information about these parameters.

```
par(mfrow = c(3, 2), las = 1)
hist(ms$sims.list$mean.phi[,1], col = "gray", main = "", xlab =
   expression(phi[A]), ylim=c(0,1300))
abline(v = phiA, col = "red")
hist(ms$sims.list$mean.phi[,2], col = "gray", main = "", xlab =
   expression(phi[B]), ylim=c(0,1300), ylab="")
abline(v = phiB, , col="red")
hist(ms$sims.list$mean.psi[,1], col = "gray", main = "", xlab =
   expression(psi[AB]), ylim=c(0,1300))
abline(v = psiAB, col="red")
hist(ms$sims.list$mean.psi[,2], col = "gray", main = "", xlab =
   expression(psi[BA]), ylab="", ylim=c(0,1300))
abline(v = psiBA, col="red")
hist(ms$sims.list$mean.p[,1], col = "gray", main = "", xlab =
   expression(p[A]), ylim=c(0,1300))
abline(v = pA, col = "red")
hist(ms$sims.list$mean.p[,2], col = "gray", main = "", xlab =
   expression(p[B]), ylab="", ylim=c(0,1300))
abline(v = pB, col = "red")
```

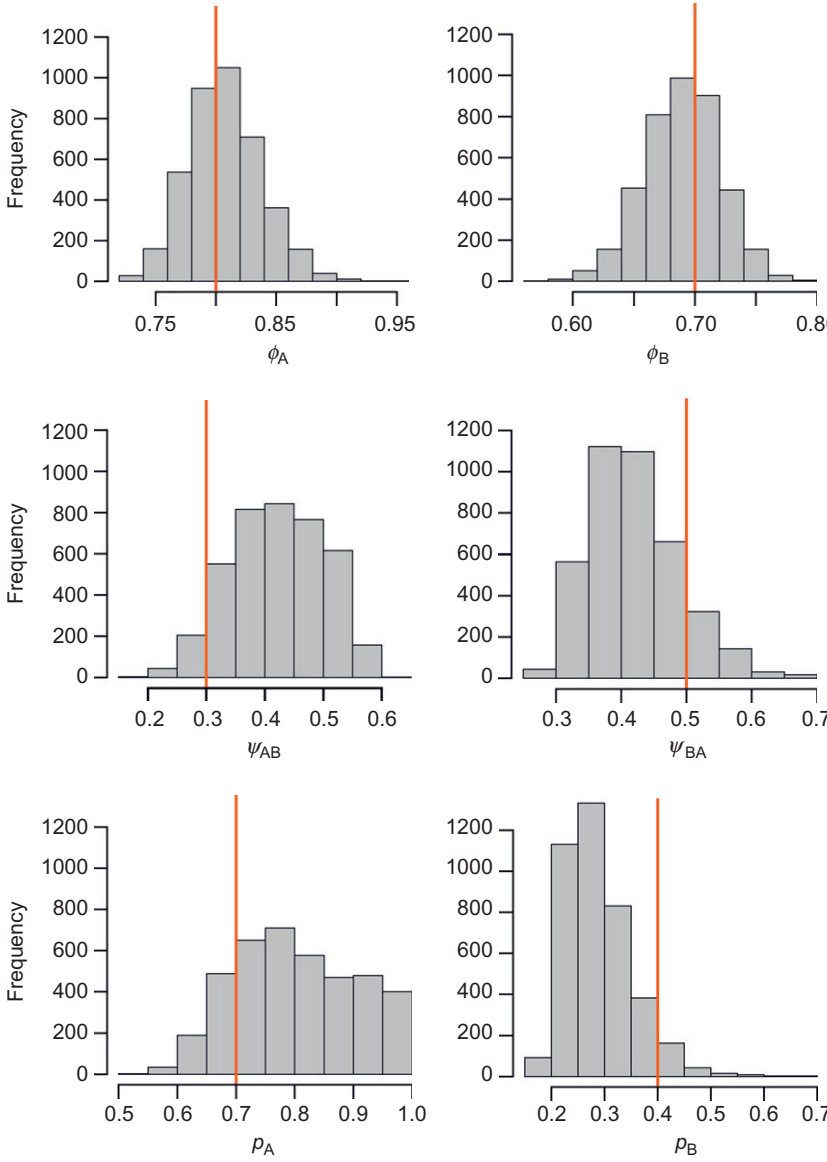

FIGURE 9.3 Posterior distributions of survival, movement, and recapture probabilities. Red lines give the values of the data generating parameters.

This basic model can now be extended in the same way as we saw for the CJS model in Chapter 7. Thus, it is straightforward to add fixed time effects, group effects, or to include individual or temporal random effects. As mentioned earlier, all we need to do is to modify the code sections called "Priors and

constraints" and "Define parameters". You may want to check the examples given for the CJS models to understand how to make these changes and study exercises 1 and 2 in Section 9.9.

The same model structure may be used when states A and B denote disease states, classes of reproductive success, or "breeder" and "non-breeder", to name a few possibilities. Moreover, this model easily extends to a larger number of locations (see Section 9.5).

This multistate model is written in such a way that the inclusion of temporal and individual effects is straightforward, that is, only the model section called "Priors and constraints" needs to be modified. However, without individual effects, the model could be written in a more efficient way, which would reduce computing time somewhat. In particular, the definition of the state-transition and observation matrices (`ps` and `po`) does not need the index for individual in that case. Thus, an alternative way to write the model for this case is as follows:

```
# Specify model in BUGS language
sink("ms.alternative1.bug")
cat("
model {
# Priors and constraints
for (t in 1:(n.occasions-1)){
   phiA[t] <- mean.phi[1]
   phiB[t] <- mean.phi[2]
   psiAB[t] <- mean.psi[1]
   psiBA[t] <- mean.psi[2]
   pA[t] <- mean.p[1]
   pB[t] <- mean.p[2]
   }
for (u in 1:2){
   mean.phi[u] ~ dunif(0, 1)    # Priors for mean state-spec. survival
   mean.psi[u] ~ dunif(0, 1)    # Priors for mean transitions
   mean.p[u] ~ dunif(0, 1)      # Priors for mean state-spec. recapture
   }
# Define state-transition and observation matrices
    # Define probabilities of state S(t+1) given S(t)
    for (t in 1:(n.occasions-1)){
       ps[1,t,1] <- phiA[t] * (1-psiAB[t])
       ps[1,t,2] <- phiA[t] * psiAB[t]
       ps[1,t,3] <- 1-phiA[t]
       ps[2,t,1] <- phiB[t] * psiBA[t]
       ps[2,t,2] <- phiB[t] * (1-psiBA[t])
       ps[2,t,3] <- 1-phiB[t]
       ps[3,t,1] <- 0
       ps[3,t,2] <- 0
       ps[3,t,3] <- 1
   # Define probabilities of O(t) given S(t)
       po[1,t,1] <- pA[t]
       po[1,t,2] <- 0
```

```
      po[1,t,3] <- 1-pA[t]
      po[2,t,1] <- 0
      po[2,t,2] <- pB[t]
      po[2,t,3] <- 1-pB[t]
      po[3,t,1] <- 0
      po[3,t,2] <- 0
      po[3,t,3] <- 1
      } #t

# Likelihood
for (i in 1:nind){
   # Define latent state at first capture
   z[i,f[i]] <- Y[i,f[i]]
   for (t in (f[i]+1):n.occasions){
      # State process: draw S(t) given S(t-1)
      z[i,t] ~ dcat(ps[z[i,t-1], t-1,])
      # Observation process: draw O(t) given S(t)
      y[i,t] ~ dcat(po[z[i,t], t-1,])
      } #t
   } #i
}
",fill = TRUE)
sink()
```

If in addition we do not need temporal effects, there is an even more efficient way to write the model: the index for time in the matrices `ps` and `po` is no longer needed. We then define priors directly for the quantities of interest (phiA, phiB, …), and we also need to change the initial values accordingly.

```
# Specify model in BUGS language
sink("ms.alternative2.bug")
cat("
model {

# Priors and constraints
   phiA ~ dunif(0, 1)      # Prior for mean survival in A
   phiB ~ dunif(0, 1)      # Prior for mean survival in B
   psiAB ~ dunif(0, 1)     # Prior for mean movement from A to B
   psiBA ~ dunif(0, 1)     # Prior for mean movement from B to A
   pA ~ dunif(0, 1)        # Prior for mean recapture in A
   pB ~ dunif(0, 1)        # Prior for mean recapture in B

# Define state-transition and observation matrices
   # Define probabilities of state S(t+1) given S(t)
      ps[1,1] <- phiA * (1-psiAB)
      ps[1,2] <- phiA * psiAB
      ps[1,3] <- 1-phiA
      ps[2,1] <- phiB * psiBA
      ps[2,2] <- phiB * (1-psiBA)
      ps[2,3] <- 1-phiB
      ps[3,1] <- 0
      ps[3,2] <- 0
      ps[3,3] <- 1
```

```
    # Define probabilities of O(t) given S(t)
      po[1,1] <- pA
      po[1,2] <- 0
      po[1,3] <- 1-pA
      po[2,1] <- 0
      po[2,2] <- pB
      po[2,3] <- 1-pB
      po[3,1] <- 0
      po[3,2] <- 0
      po[3,3] <- 1

  # Likelihood
  for (i in 1:nind){
    # Define latent state at first capture
    z[i,f[i]] <- Y[i,f[i]]
    for (t in (f[i]+1):n.occasions){
      # State process: draw S(t) given S(t-1)
      z[i,t] ~ dcat(ps[z[i,t-1],])
      # Observation process: draw O(t) given S(t)
      y[i,t] ~ dcat(po[z[i,t],])
      } #t
    } #i
  }
  ",fill = TRUE)
  sink()
```

The reduction in computing time is about 30% for the first and about 40% for the second alternative model. Despite these computational benefits, we will not use these parameterizations in the following examples but stick to the more general parameterization. However, we recommend using alternative parameterizations when computing time is an issue.

9.3 ACCOUNTING FOR TEMPORARY EMIGRATION

9.3.1 Model Description

Sometimes individuals are not available for capture at certain occasions, although they are alive *somewhere*. An example of this is when individuals skip reproduction in a particular year, and sampling is conducted only at breeding sites, or plants that are dormant in some years when they are only surveyed above ground. If the temporary absence is random (i.e., does not depend on whether an individual was available for capture at the previous occasion), then temporary emigration does not bias estimates of survival probability. It does lower recapture probability, but this parameter is usually not of direct interest. By contrast, if temporary absence is Markovian, that is, depends on whether an individual was absent one time step ago, then it does cause bias in the estimated survival probability, unless accounted for (Kendall et al., 1997; Schaub

et al., 2004a). Note that temporary emigration is different from permanent emigration. When individuals emigrate permanently, they become permanently unavailable for capture and the estimated survival probability is always biased low. That is why all survival estimated from CJS and multistate models should more accurately be called "apparent survival" (see Chapter 7).

To model temporary emigration, we define the states "alive and present", "alive and absent", and "dead". The state-transition matrix is

$$\begin{array}{c} \\ \text{present} \\ \text{absent} \\ \text{dead} \end{array} \begin{array}{c} \begin{array}{ccc} \text{present} & \text{absent} & \text{dead} \end{array} \\ \begin{bmatrix} \phi(1-\psi_{IO}) & \phi\psi_{IO} & 1-\phi \\ \phi\psi_{OI} & \phi(1-\psi_{OI}) & 1-\phi \\ 0 & 0 & 1 \end{bmatrix} \end{array},$$

where ϕ is survival probability, ψ_{IO} is the probability that an individual present ("in") at time t is absent ("out") at time $t + 1$, and ψ_{OI} is the probability that an individual absent at time t is present at time $t + 1$. If $\psi_{IO} = 1 - \psi_{OI}$, temporary emigration is random.

The list of possible observations is "seen" and "not seen". Therefore, we have a 3×2 observation matrix,

$$\begin{array}{c} \\ \text{present} \\ \text{absent} \\ \text{dead} \end{array} \begin{array}{c} \begin{array}{cc} \text{seen} & \text{not seen} \end{array} \\ \begin{bmatrix} p & 1-p \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \end{array},$$

where p is recapture probability (conditional on being alive and available for capture).

The model of temporary emigration is an example of a model with two unobserved states ("alive and absent" and "dead"). Such models often have identifiability problems. Kendall and Nichols (2002), Schaub et al. (2004a), and Bailey et al. (2010) studied the identifiability and the performance of this and more complex models with additional states, and give catalogues of which parameters can be estimated under which conditions.

9.3.2 Generation of Simulated Data

Let us assume that we have repeatedly photographed fire salamanders (Fig. 9.4) at a hibernation site in a cave. The black and yellow pattern is extremely variable among individuals; hence, these photos allow one to determine individual identity. Annual adult survival of fire salamanders is high (Schmidt et al., 2005); we here assume $\phi = 0.85$. Since salamanders do not hibernate every year at the same place, they may temporarily be absent, that is, unavailable for capture (which means, being

FIGURE 9.4 Fire salamander (*Salamandra salamandra*), Switzerland, 2008 (Photograph by T. Ott).

photographed), when sampling only takes place at one cave. We assume that the probability an individual that is present becomes absent in the next year is $\psi_{IO} = 0.2$, and that the probability a salamander that is absent becomes present the next year is $\psi_{OI} = 0.3$. Given presence in the cave, recapture probability is $p = 0.7$.

```
# Define mean survival, transitions, recapture, as well as number of
  occasions, states, observations and released individuals
phi <- 0.85
psiIO <- 0.2
psiOI <- 0.3
p <- 0.7
n.occasions <- 8
n.states <- 3
n.obs <- 2
marked <- matrix(NA, ncol = n.states, nrow = n.occasions)
marked[,1] <- rep(70, n.occasions) # Present
marked[,2] <- rep(0, n.occasions)  # Absent
marked[,3] <- rep(0, n.occasions)  # Dead

# Define matrices with survival, transition and recapture probabilities
# These are 4-dimensional matrices, with
   # Dimension 1: state of departure
```

```
   # Dimension 2: state of arrival
   # Dimension 3: individual
   # Dimension 4: time
# 1. State process matrix
totrel <- sum(marked)*(n.occasions-1)
PSI.STATE <- array(NA, dim=c(n.states, n.states, totrel,
   n.occasions-1))
for (i in 1:totrel){
   for (t in 1:(n.occasions-1)){
      PSI.STATE[,,i,t] <- matrix(c(
      phi*(1-psiIO), phi*psiIO,     1-phi,
      phi*psiOI,     phi*(1-psiOI), 1-phi,
      0,             0,             1         ), nrow=n.states,
         byrow = TRUE)
      } #t
   } #i

# 2.Observation process matrix
PSI.OBS <- array(NA, dim=c(n.states, n.obs, totrel, n.occasions-1))
for (i in 1:totrel){
   for (t in 1:(n.occasions-1)){
      PSI.OBS[,,i,t] <- matrix(c(
      p, 1-p,
      0, 1,
      0, 1      ), nrow=n.states, byrow=TRUE)
      } #t
   } #i

# Execute simulation function
sim <- simul.ms(PSI.STATE, PSI.OBS, marked)
CH <- sim$CH

# Compute vector with occasion of first capture
get.first <- function(x) min(which(x!=0))
f <- apply(CH, 1, get.first)

# Recode CH matrix: note, a 0 is not allowed!
# 1=seen alive, 2=not seen
rCH <- CH # Recoded CH
rCH[rCH==0] <- 2
```

The elements of the capture-history matrix are equal to "1" at occasions when an individual was seen and "2" otherwise. So far, we always removed the zeroes in the capture-histories; however, this would not really be necessary for zeroes that occur before first capture. Remember that in the CJS, mark-recovery, and multistate models of Chapters 7–9, the data before first capture are not modeled, that is, they do not enter the likelihood.

9.3.3 Analysis of the Model

Again, the BUGS code is straightforward to write in terms of the state-transition and observation matrices. Both matrices are specified along with the priors; no change is required in the remaining code.

```
# Specify model in BUGS language
sink("tempemi.bug")
cat("
model {

# --------------------------------
# Parameters:
# phi: survival probability
# psiIO: probability to emigrate
# psiOI: probability to immigrate
# p: recapture probability
# --------------------------------
# States (S):
# 1 alive and present
# 2 alive and absent
# 3 dead
# Observations (O):
# 1 seen
# 2 not seen
# --------------------------------

# Priors and constraints
for (t in 1:(n.occasions-1)){
   phi[t] <- mean.phi
   psiIO[t] <- mean.psiIO
   psiOI[t] <- mean.psiOI
   p[t] <- mean.p
   }
mean.phi ~ dunif(0, 1)         # Prior for mean survival
mean.psiIO ~ dunif(0, 1)       # Prior for mean temp. emigration
mean.psiOI ~ dunif(0, 1)       # Prior for mean temp. immigration
mean.p ~ dunif(0, 1)           # Prior for mean recapture

# Define state-transition and observation matrices
for (i in 1:nind){
    # Define probabilities of state S(t+1) given S(t)
    for (t in f[i]:(n.occasions-1)){
      ps[1,i,t,1] <- phi[t] * (1-psiIO[t])
      ps[1,i,t,2] <- phi[t] * psiIO[t]
      ps[1,i,t,3] <- 1-phi[t]
      ps[2,i,t,1] <- phi[t] * psiOI[t]
      ps[2,i,t,2] <- phi[t] * (1-psiOI[t])
      ps[2,i,t,3] <- 1-phi[t]
      ps[3,i,t,1] <- 0
      ps[3,i,t,2] <- 0
      ps[3,i,t,3] <- 1

      # Define probabilities of O(t) given S(t)
      po[1,i,t,1] <- p[t]
      po[1,i,t,2] <- 1-p[t]
      po[2,i,t,1] <- 0
      po[2,i,t,2] <- 1
      po[3,i,t,1] <- 0
      po[3,i,t,2] <- 1
      } #t
   } #i
```

```
# Likelihood
for (i in 1:nind){
   # Define latent state at first capture
   z[i,f[i]] <- Y[i,f[i]]
   for (t in (f[i]+1):n.occasions){
      # State process: draw S(t) given S(t-1)
      z[i,t] ~ dcat(ps[z[i,t-1], i, t-1,])
      # Observation process: draw O(t) given S(t)
      y[i,t] ~ dcat(po[z[i,t], i, t-1,])
      } #t
   } #i
}
",fill = TRUE)
sink()

# Bundle data
bugs.data <- list(y = rCH, f = f, n.occasions = dim(rCH)[2],
   nind = dim(rCH)[1], z = known.state.ms(rCH, 2))

# Initial values
inits <- function(){list(mean.phi = runif(1, 0, 1), mean.psiIO = runif(1,
   0, 1), mean.psiOI = runif(1, 0, 1), mean.p = runif(1, 0, 1), z = ms.init.z
   (rCH, f))}

# Parameters monitored
parameters <- c("mean.phi", "mean.psiIO", "mean.psiOI", "mean.p")

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

# Call WinBUGS from R (BRT 35 min)
tempemi <- bugs(bugs.data, inits, parameters, "tempemi.bug", n.chains =
   nc, n.thin = nt, n.iter = ni, n.burnin = nb, debug = TRUE, bugs.directory =
   bugs.dir, working.directory = getwd())
```

The analysis of this simulated data set requires long Markov chains to achieve convergence. This model has identifiability problems, the degree of which depends on the parameter values. Recapture and transition probabilities are not identifiable and biased if temporary emigration is random (i.e., $\psi_{IO} = 1 - \psi_{OI}$); see Kendall and Nichols (2002) and Schaub et al. (2004a). In our case, however, the posterior distributions of all parameters match up well with the values used for generating the data set, with the exception of the probability of moving back to the study area (temporary immigration; Fig. 9.5). Again, the discrepancy stems from the fact that the simulated data set is relatively small. Repeated simulations would show that the estimators are unbiased, but that their precision is low (Schaub et al., 2004a). Low precision of transition parameters is typical for multistate models with unobservable states.

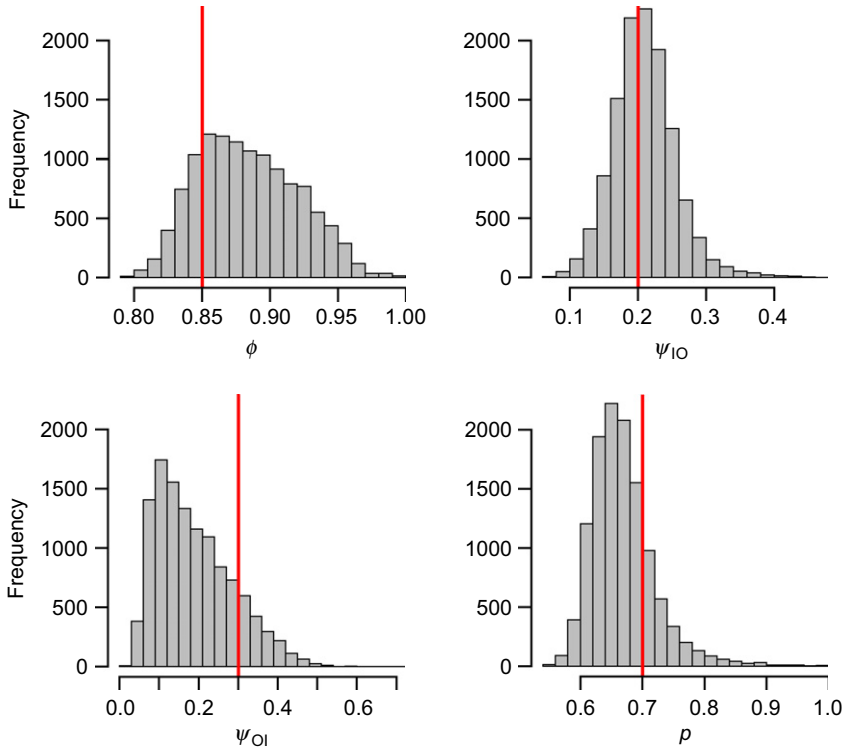

FIGURE 9.5 Posterior distributions of the estimated parameters along with the values used to simulate the data (red vertical lines).

```
print(tempemi, digits = 3)
```

```
            mean    sd  2.5%   25%   50%   75% 97.5%  Rhat n.eff
mean.phi   0.884 0.037 0.823 0.855 0.880 0.910 0.956 1.011   260
mean.psiIO 0.210 0.046 0.126 0.180 0.207 0.235 0.310 1.004   610
mean.psiOI 0.189 0.096 0.057 0.111 0.170 0.252 0.407 1.008   370
mean.p     0.669 0.052 0.592 0.634 0.661 0.693 0.796 1.003 12000
```

```
par(mfrow = c(2, 2), las = 1)
hist(tempemi$sims.list$mean.phi, col = "gray", main = "", xlab =
   expression(phi))
abline(v = phi, col = "red", lwd = 2)
hist(tempemi$sims.list$mean.psiIO, col = "gray", main = "", xlab =
   expression(psi[IO]), ylab = "")
abline(v = psiIO, col = "red", lwd = 2)
hist(tempemi$sims.list$mean.psiOI, col = "gray", main = "", xlab =
   expression(psi[OI]))
abline(v = psiOI, col = "red", lwd = 2)
hist(tempemi$sims.list$mean.p, col = "gray", main = "", xlab =
   expression(p), ylab = "")
abline(v = p, col = "red", lwd = 2)
```

9.4 ESTIMATION OF AGE-SPECIFIC PROBABILITY OF FIRST BREEDING

9.4.1 Model Description

For many long-lived species, there is an interest in estimating age-specific probability of first breeding (reproduction). This requires that newborn individuals are marked and later resighted/recaptured when they breed. The state associated with a resighting event is "seen, no breeder yet" if an individual is observed but is not breeding in a given year, nor has it ever been observed breeding before, or "seen as breeder" if the individual is seen breeding in a year. If an individual is resighted as a non-breeder in a year but was observed to breed earlier, it is also assigned to the state "seen as breeder". (Note that this assumption could be relaxed by the inclusion of a further state allowing the estimation of breeding frequency of experienced breeders.) If such data are analyzed without accounting for imperfect detection, age at first breeding is overestimated because some individuals may have first bred *before* they were first *observed* breeding. The multistate capture–recapture model described here avoids this bias by accounting for imperfect detection.

The model makes the assumption that the age at which all previous non-breeders start to breed is known. In our example, we assume that all individuals 3 years old have become breeders and define the following states in the model: "juvenile", "not yet breeding at age 1", "not yet breeding at age 2", "breeder", and "dead". The resulting state-transition matrix is

$$\begin{array}{c} \\ \text{juvenile} \\ \text{non-breeder 1} \\ \text{non-breeder 2} \\ \text{breeder} \\ \text{dead} \end{array} \begin{array}{c} \begin{array}{ccccc} \text{juvenile} & \text{non-breeder 1} & \text{non-breeder 2} & \text{breeder} & \text{dead} \end{array} \\ \begin{bmatrix} 0 & \phi_1(1-\alpha_1) & 0 & \phi_1\alpha_1 & 1-\phi_1 \\ 0 & 0 & \phi_2(1-\alpha_2) & \phi_2\alpha_2 & 1-\phi_2 \\ 0 & 0 & 0 & \phi_{ad} & 1-\phi_{ad} \\ 0 & 0 & 0 & \phi_{ad} & 1-\phi_{ad} \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \end{array}.$$

Here, ϕ denotes the age-specific survival probability and α_x the probabilities of starting to breed at age x. We define three age classes for survival, but other definitions would also be possible.

The possible observations are "seen as juv", "seen as non-breeder", "seen as breeder", and "not seen". The observation matrix is

$$\begin{array}{c} \\ \text{juvenile} \\ \text{non-breeder 1} \\ \text{non-breeder 2} \\ \text{breeder} \\ \text{dead} \end{array} \begin{array}{c} \begin{array}{cccc} \text{seen as juv} & \text{seen non-breeder} & \text{seen breeder} & \text{not seen} \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & p_{\text{NB}} & 0 & 1-p_{\text{NB}} \\ 0 & p_{\text{NB}} & 0 & 1-p_{\text{NB}} \\ 0 & 0 & p_{\text{B}} & 1-p_{\text{B}} \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{array},$$

where p_{NB} is the probability of re-encountering an individual that is not yet breeding and p_B is the probability of re-encountering a breeder. Depending on the study, p_{NB} may be zero.

The parameters estimated in this model allow the estimation of the mean age at first reproduction or of age-specific recruitment probabilities. Pradel and Lebreton (1999) show the connections between these quantities.

9.4.2 Generation of Simulated Data

Let us assume that we studied survival and age-specific recruitment in little ringed plovers (Fig. 9.2). We color-mark chicks in a population and resight them in subsequent years. For all resighted birds, we record their breeding status. In the simulation, we assume the following parameters: $\phi_1 = 0.4$, $\phi_2 = 0.7$, $\phi_{ad} = 0.8$, $\alpha_1 = 0.2$, $\alpha_2 = 0.6$, $p_{NB} = 0.5$, and $p_B = 0.7$.

```
# Define mean survival, transitions, recapture, as well as number of
  occasions, states, observations and released individuals
phi.1 <- 0.4
phi.2 <- 0.7
phi.ad <- 0.8
alpha.1 <- 0.2
alpha.2 <- 0.6
p.NB <- 0.5
p.B <- 0.7
n.occasions <- 7
n.states <- 5
n.obs <- 4
marked <- matrix(0, ncol = n.states, nrow = n.occasions)
marked[,1] <- rep(100, n.occasions) # Releases only as juveniles

# Define matrices with survival, transition and recapture probabilities
# These are 4-dimensional matrices, with
   # Dimension 1: state of departure
   # Dimension 2: state of arrival
   # Dimension 3: individual
   # Dimension 4: time
# 1. State process matrix
totrel <- sum(marked)*(n.occasions-1)
PSI.STATE <- array(NA, dim=c(n.states, n.states, totrel, n.occasions-1))
for (i in 1:totrel){
   for (t in 1:(n.occasions-1)){
      PSI.STATE[,,i,t] <- matrix(c(
      0, phi.1*(1-alpha.1), 0,                  phi.1*alpha.1, 1-phi.1,
      0, 0,                 phi.2*(1-alpha.2), phi.2*alpha.2, 1-phi.2,
      0, 0,                 0,                  phi.ad,         1-phi.ad,
      0, 0,                 0,                  phi.ad,         1-phi.ad,
      0, 0,                 0,                  0,              1), nrow =
                                                                n.states, byrow = TRUE)
      } #t
   } #i
```

```
# 2.Observation process matrix
PSI.OBS <- array(NA, dim=c(n.states, n.obs, totrel, n.occasions-1))
for (i in 1:totrel){
   for (t in 1:(n.occasions-1)){
      PSI.OBS[,,i,t] <- matrix(c(
      0, 0,    0,    1,
      0, p.NB, 0,    1-p.NB,
      0, p.NB, 0,    1-p.NB,
      0, 0,    p.B,  1-p.B,
      0, 0,    0,    1), nrow = n.states, byrow = TRUE)
      } #t
   } #i

# Execute simulation function
sim <- simul.ms(PSI.STATE, PSI.OBS, marked)
CH <- sim$CH

# Compute vector with occasion of first capture
get.first <- function(x) min(which(x!=0))
f <- apply(CH, 1, get.first)

# Recode CH matrix: note, a 0 is not allowed!
# 1 = seen as juv, 2 = seen no rep, 3 = seen rep, 4 = not seen
rCH <- CH  # Recoded CH
rCH[rCH==0] <- 4
```

The capture-histories have a "1" when a chick was marked, a "2" when an as yet non-breeding individual is observed, and a "3" when an individual is either observed breeding or it is observed non-breeding but has been observed breeding already at previous occasions. Finally, at occasions when an individual is not observed, the capture-history contains a "4".

9.4.3 Analysis of the Model

Here is the BUGS code for the analysis of the model.

```
# Specify model in BUGS language
sink("agerecruitment.bug")
cat("
model {

# -------------------------------------------------
# Parameters:
# phi.1: first year survival probability
# phi.2: second year survival probability
# phi.ad: adult survival probability
# alpha.1: probability to start breeding when 1 year old
# alpha.2: probability to start breeding when 2 years old
# p.NB: recapture probability of non-breeders
# p.B: recapture probability of breeders
# -------------------------------------------------
# States (S):
# 1 juvenile
# 2 not yet breeding at age 1 year
```

```
# 3 not yet breeding at age 2 years
# 4 breeder
# 5 dead
# Observations (O):
# 1 seen as juvenile
# 2 seen as not yet breeding
# 3 seen breeding
# 4 not seen
# ------------------------------------------------

# Priors and constraints
for (t in 1:(n.occasions-1)){
   phi.1[t] <- mean.phi1
   phi.2[t] <- mean.phi2
   phi.ad[t] <- mean.phiad
   alpha.1[t] <- mean.alpha1
   alpha.2[t] <- mean.alpha2
   p.NB[t] <- mean.pNB
   p.B[t] <- mean.pB
   }
mean.phi1 ~ dunif(0, 1)      # Prior for mean 1y survival
mean.phi2 ~ dunif(0, 1)      # Prior for mean 2y survival
mean.phiad ~ dunif(0, 1)     # Prior for mean ad survival
mean.alpha1 ~ dunif(0, 1)    # Prior for mean 1y breeding prob.
mean.alpha2 ~ dunif(0, 1)    # Prior for mean 2y breeding prob.
mean.pNB ~ dunif(0, 1)       # Prior for mean recapture non-breeders
mean.pB ~ dunif(0, 1)        # Prior for mean recapture breeders

# Define state-transition and observation matrices
for (i in 1:nind){
   # Define probabilities of state S(t+1) given S(t)
   for (t in f[i]:(n.occasions-1)){
      ps[1,i,t,1] <- 0
      ps[1,i,t,2] <- phi.1[t] * (1-alpha.1[t])
      ps[1,i,t,3] <- 0
      ps[1,i,t,4] <- phi.1[t] * alpha.1[t]
      ps[1,i,t,5] <- 1-phi.1[t]
      ps[2,i,t,1] <- 0
      ps[2,i,t,2] <- 0
      ps[2,i,t,3] <- phi.2[t] * (1-alpha.2[t])
      ps[2,i,t,4] <- phi.2[t] * alpha.2[t]
      ps[2,i,t,5] <- 1-phi.2[t]
      ps[3,i,t,1] <- 0
      ps[3,i,t,2] <- 0
      ps[3,i,t,3] <- 0
      ps[3,i,t,4] <- phi.ad[t]
      ps[3,i,t,5] <- 1-phi.ad[t]
      ps[4,i,t,1] <- 0
      ps[4,i,t,2] <- 0
      ps[4,i,t,3] <- 0
      ps[4,i,t,4] <- phi.ad[t]
      ps[4,i,t,5] <- 1-phi.ad[t]
      ps[5,i,t,1] <- 0
```

```
      ps[5,i,t,2] <- 0
      ps[5,i,t,3] <- 0
      ps[5,i,t,4] <- 0
      ps[5,i,t,5] <- 1

   # Define probabilities of O(t) given S(t)
      po[1,i,t,1] <- 0
      po[1,i,t,2] <- 0
      po[1,i,t,3] <- 0
      po[1,i,t,4] <- 1
      po[2,i,t,1] <- 0
      po[2,i,t,2] <- p.NB[t]
      po[2,i,t,3] <- 0
      po[2,i,t,4] <- 1-p.NB[t]
      po[3,i,t,1] <- 0
      po[3,i,t,2] <- p.NB[t]
      po[3,i,t,3] <- 0
      po[3,i,t,4] <- 1-p.NB[t]
      po[4,i,t,1] <- 0
      po[4,i,t,2] <- 0
      po[4,i,t,3] <- p.B[t]
      po[4,i,t,4] <- 1-p.B[t]
      po[5,i,t,1] <- 0
      po[5,i,t,2] <- 0
      po[5,i,t,3] <- 0
      po[5,i,t,4] <- 1
      } #t
   } #i

# Likelihood
for (i in 1:nind){
   # Define latent state at first capture
   z[i,f[i]] <- Y[i,f[i]]
   for (t in (f[i]+1):n.occasions){
      # State process: draw S(t) given S(t-1)
      z[i,t] ~ dcat(ps[z[i,t-1], i, t-1,])
      # Observation process: draw O(t) given S(t)
      y[i,t] ~ dcat(po[z[i,t], i, t-1,])
      } #t
   } #i
}
",fill = TRUE)
sink()
```

It is again possible to provide the known values of the latent states z as data and to estimate only the unknown values of z. However, this is trickier in this example because the labels of the observed states do not correspond with the labels of the latent states (see observation matrix above). For example, individuals that are observed and do not yet breed are labeled with "2", while they are in the latent state "2" or "3", depending on age. Furthermore, individuals that are observed breeding are labeled with "3", while they are in latent state "4". Some serious bookkeeping is

necessary to get the correct latent states for the observed capture-histories, and we avoid this here.

```
# Bundle data
bugs.data <- list(y = rCH, f = f, n.occasions = dim(rCH)[2], nind =
   dim(rCH)[1])

# Initial values (note: function ch.init is defined in section 7.3)
inits <- function(){list(mean.phi1 = runif(1, 0, 1), mean.phi2 =
   runif(1, 0, 1), mean.phiad = runif(1, 0, 1), mean.alpha1 = runif(1, 0,
   1), mean.alpha2 = runif(1, 0, 1), mean.pNB = runif(1, 0, 1), mean.pB =
   runif(1, 0, 1), z = ch.init(rCH, f))}

# Parameters monitored
parameters <- c("mean.phi1", "mean.phi2", "mean.phiad", "mean.alpha1",
   "mean.alpha2", "mean.pNB", "mean.pB")

# MCMC settings
ni <- 2000
nt <- 3
nb <- 1000
nc <- 3

# Call WinBUGS from R (BRT 2 min)
agefirst <- bugs(bugs.data, inits, parameters, "agerecruitment.bug",
   n.chains = nc, n.thin = nt, n.iter = ni, n.burnin = nb, debug = TRUE,
   bugs.directory = bugs.dir, working.directory = getwd())
```

Convergence is achieved quickly, and all parameter estimates are reasonably precise (Fig. 9.6).

```
par(mfrow = c(3, 3), las = 1)
hist(agefirst$sims.list$mean.phi1, col = "gray", main = "",
   xlab = expression(phi[1]))
abline(v = phi.1, col = "red", lwd = 2)
hist(agefirst$sims.list$mean.phi2, col = "gray", main = "",
   xlab = expression(phi[2]), ylab = "")
abline(v = phi.2, col = "red", lwd = 2)
hist(agefirst$sims.list$mean.phiad, col = "gray", main = "",
   xlab = expression(phi[ad]) , ylab = "")
abline(v = phi.ad, col = "red", lwd = 2)
hist(agefirst$sims.list$mean.alpha1, col = "gray", main = "",
   xlab = expression(alpha[1]))
abline(v = alpha.1, col = "red", lwd = 2)
hist(agefirst$sims.list$mean.alpha2, col = "gray", main = "",
   xlab = expression(alpha[2]) , ylab = "")
abline(v = alpha.2, col = "red", lwd = 2)
plot(0, type = "n", axes = F, ylab = "", xlab = "")
hist(agefirst$sims.list$mean.pNB, col = "gray", main = "",
   xlab = expression(p[NB]))
abline(v = p.NB, col = "red", lwd = 2)
hist(agefirst$sims.list$mean.pB, col = "gray", main = "",
   xlab = expression(p[B]) , ylab = "")
abline(v = p.B, col = "red", lwd = 2)
```

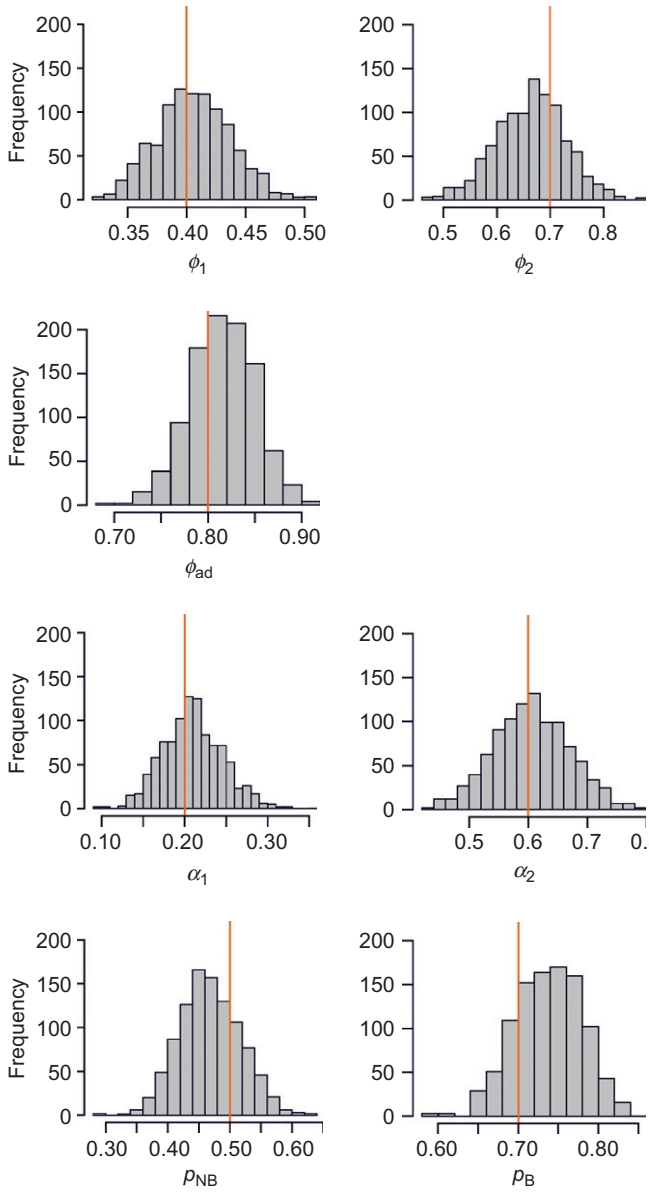

FIGURE 9.6 Posterior distribution of survival, probability of breeding for the first time, and recapture. The vertical red lines show the values used for the simulations.

9.5 JOINT ANALYSIS OF CAPTURE–RECAPTURE AND MARK-RECOVERY DATA

9.5.1 Model Description

Both capture–recapture and mark-recovery data contain information about survival. Sometimes both data types are available in the same study, for example, in bird population studies (Frederiksen and Bregnballe, 2000; Altwegg et al., 2007). It is then sensible to analyze them jointly to make use of all information about survival. The multinomial joint likelihood of this model was originally developed by Burnham (1993), Lebreton et al. (1995), and Barker (1997). More recently, Lebreton et al. (1999) showed how a joint analysis can be performed within the multistate modeling framework, and this is what we do here.

An important difference between capture–recapture and mark-recovery data is that they are often informative about two different kinds of survival probability. As we have seen in Chapter 7, capture–recapture data provide an estimate of apparent survival, which is the probability of surviving *and* remaining in the study area. By contrast, true survival can be estimated from mark-recovery data (Chapter 8). The difference between the two estimates of survival arises because sampling of marked individuals in capture–recapture studies is restricted to the study area (an exception might be studies using color marks), whereas dead recoveries can be obtained from anywhere. Apparent survival (ϕ) and true survival (s) are linked through site fidelity (F) via the relationship $\phi = s*F$. A joint analysis allows one to estimate all three parameters.

The multistate model for the joint analysis of capture–recapture and mark-recovery data has four states: "alive in the study area", "alive outside the study area", "recently dead", and "dead". It may seem strange to have two dead states; however, this is necessary because only individuals recently dead can be recovered. An individual in the state "recently dead" at occasion t has died between occasions $t-1$ and t. From occasion $t+1$ onwards, the individual can no longer be recovered. This assumption applies to the analysis of mark-recovery data as well (Section 8.2). Therefore, "recently dead" individuals move to the state "dead" at the next occasion. One says that "dead" is an absorbing state; once individuals are in it, they cannot get out anymore. The transition matrix of the joint model looks like the following:

	alive, inside	alive, outside	recently dead	dead
alive, inside	sF	$s(1-F)$	$1-s$	0
alive, outside	0	s	$1-s$	0
recently dead	0	0	0	1
dead	0	0	0	1

The parameters in the transition matrix are the true survival probability (s) and fidelity probability (F). Fidelity is defined as the probability to remain in the study area, given that an individual is alive. Thus, it is the complement of the probability to emigrate permanently from the study population ($1 - F$). Permanent emigration denotes the permanent movement of individuals outside the study area. It is different from temporary emigration, which allows multiple exits and entries to the study population (see Section 9.3). The distinction between these two types of emigration is crucial.

The possible observations are "seen alive", "recovered dead", and "not seen or recovered", and the observation matrix is

	seen alive	recovered dead	not seen or recovered
alive, inside	p	0	$1-p$
alive, outside	0	0	1
recently dead	0	r	$1-r$
dead	0	0	1

The parameters in the observation matrix are the recapture (p) and the recovery probabilities (r). The recapture probability is the probability to encounter an individual alive and in the study area, whereas the recovery probability is the probability to find and report an individual in the state "recently dead". Both parameters are defined in the same way as the corresponding parameters of the CJS (Chapter 7) and the mark-recovery model (Chapter 8).

9.5.2 Generation of Simulated Data

We assume we want to estimate the adult survival of little ringed plovers (Fig. 9.2) that are marked with ordinary rings but also with colored wing tags, so that individuals can be encountered both live and dead. We imagine that adult survival (s) is 0.8, and study site fidelity is high ($F = 0.6$). The resighting probability is moderate ($p = 0.5$), whereas the recovery probability is low ($r = 0.1$). We simulate a study over 10 years, with 100 individuals marked each year.

```
# Define mean survival, transitions, recapture, as well as number of
  occasions, states, observations and released individuals
s <- 0.8
F <- 0.6
r <- 0.1
p <- 0.5
n.occasions <- 10
n.states <- 4
n.obs <- 3
marked <- matrix(0, ncol = n.states, nrow = n.occasions)
marked[,1] <- rep(100, n.occasions)        # Releases in study area
```

```
# Define matrices with survival, transition and recapture probabilities
# These are 4-dimensional matrices, with
    # Dimension 1: state of departure
    # Dimension 2: state of arrival
    # Dimension 3: individual
    # Dimension 4: time
# 1. State process matrix
totrel<- sum(marked)*(n.occasions-1)
PSI.STATE<- array(NA, dim=c(n.states, n.states, totrel, n.occasions-1))
for (i in 1:totrel){
    for (t in 1:(n.occasions-1)){
        PSI.STATE[,,i,t] <- matrix(c(
        s*F, s*(1-F), 1-s, 0,
        0,   s,       1-s, 0,
        0,   0,       0,   1,
        0,   0,       0,   1), nrow=n.states, byrow=TRUE)
        } #t
    } #i

# 2.Observation process matrix
PSI.OBS <- array(NA, dim=c(n.states, n.obs, totrel, n.occasions-1))
for (i in 1:totrel){
    for (t in 1:(n.occasions-1)){
        PSI.OBS[,,i,t] <- matrix(c(
        p, 0, 1-p,
        0, 0, 1,
        0, r, 1-r,
        0, 0, 1), nrow=n.states, byrow=TRUE)
        } #t
    } #i

# Execute simulation function
sim <- simul.ms(PSI.STATE, PSI.OBS, marked)
CH <- sim$CH

# Compute date of first capture
get.first <- function(x) min(which(x!=0))
f <- apply(CH, 1, get.first)

# Recode CH matrix: note, a 0 is not allowed!
# 1=alive and in study are, 2=recovered dead, 3=not seen or recovered
rCH <- CH # Recoded CH
rCH[rCH==0] <- 3
```

9.5.3 Analysis of the Model

In principle, it should be possible to analyze the model in WinBUGS with code that directly translates the two matrices of the previous section, and thus in an analogous way as we have done for the other models in this chapter. However, due to a reason unknown to us, WinBUGS does not update the parameters properly, in particular the latent states z. It seems that the problem has to do with the recoveries; if they are excluded,

the problem goes away. By educated trial and error, we found out that a reparameterization of the model works fine, where the recovery probability is included in the state transition matrix:

$$\begin{bmatrix} s & s(1-F) & (1-s)r & (1-s)(1-r) \\ 0 & s & (1-s)r & (1-s)(1-r) \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The observation matrix then no longer contains the recovery probability:

$$\begin{bmatrix} p & 0 & 1-p \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

This model is then fitted in WinBUGS:

```
# Specify model in BUGS language
sink("lifedead.bug")
cat("
model {

# -----------------------------------------------
# Parameters:
# s: true survival probability
# F: fidelity probability
# r: recovery probability
# p: recapture/resighting probability
# -----------------------------------------------
# States (S):
# 1 alive in study area
# 2 alive outside study area
# 3 recently dead and recovered
# 4 recently dead, but not recovered, or dead (absorbing)
# Observations (O):
# 1 seen alive
# 2 recovered dead
# 3 neither seen nor recovered
# -----------------------------------------------

# Priors and constraints
for (t in 1:(n.occasions-1)){
   s[t] <- mean.s
   F[t] <- mean.f
   r[t] <- mean.r
   p[t] <- mean.p
   }
mean.s ~ dunif(0, 1)      # Prior for mean survival
mean.f ~ dunif(0, 1)      # Prior for mean fidelity
```

```
mean.r ~ dunif(0, 1)      # Prior for mean recovery
mean.p ~ dunif(0, 1)      # Prior for mean recapture

# Define state-transition and observation matrices
for (i in 1:nind){
   # Define probabilities of state S(t+1) given S(t)
   for (t in f[i]:(n.occasions-1)){
      ps[1,i,t,1] <- s[t]*F[t]
      ps[1,i,t,2] <- s[t]*(1-F[t])
      ps[1,i,t,3] <- (1-s[t])*r[t]
      ps[1,i,t,4] <- (1-s[t])*(1-r[t])
      ps[2,i,t,1] <- 0
      ps[2,i,t,2] <- s[t]
      ps[2,i,t,3] <- (1-s[t])*r[t]
      ps[2,i,t,4] <- (1-s[t])*(1-r[t])
      ps[3,i,t,1] <- 0
      ps[3,i,t,2] <- 0
      ps[3,i,t,3] <- 0
      ps[3,i,t,4] <- 1
      ps[4,i,t,1] <- 0
      ps[4,i,t,2] <- 0
      ps[4,i,t,3] <- 0
      ps[4,i,t,4] <- 1

      # Define probabilities of O(t) given S(t)
      po[1,i,t,1] <- p[t]
      po[1,i,t,2] <- 0
      po[1,i,t,3] <- 1-p[t]
      po[2,i,t,1] <- 0
      po[2,i,t,2] <- 0
      po[2,i,t,3] <- 1
      po[3,i,t,1] <- 0
      po[3,i,t,2] <- 1
      po[3,i,t,3] <- 0
      po[4,i,t,1] <- 0
      po[4,i,t,2] <- 0
      po[4,i,t,3] <- 1
      } #t
   } #i

# Likelihood
for (i in 1:nind){
   # Define latent state at first capture
   z[i,f[i]] <- Y[i,f[i]]
   for (t in (f[i]+1):n.occasions){
      # State process: draw S(t) given S(t-1)
      z[i,t] ~ dcat(ps[z[i,t-1], i, t-1,])
      # Observation process: draw O(t) given S(t)
      y[i,t] ~ dcat(po[z[i,t], i, t-1,])
      } #t
   } #i
}
",fill = TRUE)
sink()
```

```
# Bundle data
bugs.data <- list(y = rCH, f = f, n.occasions = dim(rCH)[2], nind = dim
   (rCH)[1])

# Initial values (note: function ch.init is defined in section 7.3)
inits <- function(){list(mean.s = runif(1, 0, 1), mean.f = runif(1, 0, 1),
   mean.p = runif(1, 0, 1), mean.r = runif(1, 0, 1), z = ch.init(CH, f))}

# Parameters monitored
parameters <- c("mean.s", "mean.f", "mean.r", "mean.p")

# MCMC settings
ni <- 40000
nt <- 10
nb <- 10000
nc <- 3

# Call WinBUGS from R (BRT 80 min)
lifedead <- bugs(bugs.data, inits, parameters, "lifedead.bug",
   n.chains = nc, n.thin = nt, n.iter = ni, n.burnin = nb, debug = TRUE,
   bugs.directory = bugs.dir)
```

The model does not converge easily, and long chains are necessary to obtain satisfactory results.

```
print(lifedead, digit = 3)
        mean    sd    2.5%    25%    50%    75%  97.5%   Rhat n.eff
mean.s 0.843 0.051   0.746  0.808  0.841  0.882  0.935  1.004   590
mean.f 0.606 0.041   0.533  0.576  0.605  0.634  0.692  1.003   770
mean.r 0.145 0.045   0.089  0.114  0.133  0.162  0.262  1.004   730
mean.p 0.515 0.028   0.461  0.496  0.515  0.534  0.572  1.001  4600
```

This model can be extended in several ways, for instance, by including mortality causes (Schaub and Pradel, 2004; Schaub, 2009; Servanthy et al., 2010), movement between sites (Kendall et al., 2006; Duriez et al., 2009), or age structures, to name just a few. Obviously, a multistate model could be used when the goal is to analyze single-state capture–recapture data alone (the true states being "alive" and "dead", and the observed states being "seen" and "not seen"), or if the goal is to analyze mark-recovery data alone (the true states being "alive", "recently dead and recovered", and "recently dead (not recovered) or dead", and the observed states being "recovered" and "not recovered").

9.6 ESTIMATION OF MOVEMENT AMONG THREE SITES

9.6.1 Model Description

In this example, we estimate movement among three sites. The list of states includes "alive at site A", "alive at site B", "alive at site C", and "dead", and the list of observations is "seen at A", "seen at B", "seen at C", and "not seen". The state transition matrix is

$$\begin{array}{c} \\ \text{site A} \\ \text{site B} \\ \text{site C} \\ \text{dead} \end{array} \begin{array}{c} \begin{array}{cccc} \text{site A} & \text{site B} & \text{site C} & \text{dead} \end{array} \\ \begin{bmatrix} \phi_A(1-\psi_{AB}-\psi_{AC}) & \phi_A\psi_{AB} & \phi_A\psi_{AB} & 1-\phi_A \\ \phi_B\psi_{BA} & \phi_B(1-\psi_{BA}-\psi_{BC}) & \phi_B\psi_{BC} & 1-\phi_B \\ \phi_C\psi_{CA} & \phi_C\psi_{CB} & \phi_C(1-\psi_{CA}-\psi_{CB}) & 1-\phi_C \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{array},$$

and the observation matrix is

$$\begin{array}{c} \\ \text{site A} \\ \text{site B} \\ \text{site C} \\ \text{dead} \end{array} \begin{array}{c} \begin{array}{cccc} \text{seen at A} & \text{seen at B} & \text{seen at C} & \text{not seen} \end{array} \\ \begin{bmatrix} p_A & 0 & 0 & 1-p_A \\ 0 & p_B & 0 & 1-p_B \\ 0 & 0 & p_C & 1-p_C \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

We here extend the multistate model for two sites (Section 9.2) because there is an additional challenge in the constraints that need to be put on the movement probabilities. For each site, we now have three movement probabilities. For instance, from site A, we have ψ_{AA}, ψ_{AB}, and ψ_{AC}. Two sets of constraints must be imposed on their estimates: first, each of them must be in the interval [0, 1] and second, they must sum to 1. With only two transition probabilities as in all examples before, this is easy to achieve: we give a uniform or a beta prior to one of them and calculate the other as the complement to 1. With more than two states, this is no longer so easy, but there are two possible solutions: the use of a multinomial logit link function for the transition probabilities or the use of a Dirichlet distribution as a prior for the transition probabilities. Below, we illustrate both.

For a multinomial link function, we specify a normal prior distribution for $n-1$ transition parameters (α). These correspond to the transition probabilities on the logit scale, and they can be modeled using the GLM framework. This is exactly equivalent to the use of the logit function when the parameters on the logit scale are modeled with a GLM. The back-transformation for each of the $n-1$ parameters is calculated as

$$\beta_j = \frac{\exp(\alpha_j)}{1+\sum_{i=1}^{n-1}\exp(\alpha_i)},$$

which ensures that each of them, as well as their sum, is <1. The last parameter is calculated as $\beta_n = 1 - \sum_{i=1}^{n-1}\beta_i$. In this model, only $n-1$ parameters can be modeled directly with a GLM, and the last parameter is modified

indirectly, being a function of the covariates used in the GLM for the $n-1$ modeled parameters.

The second option is the use of a Dirichlet prior for the βs, which automatically ensures that they are in the interval [0, 1] and sum to 1. For the Dirichlet prior distribution, hyperparameters must be chosen. With three transition probabilities that must sum to one, a vector [1, 1, 1] induces a noninformative Dirichlet prior. This vector of hyperparameters is best given as data.

The Dirichlet prior for the βs can also be specified indirectly. This option works well in WinBUGS by specification of a set of gamma (1, 1) hyperprior random variables, say, $\alpha_j \sim$ gamma(1, 1), followed by the relation $\beta_j = \alpha_j \Big/ \sum_{i=1}^{n} \alpha_i$ (Royle and Dorazio, 2008).

In the following example, we use the multinomial logit link function, whereas for the model in Section 9.7, we construct the Dirichlet prior distribution using gamma hyperpriors. The latter results in faster convergence, at least in our examples.

9.6.2 Generation of Simulated Data

We extend the previous little ringed plover example to three sites. We assume that habitat quality at site A is higher than at sites B and C, and thus annual survival at site A is higher ($\phi_A = 0.85$) than at site B ($\phi_B = 0.75$) and C ($\phi_C = 0.65$). Furthermore, movement away from A is less likely ($\psi_{AB} = 0.3$, $\psi_{AC} = 0.2$) than movement to A ($\psi_{BA} = 0.5$, $\psi_{CA} = 0.6$). Movement rates between B and C are identical ($\psi_{BC} = 0.1$, $\psi_{CB} = 0.1$). Capture effort is greater at site A than at B or C resulting in the following recapture probabilities: $p_A = 0.7$, $p_B = 0.4$, and $p_C = 0.5$. Captures at site A are labeled "1", at site B "2", and at site C "3".

```
# Define mean survival, transitions, recapture, as well as number of
  occasions, states, observations and released individuals
phiA <- 0.85
phiB <- 0.75
phiC <- 0.65
psiAB <- 0.3
psiAC <- 0.2
psiBA <- 0.5
psiBC <- 0.1
psiCA <- 0.6
psiCB <- 0.1
pA <- 0.7
pB <- 0.4
pC <- 0.5
```

```
n.occasions <- 6
n.states <- 4
n.obs <- 4
marked <- matrix(NA, ncol = n.states, nrow = n.occasions)
marked[,1] <- rep(50, n.occasions)
marked[,2] <- rep(50, n.occasions)
marked[,3] <- rep(50, n.occasions)
marked[,4] <- rep(0, n.occasions)

# Define matrices with survival, transition and recapture probabilities
# These are 4-dimensional matrices, with
   # Dimension 1: state of departure
   # Dimension 2: state of arrival
   # Dimension 3: individual
   # Dimension 4: time
# 1. State process matrix
totrel <- sum(marked)*(n.occasions-1)
PSI.STATE <- array(NA, dim=c(n.states, n.states, totrel, n.occasions-1))
for (i in 1:totrel){
   for (t in 1:(n.occasions-1)){
      PSI.STATE[,,i,t] <- matrix(c(
      phiA*(1-psiAB-psiAC), phiA*psiAB,          phiA*psiAC,          1-phiA,
      phiB*psiBA,           phiB*(1-psiBA-psiBC), phiB*psiBC,          1-phiB,
      phiC*psiCA,           phiC*psiCB,          phiC*(1-psiCA-psiCB), 1-phiC,
      0,                    0,                   0,                   1),
        nrow = n.states, byrow = TRUE)
       } #t
    } #i

# 2.Observation process matrix
PSI.OBS <- array(NA, dim=c(n.states, n.obs, totrel, n.occasions-1))
for (i in 1:totrel){
   for (t in 1:(n.occasions-1)){
      PSI.OBS[,,i,t] <- matrix(c(
      pA, 0, 0, 1-pA,
      0, pB, 0, 1-pB,
      0, 0, pC, 1-pC,
      0, 0, 0, 1), nrow = n.states, byrow = TRUE)
      } #t
   } #i

# Execute simulation function
sim <- simul.ms(PSI.STATE, PSI.OBS, marked)
CH <- sim$CH

# Compute vector with occasions of first capture
get.first <- function(x) min(which(x!=0))
f <- apply(CH, 1, get.first)

# Recode CH matrix: note, a 0 is not allowed in WinBUGS!
# 1 = seen alive in A, 2 = seen alive in B, 3, seen alive in C, 4 = not seen
rCH <- CH # Recoded CH
rCH[rCH==0] <- 4
```

9.6.3 Analysis of the Model

We first write the model with the multinomial logit link.

```
# Specify model in BUGS language
sink("ms3-multinomlogit.bug")
cat("
model {

# ------------------------------------------------
# Parameters:
# phiA: survival probability at site A
# phiB: survival probability at site B
# phiC: survival probability at site C
# psiAB: movement probability from site A to site B
# psiAC: movement probability from site A to site C
# psiBA: movement probability from site B to site A
# psiBC: movement probability from site B to site C
# psiCA: movement probability from site C to site A
# psiCB: movement probability from site C to site B
# pA: recapture probability at site A
# pB: recapture probability at site B
# pC: recapture probability at site C
# ------------------------------------------------
# States (S):
# 1 alive at A
# 2 alive at B
# 3 alive at C
# 4 dead
# Observations (O):
# 1 seen at A
# 2 seen at B
# 3 seen at C
# 4 not seen
# ------------------------------------------------

# Priors and constraints
   # Survival and recapture: uniform
   phiA ~ dunif(0, 1)
   phiB ~ dunif(0, 1)
   phiC ~ dunif(0, 1)
   pA ~ dunif(0, 1)
   pB ~ dunif(0, 1)
   pC ~ dunif(0, 1)
   # Transitions: multinomial logit
      # Normal priors on logit of all but one transition prob.
      for (i in 1:2){
         lpsiA[i] ~ dnorm(0, 0.001)
         lpsiB[i] ~ dnorm(0, 0.001)
         lpsiC[i] ~ dnorm(0, 0.001)
         }
      # Constrain the transitions such that their sum is < 1
      for (i in 1:2){
         psiA[i] <- exp(lpsiA[i]) / (1 + exp(lpsiA[1]) + exp(lpsiA[2]))
```

```
      psiB[i] <- exp(lpsiB[i]) / (1 + exp(lpsiB[1]) + exp(lpsiB[2]))
      psiC[i] <- exp(lpsiC[i]) / (1 + exp(lpsiC[1]) + exp(lpsiC[2]))
      }
    # Calculate the last transition probability
    psiA[3] <- 1-psiA[1]-psiA[2]
    psiB[3] <- 1-psiB[1]-psiB[2]
    psiC[3] <- 1-psiC[1]-psiC[2]

# Define state-transition and observation matrices
for (i in 1:nind){
  # Define probabilities of state S(t+1) given S(t)
  for (t in f[i]:(n.occasions-1)){
    ps[1,i,t,1] <- phiA * psiA[1]
    ps[1,i,t,2] <- phiA * psiA[2]
    ps[1,i,t,3] <- phiA * psiA[3]
    ps[1,i,t,4] <- 1-phiA
    ps[2,i,t,1] <- phiB * psiB[1]
    ps[2,i,t,2] <- phiB * psiB[2]
    ps[2,i,t,3] <- phiB * psiB[3]
    ps[2,i,t,4] <- 1-phiB
    ps[3,i,t,1] <- phiC * psiC[1]
    ps[3,i,t,2] <- phiC * psiC[2]
    ps[3,i,t,3] <- phiC * psiC[3]
    ps[3,i,t,4] <- 1-phiC
    ps[4,i,t,1] <- 0
    ps[4,i,t,2] <- 0
    ps[4,i,t,3] <- 0
    ps[4,i,t,4] <- 1

    # Define probabilities of O(t) given S(t)
    po[1,i,t,1] <- pA
    po[1,i,t,2] <- 0
    po[1,i,t,3] <- 0
    po[1,i,t,4] <- 1-pA
    po[2,i,t,1] <- 0
    po[2,i,t,2] <- pB
    po[2,i,t,3] <- 0
    po[2,i,t,4] <- 1-pB
    po[3,i,t,1] <- 0
    po[3,i,t,2] <- 0
    po[3,i,t,3] <- pC
    po[3,i,t,4] <- 1-pC
    po[4,i,t,1] <- 0
    po[4,i,t,2] <- 0
    po[4,i,t,3] <- 0
    po[4,i,t,4] <- 1
    } #t
  } #i

# Likelihood
for (i in 1:nind){
  # Define latent state at first capture
  z[i,f[i]] <- Y[i,f[i]]
  for (t in (f[i]+1):n.occasions){
```

```
      # State process: draw S(t) given S(t-1)
      z[i,t] ~ dcat(ps[z[i,t-1], i, t-1,])
      # Observation process: draw O(t) given S(t)
      y[i,t] ~ dcat(po[z[i,t], i, t-1,])
      } #t
   } #i
}
",fill = TRUE)
sink()

# Bundle data
bugs.data <- list(y = rCH, f = f, n.occasions = dim(rCH)[2], nind = dim
   (rCH)[1], z = known.state.ms(rCH, 4))

# Initial values
inits <- function(){list(phiA = runif(1, 0, 1), phiB = runif(1, 0, 1),
   phiC = runif(1, 0, 1), lpsiA = rnorm(2), lpsiB = rnorm(2), lpsiC =
   rnorm(2), pA = runif(1, 0, 1) , pB = runif(1, 0, 1) , pC = runif(1, 0, 1),
   z = ms.init.z(rCH, f))}

# Parameters monitored
parameters <- c("phiA", "phiB", "phiC", "psiA", "psiB", "psiC", "pA",
   "pB", "pC")

# MCMC settings
ni <- 50000
nt <- 6
nb <- 20000
nc <- 3

# Call WinBUGS from R (BRT 56 min)
ms3 <- bugs(bugs.data, inits, parameters, "ms3-multinomlogit.bug",
   n.chains = nc, n.thin = nt, n.iter = ni, n.burnin = nb, debug = TRUE,
   bugs.directory = bugs.dir, working.directory = getwd())
```

This model runs slowly and convergence is hard to achieve. To increase computation speed, we have provided the known values of the latent state `z` (see `bugs.data`). The parameter estimates look like the following:

```
print(ms3, digits = 3)
          mean    sd  2.5%   25%   50%   75% 97.5%  Rhat n.eff
phiA     0.833 0.029 0.780 0.813 0.832 0.851 0.893 1.001  8600
phiB     0.730 0.033 0.664 0.708 0.731 0.753 0.791 1.001 15000
phiC     0.676 0.036 0.607 0.651 0.676 0.699 0.747 1.001 15000
psiA[1]  0.546 0.080 0.395 0.491 0.546 0.602 0.702 1.001 15000
psiA[2]  0.310 0.070 0.173 0.262 0.310 0.357 0.449 1.002  1300
psiA[3]  0.144 0.040 0.091 0.115 0.135 0.165 0.240 1.007   320
psiB[1]  0.551 0.097 0.373 0.482 0.546 0.616 0.751 1.001 15000
psiB[2]  0.375 0.088 0.189 0.317 0.381 0.438 0.531 1.001  8400
psiB[3]  0.074 0.030 0.032 0.051 0.068 0.089 0.151 1.003  1000
psiC[1]  0.740 0.072 0.572 0.697 0.750 0.793 0.849 1.001 13000
psiC[2]  0.096 0.044 0.034 0.065 0.089 0.119 0.205 1.002  1300
psiC[3]  0.164 0.053 0.089 0.125 0.153 0.193 0.292 1.004   570
pA       0.671 0.085 0.535 0.610 0.662 0.720 0.870 1.001 15000
pB       0.456 0.132 0.280 0.366 0.427 0.514 0.817 1.002  1600
pC       0.732 0.170 0.398 0.601 0.749 0.877 0.988 1.005   450
```

9.7 REAL-DATA EXAMPLE: THE SHOWY LADY'S SLIPPER

Here, we want to estimate the survival probability of a plant species, the beautiful showy lady's slipper (Fig. 9.7). You may ask yourself why there is a need to apply multistate capture–recapture models to plant data; after all, don't plants just stand still and wait to be counted (Harper, 1977)? Couldn't we simply use logistic regression models for survival estimation? Well, this is only partly true. If a plant remains aboveground throughout its life, such as an oak, then this may be true. However, some plants are temporally unobservable because they are dormant and thus live underground for one or several years. Since individuals may also die when they are dormant, we have to use probabilistic models. Moreover, there is an interest in modeling the transition probabilities to and from the dormant state, as well as between different aboveground states, such as vegetative and reproductive. Knowing these transition probabilities is important for setting up matrix projection models for such species. Examples of the application of capture–recapture models (both single- and multistate models) for plants are Shefferson et al. (2001) and Kéry et al. (2005a).

FIGURE 9.7 Showy lady's slipper (*Cypripedium reginae*), West Virginia, 2010 (Photograph by K. Gregg).

The multistate capture–recapture data analyzed here were collected by Kathy Gregg in Big Draft (West Virginia) from 1989 to 1999. Kéry and Gregg (2004) analyzed them in the frequentist framework. Three live states can be distinguished for the showy lady's slipper: dormant, vegetative, and flowering. We would like to estimate the probabilities of state transition and of survival of this species. For the state-transition matrix, we define four states: "vegetative", "flowering", "dormant", and "dead".

$$\begin{array}{c} \\ \text{vegetative} \\ \text{flowering} \\ \text{dormant} \\ \text{dead} \end{array} \begin{array}{c} \begin{array}{cccc} \text{vegetative} & \text{flowering} & \text{dormant} & \text{dead} \end{array} \\ \begin{bmatrix} s\psi_{VV} & s\psi_{VF} & s(1-\psi_{VV}-\psi_{VF}) & 1-s \\ s\psi_{FV} & s\psi_{FF} & s(1-\psi_{FV}-\psi_{FF}) & 1-s \\ s\psi_{DV} & s\psi_{DF} & s(1-\psi_{DV}-\psi_{DF}) & 1-s \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{array}.$$

Permanent emigration is impossible, so we can estimate true survival; hence, the symbol s instead of ϕ. For the observation matrix, we define the observations "seen vegetative", "seen flowering", and "not seen". We assume that no plants were missed in the states "vegetative" and "flowering", but they cannot be observed when they are either "dormant" or "dead". Thus, the observation matrix is completely deterministic:

$$\begin{array}{c} \\ \text{vegetative} \\ \text{flowering} \\ \text{dormant} \\ \text{dead} \end{array} \begin{array}{c} \begin{array}{ccc} \text{seen veg.} & \text{seen flow.} & \text{not seen} \end{array} \\ \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} \end{array}.$$

In the data, the vegetative state is coded as 1, the flowering state as 2, and failure to observe an individual as 0. We now read the data in an R workspace

```
CH <- as.matrix(read.table("orchids.txt", sep=" ", header = F))
n.occasions <- dim(CH)[2]

# Compute vector with occasion of first capture
f <- numeric()
for (i in 1:dim(CH)[1]){f[i] <- min(which(CH[i,]!=0))}

# Recode CH matrix: note, a 0 is not allowed by WinBUGS!
# 1 = seen vegetative, 2 = seen flowering, 3 = not seen
rCH <- CH # Recoded CH
rCH[rCH==0] <- 3
```

Kéry and Gregg (2004) found that a model with year-specific survival and constant transition probabilities was most parsimonious, and we will use this parameter structure here. Since movement is possible among three live states, the transition probabilities must meet the constraints that each of them is between 0 and 1 and that they sum to 1. Here, we implement

the Dirichlet prior distribution using elemental gamma random variables as described in Section 9.6.1.

```
# Specify model in BUGS language
sink("ladyslipper.bug")
cat("
model {

# -----------------------------------------------
# Parameters:
# s: survival probability
# psiV: transitions from vegetative
# psiF: transitions from flowering
# psiD: transitions from dormant
# -----------------------------------------------
# States (S):
# 1 vegetative
# 2 flowering
# 3 dormant
# 4 dead
# Observations (O):
# 1 seen vegetative
# 2 seen flowering
# 3 not seen
# -----------------------------------------------

# Priors and constraints
   # Survival: uniform
   for (t in 1:(n.occasions-1)){
      s[t] ~ dunif(0, 1)
      }
   # Transitions: gamma priors
   for (i in 1:3){
      a[i] ~ dgamma(1, 1)
      psiD[i] <- a[i]/sum(a[])
      b[i] ~ dgamma(1, 1)
      psiV[i] <- b[i]/sum(b[])
      c[i] ~ dgamma(1, 1)
      psiF[i] <- c[i]/sum(c[])
      }

# Define state-transition and observation matrices
for (i in 1:nind)
   # Define probabilities of state S(t+1) given S(t)
   for (t in 1:(n.occasions-1)){
      ps[1,i,t,1] <- s[t] * psiV[1]
      ps[1,i,t,2] <- s[t] * psiV[2]
      ps[1,i,t,3] <- s[t] * psiV[3]
      ps[1,i,t,4] <- 1-s[t]
      ps[2,i,t,1] <- s[t] * psiF[1]
      ps[2,i,t,2] <- s[t] * psiF[2]
      ps[2,i,t,3] <- s[t] * psiF[3]
      ps[2,i,t,4] <- 1-s[t]
```

```
      ps[3,i,t,1] <- s[t] * psiD[1]
      ps[3,i,t,2] <- s[t] * psiD[2]
      ps[3,i,t,3] <- s[t] * psiD[3]
      ps[3,i,t,4] <- 1-s[t]
      ps[4,i,t,1] <- 0
      ps[4,i,t,2] <- 0
      ps[4,i,t,3] <- 0
      ps[4,i,t,4] <- 1

      # Define probabilities of O(t) given S(t)
      po[1,i,t,1] <- 1
      po[1,i,t,2] <- 0
      po[1,i,t,3] <- 0
      po[2,i,t,1] <- 0
      po[2,i,t,2] <- 1
      po[2,i,t,3] <- 0
      po[3,i,t,1] <- 0
      po[3,i,t,2] <- 0
      po[3,i,t,3] <- 1
      po[4,i,t,1] <- 0
      po[4,i,t,2] <- 0
      po[4,i,t,3] <- 1
      } #t
} #i

# Likelihood
for (i in 1:nind){
   # Define latent state at first capture
   z[i,f[i]] <- Y[i,f[i]]
   for (t in (f[i]+1):n.occasions){

      # State process: draw S(t) given S(t-1)
      z[i,t] ~ dcat(ps[z[i,t-1], i, t-1,])

      # Observation process: draw O(t) given S(t)
      y[i,t] ~ dcat(po[z[i,t], i, t-1,])
      } #t
   } #i
}
",fill=TRUE)
sink()

# Bundle data
bugs.data <- list(y = rCH, f = f, n.occasions = dim(rCH)[2],
   nind = dim(rCH)[1], z = known.state.ms(rCH, 3))

# Initial values
inits <- function(){list(s = runif((dim(rCH)[2]-1),0,1),
   z = ms.init.z(rCH, f))}

# Parameters monitored
parameters <- c("s", "psiV", "psiF", "psiD")

# MCMC settings
ni <- 5000
nt <- 3
```

```
nb <- 2000
nc <- 3

# Call WinBUGS from R (BRT 3 min)
ls <- bugs(bugs.data, inits, parameters, "ladyslipper.bug", n.chains =
   nc, n.thin = nt, n.iter = ni, n.burnin = nb, debug = TRUE,
   bugs.directory = bugs.dir, working.directory = getwd())
print(ls, digits = 3)
          mean    sd  2.5%   25%   50%   75% 97.5%  Rhat n.eff
s[1]     0.984 0.013 0.952 0.977 0.986 0.993 0.999 1.001  3000
[ ... ]
s[10]    0.982 0.013 0.951 0.975 0.984 0.992 0.999 1.001  3000
psiV[1]  0.830 0.012 0.806 0.822 0.830 0.838 0.852 1.001  3000
psiV[2]  0.151 0.011 0.130 0.143 0.151 0.159 0.174 1.002  1400
psiV[3]  0.019 0.005 0.011 0.016 0.019 0.022 0.030 1.002  1800
psiF[1]  0.183 0.018 0.149 0.171 0.183 0.195 0.221 1.001  3000
psiF[2]  0.803 0.019 0.763 0.790 0.803 0.816 0.838 1.001  3000
psiF[3]  0.014 0.006 0.005 0.010 0.013 0.018 0.028 1.001  3000
psiD[1]  0.554 0.101 0.348 0.486 0.557 0.625 0.742 1.005   990
psiD[2]  0.172 0.069 0.059 0.122 0.166 0.212 0.327 1.001  3000
psiD[3]  0.274 0.099 0.106 0.200 0.265 0.337 0.484 1.002  2000
```

Note how the estimated transitions relate to those presented in the state-transition matrix above. The estimate psiV[1] is ψ_{VV}, psiV[2] is ψ_{VF}, psiV[3] is $1 - \psi_{VV} - \psi_{VF}$, and so on. The estimated parameter estimates match very well the results from the published frequentist analysis of the model for this data set (Kéry and Gregg, 2004).

9.8 SUMMARY AND OUTLOOK

In this chapter, we have introduced an important class of capture–recapture models: multistate models. Multistate models represent a very general framework and other models such as the CJS (Chapter 7), the mark-recovery (Chapter 8), and the JS model (Chapter 10) can be described as special cases (Lebreton et al., 1999, 2009). Multistate models are extremely flexible and can be used to address a large array of very diverse ecological questions. The flexibility stems from the fact that the definition of states can involve many different quantities, as seen in the examples, which go far beyond the classical geographic locations. In addition to the examples that we have provided in this chapter, we present some further models in Appendix 2 (including data simulation and BUGS model code). To analyze multistate models in the Bayesian framework, we use the state-space formulation.

Multistate capture–recapture models can be extended in exactly the same ways as we have shown for the single-state models. Thus, we may model several groups, time-dependent parameters, and individual or temporal covariates, and group effects can be assumed fixed or

random. The data and most parts of the BUGS model code remain the same, but there are modifications in the "Priors and constraints" part of the model, in exactly the same way as was shown in Chapters 6 and 7. The exercises and their solutions provide some examples of such extensions. Multistate models have also been combined with other model classes, such as site-occupancy models (Section 13.6).

Multistate capture–recapture models sometimes suffer from identifiability problems, both intrinsic and extrinsic. Hence, it is important to check for identifiability, in particular when adopting less well-known multistate model variants. To check identifiability, the method introduced in Section 7.9 can be used. Moreover, we recommend assessing the goodness of fit using χ^2-decompositions developed by Pradel et al. (2003).

Recently, an extremely general extension of the general multistate model to include state uncertainty has been developed (Pradel, 2005). E-SURGE (Choquet et al. 2009b) is a flexible software to fit such models using maximum likelihood. These so-called multievent models belong to the class of hidden Markov models and can also be fitted using WinBUGS. An important difference is that in the conventional multistate models of this chapter, there is no error in the state assignment, whereas in multievent models, state assignment errors can occur. In other words, multistate models allow only false-negative errors, while multievent models allow both false-negative and false-positive errors. The state-space formulation used here to fit multistate models can be adapted to include state assignment errors; see also Section 13.6. The observation matrix must then be written in such a way that false state assignments are possible. A further difference is that there is also uncertainty about the state at first observation in the multievent model. We can model this uncertainty by estimating a probability of state membership at first encounter based on the observed states. This means that we have several additional parameters in the model that require priors.

9.9 EXERCISES

1. Simulate multistate capture–recapture data for two sexes (m, f) in two populations (A, B) that are connected by dispersal. Assume that movement rates between populations are the same for both sexes, but that site-specific survival and recapture differ among populations. The simulation parameters are $\phi_{A,m} = 0.5$, $\phi_{B,m} = 0.6$, $\phi_{A,f} = 0.7$, $\phi_{B,f} = 0.6$, $\psi_{AB} = 0.2$, $\psi_{BA} = 0.5$, $p_{A,m} = 0.3$, $p_{B,m} = 0.7$, $p_{A,f} = 0.4$, $p_{B,f} = 0.8$, occasions = 6, and 20 males and females are released at each population in each year. Simulate the data and analyze them.
2. Simulate multistate capture–recapture data from two populations observed over 8 occasions that exchange individuals with the following

parameter values: $\phi_A = [0.5, 0.6, 0.3, 0.7, 0.5, 0.65, 0.55]$, $\phi_B = 0.6$, $\psi_{AB} = 0.2$, $\psi_{BA} = 0.5$, $p_A = 0.3$, $p_B = 0.7$, and 20 individuals are released at each population in each year. Thus, we assume that the annual survival probabilities vary among years at location A, but not at location B. Simulate data and analyze them, a) assuming fixed year effects and b) assuming random year effects.

3. In a population of salamanders, there is nonrandom temporary emigration (with respect to one breeding site). In addition, there is strong individual heterogeneity in capture probability. Assume a 10-years study and the following parameter values: survival = 0.7, $\psi_{IO} = 0.4$, $\psi_{OI} = 0.8$, mean recapture = 0.5, and the variance among individuals of the logit of recapture $\sigma_i^2 = 0.4$. Further assume that 100 salamanders are newly marked each year. Simulate data with these characteristics and analyze them.