**Chapter X: Modeling Covariate Effects in SCR Models**

In previous chapters we showed how to fit basic spatial capture-recapture models using Bayesian analysis (in WinBUGS; chapter 4) or by classical likelihood methods (Chapter 5). These basic models involved only constant parameter values that did not vary in response to covariates of any type. However, in practice, investigators are invariably concerned with explicit factors or covariates that might influence variation in parameters. Traditionally, in the non-spatial capture recaptures literature, such models were called as “model $M\_t$”, “model $M\_h$”, or “model $M\_b$”, identifying models that account for variation in detection probability as a function of time, “individual heterogeneity” or “behavior”, where behavior often describes whether or not an individual had been previously captured.

Until this point, we have covered how to use only the basic model in various software packages and the suite of possible encounter models (e.g., the Binomial, Poisson, and Multinomial encounter models) for dealing with different types of sampling. However, we have not considered different detection functions or covariates that my affect the parameters of the detection function, including those that may arise from the individual or the trap device. In general, we can consider that most detection functions include a baseline encounter rate termed $\lambda\_0$ (or $g\_0$ for the detection probability when we use a logit link for the detection function) and a shape parameter termed $\sigma$, which takes on different interpretations depending on the selected function. For example, the most commonly used detection functions are also those used in the distance sampling literature: the half-normal, the hazard, and the negative exponential. The R package secr allows the user to access 12 different detection models, of which some are only used for simulating data (see Table 1). These detection functions can be also be coded in R, BUGS, jags, etc. We will quickly demonstrate how to model different detection functions, but then will focus on the fitting covariates to the baseline encounter rate and the shape parameter of half-normal detection function. Such covariates include time (e.g., day of year, or season), behavior (e.g., has the individual been previously captured), sex of the individual, and trap type (e.g., various camera types, or different constructions for hair snares).

In this chapter, we describe extensions of SCR models to accommodate many different kinds of covariates. We focus on the Binomial encounter model used in chapter 4 and 5 and the half-normal detection function, but the extension to other encounter and detection models is straightforward. Specifically, we consider three distinct types of covariates – those which are fixed, partially observed or completely unobserved (latent). Fixed covariates are those that are fully observed; for example, the date of all sampling occasions. Partially observed covariates are those which are not known for all observations; for example, the sex of an individual cannot always be determined from photos taken during camera trapping. Even if we are able to observe the sex of all individuals sampled, we cannot know it for those individuals never observed during the study. And finally, unobserved covariates are those which we cannot observe at all, for example, the home range size of individuals, or unstructured random “individual effects”.

Another type of covariate is a covariate that varies spatially across the landscape and we know it for every “pixel”. Such covariates might affect density. (but they could also affect encounter probability). We consider these covariates in the next chapter? One interesting thing is that you might imagine that such covariates are incompletely observed and so a 2nd stage model is needed to describe variation across the landscape, at unobserved locations, since the activity centers would be defined conditional on that covariate.

We will see that models containing all of these different types of covariates are relatively easy to describe in the BUGS language, and therefore to analyze using Bayesian analysis of the joint likelihood based on data augmentation thus providing a coherent and flexible framework for inference for all classes of SCR models. Throughout the chapter, we will continue to develop an analysis of the black bear study introduced in Chapter 3, using the bugs language. We also consider likelihood analysis of many of these models, to do so, we will demonstrate the use of the R package ‘secr’ and how to do model comparison with AIC.

Before we describe the types of covariates and demonstrate how to implement them, a brief note about the different inference approaches. In taking a Bayesian approach to analysis of covariate models, inference is always based on analysis of the “joint likelihood” based on data augmentation. That is, the conditional-on-N likelihood, with N removed by integration (as described in chapter 3 somewhere where we introduced data augmentation). However, likelihood analysis based on the conditional likelihood is often done in practice and, in particular, in the secr() package. A variant of the conditional likelihood which is kind of distinct and relevant to the individual covariates is the “Huggins-Alho” idea which is based on thinking about Horwitz-Thompson estimators involving unequal probabilities of sampling. This is not a very coherent approach in the sense that analysis of the joint likelihood is fully general and requires no modification to the manner in which the estimator is constructured. Conversely, different estimators are employed in secr() depending on which type of model is being considered. For latent covariates like finite-mixtures, you have a different estimator than if sex is the covariate (for which there are 2 or 3 estimators) and the basic null model is based on the plain old conditional estimator which integrates s[i] from the likelihood.

**8.1 Detection Functions**

Before we model covariates on the detection function parameters, we will briefly discuss a few of the common detection functions. In Chapter 4, we presented the basic capture recapture model with a rather generic detection function using “alpha1” as the shape parameter on distance. Considering the binomial encounter model, we can specify the model according to:

\begin{equation}

\mbox{logit}(p\_{ij}) = \alpha\_{0} + \alpha\_1 ||{\bf s}\_{i}-{\bf x}\_{j} ||

\label{scr0.eq.logit}

\end{equation}

where, $||{\bf s}\_{i}-{\bf x}\_{j}||$ is the distance between ${\bf s}\_{i}$ and ${\bf x}\_{j}$.

We can also consider writing the detection function such that it more closely resembles the standard distance sampling functions with a half-normal model of the form:

\[

p\_{ij} = p\_{0}\*\exp(-\alpha\_{1} \*||{\bf s}\_{i}-{\bf x}\_{j}||^2)

\]

where we can then describe $\alpha\_1$ as a function of $\sigma$ such that

\[

\alpha\_{1} = 1/(2\*\sigma^2)

\]

If we wanted to use an exponential distance function instead of the half-normal, we can rewrite the detection function as

\[

p\_{ij} = p\_{0}\*\exp(-\alpha\_{1} \*||{\bf s}\_{i}-{\bf x}\_{j}||)

\]

and modify the definition of $\alpha\_1$ to just be the inverse of $\sigma$

\[

\alpha\_{1} = 1/(\sigma)

\]

By changing the detection function and the specification of $\alpha\_1$, we can basically create any distance function for the data. It is important to note that sigma is not comparable under these different distance functions for detection. Additionally, the relationship between sigma and home range radius does not have precision definition under alternative distance functions. We demonstrate how to fit different distance functions under the Bayesian and likelihood sections below.

**8.2 Types of Covariates**

Broadly speaking, we recognize (here) 2 types of covariates: Fixed covariates which are observable and might vary by trap alone (e.g., type of trap, baited or not, disturbance regime, even habitat), sample occasion (e.g., day of season or weather conditions), or both (e.g., behavior, weather – if over a large region). The other class of covariates are those which vary at the level of the individual (and possibly also over time). As a technical matter, these are different than fixed covariates because we cannot see all of the individuals and the covariates are almost always incompletely observed (if at all). The lone exception is the behavioral response which is known for all individuals, captured or not. We noted in other chapters that space itself (i.e., the activity centers) is a type of individual covariate. We do not get to observe the activity center for any individuals, but for individuals that are encountered we get to observe some information about it in the form of which traps the individual was encountered in.

To begin, we will again assume a standard sampling design in which an array of $J$ traps is operated for $K$ time periods, which produces encounter histories for $n$ individuals. For the basic model, there are no time-varying covariates that influence encounter, there are no explicit individual-specific covariates, and there are no covariates that influence density.

For fixed effects, those which we observe fully, we can easily incorporate these into the encounter probability model, just as we would do in any standard GLM or GLMM. For example,

logit(p[i,j,k]) = \alpha\_0 + \alpha\_1\*||s[i]-x[j]|| + \alpha\_2\*C[i,j,k][\*\*]

where \alpha\_2 is a vector of coefficients and **C** is an array of covariates. How we define these covariates (e.g., trap specific versus individual specific) will influence exactly how we include them in the model. For example, the dimensions of **C** will be defined by individual, trap, or both.  We can also extend this to include session specific covariates such as time of day or season by incorporating the session $k$ information.

8.2.1 Date and Time

We might be interested in the effect of date on the detection probability, for example in a long term hair snare study, we may expect that seasonal shedding will influence our detection probabilities.   Or we may expect reproductive behaviors to influence the detection of certain species at certain times of year.  There are a number of ways to incorporate such information into the model; here we will describe two that seem most common. The first is to allow detection probability to be different for each date, but not to be a parametric function of data. In this case, we allow each sampling occasion, k, to have its own baseline detection probability, $\alpha\_0$.

logit(p0[k]) = \alpha\_0[k]

Thus, p[i,j,k] = p0[k]\*exp(- \alpha\_1\*||s[i]-x[j]||^z)

This delineation of $\alpha\_0[k]$ will return $k$ baseline detection probabilities. Thus, if we had 4 sampling occasions, we will have 4 different baseline detection probabilities. This is useful specification in situations where we have just a few sampling occasions or we do not expect a pattern in the timing of the occasions.

However, in many cases, we might expect the date to be important for a variety of reasons. For example, if we have camera traps running for an entire year and we expect mating behavior or denning behavior to change the patterns of individuals, then we might want to incorporate date as a linear or quadratic effect. This is the reason that Kery et al. (2011) incorporated a day of year covariate into their model of European wildcats; the data had been collected over a year long period and cat behavior was expected to vary seasonally thus influencing the detection probabilities. In these cases, we would specifically incorporate day of year (Date) as a continuous covariate as:  
  
 logit(p[i,j,k]) = \alpha\_0 + \alpha\_1\*||s[i]-x[j]|| + \alpha\_2\*Date[k]

It is easy to see that we could model the quadratic effect of day of years as:

logit(p[i,j,k]) = \alpha\_0 + \alpha\_1\*||s[i]-x[j]|| + \alpha\_2\*Date[k] + \alpha\_3\*Date^2[k] [\*\*]

8.2.2 Trap-specific covariate:

There are a variety reasons that traps may have a different baseline detection probability including if the trap is baited or not, if trap type varies (e.g., different camera models are used in a camera trapping study), or because of the habitat type (e.g., if the trap is located on a road/trail). For example, Sollmann et al. (2011) found a large difference in the detection probability due to traps being located on roads which the animals were using to travel along as opposed to traps placed off roads. In each of these cases, the trap type is a binary or categorical variable – on/off road, baited/non-baited, and camera model. We write this such that:

logit(p[i,j,k]) = alpha[type[j]] + \alpha\_1\*||s[i]-x[j]|| [\*\*]

Here, we use an indicator variable, “type”, that will be a numeric value for the trap-specific covariate. Thus for our example of on/off road, we would have type[j] = 1 if trap j is on a road and type[j] = 2 otherwise. This general set up allows for multiple categories, say if 3 or 4 different camera models were used.

8.2.3 Behavior or Trap Response by Individual

One of the most basic of encounter models is that which accommodates a change in encounter probability as a result of initial encounter. This is colloquially “trap happiness” or “trap shyness” which is a natural response of individuals to being captured. If a trap is baited with a food source, naturally an individual might come back for more. On the other hand, if being captured is traumatic then an individual might learn to avoid traps. Both of these types of responses can occur in most species depending on the type of encounter mechanisms being employed. Moreover, behavioral response can be either global (Gardner et al. 2010) or local (Royle et al. 2009). The local response is a trap-specific response which likely makes more sense in most spatial situations. A global response suggests that initial capture provides a net increase or decrease (across all traps).

To describe such models we can create a binary matrix that indicates if an individual has been captured previously.  For the global behavioral response, define the $nind x k$ matrix, C where $C\_i,k =1$ means that individual $i$ was captured at least once prior to session $k$, otherwise C = 0. 

logit(p[i,j,k]) = alpha + \alpha\_1 \*||s[i]-x[j]|| + \alpha\_2\*C\_i,k [\*\*]

For the local behavioral response, which is trap specific, we create an array, $C\_i,j,k$, that indicates if an individual $i$ has been previously captured in trap $j$ at time $k$. We then include this in the model in the exact same form as above:

logit(p[i,j,k]) = \alpha + \alpha\_1\*||s[i]-x[j]|| + \alpha\_2\*C\_i,j,k [\*\*]

8.2.4 Sex

Sex is a special kind of covariate because we can observe it for those individuals we encounter, but sometimes it might be missing because frequently in practice we only imperfectly determine gender of many species. We can imagine that sex impacts both the baseline encounter probability “alpha” and also it might affect the typical home range size, which should cause “alpha1” to vary by sex. The model structure could generally look like this:

logit(p[i,j,k]) = \alpha[sex[i]] + \alpha\_1[sex[i]]\*||s[i]-x[j]|| + \alpha\_2\*C\_i,j,k [\*\*]

where sex[i] is a vector of 1 or 2 indicator variables that say if individual i is male or female. However, we do not know the sex of individuals that are not observed or may not have been identifiable, making this a partially observed covariate. We deal with slightly differently based on the framework that we select (Bayesian or likelihood] and we discuss this in detail below for each modeling framework in sections 8.3.3 and 8.4.3.

8.2.5 Heterogeneity

Heterogeneity is a covariate that is completely latent. This can include many things such as an additive individual effect or an individual-specific effect of distance. We address these models separately in Section 8.5 below and show a simple example of a finite mixture model carried out in secr in Section 8.4.4.

**8.3. Bayesian Analysis of covariates**

To demonstrate how to incorporate various types of covariates using {\bf BUGS}, we will again return to the data collected during the Ft. Drum bear study. This data set was first introduced in Chapter 3, but to refresh your memory, there 38 baited hair snared that were run between June and July 2006. The snares were checked each week for a total for $K=8$ sample occasions and $n=47$ individual bears were “captured” at least once. The data are provided in the {\bf R} package \mbox{\tt scrbook} and the analysis can be set up and run as we will show throughout the chapter.

8.3.1. Detection functions

We start here by presenting the basic SCR model with no covariates and the half normal distance function.

{\small

\begin{verbatim}

library("scrbook")

data("beardata")

trapmat<-beardata$trapmat

nind<-dim(beardata$bearArray)[1]

K<-dim(beardata$bearArray)[3]

ntraps<-dim(beardata$bearArray)[2]

M=650

nz<-M-nind

Yaug <- array(0, dim=c(M,ntraps,K))

Yaug[1:nind,,]<-beardata$bearArray

y<- apply(Yaug,c(1,2),sum) # summarize by ind x traps

#center the coordinates of the trap matrix

X=as.matrix(cbind((trapmat[,2]- mean(trapmat[,2]))/1000, (trapmat[,3]- mean(trapmat[,3]))/1000))

#set up the boundary boxes

Xl=min(trapmat[,2] - mean(trapmat[,2]))/1000 - 20

Xu=max(trapmat[,2]- mean(trapmat[,2]))/1000 + 20

Yl=min(trapmat[,3]- mean(trapmat[,3]))/1000 - 20

Yu=max(trapmat[,3]- mean(trapmat[,3]))/1000 + 20

areaX=(Xl-Xu)\*(Yl-Yu)

cat("

model {

alpha0~dnorm(0,.1)

logit(p0)<- alpha0

alpha1<-1/(2\*sigma\*sigma)

sigma~dunif(0, 15)

psi~dunif(0,1)

for(i in 1:M){

z[i] ~ dbern(psi)

s[i,1]~dunif(Xl,Xu)

s[i,2]~dunif(Yl,Yu)

for(j in 1:J){

d[i,j]<- pow(pow(s[i,1]-X[j,1],2) + pow(s[i,2]-X[j,2],2),0.5)

y[i,j] ~ dbin(p[i,j],K)

p[i,j]<- z[i]\*p0\*exp(- alpha1\*d[i,j]\*d[i,j])

}

}

N<-sum(z[])

D<-N/area

}

",file = "SCR0a.txt")

data0<-list(y=y,M=M,K=K, J=ntraps, Xl=Xl, Yl=Yl, Xu=Xu, Yu=Yu, X=X, area=areaX)

params0<-list('psi','p0','N', 'D', 'sigma')

zst=as.vector(rbinom(M, 1, .5))

inits = function() {list(z=zst,psi=runif(1), sigma=runif(1),alpha0=runif(1)) }

fit0 = bugs(data0, inits, params0, model.file="SCR0a.txt",working.directory=getwd(),

debug=T, n.chains=3, n.iter=20000, n.burnin=10000, n.thin=2)

> print(fit0, digits=3)

Inference for Bugs model at "SCR0a.txt", fit using WinBUGS,

3 chains, each with 20000 iterations (first 10000 discarded), n.thin = 2

n.sims = 15000 iterations saved

mean sd 2.5% 25% 50% 75% 97.5% Rhat n.eff

psi 0.775 0.100 0.578 0.705 0.777 0.848 0.956 1.003 1100

p0 0.106 0.014 0.080 0.096 0.105 0.115 0.134 1.002 2900

N 504.262 64.264 377.000 459.000 506.000 552.000 621.000 1.003 1100

D 0.166 0.021 0.124 0.151 0.167 0.182 0.205 1.003 1100

sigma 1.996 0.129 1.766 1.908 1.988 2.077 2.272 1.001 8800

deviance 774.331 20.409 737.000 759.900 773.400 787.900 817.000 1.001 13000

For each parameter, n.eff is a crude measure of effective sample size,

and Rhat is the potential scale reduction factor (at convergence, Rhat=1).

DIC info (using the rule, pD = var(deviance)/2)

pD = 208.3 and DIC = 982.6

DIC is an estimate of expected predictive error (lower deviance is better).

\end{verbatim}

}

The output from our basic model with no covariates and the half-normal distance function provides an estimate of D = 0.167 bears per $km^2$ and sigma = 1.996. This is similar to the estimated density found under model $M\_0$ in Chapter 3.2.5, which was 0.18 bears per $km^2$. We can also see that the 97.5% percentile for N is 621, thus not reaching our M=650 value, but close enough that we may want to check that N is not truncated by this level of data augmentation.

Now, we can use the same data setup, but examine a different distance function as describe above by redefining $\alpha\_1$.. To use the exponential distance function, we modify the {\bf BUGS} model file such that:

{\small

\begin{verbatim}

cat("  
model {  
alpha0~dnorm(0,.1)  
logit(p0)<- alpha0  
alpha1<-1/(sigma)  
sigma~dunif(0, 15)  
psi~dunif(0,1)  
  
for(i in 1:M){  
 z[i] ~ dbern(psi)  
 s[i,1]~dunif(Xl,Xu)  
 s[i,2]~dunif(Yl,Yu)  
for(j in 1:J){  
d[i,j]<- pow(pow(s[i,1]-X[j,1],2) + pow(s[i,2]-X[j,2],2),0.5)  
y[i,j] ~ dbin(p[i,j],K)  
p[i,j]<- z[i]\*p0\*exp(- alpha1\*d[i,j])  
}  
N<-sum(z[])  
D<-N/area  
}  
",file = "SCR0exp.txt")

data0<-list(y=y,M=M,K=K, J=ntraps, Xl=Xl, Yl=Yl, Xu=Xu, Yu=Yu, X=X, area=areaX)

params0<-list('psi','p0','N', 'D', 'sigma')

zst=as.vector(rbinom(M, 1, .5))

inits = function() {list(z=zst,psi=runif(1), sigma=runif(1),alpha0=runif(1)) }

fitexp = bugs(data0, inits, params0, model.file="SCR0exp.txt",working.directory=getwd(),

debug=T, n.chains=3, n.iter=20000, n.burnin=10000, n.thin=2)

> print(fitexp, digits=3)

Inference for Bugs model at "SCR0exp.txt", fit using WinBUGS,

3 chains, each with 20000 iterations (first 10000 discarded)

n.sims = 30000 iterations saved

mean sd 2.5% 25% 50% 75% 97.5% Rhat n.eff

psi 0.789 0.102 0.588 0.718 0.790 0.863 0.975 1.004 760

p0 0.348 0.056 0.252 0.308 0.344 0.382 0.470 1.005 550

N 513.419 65.770 384.000 467.000 513.000 561.000 634.000 1.004 790

D 0.169 0.022 0.127 0.154 0.169 0.185 0.209 1.004 790

sigma 1.114 0.093 0.947 1.049 1.108 1.173 1.315 1.004 780

deviance 717.773 22.363 676.500 702.100 717.000 732.100 764.000 1.001 6000

For each parameter, n.eff is a crude measure of effective sample size,

and Rhat is the potential scale reduction factor (at convergence, Rhat=1).

DIC info (using the rule, pD = var(deviance)/2)

pD = 250.0 and DIC = 967.8

DIC is an estimate of expected predictive error (lower deviance is better).\end{verbatim}

}

Here, we see that density is estimated at 0.17 and is effectively the same as under the half-normal distance function model above. In this case with the exponential distance function, sigma is defined differently, so here we see the posterior mean estimate of sigma is 1.14 (0.95, 1.32), which is entirely distinct from our estimate of sigma under the half normal model. This highlights that it is important for the user to know what distance function is used and what the interpretation of sigma might be. There is not a clear way (that we know of) for sigma from the exponential model to be related to home range radius.

We leave the detection functions for now and move onto incorporating covariates into the model using the BUGS language. For this part, we will stick with the half-normal distance model shown in the SCR0.txt model file above.

8.4.1 Time

There are a number of ways in which we can incorporate time into our models. As we demonstrated above, we can easily fit a “time effect” where each occasion has its own detection probability in WinBUGS. Again, we can use the same data set up as in the previous section and just modify our WinBUGS code to now allow $\alpha\_0$ to be estimated for each time period $k$. In order to estimate time specific baseline detection, we need to use the 3-d data array which has nind x ntraps x nreps. Thus, in our list of data, we now use Yaug instead of y (the 2-d version of the data). We also update our initial values so that there are $k=8$ values generated. This ultimately means that we have put in another nested for loop in our code and the computation time will increase quite a bit (this model may take up to 20 hours or more on your machine).

{\small

\begin{verbatim}

cat("

model {

for(k in 1:K){

alpha0[k]~dnorm(0,.1)

logit(p0[k])<- alpha0[k]

}

alpha1<-1/(2\*sigma\*sigma)

sigma~dunif(0, 15)

psi~dunif(0,1)

for(i in 1:M){

z[i] ~ dbern(psi)

s[i,1]~dunif(Xl,Xu)

s[i,2]~dunif(Yl,Yu)

for(j in 1:J){

d[i,j]<- pow(pow(s[i,1]-X[j,1],2) + pow(s[i,2]-X[j,2],2),0.5)

for(k in 1:K){

y[i,j,k] ~ dbin(p[i,j,k],1)

p[i,j,k]<- z[i]\*p0[k]\*exp(- alpha1\*d[i,j]\*d[i,j])

}

}

}

N<-sum(z[])

D<-N/area

}

",file = "SCR0t.txt")

data0<-list(y=Yaug,M=M,K=K, J=ntraps, Xl=Xl, Yl=Yl, Xu=Xu, Yu=Yu, X=X, area=areaX)

params0<-list('psi','p0','N', 'D', 'sigma')

zst=as.vector(rbinom(M, 1, .5))

inits = function() {list(z=zst,psi=runif(1), sigma=runif(1),alpha0=runif(8)) }

fitt = bugs(data0, inits, params0, model.file="SCR0t.txt",working.directory=getwd(),

debug=T, n.chains=3, n.iter=20000, n.burnin=10000, n.thin=2)

> print(fitt, digits=3)

Inference for Bugs model at "SCR0t.txt", fit using WinBUGS,

3 chains, each with 20 iterations (first 10 discarded), n.thin = 2

n.sims = 15 iterations saved

mean sd 2.5% 25% 50% 75% 97.5% Rhat n.eff

psi 0.084 0.014 0.063 0.075 0.084 0.092 0.107 0.898 15

p0[1] 0.658 0.044 0.610 0.614 0.649 0.712 0.717 37.099 3

p0[2] 0.557 0.031 0.512 0.516 0.578 0.581 0.582 13.305 3

p0[3] 0.640 0.045 0.577 0.583 0.659 0.680 0.686 29.848 3

p0[4] 0.645 0.054 0.570 0.575 0.667 0.688 0.700 27.818 3

p0[5] 0.574 0.015 0.554 0.559 0.573 0.589 0.592 10.273 3

p0[6] 0.672 0.010 0.662 0.664 0.667 0.682 0.688 6.429 3

p0[7] 0.596 0.056 0.521 0.529 0.610 0.653 0.656 30.910 3

p0[8] 0.619 0.037 0.589 0.591 0.598 0.665 0.671 14.977 3

N 54.600 3.795 49.691 52.000 54.000 57.498 61.570 0.987 15

D 0.018 0.001 0.016 0.017 0.018 0.019 0.020 0.987 15

sigma 8.282 0.364 7.660 8.066 8.362 8.521 8.819 1.258 9

deviance 1498.133 40.440 1444.144 1465.999 1492.000 1526.440 1564.546 1.507 6

\end{verbatim}

}

The results from this model are very similar to those from the basic model, but now we can examine the difference in detection across time. We see that there is a clear difference in detection p0 at time k=1 than with the other time periods. Additionally, detection seems to increase for the first few time periods before stabilizing around 0.7. It is clear that by adding in a time specific detection probability, we have identified that there is heterogeneity in detection and it is important to model those differences. Our density estimates are similar however to the base model, suggesting that the variation in detection by occasion did not have a huge impact on our results.

8.3.2 Behavior

In order to model behavior, we must first create the require matrix or array that will indicate whether an individual has been previously captured or not. We can do this in one of two ways – either as a global response (was the individual previously captured in any trap) or as a local response (was the animal previously captured in this trap). In either case, we will again have to use the 3-d array of the capture histories – nind x ntraps x nreps – as we did for the time model. Thus we caution our reader that the model may take quite a bit of time to run.

{\small

\begin{verbatim}

## create the previous capture matrix that is trap specific, C[i,j,k]

C=Yaug

for(k in 2:K){

C[,,k] = Yaug[,,k] + C[,,k-1]

}

C[C >1] =1

cat("

model {

alpha0~dnorm(0,.1)

alpha2~dnorm(0,.1)

alpha1<-1/(2\*sigma\*sigma)

sigma~dunif(0, 15)

psi~dunif(0,1)

for(i in 1:M){

z[i] ~ dbern(psi)

s[i,1]~dunif(Xl,Xu)

s[i,2]~dunif(Yl,Yu)

for(j in 1:J){

d[i,j]<- pow(pow(s[i,1]-X[j,1],2) + pow(s[i,2]-X[j,2],2),0.5)

for(k in 1:K){

logit(p0[i,j,k])<- alpha0 + alpha2\*C[i,j,k]

y[i,j,k] ~ dbin(p[i,j,k],1)

p[i,j,k]<- z[i]\*p0[i,j,k]\*exp(- alpha1\*d[i,j]\*d[i,j])

}

}

}

N<-sum(z[])

D<-N/area

}

",file = "SCRb.txt")

datab<-list(y=Yaug,C=C, M=M,K=K, J=ntraps, Xl=Xl, Yl=Yl, Xu=Xu, Yu=Yu, X=X, area=areaX)

params0<-list('psi','alpha0','alpha2', 'N', 'D', 'sigma')

zst=as.vector(rbinom(M, 1, .5))

inits = function() {list(z=zst,psi=runif(1), sigma=runif(1),alpha0=runif(1),alpha2=runif(1)) }

fitb = bugs(datab, inits, params0, model.file="SCRb.txt",working.directory=getwd(),

debug=T, n.chains=3, n.iter=20, n.burnin=10, n.thin=2)

\end{verbatim}

}

WAITING FOR THE RESULTS HERE.

**8.4. Likelihood analysis in SECR**

SECR allows the user to simulate data and fit a suite of models with various detection functions and covariate responses. As we saw in Chapter 5, secr uses the standard R model specification framework, defining the dependent and independent variable relationship using tildes (e.g., y ~ x). Thus, in secr we might have g0 ~ behavior or sigma ~ time; when left unspecified or set to 1 (e.g., g0 ~ 1), this will default to a model with no covariates that is constant. . Additionally, we can specify covariates in secr on density, which are set for example as D ~ habitat.

To demonstrate a suite of models with various types of covariates using secr, we continue using the data collected on black bears in Ft. Drum, NY, USA. It should be easy for the reader to take this example and generalize it for his or her own use. First we need to read in the raw data and the trap file so that we can build the capture history and trap files required by secr. After doing this, similar to the wolverine example shown in Chapter \*5\*, we then call the command secr.fit to run the model. All of the following models should take somewhere between 30 seconds and 5 minutes to run on your computer (possibly a little longer or shorter given the specifics of your machine). To refresh your memory here how to read in and format the data and run the basic model with no covariates and a half normal distance function in secr for the Ft. Drum bear study:

{\small

\begin{verbatim}

trapmat<-read.csv("FDtrapmat.csv")

trapmat[,2:3] = trapmat[,2:3]\*1000

colnames(trapmat)<- c("trapID","x", "y")

trapfile <- read.traps(data = trapmat, detector = "proximity")

captfile <- bearraw[ , c(4,1,3,2)]

colnames(captfile) <- c("Session", "ID", "Occasion", "trapID")

bearcapt=make.capthist(captfile, trapfile, fmt = "trapID", noccasions = 8)

bear=secr.fit (bearcapt, buffer = 20000)

Detector type proximity

Detector number 38

Average spacing 1776.437 m

x-range 438789 456465 m

y-range 4874895 4887477 m

N animals : 47

N detections : 151

N occasions : 8

Mask area : 301466.0 ha

Model : D~1 g0~1 sigma~1

Fixed (real) : none

Detection fn : halfnormal

Distribution : poisson

N parameters : 3

Log likelihood : -587.1641

AIC : 1180.328

AICc : 1180.886

Beta parameters (coefficients)

beta SE.beta lcl ucl

D -6.398657 0.15502806 -6.702506 -6.094807

g0 -2.133876 0.14669415 -2.421391 -1.846361

sigma 7.587286 0.06458962 7.460692 7.713879

Variance-covariance matrix of beta parameters

D g0 sigma

D 0.0240336981 -0.0001084522 -0.002602049

g0 -0.0001084522 0.0215191733 -0.005977169

sigma -0.0026020486 -0.0059771686 0.004171819

Fitted (real) parameters evaluated at base levels of covariates

link estimate SE.estimate lcl ucl

D log 1.663791e-03 2.594918e-04 1.227831e-03 2.254545e-03

g0 logit 1.058476e-01 1.388370e-02 8.155599e-02 1.363008e-01

sigma log 1.972951e+03 1.275652e+02 1.738351e+03 2.239211e+03

\end{verbatim}

}

Just as a reminder, secr returns density in terms of individuals per hectare, so we must multiply this D by 100 to see the estimate as individuals per km2. In doing so, the resulting density is 0.166 individuals / $km ^2$, which very similar to that found in section 8.3 using WinBUGS for the basic model with a half-normal distance function on detection.

In the secr package, the detection functions are specified by changing the call to the “detectfn” within the secr.fit command. Table 1 shows the possible detection functions that secr will fit; the default is the half-normal and the exponential is 2. Thus to fit the exponential distance function, we would use the following commands.

{\small

\begin{verbatim}

bearexp = secr.fit (bearcapt, buffer = 20000, detectfn=2)

*[secr data and model summary output deleted ]*

Beta parameters (coefficients)

beta SE.beta lcl ucl

D -6.3778954 0.1575762 -6.686739 -6.0690518

g0 -0.6439777 0.2436156 -1.121455 -0.1664999

sigma 7.0065881 0.0838522 6.842241 7.1709354

Variance-covariance matrix of beta parameters

D g0 sigma

D 0.024830243 0.00312779 -0.004050057

g0 0.003127790 0.05934856 -0.015261224

sigma -0.004050057 -0.01526122 0.007031192

Fitted (real) parameters evaluated at base levels of covariates

link estimate SE.estimate lcl ucl

D log 1.698694e-03 2.693439e-04 1.247344e-03 2.313366e-03

g0 logit 3.443479e-01 5.500169e-02 2.457414e-01 4.584709e-01

sigma log 1.103882e+03 9.272586e+01 9.365855e+02 1.301061e+03

\end{verbatim}

}

Density is estimated similar to the half-normal, although the mean estimate is slightly larger. As we saw in the above sections, sigma = 1.1 which is very different from the half normal where sigma was 1.97. These results are consistent with the what we saw in the Bayesian analysis of the model.

Table 1: Distance functions fit by secr. (Table taken from the secr help files).

Code Name Parameters Function

0 halfnormal g0, sigma *g(d) = g0 \* exp{--d^2 / (2 sigma^2) }*

1 hazard rate g0, sigma, z *g(d) = g0 \* (1 -- exp(-- (d / sigma) ^(--z) ))*

2 exponential g0, sigma *g(d) = g0 \* exp(-- d / sigma)*

3 compound halfnormal g0, sigma, z *g(d) = g0 \* [1 -- {1 -- exp(--d^2 / (2 sigma^2))]^z}*

4 uniform g0, sigma *g(d) = g0, d<=sigma; g(d) = 0, otherwise*

5 w exponential g0, sigma, w *g(d) = g0, d < w; g(d) = g0 \* exp(-- (d -- w) / sigma), otherwise*

6 annular normal g0, sigma, w *g(d) = g0 \* exp(--(d-w)^2 / (2 sigma^2))*

7 cumulative lognormal g0, sigma, z *g(d) = g0 [1 -- F{(d--μ)/s)}]*

8 cumulative gamma g0, sigma, z *g(d) = g0 { 1 -- G (d; k, θ) }*

9 binary signal strength b0, b1 *g(d) = 1 -- F {-- (b0 + b1 \* d) }*

10signal strength beta0, beta1, sdS *g(d) = 1 -- F[ {c -- (beta0 + beta1 \* d)} / sdS]*

11signal strength spherical beta0, beta1, sdS *g(d) = 1 -- F[{c -- (beta0 + beta1 \* (d--1) -- 10 \* log10 ( d^2 ) ) } / sdS ]*

8.4.1 Time

Secr easily fits a “time effect” where each occasion has its own detection probability. This is reasonable when there are very few sampling occasions but becomes an unwieldy model with lots of parameters, in general, if sample occasions are very frequent (e.g., daily). In some cases it might make sense to have a smooth function of time to reflect season variation in encounter probability as show above. We saw previous that such models are easy to fit in WinBUGS. secr only fits the classical “time effect” type of model with K distinct parameters, unless an alternative approach is used such as “groups” or “sessions”.

Time specific parameters are also easy to incorporate, we just include $t$ in our model fit call and secr will automatically fit time specific estimates. For example, if we allowed baseline detection to vary by time (as we did in eq. \*\*), we simply use g0~t in our model call.

beart=secr.fit (bearcapt, model = list(D~1, g0~t, sigma~1), buffer = 20000)

beart

*[secr data and model summary output deleted ]*

Beta parameters (coefficients)

beta SE.beta lcl ucl

D -6.3984155 0.15485019 -6.70191633 -6.0949147

g0 -2.8248967 0.34791833 -3.50680406 -2.1429893

g0.t2 -0.2315323 0.49283661 -1.19747433 0.7344097

g0.t3 1.0168296 0.39928271 0.23424991 1.7994094

g0.t4 0.9923785 0.40168337 0.20509360 1.7796635

g0.t5 1.0197370 0.39931504 0.23709388 1.8023801

g0.t6 0.8267689 0.40841322 0.02629367 1.6272441

g0.t7 1.0185729 0.39931309 0.23593362 1.8012122

g0.t8 0.2871613 0.44183900 -0.57882724 1.1531498

sigma 7.5832897 0.06435693 7.45715244 7.7094270

Variance-covariance matrix of beta parameters

*[output deleted]*

Fitted (real) parameters evaluated at base levels of covariates

link estimate SE.estimate lcl ucl

D log 1.664192e-03 2.592530e-04 1.228555e-03 2.254302e-03

g0 logit 5.599354e-02 1.839036e-02 2.911925e-02 1.049882e-01

sigma log 1.965083e+03 1.265978e+02 1.732208e+03 2.229264e+03

For some reason, secr does not provide the “fitted” value for the detection probability of each time period after the initial time (t1). We can back calculate each of the values using:

>plogis(coef(beart)[2,1]) #this pulls out time t = 2

[1] 0.05599354

>plogis(coef(beart)[3,1]) #this pulls out time t = 3

[1] 0.4423741

.

.

.

> plogis(coef(beart)[6,1]) #this pulls out time t = 6

[1] 0.7349214

.

These results suggest the same as we saw in section 8.3.1, that there are differences in detection by time. Interestingly, the estimate of density remains similar to our basic model with D = 0.167 individuals / $km ^2$. One might have expected the density to change given how small the baseline detection probability was in the first session. However, the increased detection in other time periods clearly balances that initial low detection probability.

8.4.2 Behavior

The secr package allows one to incorporate a simple trap response rather easily. Not without reorganizing our data, we can just take the basic model and change the model call to:

bearb=secr.fit (bearcapt, model = list(D~1, g0~b, sigma~1), buffer = 20000)

secr.fit( capthist = bearcapt, model = list(g0 ~ b), buffer = 20000 )

secr 2.0.0, 18:39:38 14 Jul 2011

*[secr data and model summary output deleted ]*

Beta parameters (coefficients)

beta SE.beta lcl ucl

D -6.063824 0.20998468 -6.4753866 -5.652262

g0 -2.962320 0.30193079 -3.5540936 -2.370547

g0.bTRUE 1.069292 0.30169340 0.4779839 1.660600

sigma 7.580269 0.06276493 7.4572522 7.703286

Variance-covariance matrix of beta parameters

D g0 g0.bTRUE sigma

D 0.044093568 -0.038995412 0.0406985508 -0.0024012404

g0 -0.038995412 0.091162203 -0.0788400314 -0.0054049828

g0.bTRUE 0.040698551 -0.078840031 0.0910189076 -0.0006804517

sigma -0.002401240 -0.005404983 -0.0006804517 0.0039394367

Fitted (real) parameters evaluated at base levels of covariates

link estimate SE.estimate lcl ucl

D log 2.325491e-03 4.937501e-04 1.540903e-03 3.509570e-03

g0 logit 4.915745e-02 1.411255e-02 2.781168e-02 8.544641e-02

sigma log 1.959156e+03 1.230875e+02 1.732381e+03 2.215617e+03

Again, secr does not provide the “fitted” value for the detection probability of individuals that were not observed. Here the baseline detection probability is 0.049 and for those individuals that have been previously captured, we see it is:

> plogis(-2.962+1.069)

[1] 0.1309028

This is almost 3 times greater than the detection of those individuals not previously captured. It is also clear, as we saw before, that the density increases over the model with no behavioral effect (0.23 bears/km2 versus 0.16 bears/km2 in the SCR0 model). Thus as we can see, it is very easy to incorporate a trap response; however, to incorporate a trap specific behavioral response (as in Royle et al. 2009) is not so straight forward. \*\*\*B – figure this out\*\*.

Behavioral and time effects can be both incorporated by writing the standard linear regression type code within the model call:

bearbt=secr.fit (bearcapt, model = list(D~1, g0~b + t, sigma~1), buffer = 20000)

8.4.3 Sex

Incorporating sex into secr can be done a few different ways, but none of these allow us to include partial observability. Individuals that are of unknown sex must be removed from the dataset, which is different from the winbugs approach. The most common way to include sex is to code it into “session”, providing two sessions that represent males and females. This is specified using the model list within secr.fit command as show below.

bearraw$Sex<-bearsex$Sex[pmatch(bearraw$Ind, bearsex$Bear, duplicates.ok=T)]

bearraw$Session<-as.numeric(bearraw$Sex)

captfile <- bearraw[ , c(6,1,3,2)]

colnames(captfile) <- c("Session", "ID", "Occasion", "trapID")

bearcapt=make.capthist(captfile, trapfile, fmt = "trapID", noccasions = 8)

bearsex=secr.fit (bearcapt, model = list(D~session, g0~session, sigma~session), buffer = 20000)

Fitted (real) parameters evaluated at base levels of covariates

session = 1

link estimate SE.estimate lcl ucl

D log 1.192961e-03 2.400004e-04 8.073782e-04 1.762689e-03

g0 logit 1.352143e-01 2.504984e-02 9.317268e-02 1.922053e-01

sigma log 1.513867e+03 1.276755e+02 1.283591e+03 1.785454e+03

session = 2

link estimate SE.estimate lcl ucl

D log 5.376811e-04 1.081709e-04 3.638945e-04 7.944638e-04

g0 logit 9.248494e-02 1.798037e-02 6.276472e-02 1.342622e-01

sigma log 2.522380e+03 2.127309e+02 2.138699e+03 2.974894e+03

As you can see in the output, this provides two separate density estimates, which must then be combined into a total density. The resulting estimates for sigma are on par with those from WinBUGS provided in section 8.XX.

Remarks: 1) We show only 1 way in which sex can be incorporated in secr, however, there are at least two other ways that one could specify the model (M. Efford, pers. comm). One way is that we could list sex as a categorical individual covariate and then maximize the conditional likelihood. The second way is that we could specify the model as model = list(D~g, g0~g, sigma~g) and list groups = ‘sex’ where we have specified sex as a 2-level individual covariate. There is an issue with the AIC values for models with and without groups that has not been resolved so the reader should be cautious when using this latter option (M. Efford, pers. comm). \*\*\*I’m not trusting the AIC for the sex model as session either\*\*\*

8.4.4 Individual heterogeneity

Additionally, in secr, individual heterogeneity can be incorporated into the detection parameters as either a 2-part or 3-part finite mixture model with the use of “h2” or “h3”, respectively, in the model call. This allows secr to assume that the population is comprised of 2 or more latent classes, with an unknown proportion in each class. We discuss heterogeneity in more detail in the next section but provide just the basic model for secr here. To fit a 2-part finite mixture for baseline detection we use the following command call:

bearhh=secr.fit (bearcapt, model = list(g0~h2), buffer = 20000)

Remarks: 1) It is important to note that this specification of individual heterogeneity is different from that which we incorporate into WinBUGS. Here, a finite mixture model is used, which effectively puts the individuals into one of two (or three) latent classes and then assigns each class a distribution for the specified detection parameter. 2) Incorporating 3 latent classes is as easy as using h3 instead of h2. For homework, the reader should incorporate heterogeneity in sigma and using 2 and 3 classes. Take note of any warning messages or errors.

8.4.5 Model selection in secr with AIC

One practical advantage to using the secr package or likelihood inference in general is the convenience of model selection via AIC. After running our models above with various attributes (e.g., time, trap response), we can then use the AIC call to return the AIC values, delta AIC, and model weights.

AIC(bear, bearb, beart, bearbt, bearh)

model detectfn npar logLik AIC AICc dAICc AICwt

bearh D~1 g0~h2 sigma~1 pmix~h2 halfnormal 5 -570.4348 1150.870 1152.333 0.000 1

bearb D~1 g0~b sigma~1 halfnormal 4 -578.5361 1165.072 1166.025 13.692 0

bearbt D~1 g0~b + t sigma~1 halfnormal 11 -569.2231 1160.446 1167.989 15.656 0

beart D~1 g0~t sigma~1 halfnormal 10 -575.3320 1170.664 1176.775 24.442 0

bear D~1 g0~1 sigma~1 halfnormal 3 -587.1641 1180.328 1180.886 28.553 0

The results from this AIC test are pretty easy to interpret; the model with individual heterogeneity model as a finite mixture for g0 has all the model weight. Using the AIC provides a convenient mechanism for conducting model comparisons. However, the user must be left a little frustrated with these results, which indicated that individuals have some unknown source of heterogeneity which we have not identified using time and behavior. Naturally, since we found that the model with the most weight has two latent classes, we might try to explain this unknown heterogeneity by classifying the groups into sex.

**8.5. Individual heterogeneity.**

Capture-recapture models with individual heterogeneity in detection probability have a long history in classical capture recapture models (see chapter 3.4). Also, their use has been called into question by Link (2003) who noted that N may not be identifiable across arbitrary classes of mixture models. One possible way to get around this problem is to identify explicit sources of heterogeneity in detection probability and model those directly. For example, we can do this by using individual covariate models (e.g., chapter 3). Of course, spatial capture-recapture models are such a class of models which seek to explain heterogeneity in detection by describing the underlying mechanism explicitly. In particular, that mechanism is the juxtaposition of individuals with traps and the resulting heterogeneity that is induced by heterogeneity in exposure to trapping.

Model Mh has special historical relevance in the context of spatial capture-recapture models as we noted in chapter 3.4). Historically people have used Model Mh to get an estimate of N, thereby accounting for a vague sort of heterogeneity. Then they would buffer the trap array and convert Nhat under model Mh to a density estimate. Formal developments in SCR models have rendered this technology obsolete – we can model space explicitly. Despite this, it might still be desirable to accommodate individual heterogeneity of one form or another. In fact, in the context of SCR models a certain kind of heterogeneity makes eminent biological sense.

As we have stressed throughout this book, a fairly broad class of SCR models has a convenient representation as a generalized linear mixed model (GLMM) with "s" as an individual random effect (Royle et al. 2009). In particular, if the encounter rate of individuals in traps is Poisson with rate lam0\*g(x,s) where g(x,s) is a bivariate normal density, then the SCR models are individual covariate models with:

\[

cloglog(p[i,j]) = \alpha + \alpha\_1\*||s\_{i} - x\_{j}||^2

\]

where $\beta = (1/\sigma^{2})$. We could as well use a logit link here which is customary in many contexts. This GLM formulation reveals the essential connection of such models with other individual effects models including individual covariate models as well as heterogeneity models.

In all applications of such models that we are aware of the scale parameter has been constant or a function of explicit covariates (e.g., sex; Gardner et al. 2010). Conversely, it is reasonable to expect in real biological populations that there exists heterogeneity in home range size.

Here we develop and evaluate a new class of spatial capture-recapture models which allow for individual heterogeneity in encounter probability. In particular, one class of models we propose explicitly admits individual heterogeneity in home range {\it size}. In addition, we consider a standard representation for heterogeneity in which an additive individual-specific random effect is included in the linear predictor for encounter probability.

We evaluate the following questions concerning heterogeneity models: First, we evaluate the influence of ``variable home range area'' heterogeneity on estimates of density obtained under the misspecified model that does not contain such heterogeneity. Second, we evaluate how much data is needed to fit the model with heterogeneous home range area using a limited simulation study -- limited because of

computational considerations. Most studies yield sparse data and it is clear that the SCR+Ah model requires sufficient spatial recaptures of individuals to gauge home range size. Thirdly, we want to evaluate the extent to which Model SCR+Mh in some form or another yields a good approximation to heterogeneity in encounter probability that is due to heterogeneity in home range size.

**8.5.2. Models of Heterogeneity**

BETH: I WOULD LIKE TO DEVELOP THIS MATERIAL A LITTLE BIT MORE, SO NO NEED TO DO ANY EDITING BELOW THIS POINT.

An obvious model extends the SCR model by including an additive individual effect, analogous to classical ``Model $M\_{h}$''. We'll call this model ``SCR+Mh'':

\[

cloglog(p) = \alpha + \beta\*d(i,j)^2 + \eta\_{i}

\]

where $\eta\_{i}$ is an individual random effect having distribution $g(\eta|\theta)$. A popular class of models arises by assuming $\eta\_{i} \sim Normal(0,\tau^{2})$ (Coull and Agresti 1999; Dorazio and

Royle 2003; etc..). Many other random effects distributions are possible. Norris and Pollock (1996) propose a finite mixture of point supports which has been addressed considerably in the literature

(Pledger; Dorazio and Royle; Link 2003). Our view is that such models are not very realistic, yet data hugry as they require many more parameters. Heterogeneity seems naturally continuous unless one expects the heterogeneity to be due to meaningful biological groupings in which case such information would normally be collected if possible. Even so the more likely scenario is that heterogeneity is due to a lot of different sources contributing independent components of variation, and so the normal model seems sensible in that regard. We note that Efford (XXX) considered a finite-mixture type of representation for SCR models. These are fit in the R package secr() which we do in section XYZ below.

**Heterogeneity Induced by Variation in Home Range Size --** We suggest an alternative heterogeneity model, one that has more of a direct biological motivation and interpretation. Specifically , we suppose that there exists heterogeneity in home range size among individuals. This is manifest in the scale parameter of the detection function $\sigma^{2}$ or its inverse $\beta = 1/\sigma^{2}$. We might

thus assume a distribution for either $\sigma^{2}$ or its inverse, $\beta$. We thus propose ``Model SCR + Ah'' (Ah for area-induced heterogeneity).

\[

cloglog(p) = \alpha + \beta\_{i}\*d(i,j)^2

\]

This model is a model of heterogeneity in home range area. For example if we assume that $\beta\_{i} \sim \mbox{Normal}(\beta\_0,\tau^{2})$ with $\beta\_{0} = 2$ and $\tau = 0.50$. Then the population distribution of $\sigma$ in this case is given in Figure \ref{fig.one}. The motivating point of this model is that we expect such variability in natural populations. Thus we suggest this biologically sensible model of heterogeneity, which fills a methodological gap in the literature in the sense that SCR models have all been homogeneous with respect to their explicit treatment of home range morphology.

\begin{figure}[ht]

\centerline{\psfig{figure=fig1.ps,height=4in,width=4in}}

\caption{

Population distribution of $\sigma$ if $(1/\sigma^{2}) \sim \mbox{Normal}(2, 0.50)$.

}

\label{fig.one}

\end{figure}

Interesting point: $\beta\_{i}$ might have an Inverse-Gamma

distribution so we need to parameterize the IG in terms of mean and

variance.... but this is not conjugate in the present context and so

there is no compelling reason to do that. Instead, we use a normal

prior ..... Note: negative values are bad, but not nonsensical.

One idea that needs to be explicit is that if A[i] is the home range area of individual i, which is the following function of sigma[i] , then we should be able to go back and forth between distributions for A[i], sigma[i], and beta[i]. Note I did all of this stuff long ago but will never find those notes, ever!

{\bf Approximation: }

Note that ``SCR + Mh'' might be a good approximation to ``SCR + Ah''. If we write $beta\_{i} =

beta\_{0} + \eta\_{i}$ then

we can take the expectation over $\beta\_{i}$ to arrive at

\[

cloglog(p\_{ij} ) = \alpha + \beta\_{0}\*d(i,j)^2 + \eta\_{i}\*d(I,j)^2

\]

Which has this additive individual effect that varies also by trap. It might be that approximating

This by SCR+Mh is better than nothing.

This could also be viewed as suggesting an over-dispersed count model for encounter frequencies.

8.7.2. doing it in WinBUGS

Here we will simulate some data and fit SCR, SCR + Mh, SCR + Ah

8.7.3 Doing it in secr.

Secr fits the most bizarre type of heterogeneity models – they use the “finite mixture” models (Norris and Pollock 1996; Pledger 2000). These are expensive in terms of parameters and not very typically used outside of their use in secr and a few other specialized software packages that do capture-recapture things. Historically they were adopted because they are easy to compute with. More recently, continuous mixtures have been adopted in many settings because they are natural extensions of standard GLMs. We don’t favor the use of finite mixtures. Despite this we give some examples here using secr.