Introducing the OccupancyModels package

Derek Sonderegger

October 1, 2014

Department of Mathematics and Statistics
Northern Arizona University

October 1, 2014

The use of occupancy models is quite common, but the expertise to write a Bayesian model to do the analysis is a substantial burden. The package `unmarked` has helped, but it lacks some functionality, particularly for analyses that have an availability component. While the primary focus of the `OccupancyModels` package is to provide a convenient R interface to Bayesian models that include multiple detection devices at a site, we also include this missing functionality.

This vignette is primarily designed to introduce the relevant models with appropriate references, and demonstrate fitting them in the `OccupancyModel` package. If package `unmarked` contains the functionality to fit a particular model, that process will also be demonstrated.

Software issues

The package was designed to use `stan`. To learn more about `stan` and install it and the package `rstan` go to `https://github.com/stan-dev/rstan/wiki/RStan-Getting-Started`.

Introduction stuff

- Ψ = probability that a site is occupied
  - Occupied means species use on at least one day
- $y_{ij} = 1$ denotes a detection at site i on day j
- $Z_i = 1$ denotes if a site is actually occupied
  - Latent variable and not necessarily known (unless a detection is made)
- π is the probability of detection given the the site is occupied
  - $P\left(y_{ij} \mid Z_i = 1\right) = \pi$

Occurrence Models

We will use the terminology of "Occurrence Models" to address a class of models introduced by Mackenzie et al 2002.

- $\Psi_i = \text{logit}^{-1}\left(X_i^T \beta\right)$ where the X_i covariates represent site level information that increase or decrease the occurrence probability and the estimation of the β values is the primary interest of the study.
- $Z_i \sim \text{Bern}\left(\Psi_i\right)$
- $y_{ij} \sim \text{Bern}\left(\, Z_i \cdot \pi \,\right)$
- Assumes the detection probability is constant across days.

- Inappropriate in many cases
- Can aggregate multiple days into multiple sampling periods
  - What is an appropriate number of days to aggregate?

- Can be fit using `unmarked`

Occupancy / Availability models

- Similar to models in Nichols et al, 2008
- Can have a second latent variable that indicates daily availability
  - $\theta_j = 1$ probability an animal is available on day j
  - $W_{ij} = 1$ (latent variable denoting if an animal is available at site i on day j)
- $\Psi_i = \text{logit}^{-1}\left(X_i^T \beta\right)$
- $Z_i \sim \text{Bern}\left(\Psi_i\right)$
- $W_{ij} \sim \text{Bern}\left(\, Z_i \cdot \theta_j \,\right)$
- $y_{ij} \sim \text{Bern}\left(\, W_{ij} \cdot \pi \,\right)$
- Assumptions
  - Availability on day j is the same across sites.
  - Constant value of availability better?

To fit these models, users have had to write their own code in WinBUGS, JAGS, or stan. Fortunately, we have a program to do most of this automatically.

No covariates

We first introduce the produce using a model with no covariates. That is to say that Ψ_i is constant across sites. In our model, this can be coded as $X_i = 1$ for all i and $\Psi_i = \text{logit}^{-1}\left(\beta\right)$.

```r
# tools that allow me to download from GitHub
library(devtools)
install_github('dereksonderegger/OccupancyModels')
library(OccupancyModels)

# Use the make.data function to generate some simulated data
# Check out the help file!
# ?make.data
temp <- make.data(
  Occupancy.p = .4,  # thus beta = logit(.4) = -.405
  Available.p = .5,
  Detection.p = .7,
  n.sites = 40,
  n.days = 5,
  n.cameras = 3)

# What have we made...
head(temp)
```

```
##   site day camera Detections
## 1    1   1      1          0
## 2    1   1      2          0
## 3    1   1      3          0
## 4    1   2      1          0
## 5    1   2      2          0
## 6    1   2      3          0
```

The first step in the analysis is to take our detection data and turn it into a 3-dimensional array, where the dimensions correspond to `site`, `day`, and `camera`.

```r
Y <- acast(temp, site~day~camera, value.var='Detections')
str(Y)
```

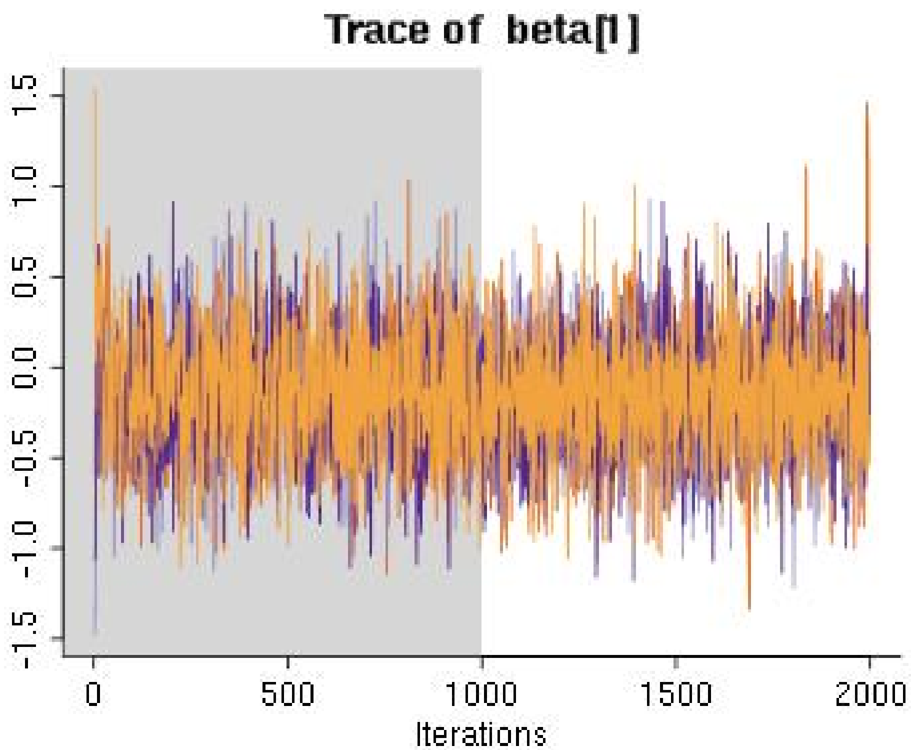
```
##  int [1:40, 1:5, 1:3] 0 0 1 0 0 0 0 0 0 0 ...
##  - attr(*, "dimnames")=List of 3
##   ..$ : chr [1:40] "1" "2" "3" "4" ...
##   ..$ : chr [1:5] "1" "2" "3" "4" ...
##   ..$ : chr [1:3] "1" "2" "3"
```

Next we can call the multiple detection occupancy model `mdOcc`.

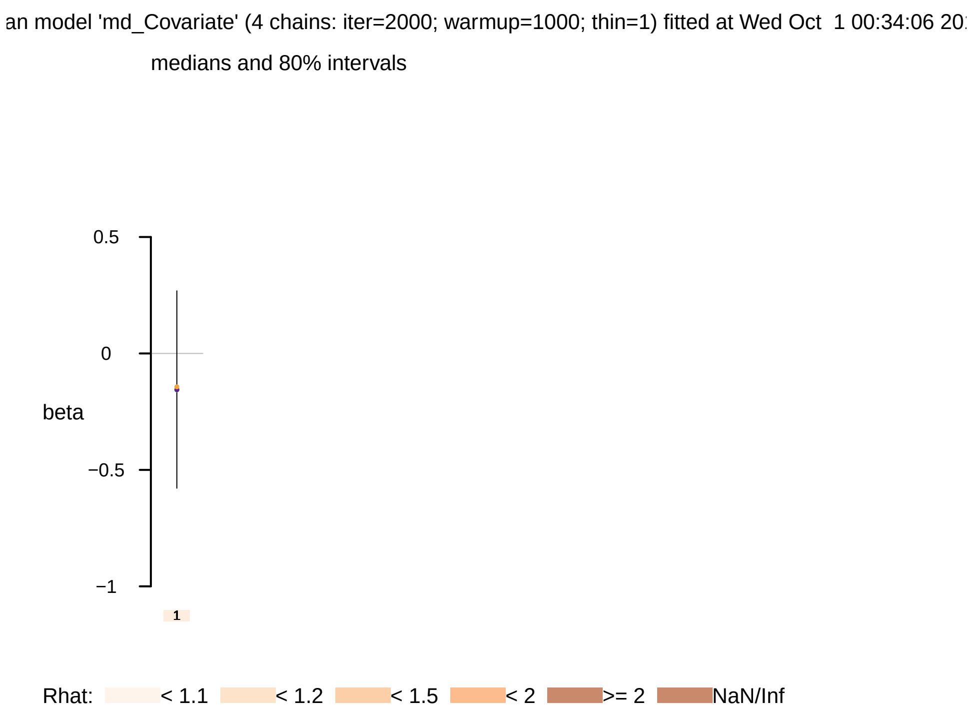
```r
model <- mdOcc(Y) # No covariates
```

We have now run the model through `stan` and can inspect the output. In particular I want to look at the values for β which is the logit transformed occupancy probability.

```r
traceplot(model, pars='beta')
```




```r
plot(model, pars='beta')
```



We can do our own summary statistics by extracting the chains for the β value (which in this simple case is just the inverse logit of the occupancy probability.

```r
beta.posterior <- extract(model, pars='beta')[[1]]
output <- apply( inv.logit( beta.posterior ),
       MARGIN=2,
       quantile, c(.025, .25, .50, .75, .975) )
output

##
##           [,1]
##   2.5%  0.3104
##   25%   0.4073
##   50%   0.4617
##   75%   0.5163
##   97.5% 0.6181
```

This shows us that the posterior distribution of Ψ, the occupancy probability, is between 0.32 and 0.64, and this interval happily includes the true value of $\Psi = 0.4$.

With Covariates

We next allow there to be covariates in the model. In this case, we'll create data with one covariate and set a positive relationship between the covariate and Ψ.

```r
# Number of plots
n <- 50

# make up some site level covariates
site.data <- data.frame(x=rnorm(n, 0, 2))

# relationship between x and Psi
psi <- inv.logit( 0 + (1/2)*site.data$x )

# Make data
sim.data <- make.data(n, n.days=20, n.cameras=3,
                      Occupancy.p=psi, Available.p=.5, Detection.p=.4 )

# reshape the data frame into a 3D array
Y <- acast(sim.data, site~day~camera, value.var='Detections')
```

```r
# Fit the model using the site level covariates
model <- mdOcc(Y, site.data, ~ x)
```

Using the `extract` function as we did before, we obtain the posterior distribution of the β values and also exam the logit^{-1} transformed values as well.

```r
beta.posterior <- extract(model, pars='beta')[[1]]
beta.summary <- apply( beta.posterior, MARGIN=2,
                       quantile, c(.025, .25, .50, .75, .975) )
inv.logit.beta.summary <- apply( inv.logit( beta.posterior ), MARGIN = 2,
                                 quantile, c(.025, .25, .50, .75, .975) )

beta.summary
```

```
##
##              [,1]   [,2]
##   2.5%  -0.50178 0.2876
##   25%   -0.07687 0.4901
##   50%    0.15689 0.6277
##   75%    0.39556 0.7732
##   97.5%  0.87446 1.1005
```

```r
inv.logit.beta.summary
```

```
##
##            [,1]   [,2]
##   2.5%  0.3771 0.5714
##   25%   0.4808 0.6201
##   50%   0.5391 0.6520
##   75%   0.5976 0.6842
##   97.5% 0.7057 0.7504
```