Lab 4 Tutorial: Hierarchical Models

Daniel Hocking

Background

Read Chapters 14, 15, 17, 18 in Powell and Gale

Key Terms

- Census
- Survey
- Count
- Abundance (estimate)
- Catch-per-unit-effort (CPUE)
- Index of abundance
- Detection probability
- Availability probability

Methods

- Distance sampling
- Double-observer
- Removal sampling (also called depletion sampling)
- Repeated counts (N-mixture model)
- Occupancy modeling

Occupancy Model

Repeated presence-absence data to account for imperfect detection

Example

formula in `unmarked` Double right-hand side formula describing covariates of detection and occupancy in that order.

```
# --------------------- Analysis of real data ----------------------------
# This a subset of point-count data from Chandler et al. (Auk 2009)
# alfl is Alder Flycatcher (Empidonax alnorum)
# PART I. Import and format data
alfl.data <- read.csv("Data/alfl05.csv", row.names=1)
str(alfl.data)
```

```
## 'data.frame':    50 obs. of  12 variables:
##  $ alfl1 : int  1 2 0 1 0 3 0 0 0 0 ...
##  $ alfl2 : int  0 3 1 0 0 1 0 0 1 0 ...
##  $ alfl3 : int  1 0 1 1 0 1 0 0 2 0 ...
##  $ struct: num  5.45 4.75 14.7 5.05 4.15 9.75 9.6 15.7 9.2 7.75 ...
##  $ woody0: int  6 1 7 6 2 8 4 0 4 3 ...
##  $ woody : num  0.3 0.05 0.35 0.3 0.1 0.4 0.2 0 0.2 0.15 ...
```

```
##  $ time.1: num  8.68 9.43 8.25 7.77 9.57 9.1 8.6 8.12 7.63 9.92 ...
##  $ time.2: num  8.73 7.4 6.7 6.23 9.55 9.12 8.62 7.92 7.43 5.67 ...
##  $ time.3: num  5.72 7.58 7.62 7.17 5.73 9.12 6.72 8.07 7.6 9.72 ...
##  $ date.1: int  6 20 20 20 8 8 8 1 1 1 ...
##  $ date.2: int  25 32 32 32 27 27 27 27 27 27 ...
##  $ date.3: int  34 54 47 47 36 36 36 36 36 36 ...
```

```
# Pull out count matrix and covert to binary
alfl.y <- alfl.data[,c("alfl1", "alfl2", "alfl3")]
alfl.y1 <- alfl.y # Make a copy
alfl.y1[alfl.y>1] <- 1
# Standardize site-covariates
woody.mean <- mean(alfl.data$woody)
woody.sd <- sd(alfl.data$woody)
woody.z <- (alfl.data$woody-woody.mean)/woody.sd
struct.mean <- mean(alfl.data$struct)
struct.sd <- sd(alfl.data$struct)
struct.z <- (alfl.data$struct-struct.mean)/struct.sd
# Create unmarkedFrame
library(unmarked)
```

```
## Loading required package: reshape

## Loading required package: lattice

## Loading required package: parallel

## Loading required package: Rcpp

## Warning: package 'Rcpp' was built under R version 3.4.4
```

```
alfl.umf <- unmarkedFrameOccu(y=alfl.y1,
    siteCovs=data.frame(woody=woody.z, struct=struct.z),
    obsCovs=list(time=alfl.data[,c("time.1", "time.2", "time.3")],
                 date=alfl.data[,c("date.1", "date.2", "date.3")]))
# Here's an easy way to standardize covariates after making the UMF
obsCovs(alfl.umf) <- scale(obsCovs(alfl.umf))
summary(alfl.umf)
```

```
## unmarkedFrame Object
##
## 50 sites
## Maximum number of observations per site: 3
## Mean number of observations per site: 3
## Sites with at least one detection: 34
##
## Tabulation of y observations:
##  0  1
## 85 65
##
## Site-level covariates:
##      woody             struct
##  Min.   :-1.5967   Min.   :-1.80997
##  1st Qu.:-0.6019   1st Qu.:-0.77096
##  Median :-0.1045   Median : 0.02358
##  Mean   : 0.0000   Mean   : 0.00000
##  3rd Qu.: 0.6417   3rd Qu.: 0.60241
```

```
##  Max.   : 2.3826   Max.   : 3.20894
##
## Observation-level covariates:
##       time                 date
##  Min.   :-2.08258   Min.   :-1.8144
##  1st Qu.:-0.78741   1st Qu.:-1.0531
##  Median : 0.00539   Median : 0.1649
##  Mean   : 0.00000   Mean   : 0.0000
##  3rd Qu.: 0.83744   3rd Qu.: 0.6979
##  Max.   : 1.85788   Max.   : 2.2204
```

```r
# PART II. Model fitting
(fm1 <- occu(~1 ~1, alfl.umf))
```

```
##
## Call:
## occu(formula = ~1 ~ 1, data = alfl.umf)
##
## Occupancy:
##  Estimate    SE    z P(>|z|)
##     0.987 0.374 2.64  0.0083
##
## Detection:
##  Estimate    SE    z P(>|z|)
##     0.384 0.235 1.63   0.102
##
## AIC: 196.3941
```

```r
backTransform(fm1, type="state")
```

```
## Backtransformed linear combination(s) of Occupancy estimate(s)
##
##  Estimate     SE LinComb (Intercept)
##     0.728 0.0739   0.987           1
##
## Transformation: logistic
```

```r
backTransform(fm1, type="det")
```

```
## Backtransformed linear combination(s) of Detection estimate(s)
##
##  Estimate     SE LinComb (Intercept)
##     0.595 0.0567   0.384           1
##
## Transformation: logistic
```

```r
(fm2 <- occu(~date ~woody, alfl.umf))
```

```
##
## Call:
## occu(formula = ~date ~ woody, data = alfl.umf)
##
## Occupancy:
##             Estimate    SE    z P(>|z|)
## (Intercept)     1.56 0.597 2.61 0.00905
## woody           2.05 0.704 2.91 0.00358
##
```

```
## Detection:
##             Estimate    SE     z  P(>|z|)
## (Intercept)     0.56 0.241  2.33 0.019958
## date           -0.82 0.242 -3.38 0.000719
##
## AIC: 170.2177
```

Distance Sampling

Example

```r
library(unmarked)

# Load and format the data
data(issj)
head(issj)
```

```
##   issj[0-100] issj(100-200] issj(200-300]        x       y elevation
## 1           0             0             2 234870.1 3767154   51.3851
## 2           0             0             0 237083.0 3766804  156.8800
## 3           0             0             0 235732.0 3766717  144.8100
## 4           0             0             0 237605.0 3766719  184.2740
## 5           0             0             0 234239.1 3766570  111.3530
## 6           0             0             0 235005.1 3766420  204.1310
##        forest chaparral
## 1 0.022053559 0.2418731
## 2 0.006730632 0.4664184
## 3 0.016182157 0.7688672
## 4 0.257625662 0.2063583
## 5 0.000716025 0.0000000
## 6 0.159100673 0.3402549
```

```r
covs <- issj[,c("elevation", "forest", "chaparral")]
area <- pi*300^2 / 10000 # just converts distance in meters to hectares
jayumf <- unmarkedFrameDS(y=as.matrix(issj[,1:3]),
                          siteCovs=data.frame(covs, area),
                          dist.breaks=c(0,100,200,300),
                          unitsIn="m", survey="point")
# Fit model
fm1 <- distsamp(~chaparral ~chaparral + elevation + offset(log(area)),
                jayumf, keyfun="halfnorm", output="abund")
fm1
```

```
##
## Call:
## distsamp(formula = ~chaparral ~ chaparral + elevation + offset(log(area)),
##     data = jayumf, keyfun = "halfnorm", output = "abund")
##
## Abundance:
##             Estimate       SE      z  P(>|z|)
## (Intercept)  -3.5150 0.313641 -11.21 3.76e-29
## chaparral     4.1180 0.626043   6.58 4.77e-11
## elevation    -0.0021 0.000725  -2.89 3.83e-03
##
```

```
## Detection:
##                  Estimate    SE     z   P(>|z|)
## sigma(Intercept)     5.02 0.162 30.94 3.53e-210
## sigmachaparral      -1.07 0.320 -3.35  8.06e-04
##
## AIC: 964.6925
```

Abundance with an N-mixture model

Example

```
# PART 1: Set-up the data for analysis
#
#
# -------------------------- Format data ---------------------------------
# This a subset of point-count data from Chandler et al. (Auk 2009)
# alfl is Alder Flycatcher (Empidonax alnorum)

# Import data and check structure
#alfl.data <- read.csv("alfl05.csv", row.names=1)
alfl.data <- read.csv("Data/alfl05.csv", row.names=1)

str(alfl.data)
```

```
## 'data.frame':    50 obs. of  12 variables:
##  $ alfl1 : int  1 2 0 1 0 3 0 0 0 0 ...
##  $ alfl2 : int  0 3 1 0 0 1 0 0 1 0 ...
##  $ alfl3 : int  1 0 1 1 0 1 0 0 2 0 ...
##  $ struct: num  5.45 4.75 14.7 5.05 4.15 9.75 9.6 15.7 9.2 7.75 ...
##  $ woody0: int  6 1 7 6 2 8 4 0 4 3 ...
##  $ woody : num  0.3 0.05 0.35 0.3 0.1 0.4 0.2 0 0.2 0.15 ...
##  $ time.1: num  8.68 9.43 8.25 7.77 9.57 9.1 8.6 8.12 7.63 9.92 ...
##  $ time.2: num  8.73 7.4 6.7 6.23 9.55 9.12 8.62 7.92 7.43 5.67 ...
##  $ time.3: num  5.72 7.58 7.62 7.17 5.73 9.12 6.72 8.07 7.6 9.72 ...
##  $ date.1: int  6 20 20 20 8 8 8 1 1 1 ...
##  $ date.2: int  25 32 32 32 27 27 27 27 27 27 ...
##  $ date.3: int  34 54 47 47 36 36 36 36 36 36 ...
```

```
# Pull out count matrix
# No need to covert to binary as we did for occupancy model

alfl.y <- alfl.data[,c("alfl1", "alfl2", "alfl3")]

# Standardize site-covariates
woody.mean <- mean(alfl.data$woody)
woody.sd <- sd(alfl.data$woody)
woody.z <- (alfl.data$woody-woody.mean)/woody.sd

struct.mean <- mean(alfl.data$struct)
struct.sd <- sd(alfl.data$struct)
struct.z <- (alfl.data$struct-struct.mean)/struct.sd
```

```r
# Create unmarkedFrame
library(unmarked)
alfl.umf <- unmarkedFramePCount(y=alfl.y,
    siteCovs=data.frame(woody=woody.z, struct=struct.z),
    obsCovs=list(time=alfl.data[,c("time.1", "time.2", "time.3")],
                 date=alfl.data[,c("date.1", "date.2", "date.3")]))
summary(alfl.umf)
```

```
## unmarkedFrame Object
##
## 50 sites
## Maximum number of observations per site: 3
## Mean number of observations per site: 3
## Sites with at least one detection: 34
##
## Tabulation of y observations:
##  0  1  2  3
## 85 42 17  6
##
## Site-level covariates:
##      woody             struct
##  Min.   :-1.5967   Min.   :-1.80997
##  1st Qu.:-0.6019   1st Qu.:-0.77096
##  Median :-0.1045   Median : 0.02358
##  Mean   : 0.0000   Mean   : 0.00000
##  3rd Qu.: 0.6417   3rd Qu.: 0.60241
##  Max.   : 2.3826   Max.   : 3.20894
##
## Observation-level covariates:
##       time            date
##  Min.   :4.900   Min.   : 1.00
##  1st Qu.:6.550   1st Qu.:11.00
##  Median :7.560   Median :27.00
##  Mean   :7.553   Mean   :24.83
##  3rd Qu.:8.620   3rd Qu.:34.00
##  Max.   :9.920   Max.   :54.00
```

```r
# Here's an easy way to standardize covariates after making the UMF
obsCovs(alfl.umf) <- scale(obsCovs(alfl.umf))
summary(alfl.umf)
```

```
## unmarkedFrame Object
##
## 50 sites
## Maximum number of observations per site: 3
## Mean number of observations per site: 3
## Sites with at least one detection: 34
##
## Tabulation of y observations:
##  0  1  2  3
## 85 42 17  6
##
## Site-level covariates:
##      woody             struct
```

```
##  Min.   :-1.5967   Min.   :-1.80997
##  1st Qu.:-0.6019   1st Qu.:-0.77096
##  Median :-0.1045   Median : 0.02358
##  Mean   : 0.0000   Mean   : 0.00000
##  3rd Qu.: 0.6417   3rd Qu.: 0.60241
##  Max.   : 2.3826   Max.   : 3.20894
##
## Observation-level covariates:
##       time                date
##  Min.   :-2.08258   Min.   :-1.8144
##  1st Qu.:-0.78741   1st Qu.:-1.0531
##  Median : 0.00539   Median : 0.1649
##  Mean   : 0.00000   Mean   : 0.0000
##  3rd Qu.: 0.83744   3rd Qu.: 0.6979
##  Max.   : 1.85788   Max.   : 2.2204
```

```r
#
#
# PART 2: Fit some models
#
# -------------------------- Model fitting  -----------------------------

(fm1 <-  pcount(~1 ~1, alfl.umf))
```

```
## Warning in pcount(~1 ~ 1, alfl.umf): K was not specified and was set to
## 103.
```

```
##
## Call:
## pcount(formula = ~1 ~ 1, data = alfl.umf)
##
## Abundance:
##  Estimate    SE    z P(>|z|)
##     0.777 0.283 2.74 0.00608
##
## Detection:
##  Estimate    SE    z P(>|z|)
##    -0.904 0.394 -2.3  0.0217
##
## AIC: 313.9004
```

```r
backTransform(fm1, type="state")
```

```
## Backtransformed linear combination(s) of Abundance estimate(s)
##
##  Estimate    SE LinComb (Intercept)
##      2.17 0.615   0.777           1
##
## Transformation: exp
```

```r
backTransform(fm1, type="det")
```

```
## Backtransformed linear combination(s) of Detection estimate(s)
##
##  Estimate     SE LinComb (Intercept)
##     0.288 0.0808  -0.904           1
##
```

```
## Transformation: logistic
```

```r
# model with date affecting detection and woody vegetation affecting abundance
(fm4 <- pcount(~date ~woody, alfl.umf))
```

```
## Warning in pcount(~date ~ woody, alfl.umf): K was not specified and was set
## to 103.
```

```
##
## Call:
## pcount(formula = ~date ~ woody, data = alfl.umf)
##
## Abundance:
##             Estimate    SE    z  P(>|z|)
## (Intercept)    0.488 0.246 1.98 4.77e-02
## woody          0.485 0.122 3.96 7.38e-05
##
## Detection:
##             Estimate    SE     z  P(>|z|)
## (Intercept)   -0.696 0.385 -1.81 0.070533
## date          -0.664 0.175 -3.80 0.000145
##
## AIC: 279.2465
```