Part II: Modeling Abundance

Abundance: A state-variable related to "# of individuals at some time and place" – population size, N

Outline:

- **A** – Poisson regression (GLM) - building block for abundance
  – likelihood analysis in **R**
- **B** – N-mixture model: Hierarchical extension of Poisson GLM
  – Observation error: Binomial counts
  – likelihood analysis: Integrated likelihood
  – analysis in `unmarked` using `pcount()`
- **C** – Observation error: Multinomial observation data
  – Point count data with "removal" protocol
  – analysis in `unmarked` using `multmix()`

A. Modeling Abundance: Poisson GLM

Data: y_i = count of individuals at sample unit i (a point, quadrat, transect) for $i = 1, 2, \ldots, M$

- Poisson Generalized Linear Model (GLM) ("Poisson regression"):

$$y_i \sim \text{Poisson}(\lambda_i)$$

$$log(\lambda_i) = \beta_0 + \beta_1 x_i$$

$$x_i = \text{some landscape/habitat covariate}$$

- Poisson likelihood:

$$\mathcal{L}(\boldsymbol{\beta}) = \prod_{i=1}^{M} exp(-\lambda_i(\boldsymbol{\beta}))\lambda_i(\boldsymbol{\beta})^{y_i}$$

- Can be maximized easily to obtain $\hat{\boldsymbol{\beta}}$
- **R** function `glm` (with `family=poisson`)

Modeling Abundance: Poisson GLM

R exercise:

Simulate data and fit a Poisson regression model using `glm`

DOWNLOAD `moduleABUN_ALFL.R` from here:

`https://sites.google.com/site/unmarkedinfo/home/webinars/2012-january/r-scripts`

Using GLM (simulated Data)

```
## PART 1: Set-up the data (in this case, simulate it):
##
# Create a covariate called vegHt
nSites <- 100
set.seed(443)                # so that we all get the same values of vegHt
vegHt <- runif(nSites, 1, 3) # uniform from 1 to 3
# Suppose that expected population size increases with vegHt
lambda <- exp(-3 + 2*vegHt)

# Now we go to 100 sites and observe the # of individuals (perfectly)
#
N <- rpois(nSites, lambda)

## PART 2: FIT SOME MODELS (in this case, one)
##
# We can fit a model that relates abundance to vegHt using the glm() function
# with "family=Poisson":

fm.glm1 <- glm(N ~ vegHt, family=poisson)

## PART 3: ANALYZE THE RESULTS
##
plot(vegHt, N, xlab="Vegetation height", ylab="Abundance (N)")
glm1.est <- coef(fm.glm1)
plot(function(x) exp(-3 + 2*x), 1, 3, add=TRUE, lwd=2)
plot(function(x) exp(glm1.est[1] + glm1.est[2]*x), 1, 3, add=TRUE,
     lwd=2, col="blue")
legend(1, 15.9, c("Truth", "Estimate"), col=c("black", "blue"), lty=1, lwd=2)
```

Extending the Poisson GLM

- When we use a Poisson GLM we think of y_i as abundance, N_i, or maybe an "index"
- In practice, it is an observation of N_i made with error.
- We would like to extend the Poisson model in way that accommodates both the desired model for actual abundance, N_i, and also *observation error*.
- This is the N-mixture model

B. N-mixture model

- Is a "hierarchical extension" of the Poisson GLM
- We use the Poisson GLM as the base model for N but we regard N as a *latent variable* (i.e., unobserved)
- We augment the Poisson GLM with a model that describes how the observations y_i are related to the latent variable N.
- The N-mixture model:

  1. Abundance model

  $$N_i \sim \text{Poisson}(\lambda_i)$$

  2. Observation model (j = replicate count)

  $$y_{ij} \sim \text{Binomial}(N_i, p)$$

- aka "Binomial/Poisson mixture"

The Real World: Analysis of ALFL Data

Typical point count data structure: replicated counts

	alfl1	alfl2	alfl3	struct	woody0	woody	time.1	time.2	time.3	date.1	date.2	date.3
crick1_05	1	0	1	5.45	6	0.30	8.68	8.73	5.72	6	25	34
his1_05	2	3	0	4.75	1	0.05	9.43	7.40	7.58	20	32	54
hisw1_05	0	1	1	14.70	7	0.35	8.25	6.70	7.62	20	32	47
hisw2_05	1	0	1	5.05	6	0.30	7.77	6.23	7.17	20	32	47
kenc1_05	0	0	0	4.15	2	0.10	9.57	9.55	5.73	8	27	36
kenc2_05	3	1	1	9.75	8	0.40	9.10	9.12	9.12	8	27	36
kenc3_05	0	0	0	9.60	4	0.20	8.60	8.62	6.72	8	27	36
kence1_05	0	0	0	15.70	0	0.00	8.12	7.92	8.07	1	27	36
kence2_05	0	1	2	9.20	4	0.20	7.63	7.43	7.60	1	27	36
kense1_05	0	0	0	7.75	3	0.15	9.92	5.67	9.72	1	27	36

Key considerations:

1. **Repeated measures:** of sites – what do we do about that?
2. **Detectability:** Variation in observed counts due to non-detection of individuals.

Poisson/binomial mixture (aka N-mixture) is a natural model for this "design" which accommodates both of these features.

N-mixture Model

Design: $t = 1, 2, \ldots, T$ surveys of $i = 1, 2, \ldots, M$ points

Data structure:

	$t = 1$	$t = 2$	N
site 1	y_{11}	y_{21}	N_1
site 2	y_{12}	y_{22}	N_2
site 3	y_{13}	y_{23}	N_3
.	.	.	.
.	.	.	.
.	.	.	.
.	.	.	.
site M	y_{1M}	y_{2M}	N_M

Model structure:

- **Observation Model -** condition on N

  Binomial sampling:

  $$y_{it} \sim \text{Binomial}(N_i, p)$$

- **Joint distribution of observations**:

  $$f(\{\mathbf{y}_i\}|\{N_i\}, p) = \prod_i^M \text{Bin}(y_{i1}; N_i, p)\text{Bin}(y_{i2}; N_i, p)$$

- N is not observed in this model – it is a **latent variable** (random effect)

- Abundance model: $N \sim \text{Poisson}(\lambda)$

Likelihood Analysis of the N-mixture Model

Marginal/Integrated Likelihood – the classical method of analyzing **random effects**

Marginal likelihood (free of random effects) for N-mixture model –

$$f(\{\mathbf{y}_i\}|p,\lambda) = \prod_i \left(\sum_{N_i=0}^{\infty} \{ \prod_{t=1}^{2} \text{Bin}(y_{it}|N_i,p)\} \text{Poisson}(N_i|\lambda) \right)$$

- $\infty \approx 100$ or so, in practice.
- Yields MLEs of λ (density) and p (det. prob.)
- Can have p_{it} and covariates
- Also λ_i and covariates
- Negative binomial or other models for over-dispersion

Likelihood optimization in R

```
lik0<-function(parms ){
    ones<-rep(1,M)
    p<-expit(parms[1])
    lam<-exp(parms[2])
    lik<-rep(NA,M)
    LIK<-matrix(NA,nrow=Nmax+1,ncol=3)

    for(i in 1:M){
       gN<-dpois(0:Nmax,lam)
       gN<-gN/sum(gN)
       dvec<-ymat[i,]
       naflag<-is.na(dvec)
       LIK[,1]<-dbinom(dvec[1],0:Nmax,p)
       LIK[,2]<-dbinom(dvec[2],0:Nmax,p)
       LIK[,3]<-dbinom(dvec[3],0:Nmax,p)
       LIK[,naflag]<-1
       binprod<-apply(LIK,1,prod)
       lik[i]<-sum(binprod*gN)
     }

-1*sum(log(lik))
}

nlm(lik0,c(0,0),hessian=TRUE)
```

N-mixture model in `unmarked`: `pcount`

R exercise:

Simulate data and fit the N-mixture model using `pcount`

DOWNLOAD `moduleABUN_ALFL.R` from here:

`https://sites.google.com/site/unmarkedinfo/home/webinars/2012-january/r-scripts`

N-mixture model in unmarked: pcount

Return to our simulated data example (basic analysis)

```
## PART 1: Set-up data (simulate it)
##
nVisits <- 4
p <- 0.6
y <- matrix(NA, nSites, nVisits)
for(i in 1:nSites) {
    y[i,] <- rbinom(nVisits, N[i], p)
}

# Format for unmarked and summarize
library(unmarked)
umf <- unmarkedFramePCount(y=y, siteCovs=as.data.frame(vegHt))
summary(umf)
```

N-mixture analysis, continued....

```
## PART 2: Fit some models
###
# pcount() function
# Detection covariates follow first tilde, then come abundance covariates

fm.nmix1 <- pcount(~1 ~vegHt, data=umf)

# Note the following warning message:
#> fm.nmix1 <- pcount(~1 ~vegHt, data=umf)
#Warning message:
#In pcount(~1 ~ vegHt, data = umf) : K was not specified and was set to 116.
# K is upper limit of summation for calculating the likelihood

# Consider other abundance models:
#  NB = negative binomial,
#  ZIP = zero-inflated Poisson
#  [currently no others available]
##
fm.nmix2<-  pcount(~1 ~vegHt, data=umf,mixture="NB")
fm.nmix3<-  pcount(~1 ~vegHt, data=umf,mixture="ZIP")
beta1 <- coef(fm.nmix1)

##
## PART 3: DO SOME ANALYSIS OF THE RESULT
##
# Or suppose you want predictions for new values of vegHt, say 1.2 and 3.1
newdat <- data.frame(vegHt=c(1.2, 3.1))
predict(fm.nmix1, type="state", newdata=newdat)
```

Real Examples

1. ALFL data

- Work flow/processing is analogous to the occupancy models
- Demonstrate goodness-of-fit

2. Swiss Bird Survey (MHB)

- Some cool prediction problems

ALFL Data

- Alder Flycatcher (*Empidonax alnorum*) from Chandler et al. (Auk 2009)

- N-mixture analysis using `pcount` is entirely analogous to the occupancy analysis using `occu`. All of the summary functions and manipulations are identical.

ALFL Data

Part 1: Set-up the data for analysis –

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

# Extract matrix of COUNTS
alfl.y <- alfl.data[,c("alfl1", "alfl2", "alfl3")]

# Standardize site-covariates. ALWAYS STANDARDIZE
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)
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")]))

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

ALFL Data

Part 2: Fit some models –

```
# -------------------------- Model fitting  ----------------------------
(fm1 <-  pcount(~1 ~1, alfl.umf))
(fm2 <- pcount(~date+time ~1, alfl.umf))
(fm3 <- pcount(~date+time ~woody, alfl.umf))
(fm4 <- pcount(~date+time ~woody+struct, alfl.umf))
(fm5 <- pcount(~date+time ~1, alfl.umf,mixture="NB"))
(fm6 <- pcount(~date+time ~1, alfl.umf,mixture="ZIP"))
(fm7 <- pcount(~date+time ~woody,alfl.umf,mixture="ZIP"))
(fm8 <- pcount(~date+time ~struct,alfl.umf,mixture="ZIP"))
(fm9 <- pcount(~date+time ~woody+struct, alfl.umf,mixture="ZIP"))
(fm10<- pcount(~date+time ~woody+struct, alfl.umf,mixture="NB"))

# -------------------------- Model selection ----------------------------
# Put the fitted models in a "fitList"
fms <- fitList("lam(.)p(.)"                    = fm1,
               "lam(.)p(date+time)"            = fm2,
               "lam(woody)p(date+time)"        = fm3,
               "lam(woody+struct)p(date+time)" = fm4,
               "lam(.)p(date+time)NB"          = fm5,
               "lam(.)p(date+time)ZIP"         = fm6,
               "lam(woody)p(date+time)ZIP"     = fm7,
               "lam(struct)p(date+time)ZIP"    = fm8,
               "lam(woody+struct)p(date+time)ZIP"=fm9,
               "lam(woody+struct)p(date+time)NB" =fm10)

# Rank them by AIC
(ms <- modSel(fms))

# Table with everything you could possibly need
coef(ms)
```

ALFL Data

Part 3: Analysis of results – PREDICTION

```
# Expected detection probability as function of time of day
# We standardized "time", so we predict over range of values on that scale
# We must fix "date" at some arbitrary value (let's use the mean)
newData1 <- data.frame(time=seq(-2.08, 1.86, by=0.1), date=0)
E.p <- predict(fm4, type="det", newdata=newData1, appendData=TRUE)

# Plot it
plot(Predicted ~ time, E.p, type="l", ylim=c(0,1),
     xlab="time of day (standardized)",
     ylab="Expected detection probability")
lines(lower ~ time, E.p, type="l", col=gray(0.5))
lines(upper ~ time, E.p, type="l", col=gray(0.5))

# Expected abundance over range of "woody"
newData2 <- data.frame(woody=seq(-1.6, 2.38,,50),struct=seq(-1.8,3.2,,50))
E.N <- predict(fm4, type="state", newdata=newData2, appendData=TRUE)
head(E.N)

# Plot predictions with 95% CI
plot(Predicted ~ woody, E.N, type="l", ylim=c(-.1,max(E.N$Predicted)),
     xlab="woody vegetation (standardized)",
     ylab="Expected abundance, E[N]")
lines(lower ~ woody, E.N, type="l", col=gray(0.5))
lines(upper ~ woody, E.N, type="l", col=gray(0.5))
```

ALFL Data

Part 3: Analysis of results – GOODNESS-OF-FIT

```
### Bootstrap GoF analysis.
### Best model is in "fm4" object

# Function returning three fit-statistics.
fitstats <- function(fm) {
    observed <- getY(fm@data)
    expected <- fitted(fm)
    resids <- residuals(fm)
    sse <- sum(resids^2)
    chisq <- sum((observed - expected)^2 / expected)
    freeTuke <- sum((sqrt(observed) - sqrt(expected))^2)
    out <- c(SSE=sse, Chisq=chisq, freemanTukey=freeTuke)
    return(out)
    }

(pb <- parboot(fm4, fitstats, nsim=100, report=1))
print(pb)

## Now lets bootstrap a summary statistic
## This is not too meaningful right now but we will do a similar thing later in
## a more relevant context

# Total population size (derived parameter)
Nhat <- function(fm) {
    N <- sum(predict(fm, type="state")$Predicted, na.rm=TRUE)
    }

(pb.N <- parboot(fm4, Nhat, nsim=25, report=5))
```

N-mixture model: Analysis of the Swiss Bird Data

R exercise:

Analyze the Swiss Bird Survey Data using `pcount`

DOWNLOAD `moduleABUN_MHB.R` from here:

`https://sites.google.com/site/unmarkedinfo/home/webinars/2012-january/r-scripts`

Application 2: Swiss Bird Data

Swiss survey of common breeding birds (abbrev "MHB").

- Observers walk route through 1 km quadrats, route length varies between about 3.5 km and 12 km.
- Observer walks same route 3 times (early, mid, late breeding season)
- Records all birds, what they're doing, where they are (on a map), and other information

Here we analyze the quadrat counts of the willow tit:

y_{it} = # of "territories" of the willow tit on quadrat i, sample t.

Swiss Bird Data

Willow tit data for 15 of 237 1 km^2 quadrats:

counts			abun covs			sampling/observation covariates								
c.1	c.2	c.3	elev	forest	length	day.1	day.2	day.3	dur.1	dur.2	dur.3	int.1	int.2	int.3
1	0	0	1300	32	6.1	35	58	75	260	270	290	42.6	44.3	47.5
2	1	0	1270	66	4.5	28	39	61	176	145	150	39.1	32.2	33.3
0	0	0	380	45	6.2	21	40	86	180	160	165	29.0	25.8	26.6
3	3	1	550	31	6.9	20	42	63	195	240	270	28.3	34.8	39.1
0	0	0	390	8	4.6	35	50	75	150	130	140	32.6	28.3	30.4
0	1	0	1380	78	3.7	54	75	99	160	160	150	43.2	43.2	40.5
0	0	0	530	37	5.7	30	50	76	225	210	225	39.5	36.8	39.5
0	0	0	1190	18	4.0	26	47	71	165	175	240	41.2	43.8	60.0
1	1	1	1490	54	5.5	47	62	77	300	325	321	54.5	59.1	58.4
0	0	0	920	6	4.3	20	40	61	170	90	98	39.5	20.9	22.8
0	0	0	620	3	3.4	17	46	61	270	271	215	79.4	79.7	63.2
0	0	0	540	27	4.2	15	45	66	180	240	180	42.9	57.1	42.9
1	0	1	820	56	5.3	37	60	90	290	315	295	54.7	59.4	55.7
2	3	4	1220	66	5.6	45	61	86	225	240	255	40.2	42.9	45.5
1	1	2	1180	43	5.1	39	60	83	325	365	350	63.7	71.6	68.6
0	0	0	730	40	6.2	23	45	58	390	391	406	62.9	63.1	65.5

Swiss Bird Data

Goals:

- What effects breeding abundance of this bird the willow tit?
- Make a breeding abundance map of the species.
- Estimate total population size.

Hierarchical model:

1. Model for quadrat population size:

$$\begin{aligned} N_i &\sim \text{Poisson}(\lambda_i) \\ log(\lambda_i) &= \beta_0 + covariates \end{aligned}$$

covariates = forest cover, elevation, route length

2. Binomial observation model

Detection covariates?

Here we use the null model with $p = \text{const.}$

Variable route length

We use **inverse** of route-length as a covariate on abundance. Idea is the "exposed population", say N_i, for a quadrat is less than the actual population of the quadrat, say M_i, but as $L \to \infty$ (quadrat becomes saturated with sampling effort) then $N_i \to M_i$.

Quadrat population size model:

$$\begin{aligned} M_i &\sim Poisson(\lambda_i) \\ log(\lambda_i) &= \beta_0 + \beta_1 elev + \beta_2 forest \\ N_i|M_i &\sim Binomial(M_i, \phi(L_i)) \end{aligned}$$

where we model

$$\phi(L_i) = exp(-\beta_3/L_i)$$

Under this model the marginal distribution of N_i is

$$N_i \sim \text{Poisson}(\phi(L_i)\lambda_i)$$

Therefore

$$log(E[N_i]) = \beta_0 + \beta_1 elev + \beta_2 forest - \beta_3(1/L_i)$$

This means:

- Use inverse of route length to account for $N_i \rightarrow M_i$ as L_i increases.
- For making predictions we set $1/L_i = 0$ so that the prediction applies to saturation sampling effort.

Swiss Bird Data: Analysis

Part 1: Set-up the data for analysis

```
library("unmarked")
#
# Read in some data from the Swiss MHB survey (Swiss equivalent of BBS)
#
mhbdata <- read.csv("http://sites.google.com/site/unmarkedinfo/home/
webinars/2012-january/data/wtmatrix.csv?attredirects=0&d=1")
# or do this:
#mhbdata<-read.csv("wtmatrix.csv")

mhb.y<-mhbdata[,c("c.1","c.2","c.3")]
mhbdata[,"length"]<-1/mhbdata[,"length"]
mhb.umf<-unmarkedFramePCount(y=mhb.y,
siteCovs=data.frame(elev=mhbdata[,"elev"],forest=mhbdata[,"forest"],
                    length=mhbdata[,"length"]),
obsCovs=list(duration=mhbdata[,c("dur.1","dur.2","dur.3")],
             day = mhbdata[,c("day.1","day.2","day.3")])  )

# standardize the covariates:
# this is extremely handy:
obsCovs(mhb.umf)<- scale(obsCovs(mhb.umf))

#
# NOTE: Do not standardize 1/length because we are using 1/length
# for a specific reason
#
siteCovs(mhb.umf)$forest<-scale(siteCovs(mhb.umf)$forest)
siteCovs(mhb.umf)$elev<-scale(siteCovs(mhb.umf)$elev)
```

Swiss Bird Data

Part 2: Fit some models – (abundance only)

```
fm1 <- pcount(~1 ~1, mhb.umf)
fm2 <- pcount(~1 ~elev,mhb.umf)
fm3 <- pcount(~1 ~forest,mhb.umf)
fm4 <- pcount(~1 ~length,mhb.umf)
fm5 <- pcount(~1 ~forest+elev,mhb.umf)
fm6 <- pcount(~1 ~forest+length,mhb.umf)
fm7 <- pcount(~1 ~elev+length,mhb.umf)
fm8 <- pcount(~1 ~forest+elev+length,mhb.umf)
fm9 <- pcount(~1 ~elev + I(elev^2),mhb.umf)
fm10<- pcount(~1 ~forest+elev+I(elev^2) + length,mhb.umf)

mspart1<- fitList(
"lam(.)p(.)"                        = fm1,
"lam(elev)p(.)"                     = fm2,
"lam(forest)p(.)"                   = fm3,
"lam(length)p(.)"                   = fm4,
"lam(forest+elev)p(.)"              = fm5,
"lam(forest+length)p(.)"            = fm6,
"lam(elev+length)p(.)"              = fm7,
"lam(forest+elev+length)p(.)"       = fm8,
"lam(elev + elev^2)p(.)"            = fm9,
"lam(forest+elev+elev^2+length)p(.)" = fm10)
```

Swiss Bird Data: Results

```
> (ms1 <- modSel(mspart1))
                                   nPars     AIC  delta    AICwt cumltvWt
lam(forest+elev+elev^2+length)p(.)     6 1209.80   0.00  1.0e+00     1.00
lam(forest+elev+length)p(.)            5 1261.66  51.85  5.5e-12     1.00
lam(forest+elev)p(.)                   4 1287.93  78.13  1.1e-17     1.00
lam(elev + elev^2)p(.)                 4 1371.72 161.92  6.9e-36     1.00
lam(elev+length)p(.)                   4 1518.77 308.97  8.1e-68     1.00
lam(elev)p(.)                          3 1579.22 369.42  6.1e-81     1.00
lam(forest+length)p(.)                 4 1640.09 430.29  3.7e-94     1.00
lam(forest)p(.)                        3 1645.05 435.25  3.1e-95     1.00
lam(length)p(.)                        3 1774.16 564.36 2.8e-123     1.00
lam(.)p(.)                             2 1784.64 574.84 1.5e-125     1.00
>

> print(coef(ms1),digits=2)
                                   lam(elev) lam(forest) lam(I(elev^2)) lam(Int) lam(length) p(Int)
lam(forest+elev+elev^2+length)p(.)      2.27        0.63          -0.86     1.22        -6.0   0.14
lam(forest+elev+length)p(.)             1.42        0.89             NA     0.60        -5.0   0.20
lam(forest+elev)p(.)                    1.41        0.97             NA    -0.50          NA   0.24
lam(elev + elev^2)p(.)                  2.43          NA          -1.38     0.50          NA   0.34
lam(elev+length)p(.)                    0.91          NA             NA     1.52        -6.9   0.43
lam(elev)p(.)                           0.80          NA             NA     0.12          NA   0.49
lam(forest+length)p(.)                    NA        0.65             NA     0.72        -2.7   0.59
lam(forest)p(.)                           NA        0.66             NA     0.16          NA   0.59
lam(length)p(.)                           NA          NA             NA     1.06        -3.3   0.64
lam(.)p(.)                                NA          NA             NA     0.37          NA   0.64
```

Swiss Bird Data

Part 3: Goodness-of-fit

```
## Does this model fit worth a darn?
#  Function returning three fit-statistics.
#  NOTE: na.rm=TRUE !!!!!!

fitstats <- function(fm) {
    observed <- getY(fm@data)
    expected <- fitted(fm)
    resids <- residuals(fm)
    sse <- sum(resids^2,na.rm=TRUE)
    chisq <- sum((observed - expected)^2 / expected,na.rm=TRUE)
    freeTuke <- sum((sqrt(observed) - sqrt(expected))^2,na.rm=TRUE)
    out <- c(SSE=sse, Chisq=chisq, freemanTukey=freeTuke)
    return(out)
    }

(pb <- parboot(fm10, fitstats, nsim=20, report=1))

# Model does not fit!
# Three choices:
#
# 1. We can expand this model or tinker with components of it (NB, ZIP)
#   [takes a long time to run these models]
# 2. We can seek out an implausible data-generating model that fits.
#   [might satisfy referee to have good p-value]
#    any model will fit if we ladel in enough random effects
#    does that make it useful?
#    plausible?
# 3. We can proceed.
#
```

Swiss Bird Data

Part 3: Analysis of Results – Elevation effect

```
## Predict response of E[N] vs. elevation -- find optimal elevation

# remember length = 0 is saturation sampling because length = 1/L
new<- data.frame(elev=seq(-1.5,2.42,,50),forest=0,length=0)
pred<-predict(fm10,type="state",newdata=new,appendData=TRUE)

plot(Predicted ~ elev, pred,type="l",xlab="elevation (standardized)",
 ylab="Expected # territories")
lines(lower ~ elev, pred,type="l",col="red")
lines(upper ~ elev, pred,type="l",col="red")

# what is primo elevation for the willow tit?
# put on your calculus hat.....
#   quadratic response:
#      y = a + b*x + c*x2
#   differentiate and set to 0:
#      dy/dx = b + 2*c*x = 0
#   solve
#      xopt = -b/(2*c)

b<- coef(ms1)[1,1]
c<- coef(ms1)[1,3]
elev.opt<- -b/(2*c)
elev.mean<- attr(siteCovs(mhb.umf)$elev,"scaled:center")
elev.sd <- attr(siteCovs(mhb.umf)$elev,"scaled:scale")
elev.opt <- elev.opt*elev.sd + elev.mean
```

Swiss Bird Data: Spatial Analysis of Results

Part 3: Analysis of Results – Mapping Abundance

Get some landscape data:

```
landscape <-
read.csv("http://sites.google.com/site/unmarkedinfo/home/webinars/
          2012-january/data/Swiss_landscape.csv?attredirects=0&d=1")
## Or do this landscape<-read.csv("Swiss_landscape.csv")

gelev<- landscape[,"medel"]   # median elevation of quadrat
gforest<-landscape[,"forest"]
grid<-landscape[,c("x","y")]

# lets plot these variables to see how they look
#
# grab utility functions including spatial.plot
source("http://sites.google.com/site/unmarkedinfo/home/webinars/
        2012-january/data/utils.R?attredirects=0&d=1")

par(mar=c(3,3,3,5),mfrow=c(2,1))
spatial.plot(grid,gelev)
spatial.plot(grid,gforest)
# this is cool
```

Make some predictions:

```
# first we have to get the landscape covariates on the same scale as
# the model covariates. i.e., standardized.

elev.mean<- attr(siteCovs(mhb.umf)$elev,"scaled:center")
elev.sd <- attr(siteCovs(mhb.umf)$elev,"scaled:scale")
gelev<- (gelev - elev.mean)/elev.sd
forest.mean<- attr(siteCovs(mhb.umf)$forest,"scaled:center")
forest.sd <- attr(siteCovs(mhb.umf)$forest,"scaled:scale")
gforest<- (gforest - forest.mean)/forest.sd

# make a dataframe for grid covariates
# remember length = 0 is saturation sampling because length = 1/L
new<- data.frame(elev=gelev,forest=gforest,length=0)
pred<-predict(fm10,type="state",newdata=new,appendData=TRUE)
head(pred)
```

Summary of Swiss Bird Survey Analysis

Distribution map:

- $\hat{N} = 212670$ breeding pairs of willow tits.
- bootstrap SE?

N-mixture Model: Summary

1. Observation model (data model, likelihood) –

$$f(\mathbf{y}_i|N_i, p) = \prod_t \left\{ \binom{N_i}{y_{it}} p^{y_{it}} (1-p)^{N_i - y_{it}} \right\}$$

2. Abundance model (stochastic part) –

$$N_i \sim \text{Poisson}(\lambda_i)$$

2b. Abundance model (deterministic part) –

$$log(\lambda_i) = \beta_0 + \beta_1 x_i$$

• Flexibility by changing the observation or abundance model components of the model.

• We focus modeling individual components of the problem independently.

PART C: Multinomial Observation Models

Multinomial Sampling Protocols

Bird sampling protocols: Many yield a **multinomial** $\mathbf{y}_i$ of counts for site $i = 1, 2, \ldots, M$.

Examples:

1. Capture/recapture

- T capture periods yields $K = 2^T - 1$ capture-history frequencies, $\mathbf{y}_i = \{y_{i,k}\}$. e.g., 2 sample periods:

history	frequency	
10	$y_{i,1}$	
01	$y_{i,2}$	K=3 **observable**
11	$y_{i,3}$	
00	$y_{i,0}$	$\longleftarrow$ **unobserved**

$$y_{i,0} = N_i - \Sigma_{k=1}^{3} y_{i,k}$$

Sampling protocols, continued

2. Multiple observer counts

- Counts are capture history frequencies
- Events indexed by observer instead of period

3. "Removal" counts

- y_{ik} = # birds first observed in time interval t

4. Distance sampling

- y_{ik} = # birds observed in distance interval k

The Observation Model

Multinomial likelihood for data from a single site:

$$f(\mathbf{y}_i|N_i, \mathbf{p}) = \frac{N_i!}{\left\{\Pi_{k=1}^{K} y_{ik}!\right\}(N_i - y_{i.})!}\left\{\prod_k \pi_k^{y_{ik}}\right\}(1-\pi.)^{N_i - y_{i.}}$$

- k is a generic index to distance class, capture-history, time interval, etc...

- π_k is a function of various (detection) parameters, $\mathbf{p}$

Difference among the various protocols is manifest in the parameterization of detectability in π_k.

Example: Capture-recapture (2 periods)

$$\begin{aligned}
\pi_1 &= \Pr('10') = \ p_1(1-p_2) \\
\pi_2 &= \Pr('01') = \ (1-p_1)p_2 \\
\pi_3 &= \Pr('11') = \ p_1 p_2
\end{aligned}$$

Example: Removal (3 periods)

$$\begin{aligned}
\pi_1 &= \Pr('1xx') = \ p_1 \\
\pi_2 &= \Pr('01x') = \ (1-p_1)p_2 \\
\pi_3 &= \Pr('001') = \ (1-p_1)(1-p_2)p_3
\end{aligned}$$

Factors that influence p –

$$\text{logit}(p_{it}) = a_0 + a_1 x_{it}$$

e.g., x_{it} = time (of day), temp., effort

The Hierarchical Model

1. Data model (the likelihood) –

$$f(\mathbf{y}_i|N_i, \mathbf{p}) \propto \frac{N_i!}{(N_i - y_{.k})!} \left\{ \prod_k \pi_k^{y_{ik}} \right\} (1 - \pi_.)^{N_i - y_{.k}} \tag{1}$$

2. Abundance model –

$$N_i \sim g(N; \theta) \tag{2}$$

- local abundance is a "random effect"

Likelihood Analysis

1. Remove them from the likelihood by integration.

The **integrated likelihood**:

$$f(\mathbf{y}|\mathbf{p},\theta) = \prod_i \left\{ \sum_{N_i=0}^{\infty} MN(\mathbf{y}_i|N_i, \boldsymbol{\pi}(\mathbf{p}))g(N_i|\theta) \right\}$$

With **Poisson** prior, simplifies to:

$$f(\mathbf{y}|\mathbf{p},\theta) = \prod_i \prod_k Poi(\lambda_i \pi_k)$$

2. Be Bayesian and analyze the posterior distribution by MCMC.
[details omitted]

Doing it in `unmarked`

`unmarked` has 2 main functions for fitting multinomial abundance models:

> `multinomPois` – Multinomial-Poisson mixture (computationally efficient)
>
> `gmultmix` – Generalized multinomial-mixture model: other abundance models and **open populations**

For set-up:

`unmarkedFrameGMM` or `unmarkedFrameMPois`: requires `type = removal` or `type = double` but other types can be described.

Multinomial-Mixture Model: Analysis of Removal Data

R exercise:

Analyze bird point count data collected using a "removal" protocol using the functions `multinomPois` and `multmix`.

DOWNLOAD `moduleABUN_multinomial.R` from here:

`https://sites.google.com/site/unmarkedinfo/home/webinars/2012-january/r-scripts`

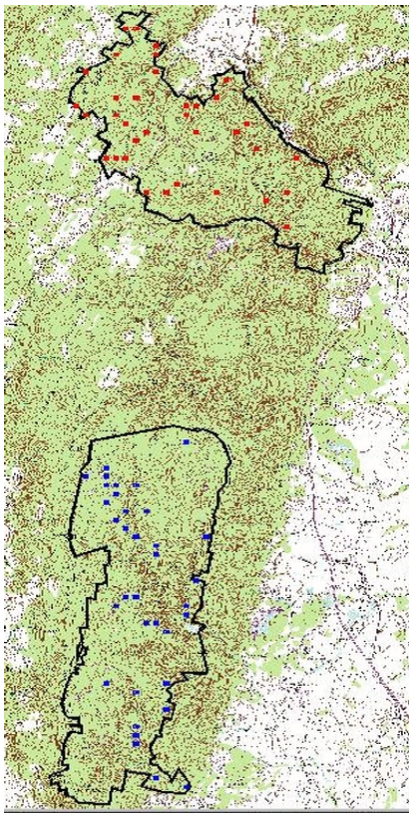
Example: Removal sampling protocol

Maryland forest comprising *Catoctin Mountain National Park*, Frederick Co. WMA, and various other land units.

Thanks to:

Deanna Dawson, USGS

Scott Bates, NPS



Sampling Details

- 70 point count locations
- Sampling protocol: "mental removal"
- 4 removal intervals
- **Habitat covariates**: Understory cover (UFC), basal area of large trees (TRBA)
- data set is provided with `unmarked`: `data(ovendata)`

Multinomial example, continued

Removal counts:

$$\mathbf{y}_i = (y_{i1}, y_{i2}, y_{i3}, y_{i4}); \quad i = 1, 2, \ldots, 70$$

Multinomial cell probabilities:

$$\begin{aligned}
\pi_1 &= p \\
\pi_2 &= (1-p)p \\
\pi_3 &= (1-p)^2 p \\
\pi_4 &= (1-p)^3 p
\end{aligned}$$

Poisson Abundance Model:

$$log(\lambda_i) = b_0 + b_1 UFC_i + b_2 BA_i$$

Ovenbird data –

	t=1	t=2	t=3	t=4
.	.	.	.	.
point 34	0	0	0	0
point 35	0	0	0	0
point 36	2	0	1	0
point 37	1	0	0	0
point 38	0	1	0	0
point 39	1	1	0	0
point 40	1	0	0	0
point 41	2	1	0	0
point 42	0	1	0	0
point 43	1	2	0	0
point 44	0	0	0	1
point 45	1	0	1	0
.	.	.	.	.
totals	49	16	5	7

←

Using multinomPois

```
##
## PART 1: Set-up the data
##

data(ovendata)  # built-in data

# note: every covariate is a "siteCovs" with a single primary sample, even if
# the covariate effects "p"

ovenFrame <- unmarkedFrameMPois(ovendata.list$data,
    siteCovs=as.data.frame(scale(ovendata.list$covariates[,-1])),
    type = "removal")
```

Using multinomPois

```
##
## PART 2:Fit a bunch of models
##

fm0 <- multinomPois(~1 ~1,ovenFrame)
fm1 <- multinomPois(~ 1 ~ ufp + trba, ovenFrame)
fm2 <- multinomPois(~ 1 ~ ufp + trba + ufp*trba, ovenFrame)
fm3 <- multinomPois(~ ufp ~ ufp + trba, ovenFrame)
fm4 <- multinomPois(~ ufp ~ ufp + trba +ufp*trba, ovenFrame)

ms<- fitList(
"lam(.)p(.)"                              = fm0,
"lam(ufp+trba)p(.)"                       = fm1,
"lam(ufp+trba+ufp*trba)p(.)"              = fm2,
"lam(ufp+trba)p(ufp)"                     = fm3,
"lam(ufp+trba+ufp*trba)p(ufp)"            = fm4)

# Rank them by AIC
(ms1 <- modSel(ms))

# Table with everything you could possibly need
coef(ms)
```

Using gmultmix

```
## numPrimary=1
## extra formula, order: lambda phi p
## DIFFERENT FORMAT AND ORDER
## LIKELIHOODS ARE SCALED DIFFERENTLY TOO (DON'T COMPARE ACROSS METHODS)
##
data(ovendata)
ovenFrame <- unmarkedFrameGMM(ovendata.list$data,
    siteCovs=as.data.frame(scale(ovendata.list$covariates[,-1])),numPrimary=1,
    type = "removal")

fm0 <- gmultmix(~1 , ~1,  ~1 ,data=ovenFrame)
fm1 <- gmultmix(~ ufp+trba , ~1 , ~ 1, data=ovenFrame)
fm2 <- gmultmix(~ ufp+trba+ufp*trba , ~1 , ~ 1, data=ovenFrame)
# maybe p also depends on understory foliage?
fm3 <- gmultmix(~ ufp +trba, ~1 , ~ ufp,data= ovenFrame)
fm4 <- gmultmix(~ ufp + trba+ufp*trba,  ~1 , ~ ufp,data= ovenFrame)

gms<- fitList(
"lam(.)p(.)"                                = fm0,
"lam(ufp+trba)p(.)"                         = fm1,
"lam(ufp+trba+ufp*trba)p(.)"                = fm2,
"lam(ufp+trba)p(ufp)"                       = fm3,
"lam(ufp+trba+ufp*trba)p(ufp)"              = fm4)

# Rank them by AIC
(gms1 <- modSel(gms))

# Table with everything you could possibly need
coef(ms)
toExport <- as(gms, "data.frame")   # generates an error -- bug needs fixed
```

Ovenbird Results

```
> ms1
                              nPars    AIC delta AICwt cumltvWt
lam(ufp+trba)p(.)                 4 326.14  0.00  0.27     0.27
lam(.)p(.)                        2 326.28  0.14  0.25     0.51
lam(ufp+trba)p(ufp)               5 326.63  0.49  0.21     0.72
lam(ufp+trba+ufp*trba)p(.)        5 327.17  1.03  0.16     0.88
lam(ufp+trba+ufp*trba)p(ufp)      6 327.72  1.58  0.12     1.00
> print(coef(ms1),digits=2)
                              lambda(Int) lambda(trba) lambda(ufp) lambda(ufp:trba) p(Int) p(ufp)
lam(ufp+trba)p(.)                    0.10        -0.17        0.10               NA   0.29     NA
lam(.)p(.)                           0.13           NA          NA               NA   0.29     NA
lam(ufp+trba)p(ufp)                  0.11        -0.17        0.13               NA   0.34  -0.34
lam(ufp+trba+ufp*trba)p(.)           0.15        -0.15        0.16             0.13   0.29     NA
lam(ufp+trba+ufp*trba)p(ufp)         0.16        -0.15        0.19             0.13   0.34  -0.33
```

Conclusion:

- positive understory foliage effect on $E[N]$
- possibly negative effect on p
- negative effect of basal area of large trees on $E[N]$.

Summary

Hierarchical abundance models –

- `unmarked` – hierarchical models for explicit modeling of both abundance and observation processes (detection probability).

- `unmarked` includes general implementations and support functions for N-mixture models (binomial observation model) and multinomial-mixture models.