

Introduction to Statistical Inference

Theme of workshop (and book): Analyzing HMs using both classical and Bayesian methods b/c if you want to be an effective user, you need to understand and be able to use both.

Topics covered here

- A. Parametric inference (Bayesian and classical)
- B. Implementation in **R** (both MLE and MCMC)
- C. Logistic Regression (not a HM)
- D. Occupancy Model (a HM)

Inference for Statistical Models

- **Parametric inference:** Explicit probability assumptions about data. Inference proceeds assuming model is truth.

Two popular flavors:

- **Classical inference:** Joint probability distribution of observations forms the **likelihood**. We maximize it to obtain MLEs and do other fun things to it.
- **Bayesian inference:** Based on the posterior distribution – proportional to the joint probability distribution of all random quantities in the model: data, random effects, parameters.

Sad Caricature of the Difference: “Parameters are random”

Sometimes you hear: “*The difference between Bayesian and frequentist inference is that, in Bayesian analysis, parameters are random but, in frequentist analysis, they are fixed but unknown.*”

- Don't ever say that!

Random parameters?

- For a given data set, parameter values are *fixed but unknown* just like a frequentist. i.e., there is a single data-generating value. But, they are regarded as a *realization* of a random variable. (nature generated from some distribution).
- Even qualified, “random variables” is not a diagnostic characteristic of Bayesianism: Random effects are common everywhere – they are features of models, not of inference paradigms.
- Posterior inference is the key element of Bayesian analysis. Regarding parameters as random allows Bayesian to compute a **posterior distribution**.

Parametric Statistical Inference

- **Observations:**

$$y_1, y_2, \dots, y_n$$

- **Probability model for observations**

e.g., logistic regression (for binary data):

- (1) Binomial probability mass function

$$y_i \sim \text{Bin}(y_i; 1, p_i)$$

with

$$\text{logit}(p_i) = \beta_0 + \beta_1 x_i$$

- (2) *and* y_1, \dots, y_n are mutually *independent*

- What do we do with this probability model to achieve formal inference about $\boldsymbol{\beta} = (\beta_0, \beta_1)$?

Classical Inference

Joint distribution:

$$f(y_1, y_2, \dots, y_n; \beta_0, \beta_1) = \left\{ \prod_{i=1}^n \text{Bin}(y_i; \boldsymbol{\beta}) \right\}$$

Likelihood = joint distribution regarded as a function of $\boldsymbol{\beta}$:

$$L(\beta_0, \beta_1; y_1, y_2, \dots, y_n) \equiv \left\{ \prod_{i=1}^n \text{Bin}(y_i | \boldsymbol{\beta}) \right\}$$

- Maximize it to obtain the MLE $\hat{\boldsymbol{\beta}}$
- 2nd derivative of $\log(L)$ w.r.t. $\hat{\boldsymbol{\beta}}$ is the *Fisher Information* – inverse is “asymptotic variance”
- Function of the parameters (not y)!
- It is **not** a probability distribution

Example: Ordinary Logistic Regression

Binary state variable:

$$z_i \sim \text{Bin}(\psi_i)$$

$$\text{logit}(\psi_i) = \beta_0 + \beta_1 x_i$$

R script `likeBayes.R`

```
# ----- Simulate data -----  
# 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 occupancy probability increases with vegHt  
# The relationship is described by an intercept of -3 and  
# a slope parameter of 2 on the logit scale  
# plogis is the inverse-logit (constrains us back to the [0-1] scale)  
psi <- plogis(-3 + 2*vegHt)  
  
# Now we go to 100 sites and observe presence or absence  
# Actually, let's just simulate the data  
z <- rbinom(nSites, 1, psi)
```

Strategy for classical estimation based on likelihood: Express the likelihood as an **R** function and then use the standard function `optim` or `nlm` to maximize it.

```
# This is the negative log-likelihood.
negLogLike <- function(beta, y, x) {
  beta0 <- beta[1]
  beta1 <- beta[2]
  psi <- plogis(beta0 + beta1*x) # inverse-logit
  likelihood <- psi^y * (1-psi)^(1-y) # same as:
# likelihood <- dbinom(y, 1, psi)
  return(-sum(log(likelihood)))
}

# Look at (negative) log-likelihood for 2 parameter sets
negLogLike(c(0,0), y=z, x=vegHt)
negLogLike(c(-3,2), y=z, x=vegHt) # Lower is better!

# Let's minimize it
starting.values <- c(beta0=0, beta1=0)
opt.out <- optim(starting.values, negLogLike, y=z, x=vegHt, hessian=TRUE)

mles <- opt.out$par      # MLEs are pretty close to truth
```

Bayesian Inference

- Parameters are realizations of random variables:

$$g(\beta_0, \beta_1) \equiv [\beta_0, \beta_1] \quad (\text{prior distribution})$$

- Joint distribution:

$$f(y_1, y_2, \dots, y_n, \beta_0, \beta_1) = \left\{ \prod_{i=1}^n \text{Bin}(y_i; \boldsymbol{\beta}) \right\} g(\beta_0, \beta_1)$$

- We can **condition on the data** – i.e., compute the conditional distribution:

$$\pi(\beta_0, \beta_1 | y_1, y_2, \dots, y_n) = \frac{f(y_1, y_2, \dots, y_n, \beta_0, \beta_1)}{f(y_1, y_2, \dots, y_n)}$$

(because everything is a r.v.)

The Posterior Distribution

$$\pi(\beta_0, \beta_1 | y_1, y_2, \dots, y_n) = \frac{f(y_1, y_2, \dots, y_n, \beta_0, \beta_1)}{f(y_1, y_2, \dots, y_n)}$$

- arises by use of basic rules of probability, because everything is a random variable
- it is a probability distribution **for the parameters!**
- characterize uncertainty in the parameter values using explicit probability statements
- e.g., $Pr(L \leq \beta_0 \leq U) =$ “Bayesian confidence interval”
- In general, report summaries of the posterior distribution: mean, mode, variance, etc..

How to do Bayesian Analysis: MCMC

The posterior:

$$\pi(\beta_0, \beta_1 | y_1, y_2, \dots, y_n) = \frac{f(y_1, y_2, \dots, y_n, \beta_0, \beta_1)}{f(y_1, y_2, \dots, y_n)}$$

Computing the denominator is computationally expensive, and sometimes not even possible.

MCMC: simulation methods for *sampling* from the posterior distribution which do not require that we know the denominator, or ever have to evaluate it. We estimate features of the posterior distribution from the posterior samples.

MCMC: The topic is too vast to cover here. We use a “Metropolis within Gibbs sampling” algorithm for everything, or let BUGS deal with it.

Metropolis-within-Gibbs Sampling

Iterative sampling of parameters from the *full conditional* distributions (one for **each** parameter):

$$[\beta_0 | \mathbf{y}, \beta_1] = [\mathbf{y} | \beta_0, \beta_1][\beta_0] / [\mathbf{y}]$$

$$[\beta_1 | \mathbf{y}, \beta_0] = [\mathbf{y} | \beta_0, \beta_1][\beta_1] / [\mathbf{y}]$$

In many/most problems we cannot compute $[\mathbf{y}]$ and therefore cannot identify the fc as a “named” distribution to simulate from. So rules are invented for drawing random variables from distributions that cannot be identified precisely.

The Metropolis algorithm is one such rule.

The Metropolis Algorithm

(1) For a candidate value β_0^* simulated from some symmetric proposal distribution: Symmetric if $h(x|y) = h(y|x)$ (**not** part of the model)

(2) Accept that value with probability

$$r = [\beta_0^*|\mathbf{y}, \beta_1]/[\beta_0|\mathbf{y}, \beta_1]$$

(3) Do also for β_1

Handy stuff:

- The marginal distribution of \mathbf{y} (i.e., denominator of the fc) cancels, so we don't need to know what it is.
- To use the Metropolis algorithm we only have to *evaluate* known distributions

A Basic MCMC Algorithm: Logistic regression

- (0) Pick starting values for β_0 and β_1 . At iteration 1 of the algorithm, these are the *current* values of the parameters
- (1) Generate a *candidate* value of β_0 from a symmetric proposal distribution. e.g., centered on the current value:

$$\beta_0^* \sim \text{Normal}(\beta_0, \delta^2)$$

(The proposal is not part of the model!)

- (2) Accept the candidate value with probability:

$$r = [\beta_0^* | \mathbf{y}, \beta_1] / [\beta_0 | \mathbf{y}, \beta_1]$$

then β_0^* becomes the current value.

- (3) Repeat for each parameter in the model.

Remarks on Metropolis-within-Gibbs Algorithm

- Heuristic: This algorithm has us simulate candidate values somehow and then accept values that have higher posterior probability (conditional on other parameters).
- The long-run frequency of “accepted” values is that of the target posterior density!
- Note: If the prior is constant, this MCMC calculation is based on repeated evaluations of the likelihood only. So, if you write a function to do MLE you can also do MCMC.
- We have a function to evaluate the likelihood for a given value of the parameters. Using MCMC we have to do this over-and-over again....

MCMC (MwiG) for Logistic Regression

1. What prior should we use?

$$\beta_0 \sim \text{Norm}(0, 10)$$

$$\beta_1 \sim \text{Norm}(0, 10)$$

2. The full conditionals look like:

$$[\beta_0 | \mathbf{y}, \beta_1] \propto \left\{ \prod_i \text{Bin}(y_i | \beta_0, \beta_1) \right\} g(\beta_0)$$

3. In **R**, this looks like:

```
beta0<- some value
```

```
beta1<- some value
```

```
fc.beta0<- exp(-1*negLogLike(c(beta0,beta1),y,x))*dnorm(beta0,0,10)
```

```
fc.beta1<- exp(-1*negLogLike(c(beta0,beta1),y,x))*dnorm(beta1,0,10)
```

Implementation for simulated data:

```
niter <- 50000
out <- matrix(NA, niter, 2) # create a matrix to save the MCMC output
colnames(out) <- c("beta0", "beta1")

# Initialize the parameters, likelihood, and priors
beta0 <- starting.values[1]
beta1 <- starting.values[2]
loglike <- -1*negLogLike(c(beta0,beta1), z, vegHt)
logprior <- dnorm(c(beta0,beta1), 0, 10, log=TRUE)

for(i in 1:niter) {
  # propose candidate values of beta
  beta0.cand <- rnorm(1, beta0, 0.3) # 0.3 is tuning parameter

  # evaluate likelihood and priors for candidates
  loglike.cand <- -1*negLogLike(c(beta0.cand,beta1), z, vegHt)
  logprior.cand <- dnorm(beta0.cand, 0, 10, log=TRUE)

  # Compute Metropolis acceptance probability
  Metrop.acceptance.prob <- exp((loglike.cand+logprior.cand) -
    (loglike + logprior[1]))

  # Keep the candidates if they meet the criterion
  if(runif(1) < Metrop.acceptance.prob) {
    beta0 <- beta0.cand
    loglike <- loglike.cand
    logprior[1] <- logprior.cand
  }
  ##### Repeat for beta1
  .
  .
  .
} # closes main MCMC loop
```

Inference for Hierarchical Models

How are things different compared to “non-hierarchical” model?

Two Canonical Examples of HMs in Ecology

HMs have 1 or more “intermediate” models/levels/stages involving a latent variable (random effect).

- Modeling species occurrence – “occupancy models”
- Modeling species abundance – “N-mixture models” (and related)

Example: Modeling occurrence of a species

- $y_i = \{0, 1\}$ observations of presence/absence at site i
- $z_i = \{0, 1\}$ state-variable true presence or absence

Observation model:

$$y_i | z_i \sim \text{Bernoulli}(z_i p)$$

p = probability of detecting species *given that it is present*

Process model:

$$z_i \sim \text{Bernoulli}(\psi_i)$$

$$\text{logit}(\psi_i) = \beta_0 + \beta_1 x_i$$

Example: Modeling Abundance from Point Counts

- y_i = count of birds at point i
- N_i = state-variable population size at point i

Observation model:

$$y_i | N_i \sim \text{Binomial}(N_i, p)$$

N_i = local population size

p = probability of encountering an *individual*

Process model:

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

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

Classical Analysis of Hierarchical Models

- What is “the likelihood”?
- Easier question: What is the joint distribution (of all the random stuff)?
- Bayesian version:

$$f(y_1, y_2, \dots, y_n, z_1, \dots, z_n, p, \boldsymbol{\beta}) = \left\{ \prod_{i=1}^n \prod_{j=1}^J [y_{ij} | z_i, p][z_i | \boldsymbol{\beta}] \right\} [\boldsymbol{\beta}]$$

Bayesian Analysis

- Nothing is special about this model. Since everything is a random variable we can compute the joint distribution:

$$f(y_1, y_2, \dots, y_n, z_1, \dots, z_n, p, \boldsymbol{\beta}) = \left\{ \prod_{i=1}^n \prod_{j=1}^J [y_{ij} | z_i, p] [z_i | \boldsymbol{\beta}] \right\} [\boldsymbol{\beta}]$$

- Posterior characterized by MCMC – Metropolis/Gibbs:

parameters:

$$\begin{aligned} & [\boldsymbol{\beta} | \mathbf{y}, \mathbf{z}] \\ & [p | \mathbf{y}, \boldsymbol{\beta}] \end{aligned}$$

latent variables:

$$[z_i | \mathbf{y}, \mathbf{z}_{-i}, p, \boldsymbol{\beta}] = [z_i | \mathbf{y}, p, \boldsymbol{\beta}]$$

The posterior is fully characterized by this set of conditional distributions

Classical Analysis of Random Effects

INTEGRATED (marginal) LIKELIHOOD:

- Remove random effects from the conditional likelihood by integration
- The distribution of \mathbf{y}_i *unconditional* on the random effect:

$$f(\mathbf{y}_i|\boldsymbol{\beta}, p) = \int \left\{ \prod_{j=1}^J f(y_{ij}|z_i, p) \right\} g(z_i|\boldsymbol{\beta}) dz_i$$

- Not a function of z_i anymore
- Maximize to obtain MLEs of $p, \boldsymbol{\beta}$
- For discrete latent variable, replace f by Σ

Example: Occupancy Model

Observation model:

$$\begin{aligned} y_{ij} &\sim \text{Bin}(J, p) && \text{if } z_i = 1 \\ y_{ij} &= 0 && \text{if } z_i = 0 \end{aligned}$$

State model:

$$\begin{aligned} z_i &\sim \text{Bin}(\psi_i) \\ \text{logit}(\psi_i) &= \beta_0 + \beta_1 x_i \end{aligned}$$

What is the marginal likelihood for y ?

Computing the Marginal Likelihood

- z is a discrete random variable
- Law of total probability

$$Pr(y) = Pr(y|z = 1)Pr(z = 1) + Pr(y|z = 0)Pr(z = 0)$$

- Marginal likelihood for total detections, y_i :

$$[y_i|p, \psi] = Bin(y_i|p)\psi + 1(y_i = 0)(1 - \psi)$$

- Zero-inflated binomial. Can be maximized easily to obtain MLEs.
- PRESENCE or **unmarked** function `occu`

Doing it in R

```
nSites <- 100
vegHt <- runif(nSites, 1, 3) # uniform from 1 to 3
psi <- plogis(-3 + 2*vegHt)

# Now we simulated true presence/absence for 100 sites
z <- rbinom(nSites, 1, psi)

## Now generate observations
p<- 0.6
J<- 3 # sample each site 3 times
y<-rbinom(nSites,J,p*z)

# This is the negative log-likelihood.
negLogLikeocc <- function(beta, y, x,J) {
  beta0 <- beta[1]
  beta1 <- beta[2]
  p<- plogis(beta[3])

  psi <- plogis(beta0 + beta1*x)

  marg.likelihood <- dbinom(y, J,p)*psi + ifelse(y==0,1,0)*(1-psi)
  return(-sum(log(marg.likelihood)))
}

starting.values <- c(beta0=0, beta1=0,logitp=0)
opt.out <- optim(starting.values, negLogLikeocc, y=y, x=vegHt,J=J,
                hessian=TRUE)
```

Continuous Case: Numerical integration

- We use **R** function `integrate()` which takes the function to be integrated as an argument.
- Example: Binomial/logit-normal mixture

$$y_i \sim \text{Bin}(J, p_i)$$
$$\text{logit}(p_i) \sim \text{Normal}(\mu, \sigma)$$

- Snowshoe hare data:

```
# FREQUENCIES captured 0, J=14 times:
nx<-c(14,34, 16, 10, 4, 2, 2,0,0,0,0,0,0,0)
nind<-sum(nx)
J<-14

Mhlik<-function(parms){
mu<-parms[1]
sigma<-exp(parms[2])

il<-rep(NA,J+1)
for(k in 0:J){
il[k+1]<-integrate(
function(x){ dbinom(k,J,plogis(x))*dnorm(x,mu,sigma)
},lower=-Inf,upper=Inf)$value
}
-1*( sum(nx*log(il)) )
}
tmp<-nlm(Mhlik,c(-1,-1 ),hessian=TRUE)
sqrt(diag(solve(tmp$hessian)))
```

Prediction

- **Estimation** of fixed-effects or parameters
- **Prediction** is estimation of a random variable. (also estimation of a response surface)
 - “Best Unbiased Prediction” (BUP) use Bayes rule to compute $E[z|\mathbf{data}]$
 - Response surface: $E[z]$

Prediction in Occupancy Models

Occupancy model

$$y_i | z_i \sim \text{Bin}(p * z_i)$$

$$z_i \sim \text{Bern}(\psi_i)$$

Types of prediction problems:

- Predict z_i where $y_i = 0$ (observed)

use $E[z | y_i = 0]$ (use Bayes rule)

Or simulate new z in the MCMC algorithm

- Predict z_i for unsampled sites. There is no y to condition on.

use $\hat{\psi}_i$

Summary Thoughts on Bayesian vs. Classical Inference

Both inference paradigms useful for analysis of hierarchical models

- Bayesian:
 - Completely general methods for implementation (MCMC) which always work. Sometimes BUGS implementations don't work, so its good to know how to do it.
 - Takes more math/programming know-how????
 - Sometimes slower due to more calculations
 - Inferences are *not* asymptotic, apply to arbitrary n
 - Prediction is more coherent – comes “for free”
- Classical:
 - Integrated likelihood sometimes not feasible. (community model)
 - But very accurate (no MC error)
 - Automatic model selection (AIC)

“Bayesian” Hierarchical Model??

- Hierarchical modeling is a conceptual and technical framework for formulating models
- The method of inference is independent of model formulation
- HMs can be analyzed by Bayesian and non-Bayesian methods.
- $\text{HM} \neq \text{Bayesian!!!}$ A model is not Bayesian or frequentist – what you do to that model is Bayesian or frequentist.
- Models don't have political views (people do)

Idealized Structure of Workshop/Book

- Introduction to a class of models
- likelihood analysis of models in unmarked
- stressing consistent workflow and ease of doing standard things like prediction and model selection
- Bayesian analysis in BUGS
- Illustration of a type of model that can't be done (easily, or in **unmarked**) using likelihood methods.