

Introduction to Occupancy Models

Statistical Toolbox for Analysing
Citizen Science Data

Swantje Löbel

GFÖ Conference 2021, Pre-Meeting Workshop
29.08.2021

Introduction to Occupancy Models

2

The **occurrence** of a species at a given location in your data is the result of two processes:

1. The species is actually present



&

2. You detect the species in your survey



The **absence** of a species at a given location in your data is either the result of:

1. The species is actually absent



|

2. You did not detect the species in your survey



Introduction to Occupancy Models

3

The **occurrence** of a species at a given location in your data is the result of two processes:

1. The species is actually present

2. You detect the species in your survey



&



In occupancy models, we quantify this uncertainty in detection!

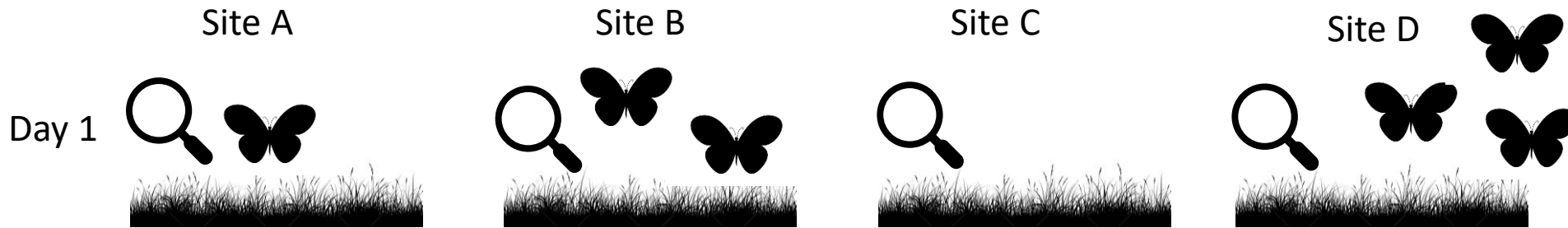


|



Study design to estimate detection probability

4

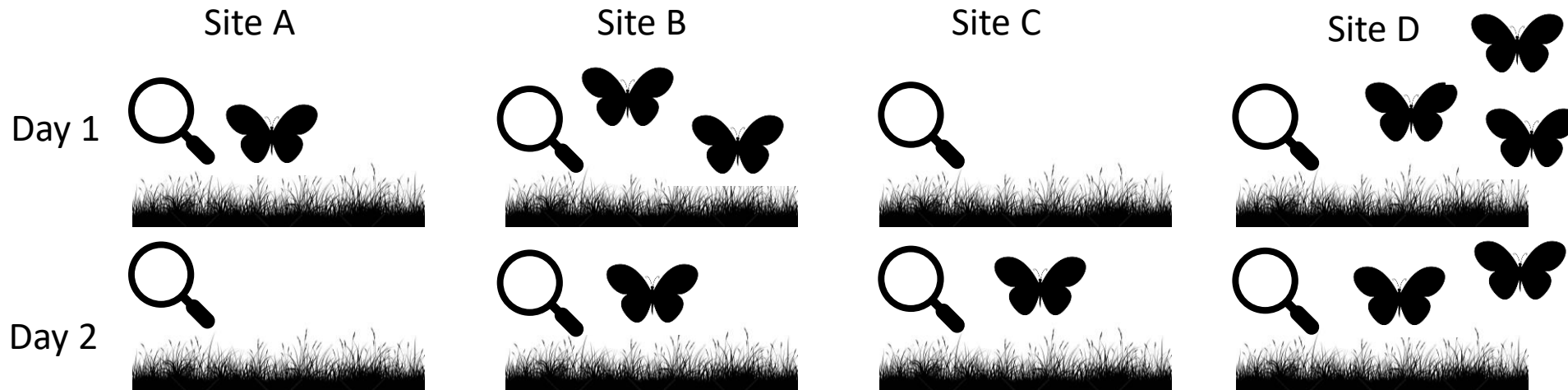


- We visit 4 sites – each site is visited once
- At each site, we count the number of butterflies we see
- Can we separate detection rate from true occurrence rate? **NO !**



Study design to estimate detection probability

5

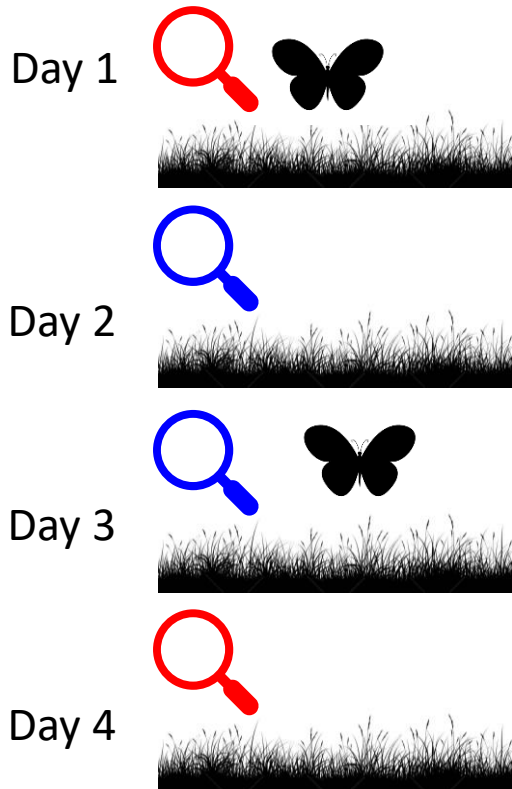


- We visit 4 sites – each site is visited twice on different days
- At each site and on each day, we count the number of butterflies we see
- Can we separate detection rate from true occurrence rate? **YES !**

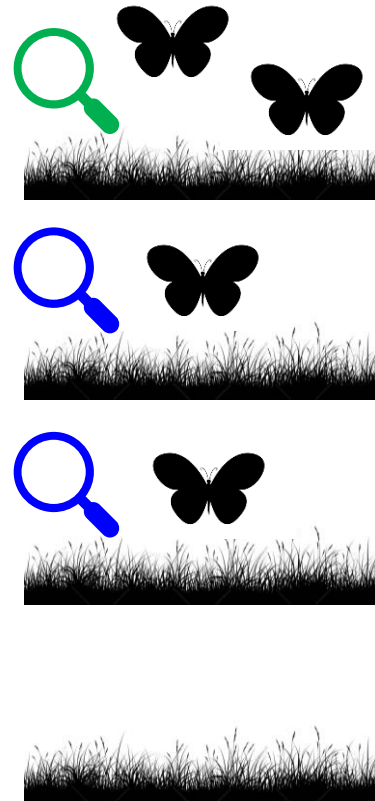
Study design to estimate detection probability

6

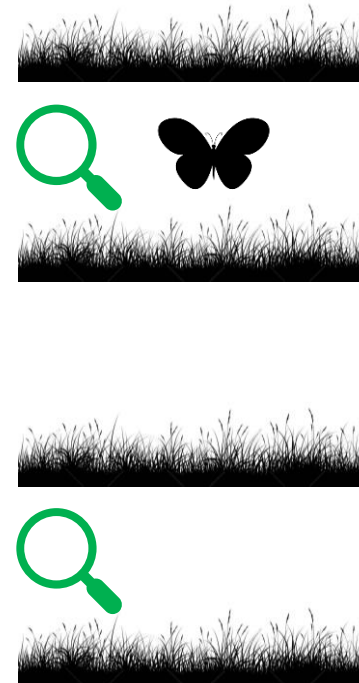
Site A



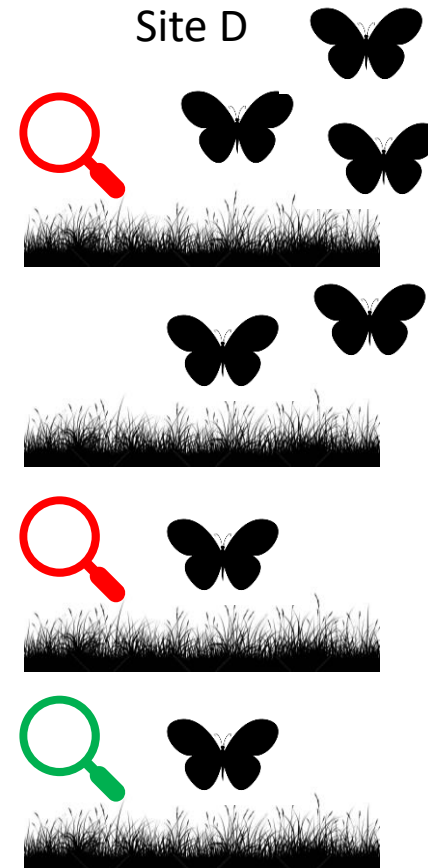
Site B



Site C



Site D



	A	B	C	D
D 1	1	2	NA	3
D 2	0	1	1	2
D 3	1	1	NA	1
D 4	0	NA	0	1

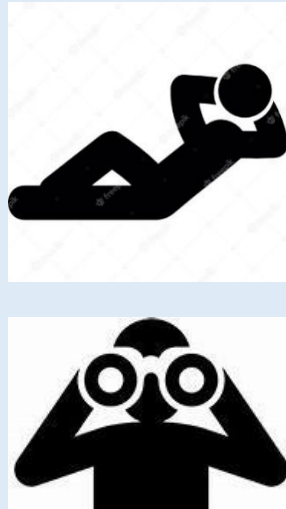
Typical design while working with Citizen Science Data

Assumptions:

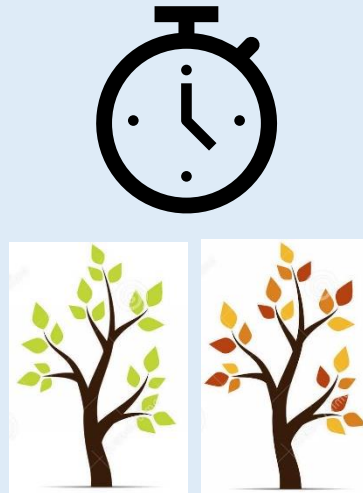
- Detection probability is constant or explained by covariates.



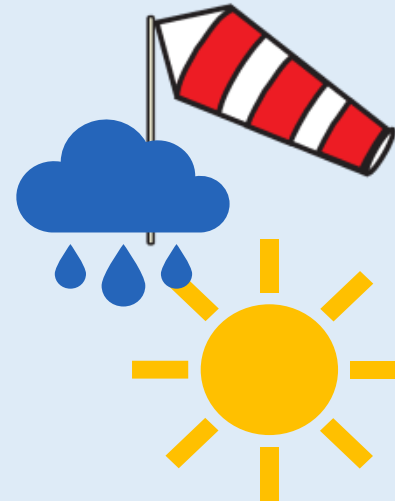
Observer



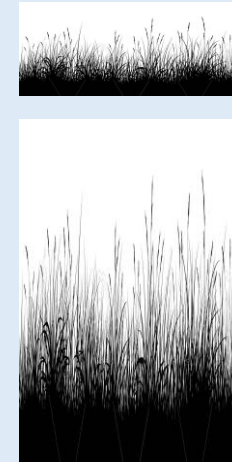
Sampling effort



Timing

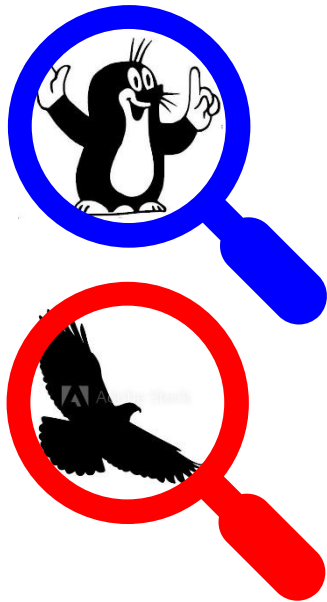


Climate/
Weather



Habitat

Jonston et al., 2018.
Methods in Ecology and
Evolution, 9, 88–97.



Received: 31 January 2017 | Accepted: 27 April 2017

DOI: 10.1111/2041-210X.12838

RESEARCH ARTICLE

Methods in Ecology and Evolution



Estimates of observer expertise improve species distributions from citizen science data

Alison Johnston^{1,2,3}  | Daniel Fink¹ | Wesley M. Hochachka¹ | Steve Kelling¹

¹Cornell Lab of Ornithology, Ithaca, NY, USA

²British Trust for Ornithology, The Nunnery, Thetford, Norfolk, UK

³Department of Zoology, Conservation Science Group, University of Cambridge, Cambridge, UK

Correspondence

Alison Johnston

Email: aj327@cornell.edu

Funding information

Leon Levy Foundation; Wolf Creek Foundation; Seaver Institute; National Aeronautics and Space Administration, Grant/Award Number: NNN12ZDA001N-ECOF; National Science Foundation, Grant/Award Number: DBI-1356308, IIS-120589 and CNS-1059284

Handling Editor: Nick Isaac

Abstract

1. Citizen science data are increasingly making valuable contributions to ecological studies. However, many citizen science surveys are also designed to encourage wide participation and therefore the participants have a range of natural history expertise, leading to variation and potentially bias in the data.
2. We assessed a recently proposed measure of observer expertise, calculated based on the average numbers of species recorded by observers. We investigated if this observer expertise score is associated with how often an observer reports any individual species. Species reporting rates increased monotonically with the observer's expertise score for 197 of 200 species, suggesting that this expertise score describes inter-observer variation in the detectability of individual species.
3. Expertise scores were incorporated into single-species occupancy models as a covariate, to explain inter-observer variation in detectability. Including expertise as a detectability covariate led to improved model fit and improved predictive performance

Assumptions:

- Detection probability is constant or explained by covariates.
- **Closure:** We require that the (true) presence/absence state of Site i does not change over the course of the study (or one season) !
- No false positive errors
- Independence of occurrence and independence of detection
- No residual spatial autocorrelation

- Process or state model

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

Ecological process yields true state (Z)

Latent variable – true species presence or absence

→ simple Bernoulli model with an occurrence probability (ψ).

- Observation or detection model

$$y_{ij} \mid Z_i \sim \text{Bernoulli}(Z_i * p_{ij})$$

Observation model links the observations (y = our data) to the state variable (Z).

→ standard observation model is a Bernoulli model. p is the estimated detection probability.

- Process or state model

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

We can add linear predictors to both the occurrence probability (ψ , psi) and detection probability (p)

$$\text{logit}(\psi_i) = \beta_0 + \beta_1 * \text{occupancy.covar}_i + \dots$$

→ simple Bernoulli model with an occurrence probability (ψ).

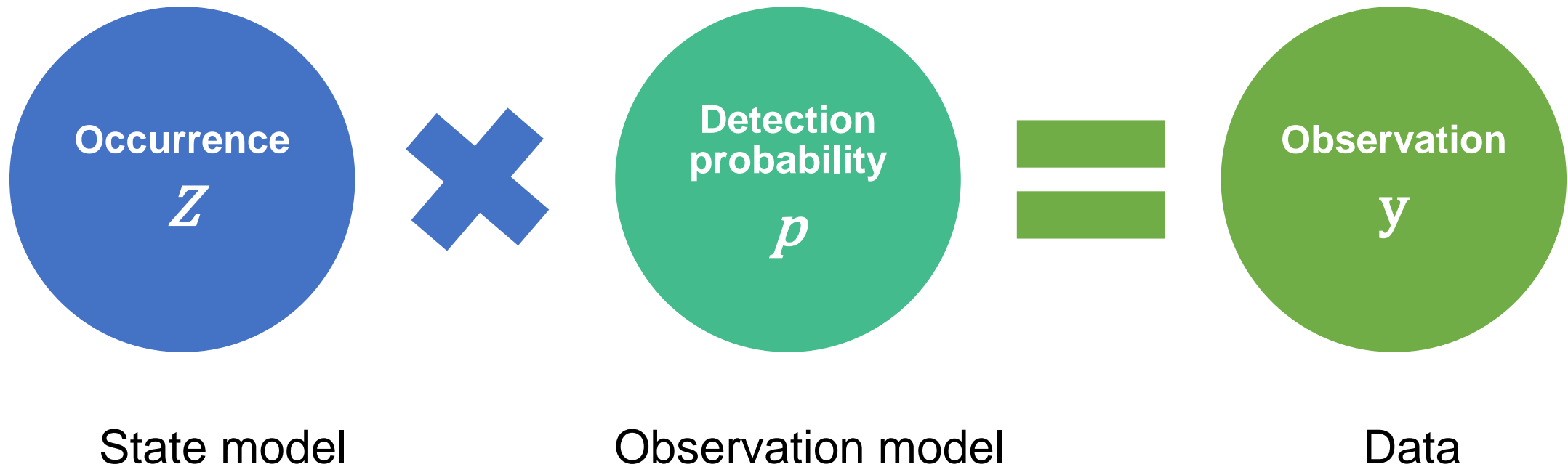
- Observation or detection model

$$y_{ij} | Z_i \sim \text{Bernoulli}(Z_i * p_{ij})$$

$$\text{logit}(p_{ij}) = \alpha_0 + \alpha_1 * \text{detection.covar}_{ij} + \dots$$

→ standard observation model is a Bernoulli model. p is the estimated detection probability.

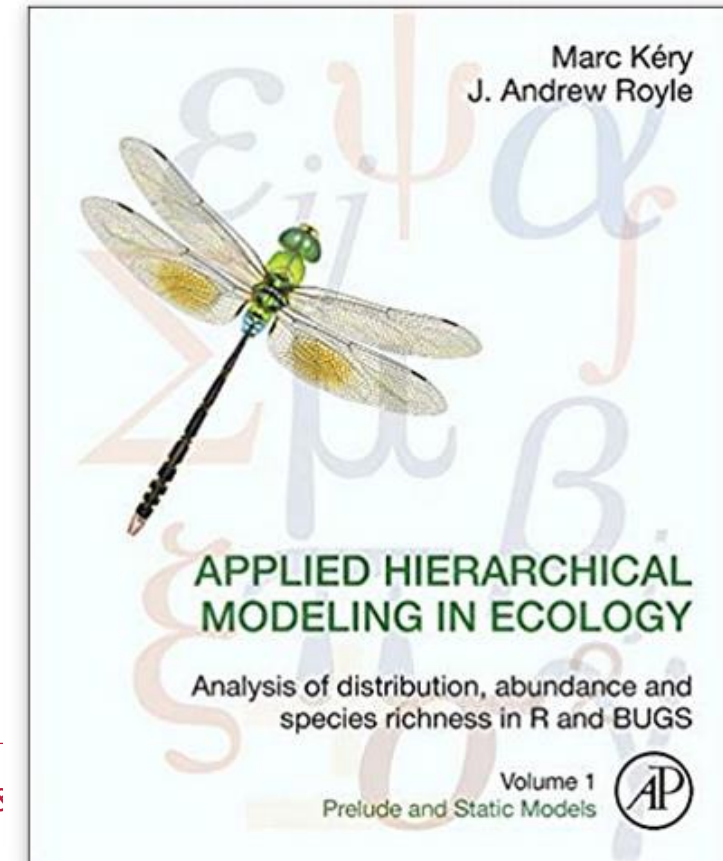
You can include the same or separate covariates in the detection and occupancy components of the model !



Authors: Ian Fiske, Richard Chandler, Andy Royle, Marc Kéry, David Miller, and Rebecca Hutchinson

“Fits hierarchical models of animal abundance and occurrence to data collected using survey methods such as point counts, site occupancy sampling, distance sampling, removal sampling, and double observer sampling. Parameters governing the state and observation processes can be modeled as functions of covariates.”

- easy to use for experienced R-Users
- Bayesian knowledge is not required (frequentists statistical package !)
- less flexible than coding in traditional BUGS-Language
- often less efficient while working with sparse detection-nondetection matrices



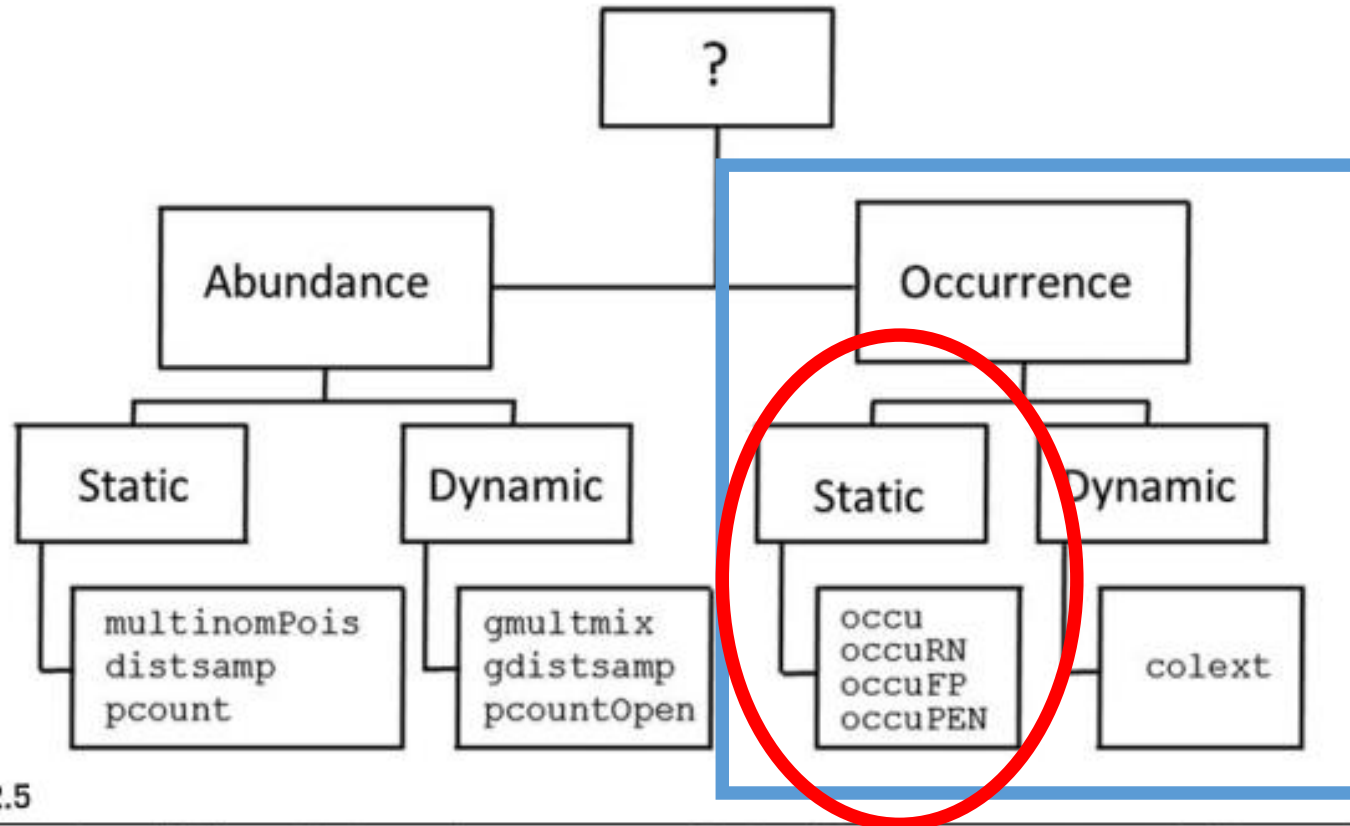


FIGURE 2.5

Decision tree for available models in R package *unmarked*. A scientific or management question (the question mark in the top square) first suggests a focus on either abundance or occurrence; second, the system is static or dynamic; and third, you use a particular sampling method; all of these determine the appropriate model for your study. (Figure courtesy of R. Chandler.)

From: Kéry & Royle (2016)

- (1) Processing and packaging the data into an “unmarked frame” using standard constructor functions that ensure data are in the proper format (`unmarkedFrameOccu()`).

```
library(unmarked)
umf <- unmarkedFrameOccu(
  y = y,                                     # Pres/Abs measurements
  siteCovs = data.frame(vegHt = vegHt, hab = hab), # Site covariates
  obsCovs = list(wind = wind, obsID = observerID)) # Observation covariates
summary(umf)
```

Table 2.1 Typical data structure for the classes of HMs implemented in the `unmarked` package.

	Detection Data			Site Covariate	Observation Covariate		
	Visit1	Visit2	Visit3	Habitat	Date1	Date2	Date3
Site 1	1	1	1	Good	3	6	10
Site 2	0	0	0	Good	1	7	11
Site 3	1	0	0	Bad	2	9	12
Site 4	0	0	1	Bad	5	6	10



- (1) Processing and packaging the data into an “unmarked frame” using standard constructor functions that ensure data are in the proper format (`unmarkedFrameOccu()`).

```
library(unmarked)
umf <- unmarkedFrameOccu(
  y = y,                                     # Pres/Abs measurements
  siteCovs = data.frame(vegHt = vegHt, hab = hab), # Site covariates
  obsCovs = list(wind = wind, obsID = observerID)) # Observation covariates
```

- (2) Utilization of a standard model fitting function that produces parameter estimates, standard errors, AIC, and other summary statistics.

```
library(unmarked)
fm1 <- occu( ~ Detection_Covariates ~ Occupancy_Covariates, data = umf)
```

Detection model Occupancy model

- (3) Summary analyses that include producing model selection tables, goodness-of fit analyses (e.g., using parametric bootstrapping), and plotting predictions or fitted values.

```
fm  <- occu( ~ Detection_Covariates ~ Occupancy_Covariates, data = umf)

fm0 <- occu(~ 1 ~ 1, data=umf)           # intercepts only
fm1 <- occu(~ wind ~ 1, data=umf)        # covariates on detection
fm2 <- occu(~ 1 ~ habitat, data=umf)     # covariates on occupancy
fm3 <- occu(~ wind + observer ~ habitat, data=umf) # covariates on both
```

```
summary(fm3)
```

```
call:
occu(formula = ~wind ~ habitat, data = umf)

Occupancy (logit-scale):
      Estimate      SE      z  P(>|z|)
(Intercept) -0.232 0.303 -0.765 4.44e-01
habitat      2.598 0.648  4.012 6.03e-05

Detection (logit-scale):
      Estimate      SE      z  P(>|z|)
(Intercept) -1.04 0.230 -4.53 5.83e-06
wind        -2.51 0.378 -6.65 3.02e-11
```

Swan

e


```
fm  <- occu( ~ Detection_Covariates ~ Occupancy_Covariates, data = umf)

fm0 <- occu(~ 1 ~ 1, data=umf)           # intercepts only
fm1 <- occu(~ wind ~ 1, data=umf)        # covariates on detection
fm2 <- occu(~ 1 ~ habitat, data=umf)     # covariates on occupancy
fm3 <- occu(~ wind + observer ~ habitat, data=umf) # covariates on both

summary(fm3)
fm3@AIC
LRT(fm2, fm3)

pred.occ <- predict(fm3, type="state", newdata=newdat)
pred.det <- predict(fm3, type="det", newdata=newdat)
```



ranef(fm3)

State process $Z_i \sim \text{Bernoulli}(\psi_i)$

	Mean	Mode	2.5%	97.5%
[1,]	0.015231766	0	0	0
[2,]	0.017820736	0	0	0
[3,]	0.058991790	0	0	1
[4,]	1.000000000	1	1	1
[5,]	0.011251114	0	0	0
[6,]	0.072711276	0	0	1
[7,]	0.015363369	0	0	0
[8,]	0.058627194	0	0	1
[9,]	1.000000000	1	1	1

- These random effects are the presence/absence state at each site (Z), and their estimates represent our best guess of whether a particular site is occupied or not.
- These predictions of the random effects Z are also called **conditional** occupancy probability, where conditional means “given the observed data at that site”.

ranef(fm3)

State process $Z_i \sim \text{Bernoulli}(\psi_i)$

➤ If species was detected at a site, then Z is 1 with no uncertainty!!

	Mean	Mode	2.5%	97.5%
[1,]	0.015231766	0	0	0
[2,]	0.017820736	0	0	0
[3,]	0.058991790	0	0	1
[4,]	1.000000000	1	1	1
[5,]	0.011251114	0	0	0
[6,]	0.072711276	0	0	1
[7,]	0.015363369	0	0	0
[8,]	0.058627194	0	0	1
[9,]	1.000000000	1	1	1



ranef(fm3)

State process $Z_i \sim \text{Bernoulli}(\psi_i)$

	Mean	Mode	2.5%	97.5%
[1,]	0.015231766	0	0	0
[2,]	0.01782	y[1,]	[1]	0 0 0
[3,]	0.058991750	0	0	1
[4,]	1.000000000	1	1	1
[5,]	0.011251114	0	0	0
[6,]	0.072711276	0	0	1
[7,]	0.015363369	0	0	0
[8,]	0.058627194	0	0	1
[9,]	1.000000000	1	1	1

➤ Predicting Z at a site when species never detected

$$\Pr(z_i = 1 | \{y_i\} = 0) = \frac{\psi(1-p)^J}{(1-\psi) + \psi(1-p)^J}$$

➤ Model will predict some probability that the species was there

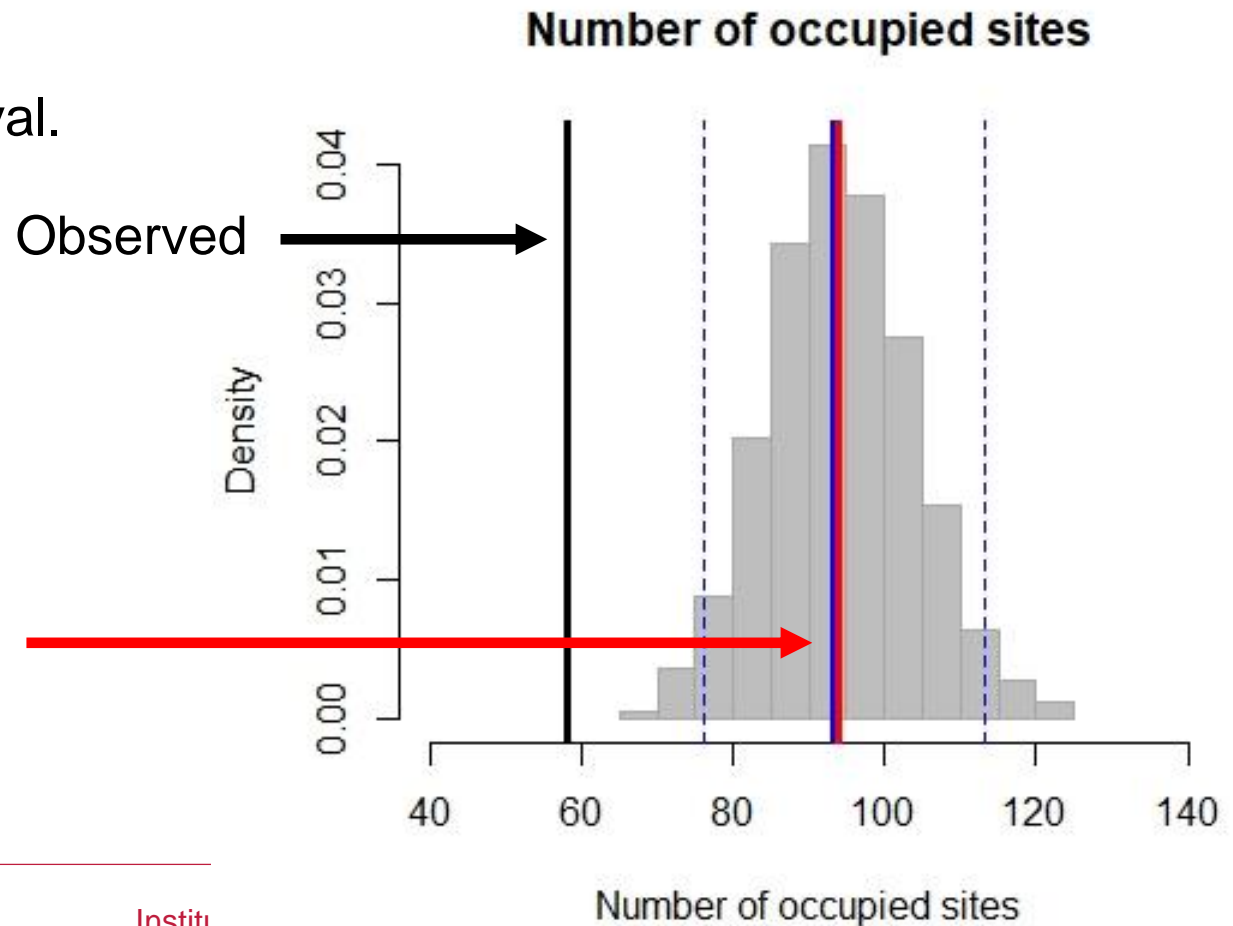
➤ We have higher confidence in its presence ($Z=1$) despite the negative survey results ($y=0,0,0$) when

1. it is widespread overall (occurrence probability ψ is high),
2. it is elusive (detection probability p is small),
3. the number of times we have looked for it (J) is small.

- In **unmarked**, we can obtain the number of occupied sites by summing over the estimates of the random effects Z .
- Parametric bootstrapping can be used to obtain a confidence interval.

Species was only seen at 58 sites
but was estimated to be present
at ~ 93 sites by the model

True number of occupied sites = 94



- For complicated functions, we often compute standard errors and confidence intervals using the method of bootstrapping.
- Parametric bootstrapping - we simulate data sets under the assumed model with parameters equal to the MLEs obtained from our observed data.
- For each simulated data set, we refit the model and obtain the (new) MLEs for the simulated data set. Each set of MLEs thus obtained is called a bootstrap sample, and together they form the bootstrap distribution.
- We can summarize to obtain quantities of interest. For example, the variance of the bootstrap samples is an estimate of the sampling variance. The 0.025 and 0.975 percentiles of the bootstrap distribution form a 95% confidence interval.



What is JAGS?

- JAGS is ‘Just Another Gibbs Sampler’: a software package for performing Bayesian inference Using Gibbs Sampling
- Using a dialect of the BUGS language (WinBUGS, OpenBUGS, STAN)
- It estimates parameter posterior distributions for Bayesian statistical models, of (almost) arbitrary complexity, defined by the relationships between variables.
- JAGS uses MCMC simulation to converge on the posterior distribution of our parameters.

We will access the **JAGS** functionalities within **R**. The package **rjags** provides an interface that allows us to handle pre-processing of data, the model estimation and the post-processing of results all together in one R session.

10 basic steps of a Bayesian model analysis with JAGS

- (1) Get an idea of your data and your research question
- (2) Formulate a model and code it in the BUGS syntax



- A BUGS model basically describes relationships between some variables in terms of conditional probability distributions.
- BUGS uses a syntax looking quite similar to R. But be aware: the model description is nothing like a run-able script but just describes the relations between variables in the model!
- Most important elements of the syntax include:
 - $x \leftarrow \dots$ **'logical node'** describes x as a deterministic function of some other variables
 - $x \sim \dots$ **'stochastic node'** describes a probability distribution of x conditional on some other variables
- Variety of in-build probability distributions: $\text{dnorm}(\mu, \tau)$, $\text{dbin}(p, n)$, $\text{dbern}(p)$, ...
- The BUGS language is not really vectorized - we need loops to define the model for every element in vectors or multidimensional arrays.

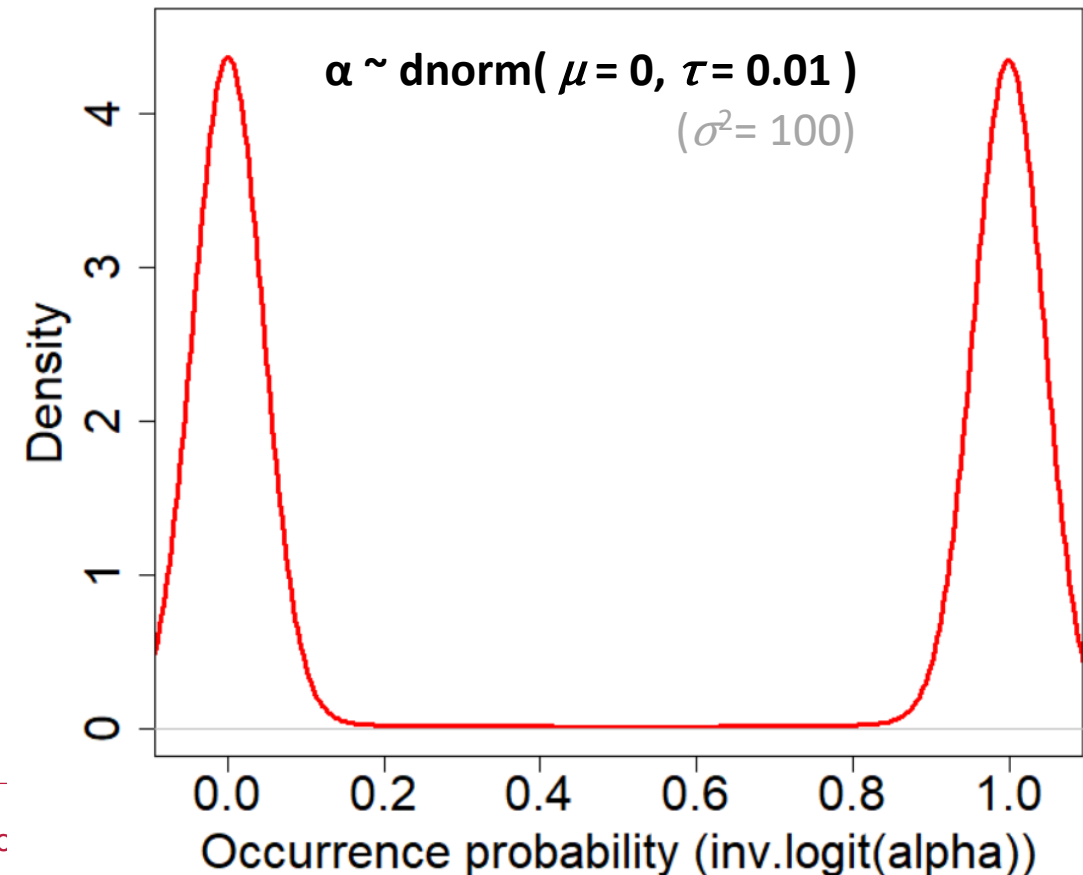


10 basic steps of a Bayesian model analysis with JAGS

- (1) Get an idea of your data and your research question
- (2) Formulate a model and code it in the BUGS syntax
- (3) Think about the necessary prior distributions

Most ecologists use vague or uninformative priors, but be careful... priors might not always be as uninformative as you think !

Probability distributions in BUGS are defined by μ and precision $\tau (= 1/\sigma^2)$!



10 basic steps of a Bayesian model analysis with JAGS

- (1) Get an idea of your data and your research question
- (2) Formulate a model and code it in the BUGS syntax
- (3) Think about the necessary prior distributions

Most ecologists use vague or uninformative priors, but be careful... priors might not always be as uninformative as you think !

≈ flat priors at probability scale
(e.g., for intercept parameters in logistic models)

$\alpha \sim \text{dlogis}(0, 1)$ Northrup & Gerber (2018)

$\text{mean.p} \sim \text{dunif}(0, 1)$ Kéry & Royle (2016)

$\alpha \leftarrow \text{logit}(\text{mean.p})$

$\alpha \sim \text{dnorm}(0, 0.5)$ Hobbs & Hooten (2015)

Choice of priors for slope parameters is often less problematic, typical priors are

$\beta \sim \text{dnorm}(0, 0.1) \rightarrow \sigma^2 = 10$

$\beta \sim \text{dnorm}(0, 0.01) \rightarrow \sigma^2 = 100$

$\beta \sim \text{dunif}(-10, 10)$

$\beta \sim \text{dunif}(-20, 20)$

The less information your data contains, the more impact has the choice of prior!

Recommendation: Perform a sensitivity analysis with different priors. If your posterior samples differ much – you should think further about your choice of prior.

Northrup & Gerber
2021. PLoS ONE, 13,
e0192819.



RESEARCH ARTICLE

A comment on priors for Bayesian occupancy models

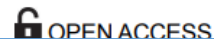
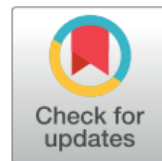
Joseph M. Northrup^{1,2*}, Brian D. Gerber³

1 Forest Biodiversity Research Network, Department of Forest Ecosystems and Society, Oregon State University, Corvallis, Oregon, United States of America, **2** Wildlife Research and Monitoring Section, Ontario Ministry of Natural Resources and Forestry, Peterborough, Ontario, Canada, **3** Department of Natural Resources Science, University of Rhode Island, Kingston, Rhode Island, United States of America

* joe.northrup@gmail.com

Abstract

Understanding patterns of species occurrence and the processes underlying these patterns is fundamental to the study of ecology. One of the more commonly used approaches to investigate species occurrence patterns is occupancy modeling, which can account for imperfect detection of a species during surveys. In recent years, there has been a proliferation of Bayesian modeling in ecology, which includes fitting Bayesian occupancy models. The Bayesian framework is appealing to ecologists for many reasons, including the ability to incorporate prior information through the specification of prior distributions on parameters. While ecologists almost exclusively intend to choose priors so that they are “uninformative” or “vague” such priors can easily be unintentionally highly informative. Here we report on



Model specification in the BUGS syntax

30

```
cat(file = "StaticOccupancyModel.txt", "
model{

  ## Specify priors
  beta0.occ ~ dlogis(0,1)          ## prior on occupancy intercept
  beta1.occ ~ dnorm(0,0.01)        ## prior on slope occupancy covariate
  alpha0.det ~ dlogis(0,1)         ## prior on detection intercept
  alpha1.det ~ dnorm(0,0.01)       ## prior on slope detection covariate

  for (i in 1:nSites) {           ## Loop over Sites
    #--- True state model for the partially observed true state ---#
    z[i] ~ dbern(psi[i])           ## True occupancy z at site i
    logit(psi[i]) <- beta0.occ + beta1.occ * occ.cov[i]

    for (j in 1:nVisits) {        ## Loop over Visits
      #--- Observation model for the actual observations ---#
      y[i,j] ~ dbern(muy[i,j])     ## Detection-nondetection at i and j
      muy[i,j] <- z[i] * p[i,j]
      logit(p[i,j]) <- alpha0.det + alpha1.det * det.cov[i,j]

    } #j
  } #i
}
```

10 basic steps of a Bayesian model analysis with JAGS

- (1) Get an idea of your data and your research question
- (2) Formulate a model and code it in the BUGS syntax
- (3) Think about the necessary prior distributions
- (4) Assemble your data
 - format your data as a named list

```
BugsData <- list(y=y, nSites=dim(y)[1], nVisits=dim(y)[2],  
                occ.cov=VegHeight, det.cov=wind)
```



10 basic steps of a Bayesian model analysis with JAGS

- (1) Get an idea of your data and your research question
- (2) Formulate a model and code it in the BUGS syntax
- (3) Think about the necessary prior distributions
- (4) Assemble your data
- (5) Generate (overdispersed) initial parameter values

```
## Inits function
```

```
zst <- apply(y, 1, max)    ## Starting values for latent states  
inits.fn <- function() {list(z = zst, beta0.occ=runif(1, -5, 5),  
                             alpha0.det = runif(1, -5, 5)), ...}
```



10 basic steps of a Bayesian model analysis with JAGS

- (1) Get an idea of your data and your research question
- (2) Formulate a model and code it in the BUGS syntax
- (3) Think about the necessary prior distributions
- (4) Assemble your data
- (5) Generate (overdispersed) initial parameter values
- (6) Set the MCMC parameters (n.chains, n.iter, thin, ...)

```
n.chains = 3          # Number of chains (typically 3)
n.iter = 5000         # Total number of iterations (start with few)
n.burnin = 2500       # Burnin-length (Samples before will be discarded)
n.thin = 1            # e.g., thin = 1 → All samples are kept, thin = 10
                      # → every 10th sample will be kept
parameters.to.save = c(beta0.occ, alpha0.det, ... )
                      # which parameters should be saved?
```



10 basic steps of a Bayesian model analysis with JAGS

- (1) Get an idea of your data and your research question
- (2) Formulate a model and code it in the BUGS syntax
- (3) Think about the necessary prior distributions
- (4) Assemble your data
- (5) Generate (overdispersed) initial parameter values
- (6) Set the MCMC parameters (n.chains, n.iter, thin, ...)
- (7) Call and run JAGS

```
jagsModel <- jags(model.file="StaticOccupancyModel.txt",  
                  data=BugsData, inits = inits.fn, n.chains = 3,  
                  n.iter=5000, n.burnin=2500,n.thin=1,  
                  parameters.to.save=para.names, DIC=T)
```



10 basic steps of a Bayesian model analysis with JAGS

- (1) Get an idea of your data and your research question
- (2) Formulate a model and code it in the BUGS syntax
- (3) Think about the necessary prior distributions
- (4) Assemble your data
- (5) Generate (overdispersed) initial parameter values
- (6) Set the MCMC parameters (n.chains, n.iter, thin, ...)
- (7) Call and run JAGS
- (8) Check your obtained posterior samples for convergence (traceplot, $R\text{-hat} < 1.1$)

```
print(jagsModel)
```

```
traceplot(jagsModel)
```

```
gelman.plot(jagsModel)
```



```
print(jagsModel)
```

JAGS output for model 'StaticOcc_Veg_WindObsID.txt', generated by jagsUI.
Estimates based on 3 chains of 5000 iterations, ...

	mean	sd	2.5%	50%	97.5%	overlap0	f	Rhat	n.eff
beta0.occ	0.033	0.327	-0.564	0.013	0.724	TRUE	0.517	1.001	3609
beta1.occ	3.116	0.769	1.792	3.057	4.821	FALSE	1.000	1.002	1215
alpha0.det	-2.213	0.420	-3.042	-2.219	-1.393	FALSE	1.000	1.001	7500
alpha1.det	-2.715	0.390	-3.503	-2.697	-1.992	FALSE	1.000	1.003	1528
occ.fs	92.491	7.249	79.000	92.000	107.000	FALSE	1.000	1.000	7500
deviance	237.745	15.012	209.222	237.347	267.742	FALSE	1.000	1.000	7500

Successful convergence based on Rhat values (all < 1.1).

Rhat is the potential scale reduction factor (at convergence, Rhat=1).
For each parameter, n.eff is a crude measure of effective sample size.

...

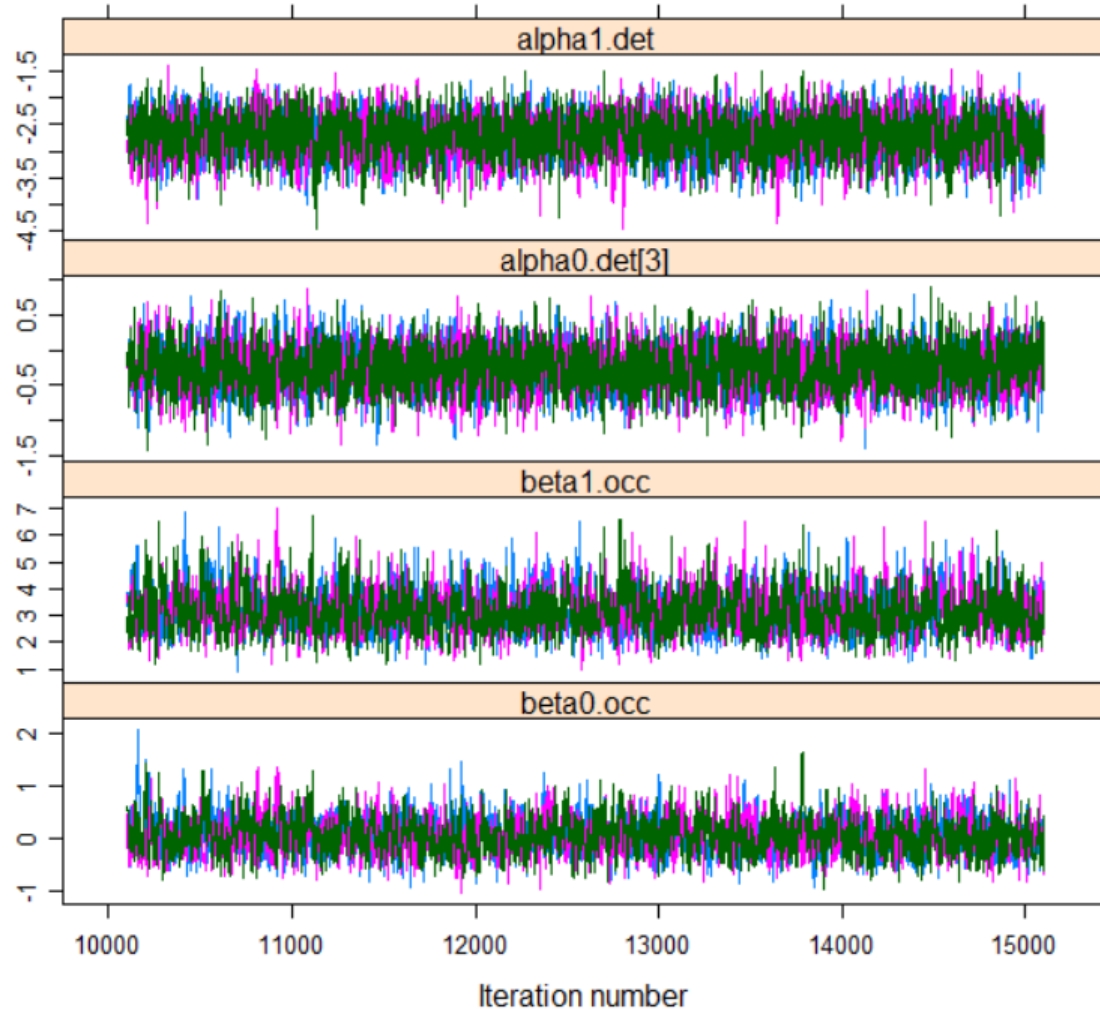
pD = 112.7 and DIC = 350.432

R-hat = Gelman and Rubin's shrink factor
→ Should be < 1.1 for all parameters

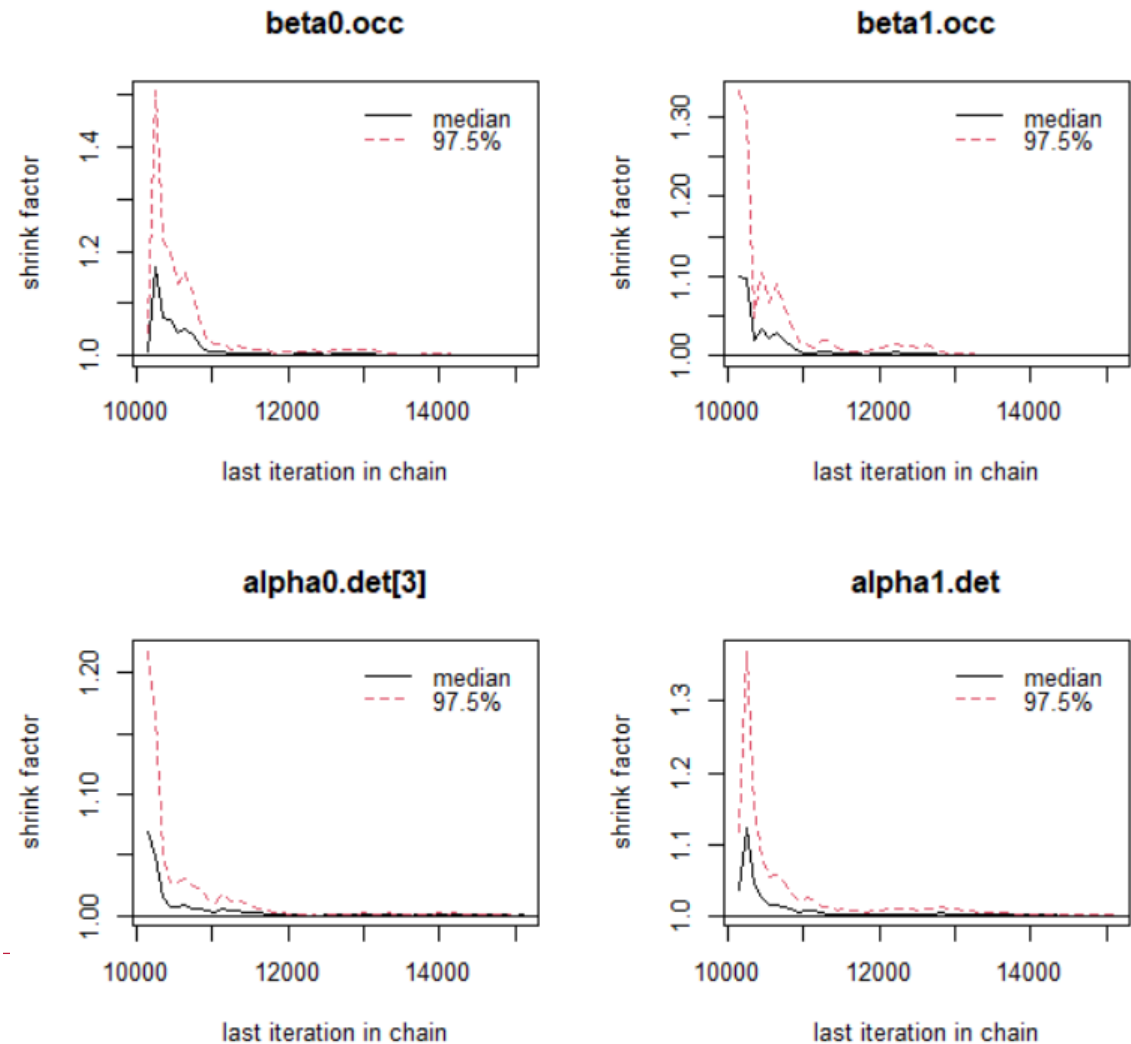
Fitting occupancy models in *JAGS*

37

```
traceplot(jagsModel) #or  
xyplot(jagsModel$samples)
```



```
gelman.plot(jagsModel$samples)  
#Gelman and Rubin's shrink factor (R-hat)
```



10 basic steps of a Bayesian model analysis with JAGS

















- (1) Get an idea of your data and your research question
- (2) Formulate a model and code it in the BUGS syntax
- (3) Think about the necessary prior distributions
- (4) Assemble your data
- (5) Generate (overdispersed) initial parameter values
- (6) Set the MCMC parameters (n.chains, n.iter, thin, ...)
- (7) Call and run JAGS
- (8) Check your obtained posterior samples for convergence (traceplot, $R\text{-hat} < 1.1$)
- (9) If necessary, continue with further iterations until convergence
- (10) Inspect the posterior sample and analyse it with respect to your research question 1



CodingSession_StaticOccupancyModels_SL.R

Truth

180 sites

1	2	3	...	180
			...	
			...	
			...	
			...	

3 repeated
visits to
each site

True state is affected
by vegetation height

Variation in detections is
used to estimate detection
probability, affected by

+

—

