Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II. *doi: 10.18637/jss.v000.i00*

Estimating Animal Abundance with N-Mixture Models Using the R-INLA Package for R

Timothy D. Meehan
National
Audubon Society

Nicole L. Michel
National
Audubon Society

Hävard Rue
King Abdulla University
of Science and Technology

Abstract

Successful management of wildlife populations requires accurate estimates of abundance. Abundance estimates can be confounded by imperfect detection during wildlife surveys. N-mixture models enable quantification of detection probability and often produce abundance estimates that are less biased. The purpose of this study was to demonstrate the use of the **R-INLA** package to analyze N-mixture models and to compare performance of **R-INLA** to two other common approaches – JAGS (via the **runjags** package), which uses Markov chain Monte Carlo and allows Bayesian inference, and **unmarked**, which uses Maximum Likelihood and allows Frequentist inference. We show that **R-INLA** is an attractive option for analyzing N-mixture models when (1) familiar model syntax and data format (relative to other R packages) are desired, (2) survey level covariates of detection are not essential, (3) fast computing times are necessary (**R-INLA** is 10 times faster than **unmarked**, 300 times faster than JAGS), and (4) Bayesian inference is preferred.

Keywords: abundance, detection, JAGS, N-mixture model, R, **R-INLA**, **unmarked**, wildlife.

1. Introduction

1.1. Background

Successful management of wildlife species requires accurate estimates of abundance (Yoccoz, Nichols, and Boulinier 2001). One common method for estimating animal abundance is direct counts (Pollock, Nichols, Simons, Farnsworth, Bailey, and Sauer 2002). Efforts to obtain accurate abundance estimates via direct counts can be hindered by the cryptic nature of many wildlife species and by other factors such as observer expertise, weather, and habitat structure (Denes, Silveira, and Beissinger 2015). The lack of perfect detection in wildlife

surveys is common, and can cause abundance to be greatly underestimated (Wenger and Freeman 2008; Joseph, Elkin, Martin, and Possingham 2009).

In recent years, new sampling schemes and modeling approaches have enabled improved estimates of animal abundance that are less biased by non-detection (Denes *et al.* 2015). One such sampling scheme, termed a metapopulation design (Kery and Royle 2010), involves repeat visits in rapid succession to each of multiple study sites in a study area. If, during repeat visits, the population is assumed to be closed (no immigration , emigration, or reproduction; static abundance), then the ratio of detections to non-detections during repeated counts can inform an estimate of detection probability. This detection probability can, in effect, be used to correct abundance estimates for imperfect detection (Royle 2004).

Data resulting from this sampling scheme are often modeled using an explicitly hierarchical statistical model referred to as an N-mixture model (Royle and Nichols 2003; Dodd and Dorazio 2004; Royle 2004; Kery, Royle, and Schmid 2005). A simple form of an N-mixture model, a binomial mixture model, describes observed counts Y at site i during survey j as coming from a binomial distribution with parameters for abundance N and detection probability p, where N per site is drawn from a Poisson distribution with an expected value, lambda (λ). Specifically,

$$N_i \sim \text{Poisson}(\lambda) \qquad \text{and} \qquad Y_{i,j}|N_i \sim \text{binomial}(N_i, p).$$

λ is commonly modeled as a log-linear function of site covariates, as $\log(\lambda_i) = \beta_0 + \beta_1 * x_i$. Similarly, p is commonly modeled as $\text{logit}(p_{i,j}) = \alpha_0 + \alpha_1 * x_{i,j}$, a logit-linear function of site-survey covariates.

This estimation approach can be extended to cover K distinct breeding seasons, which correspond with distinct years for wildlife species that breed annually (Kery, Dorazio, Soldaat, Van Strien, Zuiderwijk, and Royle 2009). In this case, population closure is assumed across J surveys within year k, but is relaxed across years (Kery *et al.* 2009). A simple specification of a multiple-year model is $N_{i,k} \sim \text{Poisson}(\lambda_{i,k})$, $Y_{i,j,k}|N_{i,k} \sim \text{binomial}(N_{i,k}, p_{i,k})$. Like the single-year specification, λ is commonly modeled using site and site-year covariates, and p using site-survey-year covariates. There are other variations of N-mixture models that accommodate overdispersed counts through use of a negative binomial distribution (Kery and Royle 2010), a zero-inflated Poisson distribution (Wenger and Freeman 2008), or survey-level random intercepts (Kery and Schaub 2011). Yet other variations account for non-independent detection probabilities through use of a beta-binomial distribution (Martin, Royle, Mackenzie, Edwards, Kery, and Gardner 2011), parse different components of detection through the use of unique covariates (O'Donnell, Thompson, and Semlitsch 2015), or relax assumptions of population closure (Chandler, Royle, and King 2011; Dail and Madsen 2011). We do not discuss all of these variations here, but refer interested readers to Denes *et al.* (2015) for an overview.

The development of metapopulation designs and N-mixture models represents a significant advance in quantitative wildlife ecology. However, there are practical issues that sometimes act as barriers to adoption. Many of the examples of N-mixture models in the wildlife literature have employed Bayesian modeling software such as WinBUGS, OpenBUGS, or JAGS (Lunn, Jackson, Best, Thomas, and Spiegelhalter 2012). These are extremely powerful and flexible platforms for analyzing hierarchical models, but they come with a few important challenges. First, many wildlife biologists are not accustomed to coding statistical models using the BUGS

modeling syntax. While there are several outstanding resources aimed at teaching this skill (Royle and Dorazio 2008; Kery 2010; Kery and Schaub 2011; Kery and Royle 2015) it is, nonetheless, a considerable commitment. Second, while Markov chain Monte Carlo (MCMC) chains converge quickly for relatively simple N-mixture models, convergence for more complex models can take hours to days, and may not occur at all.

There are other tools available for analyzing N-mixture models that alleviate some of these practical issues. The **unmarked** package (Fiske, Chandler *et al.* 2011) for R statistical computing software (Team 2016) offers several options for analyzing N-mixture models within a Frequentist framework, with the capacity to accommodate overdispersion and dynamic populations. The model coding syntax used in **unmarked** is a simple extension of the standard R syntax. Models are analyzed using a Maximum Likelihood (ML), so model analysis is often completed in a fraction of the time taken using an MCMC approach. The familiar model syntax and rapid model evaluation of **unmarked** has undoubtedly contributed to the broader adoption of N-mixture models by wildlife biologists. However, it comes at a cost – loss of the intuitive inferential framework associated with Bayesian analysis.

Here we discuss analysis of N-mixture models using the **R-INLA** package (Rue, Martino, Lindgren, Simpson, and Riebler 2013) for R. The **R-INLA** package uses integrated nested Laplace approximation (INLA) to derive posterior distributions for a large class of Bayesian statistical models that can be formulated as latent Gaussian models (Rue, Martino, and Chopin 2009). INLA was developed to allow estimation of posterior distributions in a fraction of the time taken by MCMC. Like **unmarked**, the model syntax used in the **R-INLA** package is a straightforward extension of the modeling syntax commonly used in R. Also, like **unmarked**, the computational cost of analyzing models with **R-INLA** is relatively low. The **R-INLA** approach is different from **unmarked** in that inference about model parameters falls within a Bayesian framework.

1.2. Overall objectives

The purpose of this manuscript is to demonstrate analysis of N-mixture models using the **R-INLA** package. In the process, we employ simulated and real count datasets, and analyze them using JAGS, via the **runjags** package (Denwood 2016) for R, the **unmarked** package, and the **R-INLA** package. In each case, we demonstrate how data are formatted, how models are specified, how model estimates compare to simulation inputs, and how methods compare in terms of computational performance. We also explore a limitation of the **R-INLA** approach related to model specification. Specifically, while it is possible to specify survey level covariates for detection using JAGS and **unmarked**, this is not possible using **R-INLA**. Rather, survey level covariates of detection must be averaged to the site or site-year level. Using an averaged detection covariate does allow accounting for site-level differences in survey conditions, should they occur. However, in the process of averaging, information related to detection within a site or site-year combination is discarded, which could lead to biased detection and abundance estimates under certain conditions.

Much of the code used to conduct analyses is shown in the body of this manuscript. However, some repeated code, and code related to generating tables and figures, is not shown for brevity. All code can be accessed via `https://github.com/tmeeha/inlaNMix`.

1.3. Simulated data

The data simulated for this analysis were intended to represent a typical wildlife population study. To put the simulation in context, consider an effort to estimate the abundance of a bird species in a national park, within which are located 72 study sites. At each site, 3 replicate surveys are conducted within 6 weeks, during the peak of the breeding season, when birds are likely to be singing. In order to estimate a trend in abundance over time, clusters of repeated surveys are conducted each breeding season over a 9-year period.

In this scenario, the abundance of the species is thought to vary with two site-level covariates (λ covariates 1 and 2) that represent habitat characteristics at a site and do not change appreciably over time. The detection probability is also believed to vary according to two covariates (p covariates 1 and 3). The first covariate for detection, p covariate 1, is the same site-level covariate 1 that affects abundance, although it has the opposite effect on detection. The other detection covariate, p covariate 3, is a site-survey-year variable that could be related to weather conditions during an individual survey.

As is commonly the case, we assume that the counts are overdispersed due to the effects of unknown variables. Overdispersion was generated and modeled using a negative binomial distribution for the count component of the model. The specific model parameters and their simulated values were: p.b0 = 1.0 (logit(p) intercept), p.b1 = -2.0 (effect of covariate 1 on logit(p)), p.b3 = 1.0 (effect of covariate 3 on logit(p)), lam.b0 = 2.0 (log(lambda) intercept), lam.b1 = 2.0 (effect of covariate 1 on log(lambda)), lam.b2 = -3.0 (effect of covariate 2 on log(lambda)), lam.b4 = 1.0 (effect of year on log(lambda)), disp.size = 3.0 (size of the overdispersion parameter). All independent variables in the simulation were centered at zero to alleviate computational difficulties and to make model intercepts more easily interpreted.

We simulated data for this study using the following function. Note that the function produces two versions of detection covariate 3, x.p.3 and x.p.3.mean, and two versions of the count matrix, y1 and y2. x.p.3 is the site-survey-year variable described above. It is used to generate y2, which is used in Example II. x.p.3.mean is derived from x.p.3, where values are unique to site and year, but averaged over surveys. It is used to generate y1, which is used in Example I.

```
R> data4sim <- function(n.sites = 72,   # number study sites
+                       n.surveys = 3,  # number replicate surveys
+                       n.years = 9,    # number years
+                       lam.b0 = 2.0,   # intercept for log lambda
+                       lam.b1 = 2.0,   # slope for log lambda cov 1
+                       lam.b2 = -3.0,  # slope for log lambda cov 2
+                       lam.b4 = 1.0,   # slope for log lambda year
+                       p.b0 = 1.0,     # intercept for logit p
+                       p.b1 = -2.0,    # slope for logit p cov 1
+                       p.b3 = 1.0,     # slope for logit p cov 3
+                       disp.size = 3.0 # size of overdispersion
+                       ){
+  # setup
+  if(n.years %% 2 == 0) {n.years <- n.years + 1}     # make years odd
+  N.tr <- array(dim = c(n.sites, n.years))           # for abund
+  y1 <- array(dim = c(n.sites, n.surveys, n.years))  # for ex1 counts
```

```
+  y2 <- array(dim = c(n.sites, n.surveys, n.years))  # for ex2 counts
+  # abundance covariate values
+  x.lam.1 <- array(as.numeric(scale(runif(n = n.sites, -0.5, 0.5),
+                   scale = F)), dim = c(n.sites, n.years))
+  x.lam.2 <- array(as.numeric(scale(runif(n = n.sites, -0.5, 0.5),
+                   scale = F)), dim = c(n.sites, n.years))
+  yrs <- 1:n.years
+  yrs <- (yrs - mean(yrs)) / (max(yrs - mean(yrs))) / 2
+  yr <- array(rep(yrs, each = n.sites), dim = c(n.sites, n.years))
+  # fill abundance array
+  lam.tr <- exp(lam.b0 + lam.b1*x.lam.1 + lam.b2*x.lam.2 + lam.b4*yr)
+  for(i in 1:n.sites){
+    for(k in 1:n.years){
+    N.tr[i, k] <- rnbinom(n = 1, mu = lam.tr[i, k], size = disp.size)
+  }}
+  # detection covariate values
+  x.p.1 <- array(x.lam.1[,1], dim = c(n.sites, n.surveys, n.years))
+  x.p.3 <- array(as.numeric(scale(runif(n = n.sites*n.surveys*n.years,
+                                        -0.5, 0.5), scale=F)),
+                 dim=c(n.sites, n.surveys, n.years))
+  # average x.p.3 per site-year
+  x.p.3.mean <- apply(x.p.3, c(1,3), mean, na.rm=F)
+  out1 <- c()
+  for(k in 1:n.years){
+   chunk1 <- x.p.3.mean[,k]
+   chunk2 <- rep(chunk1, n.surveys)
+   out1 <- c(out1, chunk2)
+  }
+  x.p.3.arr <- array(out1, dim=c(n.sites, n.surveys, n.years))
+  # fill count array with site-yr x.p.3
+  p.tr1 <- plogis(p.b0 + p.b1*x.p.1 + p.b3*x.p.3.arr)
+  for (i in 1:n.sites){
+    for (k in 1:n.years){
+      for (j in 1:n.surveys){
+        y1[i,j,k] <- rbinom(1, size = N.tr[i,k], prob = p.tr1[i,j,k])
+  }}}
+  # fill count array with site-surv-yr x.p.3
+  p.tr2 <- plogis(p.b0 + p.b1 * x.p.1 + p.b3 * x.p.3)
+  for (i in 1:n.sites){
+    for (k in 1:n.years){
+      for (j in 1:n.surveys){
+        y2[i,j,k] <- rbinom(1, size = N.tr[i,k], prob = p.tr2[i,j,k])
+  }}}
+  # return data
+  return(list(n.sites = n.sites, n.surveys = n.surveys, n.years = n.years,
+              x.p.1 = x.p.1[,1,1], x.p.3 = x.p.3, x.p.3.mean = x.p.3.mean,
+              x.p.3.arr = x.p.3.arr, x.lam.1 = x.lam.1[,1],
```

```
+              x.lam.2 = x.lam.2[,1], yr = yr[1,], y1 = y1, y2 = y2,
+              lam.tr = lam.tr, N.tr = N.tr))
+  } #end function
R> set.seed(12345) # make results reproducible
R> sim.data <- data4sim() # create datasets
```

1.4. Real data

In addition to simulated data, we also demonstrate the use of **R-INLA** with a real dataset in Example III. This dataset comes from a study by Kery *et al.* (2005) and is publicly available as part of the **unmarked** package. The dataset includes mallard duck (*Anas platyrhynchos*) counts, conducted at 239 sites on 2 or 3 occasions during the summer of 2002 as part of a Swiss program that monitors breeding bird abundance (Monitoring Häufige Brutvögel). In addition to counts, the dataset also includes 2 site-survey covariates related to detection (survey effort and survey date), and 3 site level covariates related to abundance (route length, elevation, and forest cover). Full dataset details are given in Kery *et al.* (2005).

2. Example I

2.1. Goals

In Example I, we demonstrate the use of **R-INLA** and compare use and performance to similar analyses using JAGS and **unmarked**. In this exercise, the functional forms of JAGS, **unmarked**, and **R-INLA** models match the data generating process. Specifically, we used the covariate x.p.3.mean to generate the count matrix y1 and analyzed the data with models that use x.p.3.mean as a covariate. This exercise was intended to demonstrate the differences and similarities in use, computation time, and estimation results across the three methods when the specified model was reasonably close to the data generating process.

2.2. Analysis with JAGS

We first analyzed the simulated data using JAGS, via the **runjags** package. In defining the model, we specified a negative binomial distribution for the abundance component, and used vague normal priors for the intercepts and the global effects of the covariates of λ and p.

```
R> jags.model.string <- "
+    model {
+    # priors
+    intP ~ dnorm(0, 0.01)        # detection intercept
+    bCov1P ~ dnorm(0, 0.01)      # detection cov 1 effect
+    bCov3P ~ dnorm(0, 0.01)      # detection cov 3 effect
+    intLam ~ dnorm(0, 0.01)      # lambda intercept
+    bCov1Lam ~ dnorm(0, 0.01)    # lambda cov 1 effect
+    bCov2Lam ~ dnorm(0, 0.01)    # lambda cov 2 effect
+    bYr ~ dnorm(0, 0.01)         # year effect
```

```
+    overDisEst ~ dunif(0, 5)    # overdispersion size
+    # abundance component
+    for (k in 1:nYears){
+      for (i in 1:nSites){
+        N[i, k] ~ dnegbin(prob[i, k], overDisEst) # negative binomial
+        prob[i, k] <- overDisEst / (overDisEst + lambda[i, k])
+        log(lambda[i, k]) <- intLam + (bCov1Lam * x.lam.1[i]) +
+                             (bCov2Lam * x.lam.2[i]) + (bYr * yr[k])
+    # detection component
+        for (j in 1:nSurveys){
+          y[i, j, k] ~ dbin(p[i,j,k], N[i,k])
+          p[i, j, k] <- exp(lp[i,j,k]) / (1 + exp(lp[i,j,k]))
+          lp[i, j, k] <- intP + (bCov1P * x.p.1[i]) + (bCov3P * x.p.3[i, k])
+        } # close j loop
+      } # close i loop
+    } # close k loop
+  } # close model loop
+  "
```

Next, we defined the parameters to be monitored during the MCMC runs, bundled the data for JAGS, and created a function for drawing random initial values for the model parameters. The initial values for abundance were made to avoid values of "NA" and zero, as these would cause computational problems.

```
R> params <- c("intP", "bCov1P", "bCov3P", "intLam", "bCov1Lam","bCov2Lam",
+              "bYr", "overDisEst") # parameters to monitor
R> jags.data <- list(y = sim.data$y1, x.lam.1 = sim.data$x.lam.1,
+               x.lam.2 = sim.data$x.lam.2, yr = sim.data$yr,
+               x.p.1 = sim.data$x.p.1, x.p.3 = sim.data$x.p.3.mean,
+               nSites = sim.data$n.sites, nSurveys = sim.data$n.surveys,
+               nYears = sim.data$n.years)
R> N.init <- sim.data$y1 # initial count values
R> N.init[is.na(N.init)] <- 1 # clean up NA's
R> N.init <- apply(N.init, c(1, 3), max) + 1 # zero values cause trouble
R> inits <- function() list(N = N.init, intLam = rnorm(1, 0, 0.01),
+               intP = rnorm(1, 0, 0.01), bCov1P = rnorm(1, 0, 0.01),
+               bCov2Lam = rnorm(1, 0, 0.01), bCov1Lam = rnorm(1, 0, 0.01),
+               bCov3P = rnorm(1, 0, 0.01), bYr = rnorm(1, 0, 0.01),
+               overDisEst = runif(1, 0.5, 2.5))
```

Finally, we set the run parameters and started the MCMC process. Run parameters were chosen such that MCMC diagnostics indicated converged chains (potential scale reduction factors < 1.05) and reasonably robust posterior distributions (effective sample sizes > 1000). Note that the recommended number of effective samples for particularly robust inference is closer to 6000 (Gong and Flegal 2016). So MCMC processing times reported here could be considered low estimates.

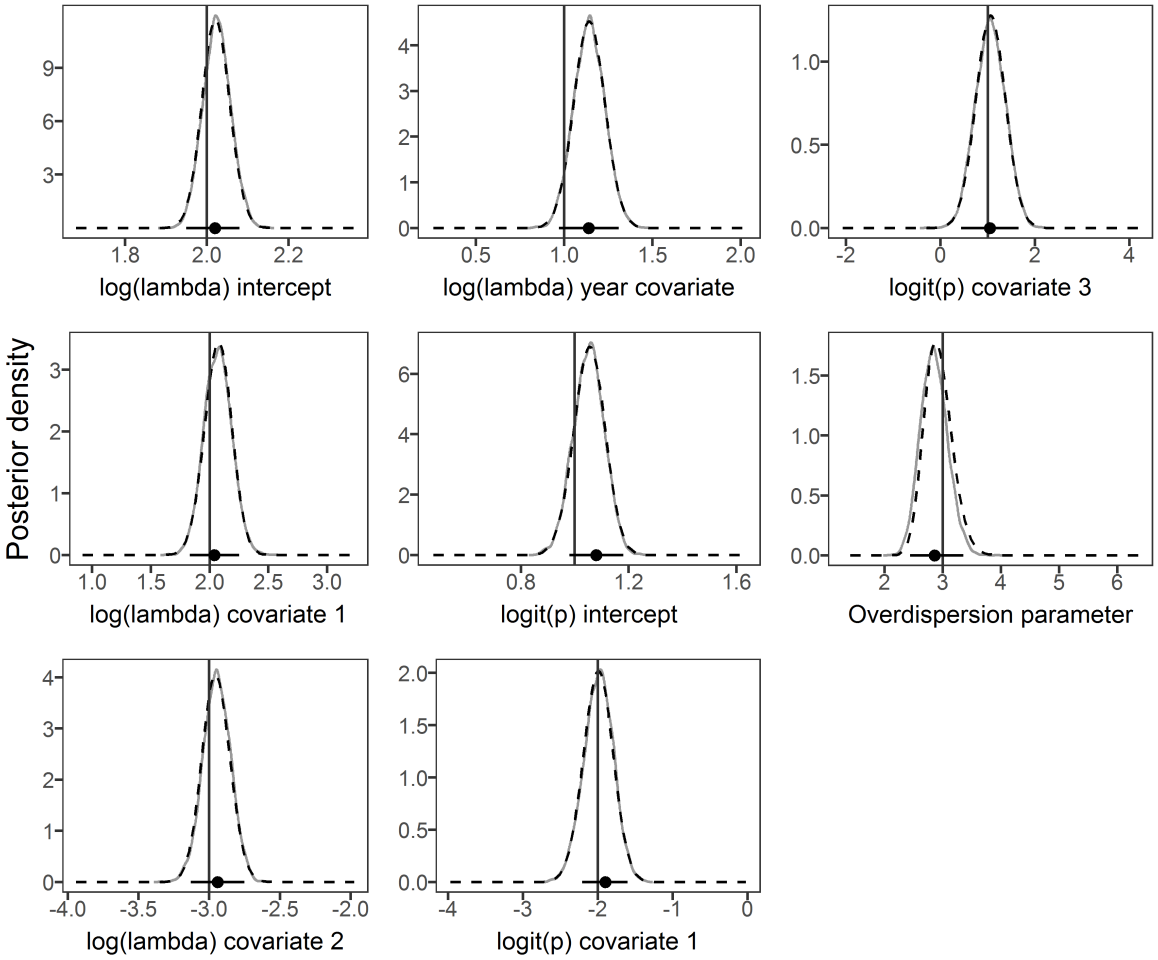

Figure 1: Marginal posteriors from JAGS (solid gray lines) and **R-INLA** (dashed black lines), along with Maximum Liklihood estimates (black circles) and 95% confidence intervals (horizontal black lines) from **unmarked**. True input values are represented by vertical black lines.

```
R> nc <- 3; na <- 2500; nb <- 2500; ni <- 5000; nt <- 10 # run parameters
R> out.jags <- run.jags(model = jags.model.string, data = jags.data,
+                       monitor = params, n.chains = nc, inits = inits,
+                       burnin = nb, adapt = na, sample = ni, thin = nt,
+                       modules = "glm on", method = "parallel")
```

Mean parameter estimates from the JAGS model were reasonably close to, and not significantly different from, the input values used to generate the data (Fig. 1). The potential scale reduction factor for all variables was < 1.01, and the effective sample size for all variables was > 2409. The simulation ran in parallel on 3 virtual cores, 1 MCMC chain per core, and took approximately 2104 seconds.

2.3. Analysis with unmarked

Next, we prepared the simulated data for the **unmarked** analysis, which involved slight mod-

ification of the data in the sim.data object. Here, the count data was changed from a 3-dimensional $I * J * K$ array to a 2-dimensional $I * K$ row by J column matrix. Each static site level variable was duplicated and stacked K times to form a single-column vector. A column vector was created to identify each year in the stacked data. The site-year variable, x.p.3.mean, was transformed from 2-dimensional matrix to a single column vector. Reformatted variables were then assembled in an **unmarked** data structure called an unmarked frame.

```
R> y.unmk <- sim.data$y1[ , , 1]
R> for(i in 2:sim.data$n.years){
+    y.chunk <- sim.data$y1[ , , i]
+    y.unmk <- rbind(y.unmk, y.chunk)
+  }
R> x.lam.1.unmk <- rep(sim.data$x.lam.1, sim.data$n.years)
R> x.lam.2.unmk <- rep(sim.data$x.lam.2, sim.data$n.years)
R> yr.unmk <- rep(sim.data$yr, each = sim.data$n.sites)
R> x.p.1.unmk <- rep(sim.data$x.p.1, sim.data$n.years)
R> x.p.3.unmk <- c(sim.data$x.p.3.mean)
R> site.covs.unmk <- data.frame(x.lam.1.unmk, x.lam.2.unmk, yr.unmk,
+                               x.p.1.unmk, x.p.3.unmk)
R> unmk.data <- unmarkedFramePCount(y = y.unmk, siteCovs = site.covs.unmk)
```

The first argument in the call to the pcount() function was the model formula, which specified the covariates for detection and then the covariates for abundance. This was followed by an argument identifying the appropriate unmarked frame, and the form of the mixture model, negative binomial-binomial.

```
R> out.unmk <- pcount(~ x.p.1.unmk + x.p.3.unmk
+                     ~ x.lam.1.unmk + x.lam.2.unmk + yr.unmk,
+                       data = unmk.data, mixture = "NB")
```

Maximum Likelihood estimates from the **unmarked** analysis are also close to, and not significantly different from, input values (Fig. 1) The **unmarked** estimates were produced in approximately 77 seconds.

2.4. Analysis with R-INLA

Next, we prepared data for **R-INLA**, which uses data formats similar to those used by **unmarked**. The object counts.and.count.covs is an **R-INLA** object that includes an $I * K$ row by J column matrix of counts. Next comes a value of 1, which is turned into an $I * K$ length vector of ones to specify that the model has a global intercept for λ. Finally, there are two $I * K$ length vector of values for the two static site covariates of abundance, where the vector of values for I sites is stacked K times. In addition to the counts.and.count.covs object, we also defined x.p.1.inla, which is a copy of x.lam.1.inla, and x.p.3.inla, which is the $I * K$ length vector corresponding with x.p.3.mean.

```
R> y.inla <- sim.data$y1[ , , 1]
```

```
R> for(i in 2:sim.data$n.years){
+    y.chunk <- sim.data$y1[ , , i]
+    y.inla <- rbind(y.inla, y.chunk)
+  }
R> x.lam.1.inla <- rep(sim.data$x.lam.1, sim.data$n.years)
R> x.lam.2.inla <- rep(sim.data$x.lam.2, sim.data$n.years)
R> yr.inla <- rep(sim.data$yr, each = sim.data$n.sites)
R> x.p.1.inla <- rep(sim.data$x.p.1, sim.data$n.years)
R> x.p.3.inla <- c(sim.data$x.p.3.mean)
R> counts.and.count.covs <- inla.mdata(y.inla, 1, x.lam.1.inla,
+                                      x.lam.2.inla, yr.inla)
```

The call to the inla() function includes several components. First is the model statement. On the left side of the formula is the counts.and.count.covs object that includes the matrix of counts and the covariates related to λ. On the right side of the formula is a 1 to specify a global intercept for p and the two covariates for p. Note that a wide range of random effects for p could be added to the right side of the formula using the f() syntax (Rue *et al.* 2013). The second argument describes the data, provided here as a list that corresponds with the model formula. Third is the likelihood family, which can take values of "nmix" for a Poisson-binomial mixture and "nmixnb" for a negative binomial-binomial mixture. The fourth (control.fixed, for detection parameters) and fifth (control.family, for abundance and overdispersion parameters) arguments specify the priors for the two model components. In this case, the priors are specified similarly to those in the JAGS model, but a variety of other options are available. Several other characteristics of the analysis can be modified in the call to inla(). See Rue *et al.* (2013) for details.

```
R> out.inla <- inla(counts.and.count.covs ~ 1 + x.p.1.inla + x.p.3.inla,
+           data = list(counts.and.count.covs = counts.and.count.covs,
+                       x.p.1.inla = x.p.1.inla, x.p.3.inla = x.p.3.inla),
+           family = "nmixnb",
+           control.fixed = list(mean = 0, mean.intercept = 0, prec = 0.01,
+                                prec.intercept = 0.01),
+           control.family = list(hyper =
+                                   list(theta1 = list(param = c(0, 0.01)),
+                                        theta2 = list(param = c(0, 0.01)),
+                                        theta3 = list(param = c(0, 0.01)),
+                                        theta4 = list(param = c(0, 0.01)),
+                                        theta5 = list(prior = "flat",
+                                                      param = numeric()))))
```

Mean parameter estimates from the **R-INLA** model are also close to, and not significantly different from, input values (Fig. 1). Full marginal posterior distributions from **R-INLA** are shown in Fig. 1. The analysis took approximately 6 seconds.

2.5. Example I summary

Example I demonstrated basic use of **R-INLA** and highlighted similarities and differences between it and two other commonly used approaches. In demonstrating the use of **R-INLA**, we

show that the input data format is not complicated, and that the formatting process can be accomplished with relatively few lines of code. Similarly, model specification uses a straightforward extension of the standard syntax in R, where the count matrix and covariates for λ are specified through an **R-INLA** object included on the left side of the formula, and covariates for p are specified on the right side of the formula. The data format and model specification syntax of **R-INLA** are not too different from **unmarked**, which are both considerably different from JAGS and other BUGS-oriented MCMC software.

Regarding performance, **R-INLA**, **unmarked**, and JAGS successfully extracted simulation input values. Fig. 1 shows marginal posterior distributions produced by JAGS and **R-INLA**, and estimates and 95% confidence intervals from **unmarked**. These results derive from data from one random manifestation of the input values. Thus, we do not expect the posterior distributions for the estimates to be centered at the input values, which would be expected if the simulation was repeated many times. However, we do expect the input values to fall somewhere within the posterior distributions and 95% confidence limits, which is what occurs here. Fig. 1 shows that, for similarly specified models, **R-INLA** (dashed black lines) and JAGS (solid gray lines) yielded practically identical marginal posterior distributions for model parameters.

Where **R-INLA**, **unmarked**, and JAGS differed was in computing time. In this example, **R-INLA** took 6 seconds, **unmarked** took 77 seconds, and JAGS took 2104 seconds to produce results. Thus, **R-INLA** was approximately 10 times faster than **unmarked** and 300 times faster than JAGS. This was the case despite the fact that **unmarked** produces ML estimates and the JAGS model was run in parallel with each of three MCMC chains simulated on a separate virtual computing core. If parallel computing had not been used with JAGS, processing the JAGS model would have taken approximately twice as long. If MCMC simulations were run until an effective sample size of 6000 was reached, processing times would have doubled again.

In sum, when compared to other tools, **R-INLA** is relatively easy to implement and produces accurate estimates of Bayesian posteriors very quickly. Its utility depends on the degree to which the data generating process can be captured accurately in model specification. However, as mentioned above, certain N-mixture models can not be specified using **R-INLA**. For the data in Example I, the count matrix was produced using a detection covariate that was averaged to the site-year level. This averaged covariate was subsequently specified in the model. But what happens when the site-survey-year covariate is an important component of the data generating process, and it can't be entered into the model in this form? This is the question explored in Example II.

3. Example II

3.1. Goals

In Example 2, we show the consequences of not being able to specify a site-survey-year covariate under a range of conditions. Specifically, we conducted a Monte Carlo simulation where, for each iteration, the count matrix for the analysis, y2, was generated with the data generating function described in 1.3, using the site-survey-year covariate x.p.3. The count data were then analyzed with two JAGS models. One model incorporated an averaged site-survey x.p.3.mean as a covariate, such as in 2.1. The other model incorporated the full site-survey-

year covariate instead. In each iteration we randomly varied the size of the effect for x.p.3. We expected that the simpler model, with x.p.3.mean, would yield biased estimates when the effect of x.p.3 was relatively large, and unbiased estimates when the effect was relatively small.

3.2. Analysis with JAGS

Given the long processing time associated with JAGS models, we only ran 3000 iterations (no thinning, after 1000 adaptive and 1000 burn-in iterations) during each simulation. This number is not sufficient for drawing inference from marginal posteriors, but was sufficient for looking at qualitative patterns in posterior means. In each of the 50 iterations, the size of the effect for x.p.3 was drawn from a uniform distribution that ranged from -3 to 3. The results of the simulations are depicted in Fig. 2.

3.3. Example II summary

As expected, biases in estimates of average abundance (log(lambda) intercept) and the effects of abundance covariates (log(lambda) covariate 1) increased with the effect strength of the site-survey-year covariate for p (x.p.3; Fig. 2). When the effect size was small, with an absolute value less than 1, the bias was negligible. When the effect size was large, with an absolute value greater than 2, the bias was considerable (Fig. 2). When interpreting the effect size, bear in mind that x.p.3 ranged from -0.5 to 0.5.

4. Example III

4.1. Goals

In Example III, we explore the performance of **R-INLA** using real data – a publicly available dataset on mallard duck abundance in Switzerland in 2002. By employing real data, we hoped to evaluate (1) the performance of **R-INLA** using data that were not predictable by design and (2) the practical outcome of not being able to specify site-survey covariates in **R-INLA**. The dataset is available as a demonstration dataset in **unmarked**, so we compared the performance of **R-INLA** with **unmarked**, using the demonstration model and analysis settings, in Example III.

4.2. Analysis with unmarked

The **unmarked** model was specified with linear effects of transect length (length), elevation (elev), and forest cover (forest) on abundance, and a linear effect and quadratic effect of sampling intensity (ivel) and sampling date (date), respectively, on detection. Both sampling intensity and sampling date were site-survey, or survey level covariates.

```
R> data("mallard") # get mallard data from unmarked package
R> mallard.y.unmk <- mallard.y # format unmarked data
R> mallard.site.unmk <- mallard.site
R> mallard.obs.unmk <- mallard.obs
```

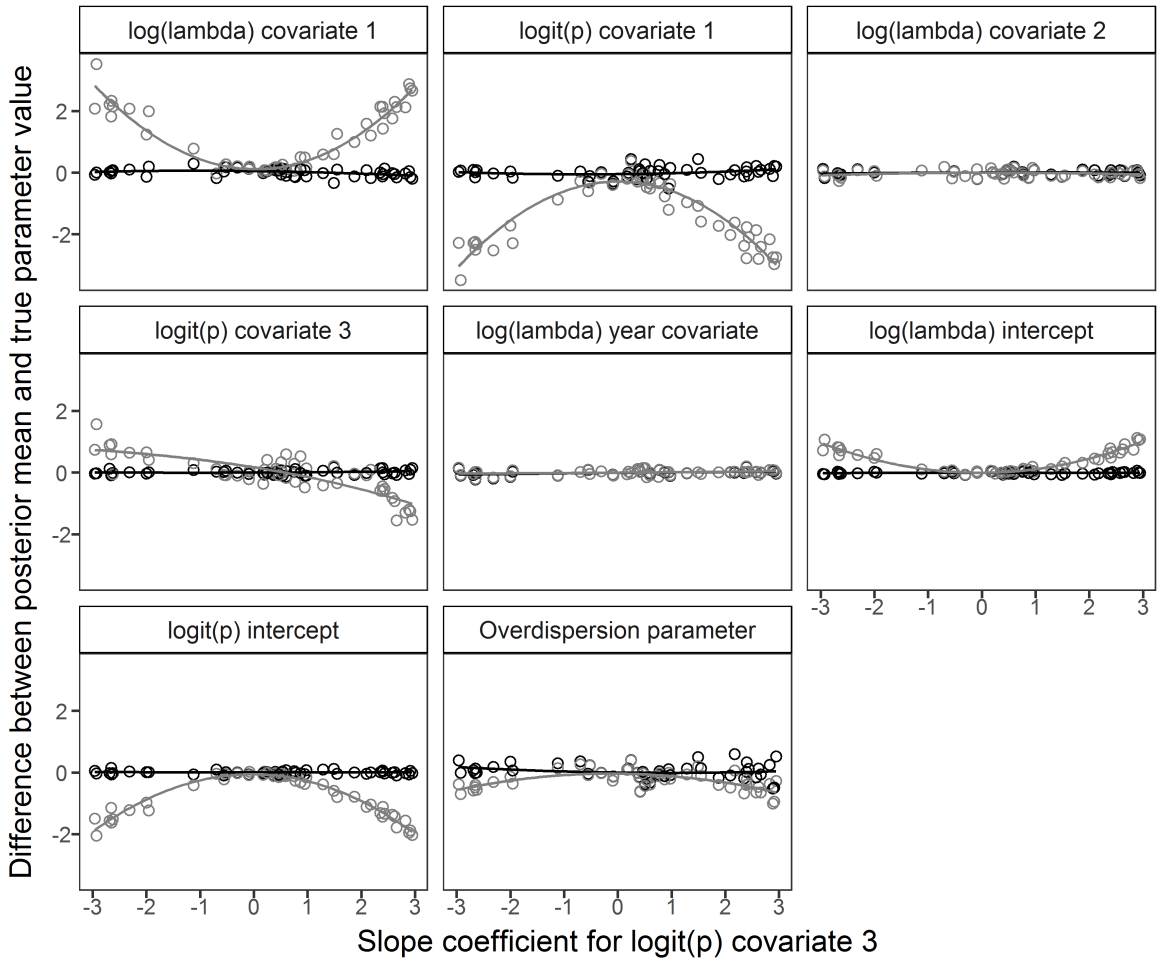

Figure 2: Differences between the estimated posterior mean values and the true input parameter values as a function of the slope coefficient (effect size) for the site-survey-year covariate, x.p.3. Black circles and lines are output from the model with the site-survey-year covariate, x.p.3, and gray circles and lines are from the model with an averaged site-year covariate, x.p.3.mean. Parameter name and true input value is given in the strip across the top of each panel.

```
R> mallardUMF <- unmarkedFramePCount(mallard.y.unmk,
+                                    siteCovs = mallard.site.unmk,
+                                    obsCovs = mallard.obs.unmk)
R> mallard.out.unmk <- pcount(~ ivel+ date + I(date^2)
+                              ~ length + elev + forest,
+                                mixture = "NB", mallardUMF, K=30)
```

4.3. Analysis with R-INLA

The **R-INLA** model used a similar model structure. The only difference was that the **R-INLA** model was not able to process site-survey covariates for detection, so averaged site level covariates were used, instead. These averages were attained using the rowMeans() function.

```
R> mallard.y.inla <- mallard.y # format inla data
R> mallard.length.inla <- mallard.site[,2]
R> mallard.elev.inla <- mallard.site[,1]
R> mallard.forest.inla <- mallard.site[,3]
R> mallard.ivel.inla <- rowMeans(mallard.obs$ivel, na.rm=T) # site average
R> mallard.ivel.inla[is.na(mallard.ivel.inla)] <-
+                                      mean(mallard.ivel.inla, na.rm = T)
R> mallard.date.inla <- rowMeans(mallard.obs$date, na.rm=T) # site average
R> mallard.date2.inla <- mallard.date.inla^2
R> counts.and.count.covs <- inla.mdata(mallard.y.inla, 1,
+                                      mallard.length.inla,
+                                      mallard.elev.inla,
+                                      mallard.forest.inla)
R> mallard.out.inla <- inla(counts.and.count.covs ~ 1 + mallard.ivel.inla +
+                            mallard.date.inla + mallard.date2.inla,
+      data = list(counts.and.count.covs = counts.and.count.covs,
+                            mallard.ivel.inla = mallard.ivel.inla,
+                            mallard.date.inla = mallard.date.inla,
+                            mallard.date2.inla = mallard.date2.inla),
+      family = "nmixnb",
+      control.fixed = list(mean = 0, mean.intercept = 0, prec = 0.01,
+                                 prec.intercept = 0.01),
+      control.family = list(hyper = list(theta1 = list(param = c(0, 0.01)),
+                                      theta2 = list(param = c(0, 0.01)),
+                                      theta3 = list(param = c(0, 0.01)),
+                                      theta4 = list(param = c(0, 0.01)),
+                                      theta5 = list(prior = "flat",
+                                                    param = numeric()))))
```

The 95% credible intervals from the **R-INLA** model overlapped broadly with the 95% confidence intervals from **unmarked** model, so parameter estimates were not significantly different from one another (Fig. 3). Also, the parameters with estimates significantly different from zero were the same regardless of technique (Fig. 3). Using both techniques, detection decreased linearly as the season progressed, and abundance decreased as linear functions of

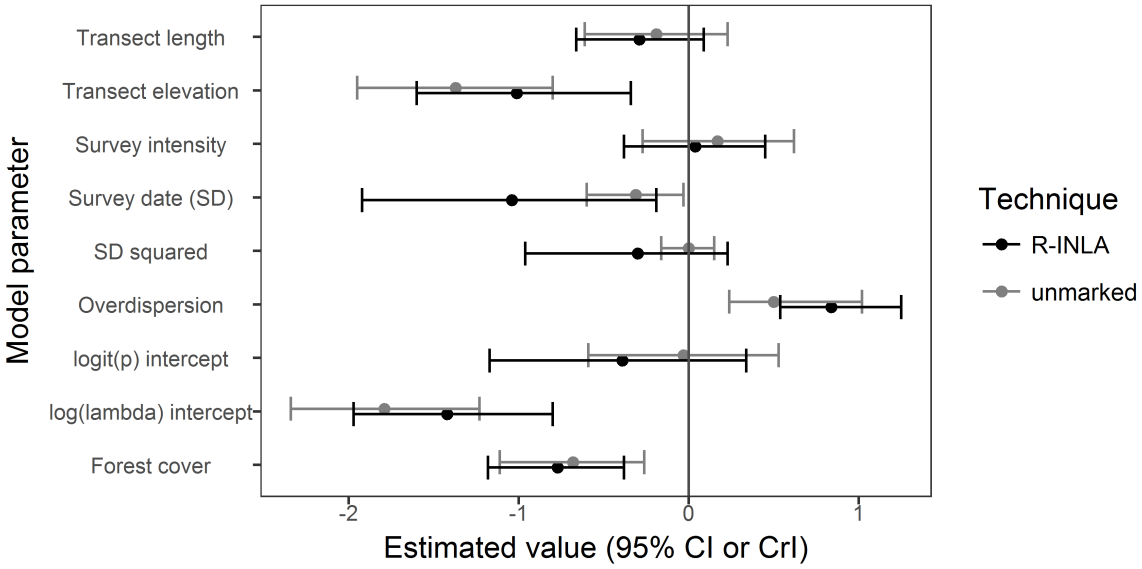

Figure 3: Estimates and 95% confidence intervals from **unmarked** (gray circles and lines) and posterior means and 95% credible intervals from **R-INLA** (black lines) from an N-mixture model of mallard duck abundance. The **unmarked** model included site-survey covariates for survey effort and survey date, while the **R-INLA** model included site-averaged versions. A value of zero is depicted by the vertical gray line.

increasing forest cover and elevation gain. Parameter estimates and biological conclusions were similar despite the fact that site-survey detection covariates were used for **unmarked** and site-averaged detection covariates were used for **R-INLA**. Note that both approaches estimated moderate effects of detection covariates which, according to the results in Example II, would indicate that other parameter estimates were not substantially biased. These results may have been different with a different real dataset, where detection covariates had very strong effects.

5. Discussion

The purpose of this study was to demonstrate the use of the **R-INLA** package (Rue *et al.* 2013) to analyze N-mixture models and to compare performance of **R-INLA** to two other common approaches, JAGS (Lunn *et al.* 2012), via the **runjags** package (Denwood 2016), which employs MCMC methods and allows Bayesian inference, and the **unmarked** package (Fiske *et al.* 2011), which uses Maximum Likelihood and allows Frequentist inference.

Results showed that **R-INLA** can be a complementary tool in the wildlife biologist's analytical tool kit. Strengths of **R-INLA** include Bayesian inference, based on highly accurate approximations of posterior distributions, which are derived over 300 times faster than MCMC methods, where models are specified using a syntax that should be familiar to R users, and where data are formatted in a straightforward way with relatively few lines of code. The straightforward model syntax and data format could help remove barriers to adoption of N-mixture models for biologists who are not committed to learning the BUGS syntax. The

substantial decrease in computation time should facilitate use of a wider variety of model and variable selection techniques (e.g., cross validation and model averaging), ones that are not commonly used in an MCMC context due to practical issues related to computing time (Kery and Schaub 2011).

Limitations of **R-INLA** are mainly related to the more restricted set of N-mixture models that can be specified. Of the approaches explored here, BUGS-based approaches allow users ultimate flexibility in specifying models. For example, with JAGS, site-survey-year covariates for detection are possible, multiple types of mixed distributions are available (Joseph *et al.* 2009; Martin *et al.* 2011), and a variety of random effects can be specified for both λ and p (Kery and Schaub 2011). In comparison, the current version of **R-INLA** does not handle site-survey covariates, employs only Poisson-binomial and negative binomial-binomial mixtures, and handles random effects (exchangeable, spatially and temporally structured) for p only. In cases where site-survey covariates are particularly important and not otherwise controlled in the sampling design, **R-INLA** will not be the appropriate tool (Fig. 2).

When compared to **unmarked**, the **R-INLA** approach is similar in regards to model syntax and data format. The approaches are also similar in that both yield results much faster than MCMC. The two approaches differ in that **R-INLA** is roughly 10 times faster than **unmarked**, possibly due to the different approach used to compute model likelihoods (see Appendix A). They also differ in that **unmarked** can accommodate site-survey covariates and can analyze dynamic N-mixture models (Chandler *et al.* 2011; Dail and Madsen 2011).

In conclusion, JAGS (and WinBUGS, OpenBUGS), **unmarked**, and **R-INLA** all allow users to analyze N-mixture models for estimating wildlife abundance while accounting for imperfect detection. Each method has its strengths and limitations. **R-INLA** appears to be an attractive option when survey level covariates are not essential, familiar model syntax and data format are desired, fast computing time is needed, and Bayesian inference is preferred.

Acknowledgments

We thank the following individuals for commenting on previous drafts of this manuscript.

References

Chandler RB, Royle JA, King DI (2011). "Inference About Density and Temporary Emigration in Unmarked Populations." *Ecology*, **92**(7), 1429–1435.

Dail D, Madsen L (2011). "Models for Estimating Abundance from Repeated Counts of an Open Metapopulation." *Biometrics*, **67**(2), 577–587.

Denes FV, Silveira LF, Beissinger SR (2015). "Estimating Abundance of Unmarked Animal Populations: Accounting for Imperfect Detection and Other Sources of Zero Inflation." *Methods in Ecology and Evolution*, **6**(5), 543– 556.

Denwood MJ (2016). "**runjags**: An R Package Providing Interface Utilities, Model Templates, Parallel Computing Methods and Additional Distributions for MCMC Models in JAGS." *Journal of Statistical Software*, **71**(9). URL https://www.jstatsoft.org/article/view/v071i09.

Dodd CK, Dorazio RM (2004). "Using Counts to Simultaneously Estimate Abundance and Detection Probabilities in a Salamander Community." *Herpetologica*, **60**(4), 468–478.

Fiske I, Chandler R, *et al.* (2011). "**unmarked**: An R Package for Fitting Hierarchical models of Wildlife Occurrence and Abundance." *Journal of Statistical Software*, **43**(10), 1–23. URL https://www.jstatsoft.org/article/view/v043i10.

Gong L, Flegal JM (2016). "A Practical Sequential Stopping Rule for High- Dimensional Markov Chain Monte Carlo." *Journal of Computational and Graphical Statistics*, **25**(3), 684–700. ISSN 1061-8600. doi:10.1080/10618600.2015.1044092.

Joseph LN, Elkin C, Martin TG, Possingham HP (2009). "Modeling Abundance Using N-Mixture Models: The Importance of Considering Ecological Mechanisms." *Ecological Applications*, **19**(3), 631–642.

Kery M (2010). *Introduction to WinBUGS for Ecologists: Bayesian Approach to Regression, ANOVA, Mixed Models and Related Analyses*. Academic Press.

Kery M, Dorazio RM, Soldaat L, Van Strien A, Zuiderwijk A, Royle JA (2009). "Trend Estimation in Populations with Imperfect Dtection." *Journal of Applied Ecology*, **46**(6), 1163–1172.

Kery M, Royle JA (2010). "Hierarchical Modelling and Estimation of Abundance and Population Trends in Metapopulation Designs." *Journal of Animal Ecology*, **79**(2), 453– 461.

Kery M, Royle JA (2015). *Applied Hierarchical Modeling in Ecology: Analysis of Distribution, Abundance and Species Richness in R and BUGS: Volume 1: Prelude and Static Models*. Academic Press.

Kery M, Royle JA, Schmid H (2005). "Modeling Avian Abundance from Replicated Counts using Binomial Mixture Models." *Ecological Applications*, **15**(4), 1450–1461.

Kery M, Schaub M (2011). *Bayesian Population Analysis using WinBUGS: A Hierarchical Perspective*. Academic Press.

Lunn D, Jackson C, Best N, Thomas A, Spiegelhalter D (2012). *The Bugs Book: A Practical Introduction to Bayesian Analysis*. CRC press.

Martin J, Royle JA, Mackenzie DI, Edwards HH, Kery M, Gardner B (2011). "Accounting for Non-Independent Detection when Estimating Abundance of Organisms with a Bayesian Approach." *Methods in Ecology and Evolution*, **2**(6), 595–601.

O'Donnell KM, Thompson FR, Semlitsch RD (2015). "Partitioning Detectability Components in Populations Subject to Within-Season Temporary Emigration using Binomial Mixture Models." *PLoS ONE*, **10**(3), e0117216. URL http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0117216.

Pollock KH, Nichols JD, Simons TR, Farnsworth GL, Bailey LL, Sauer JR (2002). "Large Scale Wildlife Monitoring Studies: Statistical Methods for Design and Analysis." *Environmetrics*, **13**(2), 105–119.

Royle JA (2004). "N-Mixture Models for Estimating Population Size from Spatially Replicated Counts." *Biometrics*, **60**(1), 108–115.

Royle JA, Dorazio RM (2008). *Hierarchical Modeling and Inference in Ecology: the Analysis of Data from Populations, Metapopulations, and Communities*. Academic Press.

Royle JA, Nichols JD (2003). "Estimating Abundance from Repeated Presence- Absence Data or Point Counts." *Ecology*, **84**(3), 777–790.

Rue H, Martino S, Chopin N (2009). "Approximate Bayesian Inference for Latent Gaussian Models by Using Integrated Nested Laplace Approximations." *Journal of the Royal Statistical Society B*, **71**(2), 319–392.

Rue H, Martino S, Lindgren F, Simpson D, Riebler A (2013). ***R-INLA**: Approximate Bayesian Inference using Integrated Nested Laplace Approximations*.

Team RC (2016). "R: A Language and Environment for Statistical Computing." URL http://www.R-project.org.

Wenger SJ, Freeman MC (2008). "Estimating Species Occurrence, Abundance, and Detection Probability using Zero-Inflated Distributions." *Ecology*, **89**(10), 2953–2959.

Yoccoz NG, Nichols JD, Boulinier T (2001). "Monitoring of Biological Diversity in Space and Time." *Trends in Ecology and Evolution*, **16**(8), 446–453.

A. Recursive computations of the "nmix" likelihood

The likelihood for the simplest case is

$$\text{Probability}(y) = \sum_{n=y}^{\infty} \text{Poisson}(n;\lambda) \ \times \ \text{binomial}(y;n,p)$$

where $\text{Poisson}(n;\lambda)$ is the density for the Poisson distribution with mean λ, $\lambda^n \exp(-\lambda)/n!$, and $\text{binomial}(y;n,p)$ is the density for the binomial distribution with n trials and probability p, $\binom{n}{y}p^y(1-p)^{n-p}$. Although the likelihood can be computed directly when replacing the infinite limit with a finite value, we will demonstrate here that we can easily evaluate it using a recursive algorithm that is both faster and more numerical stable. The same idea is also applicable to the negative binomial case, and the case where we have replicated observations of the same n. We leave it to the reader to derive these straight forward extensions.

The key observation is that both the Poisson and the binomial distribution can be evaluated recursively in n,

$$\text{Poisson}(n;\lambda) = \text{Poisson}(n-1;\lambda)\frac{\lambda}{n}$$

and

$$\text{binomial}(y;n,p) = \text{binomial}(y;n-1,p)\frac{n}{n-y}(1-p),$$

and then also for the Poisson-binomial product

$$\text{Poisson}(n;\lambda)\ \text{binomial}(y;n,p) = \text{Poisson}(n-1;\lambda)\ \text{binomial}(y;n-1,p)\frac{\lambda}{n-y}(1-p).$$

If we define $f_i = \lambda(1-p)/i$ for $i = 1, 2, \ldots$, we can make use of this recursive form to express the likelihood with a finite upper limit as

$$\begin{aligned}
\text{Prob}(y) &= \sum_{n=y}^{n_{\max}} \text{Poisson}(n;\lambda)\ \text{binomial}(y;n,p) \\
&= \text{Poisson}(y;\lambda)\ \text{binomial}(y;y,p)\Big\{1 + f_1 + f_1 f_2 + \ldots + f_1 \cdots f_{n_{\max}}\Big\} \\
&= \text{Poisson}(y;\lambda)\ \text{binomial}(y;y,p)\Big\{1 + f_1(1 + f_2(1 + f_3(1 + \ldots)))\Big\}
\end{aligned}$$

The log-liklihood can then be evaluated using the following simple R-code.

```
R> fac <- 1; ff <- lambda * (1-p)
R> for(i in (n.max - y):1){
+    fac <- 1 + fac * ff / i
+  }
R> log.L <- dpois(y, lambda, log = TRUE) + dbinom(y, y, p, log = TRUE) +
+           log(fac)
```

Since this evaluation is recursive in decreasing n, we have to choose the upper limit $n_{\max}$ up-front, for example as an integer larger than y so that $\frac{\lambda(1-p)}{n_{\max}-y}$ is small. Note that we are computing `fac` starting with the smallest contributions, which are more numerically stable.

Affiliation:

Timothy D. Meehan
National Audubon Society
Boulder, Colorado USA
E-mail: tmeehan@audubon.org

Nicole L. Michel
National Audubon Society
San Francisco, California USA
E-mail: nmichel@audubon.org

Hävard Rue
King Abdulla University of Science and Technology
Thuwal, SAU
E-mail: haavard.rue@kaust.edu.sa

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
MMMMMM YYYY, Volume VV, Issue II *Submitted:* yyyy-mm-dd
doi:10.18637/jss.v000.i00 *Accepted:* yyyy-mm-dd