Appendix S1. Estimating Aplomado Falcon Vital Rates using Cormack-Jolly-Seber and Integrated Population Models

Brian W. Rolek

12 July 2022

Contents

1 Preliminary Analyses **1**
1.1 Overview: Selecting Covariates for Survival, Recruitment, and Resight Probabilities 1
1.2 Overview: Immigration and Emigration Rates 2
1.3 Formulation of Multi-state Cormack-Jolly-Seber (CJS) Model with Emigration 2
1.4 Implementation of the Multi-state CJS Model with Global Constraints 4
1.5 Implementation of the Integrated Population Model with Immigration 16

2. Implementation of the Integrated Population Model without Immigration for Evaluation of Vital Rates **30**
2.1 JAGS and R Code for the IPM without Immigration 30
2.2 Postprocess Model Output 40

3. Goodness-of-fit tests for the IPM **47**

4. Literature cited **49**

Supplemental materials for: B. W. Rolek, B. Pauli, A. Macias-Duarte, B. Mutch, P. Juergens, T. Anderson, C. Parish, J. Johnson, B. Millsap, C. J. W. McClure. 2022. Long-term Demography of a Reintroduced and Isolated Population of Endangered Falcons. Global Ecology and Conservation.

Metadata (see Readme), data, and scripts used in analyses can be found at https://github.com/The-Peregrine-Fund/Aplomado-Falcon-IPM

1 Preliminary Analyses

1.1 Overview: Selecting Covariates for Survival, Recruitment, and Resight Probabilities

Preliminary analyses had two primary components: (1) an evaluation of the probabilities of survival, recruitment, and resight in response to explanatory variables (i.e. hacked, sex, and effort) for each life stage; and (2) an evaluation of emigration and immigration probabilities. For the evaluation of survival, recruitment, and resight response to explanatory variables (1) we used a state-space formulation of a Cormack-Jolly-Seber survival model (hereafter CJS, Gimenez et al. 2007, Royle 2008) with capture-mark-resight-dead recovery data (Kéry and Schaub 2012).

We implemented a preliminary CJS model using the global model that included all discrete explanatory variables considered here as fixed effects for survival, recruitment, and resight probabilities. The explanatory variable 'sex' indexed the mean intercept of survival, recruitment, and resight probabilities as either female or male; 'hacked status' indexed the mean intercept of survival, recruitment, and resight probabilities as either wild or hacked; and 'survey effort' indexed the mean intercept of resight probabilities in a given year

as low or high effort. Additionally, we included these explanatory variables as indices of the standard errors (σ) associated with the random factor for each year, allowing groups (e.g. male and female) to have different standard errors over time.

We used the preliminary CJS model that included all explanatory variables to estimate whether parameters were important by calculating the difference between pairs of groups (e.g. male and female, Kruschke 2011, McClure et al. 2017b) using their posterior distributions and retained important covariates when 85% highest density intervals (HDIs) of these differences did not intersect zero. We retained important explanatory variables within a CJS survival model for use in subsequent IPMs (Appendix S1: Table S1 and Fig. S1). This CJS model included five observation states and seven true states that allowed for the estimation of emigration and true survival (see S1 Section 1 for details).

1.2 Overview: Immigration and Emigration Rates

We estimated emigration from an integrated population model by using mark-resight-dead recovery data (p. 241, Kéry and Schaub 2012). We did not have explicit data on immigration; however, IPMs can estimate a latent vital rate when all other vital rates were measured empirically (Kéry and Schaub 2012); therefore, we used a model-based method to estimate the number of immigrants (N^I, denoted with superscript I) and an immigration rate (ω, Abadi et al. 2010) of adults (non-breeders and breeders). First-years must be recruited as either breeders or non-breeders during the following post-breeding survey; therefore, we did not include immigration of first-years. We constructed an identical IPM to the one described below (see Integrated Population Model), except that we included an immigration rate parameter. We used this preliminary IPM analysis to assess whether inclusion of immigration rates could bias estimation of vital rates (Schaub and Fletcher 2015), and excluded immigration rates if they were negligible (median immigration rate 0.01), and when density plots of posterior distributions of survival did not visually differ.

Including latent immigration rates in IPMs when populations are isolated could bias estimates (Schaub and Fletcher 2015; Paquet et al. 2021) because immigration rates were constrained to positive values. Therefore, we excluded immigration from final analyses if estimates were too imprecise or biologically implausible.

1.3 Formulation of Multi-state Cormack-Jolly-Seber (CJS) Model with Emigration

The formulation of the CJS model with emigration is included in the Methods section of the manuscript. We review the basic structure here.

1.3.1 Observation Matrix

We specified the observation matrix (PO) that relates observed states (columns) to real states (rows) of Northern Aplomado Falcons.
Observed states (1–5, columns left to right):
1. Seen first-year
2. Seen non-breeder
3. Seen breeder
4. Seen dead
5. Not seen

True states (1–7, rows top to bottom):
1. First-year
2. Non-breeder
3. Breeder
4. Recovered recently dead
5. Dead not recovered
6. Emigrated and alive
7. Emigrated and dead

where superscripts with capital letters are used as labels for life cycle states: first-year (O), non-breeder (A), and breeder (B). Observations are related to true states using transition and resight probabilities (p) for non-breeder (p^A) and breeders (p^B):

$$PO = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & p^A & 0 & 0 & 1-p^A \\ 0 & 0 & p^B & 0 & 1-p^B \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

We assume perfect resight probability (1.0 at row 1 and column 1) because these birds must be captured during their first observation.

1.3.2 State-Transition Matrix

Next, we specified the state-transition matrix (Ω) that governs dynamics between true states over time. Rows of the matrix represent true states during time step (t), and columns represent true states during the following time step ($t+1$). Each row and column number correspond to true states listed above. For example, row 1 corresponds with first-years at time t and column 1 corresponds with first-years at time t+1. Here, r is the recapture probability of a recently dead bird and δ is the probability of emigration.

$$\Omega = \begin{bmatrix} 0 & \phi^0(1-\psi^{0B})(1-\delta) & \phi^0\psi^{0B}(1-\delta) & (1-\phi^0)r(1-\delta) & (1-\phi^0)(1-r)(1-\delta) & \phi^0\delta & (1-\phi^0)(1-r)\delta \\ 0 & \phi^{AB}(1-\psi^{AB})(1-\delta) & \phi^A\psi^{AB}(1-\delta) & (1-\phi^A)r(1-\delta) & (1-\phi^A)(1-r)(1-\delta) & \phi^A\delta & (1-\phi^A)(1-r)\delta \\ 0 & \phi^B\psi^{BA}(1-\delta) & \phi^B(1-\psi^{BA})(1-\delta) & (1-\phi^B)r(1-\delta) & (1-\phi^B)(1-r)(1-\delta) & \phi^B\delta & (1-\phi^B)(1-r)\delta \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

1.3.3 CJS Model with Global Constraints (Includes All Group Covariates)

We implemented the global CJS model in Just Another Gibbs Sampler (JAGS). The "global survival model" includes all covariates considered here. Probabilities of survival (ϕ), recruitment (ψ), and resight (p) are contrained to vary by sex (s, male or female), hacked (h, wild-born or captive bred). Each life stage (first-year, non-breeder, breeder) and group (sex, hacked, effort) has a unique variance parameter over time. We used the same constraints for first-year, non-breeder, and breeder birds. Superscripts with Greek symbols are used to label parameters.

$$logit(\phi_{i,t}) = \mu^{\phi}_{s,h} + \epsilon^{\phi}_{s,h,t}$$

$$\epsilon^{\phi}_{s,h,t} = normal(0, \tau = 1/\sigma^{\phi,2}_{s,h})$$

$$logit(\psi_{i,t}) = \mu^{\psi}_{s,h} + \epsilon^{\psi}_{s,h,t}$$

$$\epsilon^{\psi}_{s,h,t} = normal(0, \tau = 1/\sigma^{\psi,2}_{s,h})$$

Furthermore, probability of resight varied by survey effort (e) for non-breeders and breeders.

$$logit(p_{i,t}) = \mu^{p}_{s,h,e} + \epsilon^{p}_{s,h,t}$$

$$\epsilon^{p}_{s,h,t} = normal(0, \tau = 1/\sigma^{p,2}_{s,h})$$

Probabilities of emigration (δ) and dead recovery (r) are contrained to an overall mean μ.

$$logit(\delta_{i,t}) = \mu^{\delta}$$

Including emigration within this model allowed us to estimate *true* survival rather than *apparent* survival, because apparent survival is confounded with emigration (1-site fidelity) in some CJS models.

1.4 Implementation of the Multi-state CJS Model with Global Constraints

1.4.1 JAGS and R Code for Implementing the CJS Model with Global Constraints

```
# JAGS must be installed on your computer
library (jagsUI)
load("./data/data-7states.Rdata")
m<- c("surv7-jags-global")
modfl <- paste("./", m, ".txt", sep="")

sink(modfl)
cat("
    model{
    ######################################################
    ######################################################
    # Mark-resight-recovery data
    #   Observations (po) = y
    #     1 seen first year (age 0)
    #     2 seen nonbreeder
    #     3 seen breeder
    #     4 recovered dead
    #     5 not seen
    #   States (ps)
    #     1 alive first year
    #     2 alive nonbreeder
    #     3 alive breeder
    #     4 dead
    #     5 dead not recovered
    #     6 emigrated alive
    #     7 emigrated dead
    #   Groups
    #     1 wild born
    #     2 hacked
    #   Sex
    #     1 female
    #     2 male
    #   Effort
    #     1 low
    #     2 high
    ######################################################
    # PARAMETERS
    #   sO: survival probability first year
    #       (sO as in the letter O rather than zero so jags can parse)
    #   sA: survival probability nonbreeders
    #   sB: survival probability breeders
    #   psiOB: recruitment probability from first-year to breeder
    #   psiAB: recruitment probability from nonbreeders to breeder
    #   psiBA: recruitment probability from breeder to nonbreeders
    #   pO: resight probability first-year
    #   pA: resight probability nonbreeders
    #   pB: resight probability breeder
    #   em: probability of emigration
    #   r: probability of dead recovery
    ######################################################
```

```
# Priors and constraints
#####################################################
l.mu.em <- logit(mu.em)
mu.em ~ dunif(0,1)
r ~ dunif(0, 1)

for (h in 1:2){
for (s in 1:2){
for (t in 1:(n.yr-1)){
logit(eta.OSalpha[s,h,t]) <- OSalpha[s,h] + eps.OS.s[s,h,t]
logit(eta.ASalpha[s,h,t]) <- ASalpha[s,h] + eps.AS.s[s,h,t]
logit(eta.BSalpha[s,h,t]) <- BSalpha[s,h] + eps.BS.s[s,h,t]
logit(eta.ABRalpha[s,h,t]) <- ABRalpha[s,h] + eps.ABR.psi[s,h,t]
logit(eta.OBRalpha[s,h,t]) <- OBRalpha[s,h] + eps.OBR.psi[s,h,t]
logit(eta.BARalpha[s,h,t]) <- BARalpha[s,h] + eps.BAR.psi[s,h,t]
logit(eta.pA[s,h,t]) <- mu.pA[s,h,effort[t]] + eps.pA[s,h,t]
logit(eta.pB[s,h,t]) <- mu.pB[s,h,effort[t]] + eps.pB[s,h,t]

eps.BS.s[s,h,t] ~ dnorm(0, 1/(sigma.BS.s[s,h]*sigma.BS.s[s,h]))
eps.AS.s[s,h,t] ~ dnorm(0, 1/(sigma.AS.s[s,h]*sigma.AS.s[s,h]))
eps.OS.s[s,h,t] ~ dnorm(0, 1/(sigma.OS.s[s,h]*sigma.OS.s[s,h]))
eps.OBR.psi[s,h,t] ~ dnorm(0, 1/(sigma.OBR.psi[s,h]*sigma.OBR.psi[s,h]))
eps.ABR.psi[s,h,t] ~ dnorm(0, 1/(sigma.ABR.psi[s,h]*sigma.ABR.psi[s,h]))
eps.BAR.psi[s,h,t] ~ dnorm(0, 1/(sigma.BAR.psi[s,h]*sigma.BAR.psi[s,h]))
eps.pA[s,h,t] ~ dnorm(0, 1/(sigma.pA[s,h]*sigma.pA[s,h]))
eps.pB[s,h,t] ~ dnorm(0, 1/(sigma.pB[s,h]*sigma.pB[s,h]))
} } } #s sex #h hacked #t time

for (h in 1:2){
for (s in 1:2){

for (k in 1:2){
mu.pB[s,h,k]<- logit(mu.pB1[s,h,k])
mu.pB1[s,h,k] ~ dunif(0, 1)
mu.pA[s,h,k] <- logit(mu.pA1[s,h,k])
mu.pA1[s,h,k] ~ dunif(0,1)
} #k effort

OSalpha[s,h] <- logit(OSalpha1[s,h])
OSalpha1[s,h] ~ dunif(0, 1)
ASalpha[s,h] <- logit(ASalpha1[s,h])
ASalpha1[s,h] ~ dunif(0, 1)
BSalpha[s,h] <- logit(BSalpha1[s,h])
BSalpha1[s,h] ~ dunif(0, 1)
ABRalpha[s,h] <- logit(ABRalpha1[s,h])
ABRalpha1[s,h] ~ dunif(0, 1)
BARalpha[s,h] <- logit(BARalpha1[s,h])
BARalpha1[s,h] ~ dunif(0, 1)
OBRalpha[s,h] <- logit(OBRalpha1[s,h])
OBRalpha1[s,h] ~ dunif(0, 1)

sigma.OS.s[s,h] ~ dnorm(0, 1/(2*2) )T(0,)
sigma.AS.s[s,h] ~ dnorm(0, 1/(2*2) )T(0,)
```

```
sigma.BS.s[s,h] ~ dnorm(0, 1/(2*2) )T(0,)
sigma.OBR.psi[s,h] ~ dnorm(0, 1/(2*2) )T(0,)
sigma.ABR.psi[s,h] ~ dnorm(0, 1/(2*2) )T(0,)
sigma.BAR.psi[s,h] ~ dnorm(0, 1/(2*2) )T(0,)
sigma.pB[s,h] ~ dnorm(0, 1/(2*2) )T(0,)
sigma.pA[s,h] ~ dnorm(0, 1/(2*2) )T(0,)
}} # s h

for (t in 1:(n.yr-1)){
logit(em[t]) <-  l.mu.em
} #t time

################################
# Likelihood for survival
################################
for (i in 1:nind){
for (t in 1:(n.yr-1)){
#Survival
sO[i,t] <- eta.OSalpha[sex[i],hacked[i], t] # first year
sA[i,t] <- eta.ASalpha[sex[i],hacked[i], t] # nonbreeder
sB[i,t] <- eta.BSalpha[sex[i],hacked[i], t] # breeder
#Recruitment
psiOB[i,t] <- eta.OBRalpha[sex[i],hacked[i], t] # first year to breeder
psiAB[i,t] <- eta.ABRalpha[sex[i],hacked[i], t] # nonbreeder to breeder
psiBA[i,t] <- eta.BARalpha[sex[i],hacked[i], t] # breeder to nonbreeder
#Re-encounter
pA[i,t] <- eta.pA[sex[i],hacked[i], t] # resight of nonbreeders
pB[i,t] <- eta.pB[sex[i],hacked[i], t]  # resight of breeders
}#t
}#i

# Define state-transition and observation matrices
for (i in 1:nind){
# Define probabilities of state S(t+1) given S(t)
for (t in first[i]:(n.yr-1)){
ps[1,i,t,1] <- 0
ps[1,i,t,2] <- sO[i,t] * (1-psiOB[i,t]) * (1-em[t])
ps[1,i,t,3] <- sO[i,t] * psiOB[i,t] * (1-em[t])
ps[1,i,t,4] <- (1-sO[i,t]) * r * (1-em[t])
ps[1,i,t,5] <- (1-sO[i,t]) * (1-r) * (1-em[t])
ps[1,i,t,6] <- sO[i,t] * em[t]
ps[1,i,t,7] <- (1-sO[i,t]) * (1-r) * em[t]

ps[2,i,t,1] <- 0
ps[2,i,t,2] <- sA[i,t] * (1-psiAB[i,t]) * (1-em[t])
ps[2,i,t,3] <- sA[i,t] * psiAB[i,t] * (1-em[t])
ps[2,i,t,4] <- (1-sA[i,t]) * r * (1-em[t])
ps[2,i,t,5] <- (1-sA[i,t]) * (1-r) * (1-em[t])
ps[2,i,t,6] <- sA[i,t] * em[t]
ps[2,i,t,7] <- (1-sA[i,t]) * (1-r) * em[t]

ps[3,i,t,1] <- 0
ps[3,i,t,2] <- sB[i,t] * psiBA[i,t] * (1-em[t])
```

```
    ps[3,i,t,3] <- sB[i,t] * (1-psiBA[i,t]) * (1-em[t])
    ps[3,i,t,4] <- (1-sB[i,t]) * r * (1-em[t])
    ps[3,i,t,5] <- (1-sB[i,t]) * (1-r) * (1-em[t])
    ps[3,i,t,6] <- sB[i,t] * em[t]
    ps[3,i,t,7] <- (1-sB[i,t]) * (1-r) * em[t]

    ps[4,i,t,1] <- 0
    ps[4,i,t,2] <- 0
    ps[4,i,t,3] <- 0
    ps[4,i,t,4] <- 0
    ps[4,i,t,5] <- 1
    ps[4,i,t,6] <- 0
    ps[4,i,t,7] <- 0

    ps[5,i,t,1] <- 0
    ps[5,i,t,2] <- 0
    ps[5,i,t,3] <- 0
    ps[5,i,t,4] <- 0
    ps[5,i,t,5] <- 1
    ps[5,i,t,6] <- 0
    ps[5,i,t,7] <- 0

    ps[6,i,t,1] <- 0
    ps[6,i,t,2] <- 0
    ps[6,i,t,3] <- 0
    ps[6,i,t,4] <- 0
    ps[6,i,t,5] <- 0
    ps[6,i,t,6] <- 1
    ps[6,i,t,7] <- 0

    ps[7,i,t,1] <- 0
    ps[7,i,t,2] <- 0
    ps[7,i,t,3] <- 0
    ps[7,i,t,4] <- 0
    ps[7,i,t,5] <- 0
    ps[7,i,t,6] <- 0
    ps[7,i,t,7] <- 1

    # Define probabilities of O(t) given S(t)
    po[1,i,t,1] <- 1
    po[1,i,t,2] <- 0
    po[1,i,t,3] <- 0
    po[1,i,t,4] <- 0
    po[1,i,t,5] <- 0

    po[2,i,t,1] <- 0
    po[2,i,t,2] <- pA[i,t]
    po[2,i,t,3] <- 0
    po[2,i,t,4] <- 0
    po[2,i,t,5] <- 1-pA[i,t]

    po[3,i,t,1] <- 0
    po[3,i,t,2] <- 0
```

```
    po[3,i,t,3] <- pB[i,t]
    po[3,i,t,4] <- 0
    po[3,i,t,5] <- 1-pB[i,t]

    po[4,i,t,1] <- 0
    po[4,i,t,2] <- 0
    po[4,i,t,3] <- 0
    po[4,i,t,4] <- 1
    po[4,i,t,5] <- 0

    po[5,i,t,1] <- 0
    po[5,i,t,2] <- 0
    po[5,i,t,3] <- 0
    po[5,i,t,4] <- 0
    po[5,i,t,5] <- 1

    po[6,i,t,1] <- 0
    po[6,i,t,2] <- 0
    po[6,i,t,3] <- 0
    po[6,i,t,4] <- 0
    po[6,i,t,5] <- 1

    po[7,i,t,1] <- 0
    po[7,i,t,2] <- 0
    po[7,i,t,3] <- 0
    po[7,i,t,4] <- 0
    po[7,i,t,5] <- 1
    } #t
    } #i

    # Likelihood
    for (i in 1:nind){
    # Define latent state at first capture
    z[i,first[i]] <- y[i,first[i]]
    for (t in (first[i]+1):n.yr){
    # State process: draw S(t) given S(t-1)
    z[i,t] ~ dcat(ps[z[i,t-1], i, t-1, 1:7])
    # Observation process: draw O(t) given S(t)
    y[i,t] ~ dcat(po[z[i,t], i, t-1, 1:5])
    } #t
    } #i
    } #model
    ",fill = TRUE)
sink()

# Initial values
get.first <- function(x) min(which(x!=5))
f <- apply(datl$y, 1, get.first)
get.last <- function(x) max(which(x!=5))
l <- apply(datl$y, 1, get.last)
TFmat <- is.na(z.inits) & is.na(datl$z)
for (i in 1:dim(TFmat)[1]){  TFmat[i,1:f[i]] <- FALSE }
z.inits[TFmat] <- sample(size=445, c(2,3), replace=T, prob=c(0.5, 0.5) )
```

```
inits <- function(){list(z = z.inits)}

params <- c(
  "r", "l.mu.em", "mu.em", "em",
  "OSalpha", "ASalpha", "BSalpha", "OBRalpha", "ABRalpha", "BARalpha",  "mu.pA", "mu.pB",
  "OSalpha1", "ASalpha1", "BSalpha1","OBRalpha1", "ABRalpha1", "BARalpha1","mu.pA1", "mu.pB1",
  "eps.OS.s", "eps.AS.s", "eps.BS.s", "eps.OBR.psi", "eps.ABR.psi", "eps.BAR.psi", "eps.pA", "eps.pB",
  "eta.OSalpha", "eta.ASalpha", "eta.BSalpha", "eta.OBRalpha", "eta.ABRalpha", "eta.BARalpha", "eta.pA"
  "sigma.OS.s", "sigma.AS.s", "sigma.BS.s", "sigma.OBR.psi", "sigma.ABR.psi", "sigma.BAR.psi", "sigma.p
)

# MCMC settings
ni <- 200000; nt <- 100; nb <- 100000; nc <- 3; na <- 10000 # actual run # takes about 1 week on a high
out <- jags(datl, inits, params, modfl,
            n.chains = nc, n.thin = nt, n.burnin = nb, n.adapt=na, n.iter=ni,
            parallel=T, module=c("glm", "bugs"))
# save(file=paste("./", m, ".Rdata", sep=""), list="out")
```

1.4.2 Postprocess Global Model

We postprocess the output from JAGS to examine differences between sex, hacked, and effort estimates for probabilities of survival, recruitment, resight, and estimate emigration.

```
library (jagsUI)
```

1.4.2.1 Functions to Summarize Posterior Distributions

```
## Warning: package 'jagsUI' was built under R version 4.0.5
```

```
library (ggplot2)
library (gridExtra)

# Function to compute highest density interval. From Kruschke 2011.
HDIofMCMC = function( sampleVec , credMass=0.95 ) {
  sortedPts = sort( sampleVec )
  ciIdxInc = floor( credMass * length( sortedPts ) )
  nCIs = length( sortedPts ) - ciIdxInc
  ciWidth = rep( 0 , nCIs )
  for ( i in 1:nCIs ) {
    ciWidth[ i ] = sortedPts[ i + ciIdxInc ] - sortedPts[ i ]
  }
  HDImin = sortedPts[ which.min( ciWidth ) ]
  HDImax = sortedPts[ which.min( ciWidth ) + ciIdxInc ]
  HDIlim = c( HDImin , HDImax )
  return( HDIlim )
}

# Function to compute summary stats over posteriors
data_summary <- function(x,...) {
  m <- median(x)
  ymin <- HDIofMCMC(x, credMass=0.95)[[1]]
  ymax <- HDIofMCMC(x, credMass=0.95)[[2]]
  return(c(y=m,ymin=ymin,ymax=ymax))
```

```
}

data_summary2 <- function(x,...) {
  m <- median(x)
  ymin <- HDIofMCMC(x, credMass=0.85)[[1]]
  ymax <- HDIofMCMC(x, credMass=0.85)[[2]]
  return(c(y=m,ymin=ymin,ymax=ymax))
}
```

1.4.2.2 Calculate Differences Between Groups We load the results from the global CJS model to assess which covariates had important effects on parameters.

```
load(".\\outputs\\surv7-em-global.Rdata")


O.surv <- O.rec <- A.surv <- A.rec <- B.surv <- B.rec <- array(NA, dim=c(dim(out$sims.list$OSalpha1)[1]
A.resight <- B.resight <-array(NA, dim=c(dim(out$sims.list$mu.pA1)[1],3))

############################
# First-year (O) capital letter o, rather than zero so code works
############################
# Survival
# wild - hacked
O.surv[,1]<- (out$sims.list$OSalpha1[,1,1] +         out$sims.list$OSalpha1[,2,1])/2 -
  (out$sims.list$OSalpha1[,1,2] + out$sims.list$OSalpha1[,2,2])/2
# female - male
O.surv[,2]<- (out$sims.list$OSalpha1[,1,1] + out$sims.list$OSalpha1[,1,2])/2 -
  (out$sims.list$OSalpha1[,2,1] + out$sims.list$OSalpha1[,2,2])/2
# Recruitment
# wild - hacked
O.rec[,1]<- (out$sims.list$OBRalpha1[,1,1] + out$sims.list$OBRalpha1[,2,1])/2 -
  (out$sims.list$OBRalpha1[,1,2] + out$sims.list$OBRalpha1[,2,2])/2
# female - male
O.rec[,2]<- (out$sims.list$OBRalpha1[,1,1] + out$sims.list$OBRalpha1[,1,2])/2 -
  (out$sims.list$OBRalpha1[,2,1] + out$sims.list$OBRalpha1[,2,2])/2

############################
# Adult Nonbreeders (A)
############################
# Survival
# wild - hacked
A.surv[,1]<- (out$sims.list$ASalpha1[,1,1] + out$sims.list$ASalpha1[,2,1])/2 -
  (out$sims.list$ASalpha1[,1,2] + out$sims.list$ASalpha1[,2,2])/2
# female - male
A.surv[,2]<- (out$sims.list$ASalpha1[,1,1] + out$sims.list$ASalpha1[,1,2])/2 -
  (out$sims.list$ASalpha1[,2,1] + out$sims.list$ASalpha1[,2,2])/2
# Recruitment
# wild - hacked
A.rec[,1]<- (out$sims.list$ABRalpha1[,1,1] + out$sims.list$ABRalpha1[,2,1])/2 -
  (out$sims.list$ABRalpha1[,1,2] + out$sims.list$ABRalpha1[,2,2])/2
# female - male
A.rec[,2]<- (out$sims.list$ABRalpha1[,1,1] + out$sims.list$ABRalpha1[,1,2])/2 -
  (out$sims.list$ABRalpha1[,2,1] + out$sims.list$ABRalpha1[,2,2])/2
# Resight prob
```

```
# wild-hacked
A.resight[,1]<- (out$sims.list$mu.pA1[,1,1,1] + out$sims.list$mu.pA1[,1,1,2] + out$sims.list$mu.pA1[,2,
              (out$sims.list$mu.pA1[,1,2,1] + out$sims.list$mu.pA1[,1,2,2]+
                out$sims.list$mu.pA1[,2,2,1] + out$sims.list$mu.pA1[,2,2,2])/4
# female - male
A.resight[,2]<- (out$sims.list$mu.pA1[,1,1,1] + out$sims.list$mu.pA1[,1,1,2]+
                 out$sims.list$mu.pA1[,1,2,1] + out$sims.list$mu.pA1[,1,2,2])/4 -
              (out$sims.list$mu.pA1[,2,1,1] + out$sims.list$mu.pA1[,2,1,2]+
                 out$sims.list$mu.pA1[,2,2,1] + out$sims.list$mu.pA1[,2,2,2])/4

# high effort - low effort
A.resight[,3]<- (out$sims.list$mu.pA1[,1,1,2] + out$sims.list$mu.pA1[,2,1,2]+
                 out$sims.list$mu.pA1[,1,2,2] + out$sims.list$mu.pA1[,2,2,2])/4 -
              (out$sims.list$mu.pA1[,1,1,1] + out$sims.list$mu.pA1[,2,1,1]+
                out$sims.list$mu.pA1[,1,2,1] + out$sims.list$mu.pA1[,2,2,1])/4

##########################
# Adult Breeders (B)
##########################
# Survival
# wild - hacked
B.surv[,1]<- (out$sims.list$BSalpha1[,1,1] + out$sims.list$BSalpha1[,2,1])/2 -
  (out$sims.list$BSalpha1[,1,2] + out$sims.list$BSalpha1[,2,2])/2
# female - male
B.surv[,2]<- (out$sims.list$BSalpha1[,1,1] + out$sims.list$BSalpha1[,1,2])/2 -
  (out$sims.list$BSalpha1[,2,1] + out$sims.list$BSalpha1[,2,2])/2
# Recruitment
# wild - hacked
B.rec[,1]<- (out$sims.list$BARalpha1[,1,1] + out$sims.list$BARalpha1[,2,1])/2 -
  (out$sims.list$BARalpha1[,1,2] + out$sims.list$BARalpha1[,2,2])/2
# female - male
B.rec[,2]<- (out$sims.list$BARalpha1[,1,1] + out$sims.list$BARalpha1[,1,2])/2 -
  (out$sims.list$BARalpha1[,2,1] + out$sims.list$BARalpha1[,2,2])/2
# resight prob
# wild-hacked
B.resight[,1]<- (out$sims.list$mu.pB1[,1,1,1] + out$sims.list$mu.pB1[,1,1,2] + out$sims.list$mu.pB1[,2,
              (out$sims.list$mu.pB1[,1,2,1] + out$sims.list$mu.pB1[,1,2,2]+
                out$sims.list$mu.pB1[,2,2,1] + out$sims.list$mu.pB1[,2,2,2])/4
# female - male
B.resight[,2]<- (out$sims.list$mu.pB1[,1,1,1] + out$sims.list$mu.pB1[,1,1,2]+
                 out$sims.list$mu.pB1[,1,2,1] + out$sims.list$mu.pB1[,1,2,2])/4 -
              (out$sims.list$mu.pB1[,2,1,1] + out$sims.list$mu.pB1[,2,1,2]+
                 out$sims.list$mu.pB1[,2,2,1] + out$sims.list$mu.pB1[,2,2,2])/4

# high effort - low effort
B.resight[,3]<- (out$sims.list$mu.pB1[,1,1,2] + out$sims.list$mu.pB1[,2,1,2]+
                 out$sims.list$mu.pB1[,1,2,2] + out$sims.list$mu.pB1[,2,2,2])/4 -
              (out$sims.list$mu.pB1[,1,1,1] + out$sims.list$mu.pB1[,2,1,1]+
                out$sims.list$mu.pB1[,1,2,1] + out$sims.list$mu.pB1[,2,2,1])/4

stats <- data.frame(pval=NA, lci=NA, uci=NA)

# combine survival posteriors
```

```r
temp.df<- data.frame(Draws=O.surv[,1], Cat="Hacked", State="First-year")
temp.df1<- data.frame(Draws=O.surv[,2], Cat="Sex", State="First-year")
temp.df2<- data.frame(Draws=B.surv[,1], Cat="Hacked", State="Breeder")
temp.df3<- data.frame(Draws=B.surv[,2], Cat="Sex", State="Breeder")
temp.df4<- data.frame(Draws=A.surv[,1], Cat="Hacked", State="Non-breeder")
temp.df5<- data.frame(Draws=A.surv[,2], Cat="Sex", State="Non-breeder")
df.surv<- rbind(temp.df, temp.df1, temp.df2, temp.df3, temp.df4, temp.df5)
df.surv$combined<- factor(paste(df.surv$State, df.surv$Cat))
df.surv$combined <- factor(df.surv$combined, levels=levels(df.surv$combined)[c(3,4,5,6,1,2)])

# combine recruitment posteriors
temp.df<- data.frame(Draws=O.rec[,1], Cat="Hacked", State="First-year")
temp.df1<- data.frame(Draws=O.rec[,2], Cat="Sex", State="First-year")
temp.df2<- data.frame(Draws=B.rec[,1], Cat="Hacked", State="Breeder")
temp.df3<- data.frame(Draws=B.rec[,2], Cat="Sex", State="Breeder")
temp.df4<- data.frame(Draws=A.rec[,1], Cat="Hacked", State="Non-breeder")
temp.df5<- data.frame(Draws=A.rec[,2], Cat="Sex", State="Non-breeder")
df.rec<- rbind(temp.df, temp.df1, temp.df2, temp.df3, temp.df4, temp.df5)
df.rec$combined<- factor(paste(df.rec$State, df.rec$Cat))
df.rec$combined <- factor(df.rec$combined, levels=levels(df.rec$combined)[c(3,4,5,6,1,2)])

# combine resight posteriors
temp.df2<- data.frame(Draws=B.resight[,1], Cat="Hacked", State="Breeder")
temp.df3<- data.frame(Draws=B.resight[,2], Cat="Sex", State="Breeder")
temp.df4<- data.frame(Draws=B.resight[,3], Cat="Effort", State="Breeder")
temp.df5<- data.frame(Draws=A.resight[,1], Cat="Hacked", State="Non-breeder")
temp.df6<- data.frame(Draws=A.resight[,2], Cat="Sex", State="Non-breeder")
temp.df7<- data.frame(Draws=A.resight[,3], Cat="Effort", State="Non-breeder")
df.resight<- rbind(temp.df2, temp.df3, temp.df4, temp.df5, temp.df6, temp.df7)
df.resight$combined<- factor(paste(df.resight$State, df.resight$Cat))
df.resight$combined <- factor(df.resight$combined, levels=c(levels(df.resight$combined)[c(4,5,6,1,2,3)]
# print median and 95% HDIs
ests <- data.frame(round(rbind(
  do.call(rbind, tapply(df.surv$Draws, df.surv$combined, data_summary)),
  do.call(rbind, tapply(df.rec$Draws, df.rec$combined, data_summary)),
  do.call(rbind, tapply(df.resight$Draws, df.resight$combined, data_summary))
),3))
ests2 <- data.frame(round(rbind(
  do.call(rbind, tapply(df.surv$Draws, df.surv$combined, data_summary2)),
  do.call(rbind, tapply(df.rec$Draws, df.rec$combined, data_summary2)),
  do.call(rbind, tapply(df.resight$Draws, df.resight$combined, data_summary2))
),3))
ests <- cbind(ests, ests2[, c(2,3)])
colnames(ests) <- c("median", "LHDI_95", "UHDI_95", "LHDI_85", "UHDI_85")
ests <- cbind(c(rep("survival", 6), rep("recruitment", 6), rep("resight", 6) ), ests)
colnames(ests)[1] <- "param"
ests$imp95 <- ifelse((ests$LHDI_95 >= 0 & ests$UHDI_95>=0) |                    (ests$LHDI_95 <= 0 & ests
ests$imp85 <- ifelse((ests$LHDI_85 >= 0 & ests$UHDI_85>=0) |                    (ests$LHDI_85 <= 0 & ests
```

```r
library (knitr)
knitr::kable(ests, caption="Table S1. Estimates of differences between groups for survival, recruitment
```

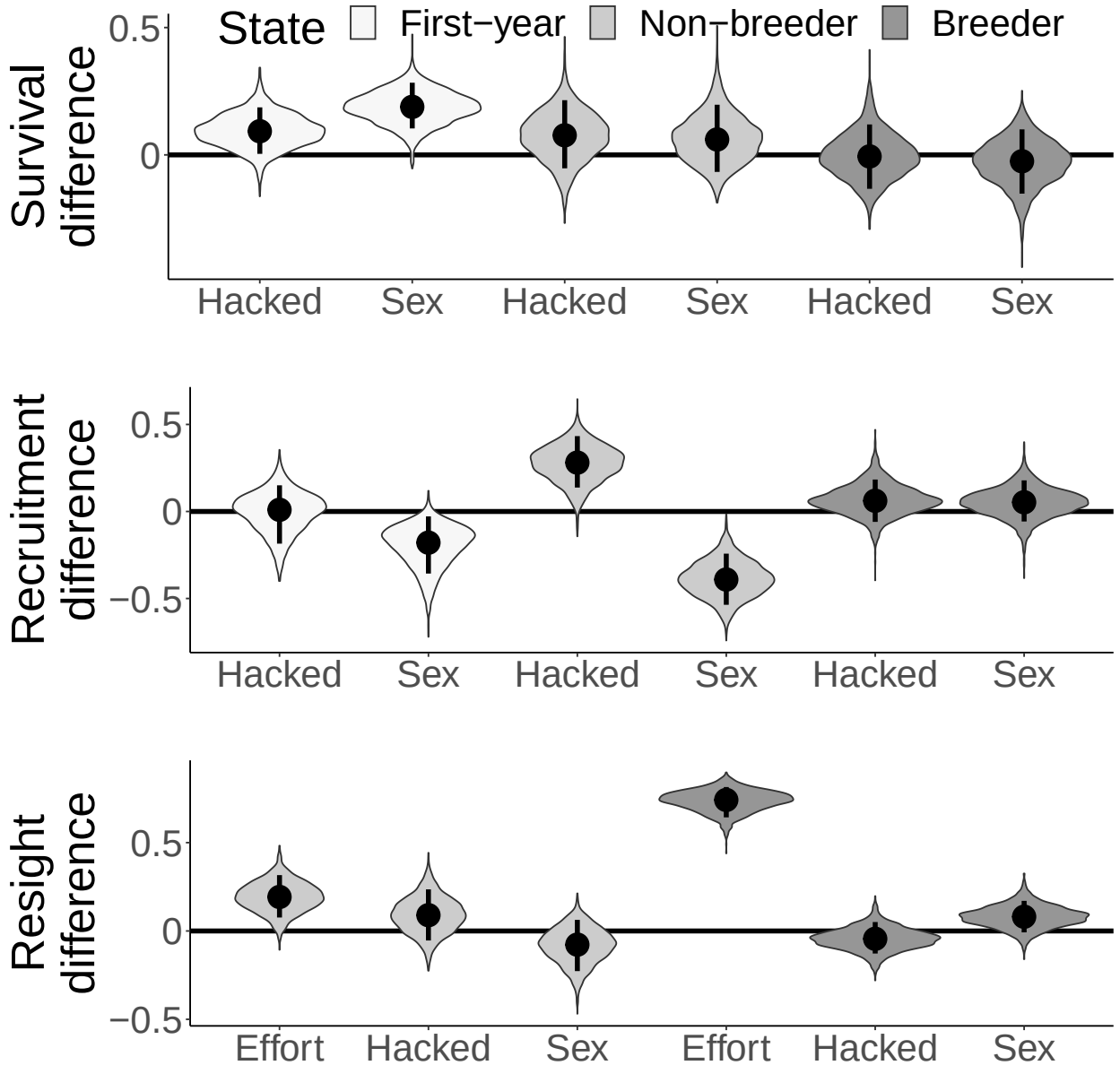
1.4.2.3 The Importance of Covariates in the CJS Model with Global Constraints

Table 1: Table S1. Estimates of differences between groups for survival, recruitment, and resight probabilities. We present medians, 95% HDIs, and 85% HDIs. Differences were considered potentially important when 85% highest density intervals did not intersect zero.

	param	median	LHDI_95	UHDI_95	LHDI_85	UHDI_85	imp95	imp85
First.year.Hacked	survival	0.093	-0.030	0.227	0.004	0.187	no	yes
First.year.Sex	survival	0.189	0.066	0.312	0.104	0.283	yes	yes
Non.breeder.Hacked	survival	0.077	-0.121	0.249	-0.053	0.215	no	no
Non.breeder.Sex	survival	0.061	-0.117	0.249	-0.067	0.197	no	no
Breeder.Hacked	survival	-0.006	-0.181	0.174	-0.133	0.120	no	no
Breeder.Sex	survival	-0.025	-0.207	0.147	-0.152	0.100	no	no
First.year.Hacked.1	recruitment	0.010	-0.249	0.222	-0.185	0.150	no	no
First.year.Sex.1	recruitment	-0.179	-0.468	-0.008	-0.357	-0.028	yes	yes
Non.breeder.Hacked.1	recruitment	0.281	0.067	0.475	0.138	0.433	yes	yes
Non.breeder.Sex.1	recruitment	-0.391	-0.611	-0.200	-0.536	-0.242	yes	yes
Breeder.Hacked.1	recruitment	0.062	-0.103	0.253	-0.060	0.184	no	no
Breeder.Sex.1	recruitment	0.053	-0.138	0.207	-0.058	0.179	no	no
Non.breeder.Effort	resight	0.193	0.019	0.353	0.076	0.316	yes	yes
Non.breeder.Hacked.2	resight	0.089	-0.109	0.289	-0.054	0.236	no	no
Non.breeder.Sex.2	resight	-0.077	-0.273	0.109	-0.227	0.063	no	no
Breeder.Effort	resight	0.742	0.621	0.856	0.643	0.815	yes	yes
Breeder.Hacked.2	resight	-0.044	-0.180	0.076	-0.128	0.051	no	no
Breeder.Sex.2	resight	0.081	-0.039	0.214	-0.008	0.170	no	no

```
txt <- 30
lwd <- 1.5
df.surv$State <- factor(df.surv$State, levels=c("First-year", "Non-breeder", "Breeder"))

ps <- ggplot(df.surv, aes(x = combined, y = Draws, group=combined )) +
  scale_y_continuous(breaks=c(-0.5, 0, 0.5),  labels=c(-0.5, 0, 0.5)) +
  geom_hline(yintercept=0, linetype="solid", size=lwd) +
  geom_violin(aes(fill=State)) + scale_fill_manual(values=c("First-year"="#f7f7f7",
    "Non-breeder"="#cccccc", "Breeder"="#969696")) +
  stat_summary(fun.data=data_summary2,  geom="pointrange", size=lwd) +
  theme_classic() + theme (text = element_text(size=txt)) +
  ylab("Survival\ndifference") + xlab ("") +
  scale_x_discrete( labels=c("Hacked", "Sex", "Hacked", "Sex","Hacked", "Sex" ))

pt <- ggplot(df.rec, aes(x = combined, y = Draws, group=combined )) +
  scale_y_continuous(breaks=c(-0.5, 0, 0.5),  labels=c(-0.5, 0, 0.5)) +
  geom_hline(yintercept=0, linetype="solid", size=lwd) +
  geom_violin(aes(fill=State)) +  scale_fill_manual(values=c("First-year"="#f7f7f7",
    "Non-breeder"="#cccccc", "Breeder"="#969696")) +
  stat_summary(fun.data=data_summary2,  geom="pointrange", size=lwd) +
  theme_classic() + theme (text = element_text(size=txt)) +
  ylab("Recruitment\ndifference") + xlab ("") +
  scale_x_discrete( labels=c("Hacked", "Sex", "Hacked", "Sex","Hacked", "Sex" ))

pr <- ggplot(df.resight, aes(x = combined, y = Draws, group=combined)) +
```

```
  scale_y_continuous(breaks=c(-0.5, 0, 0.5),  labels=c(-0.5, 0, 0.5)) +
  geom_hline(yintercept=0, linetype="solid", size=lwd) +
  geom_violin(aes(fill=State)) +  scale_fill_manual(values=c("First-year"="#f7f7f7",
    "Non-breeder"="#cccccc", "Breeder"="#969696")) +
  stat_summary(fun.data=data_summary2,  geom="pointrange", size=lwd) +
  theme_classic() + theme (text = element_text(size=txt)) +
  ylab("Resight\ndifference") + xlab ("") +
  scale_x_discrete( labels=c("Effort", "Hacked", "Sex", "Effort", "Hacked", "Sex" ))

grid.arrange(ps + theme(legend.position=c(0.5,0.95), legend.direction="horizontal",legend.background = e
             pt + theme(legend.position="none"),
             pr + theme(legend.position="none"),
             nrow=3,
             layout_matrix = rbind(c(1, 1, 1, 1, 1, 1),
                                   c(2, 2, 2, 2, 2, 2),
                                   c(3, 3, 3, 3, 3, 3)))
```



1.4.2.4 The Importance of Emigration

```
hist(out$sims.list$em, main="Histogram of Emigration", xlab="Posterior draws of emigration")
abline(v=median(out$sims.list$em), lwd=2)
abline(v=HDIofMCMC(out$sims.list$em, cred=0.95), lty=2, lwd=2)
```

```
median(out$sims.list$em)
```

```
## [1] 0.0008555531
```

```
HDIofMCMC(out$sims.list$em, cred=0.95)
```

```
## [1] 8.894633e-07 2.669040e-03
```

```
# plot prior of dunif(0,1) and posterior density of emigration
x <- seq(-0.5, 1.5, length=1000)
y <- dunif(x, min = 0, max = 1)
```

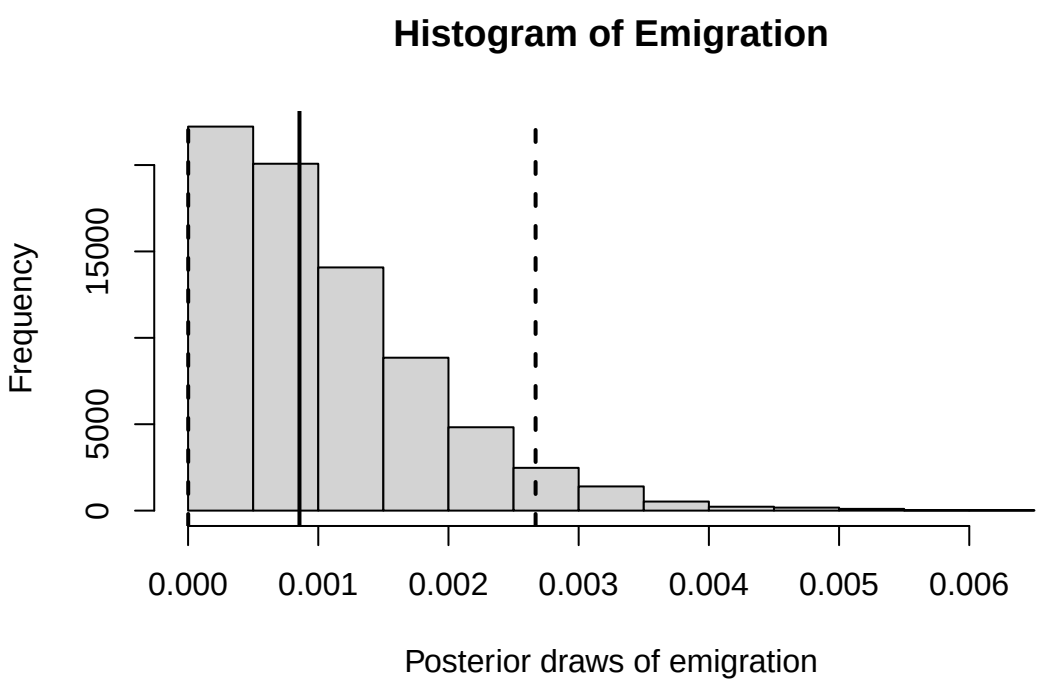


Figure 1: Fig. S2. Histogram of emigration. Median (solid line) and 95% HDIs (dashed line) are depicted.

```
par(mar=c(5,5,2,5))
plot(x, y, type="l", lwd=2, xlim=c(0,1.1),
     ylab="Prior density (black line)\n Uniform(0,1)",
     xlab=expression(paste("Emigration rate (",delta,")", sep="")),
     main="")
par(new=TRUE)
plot(density(out$sims.list$em), type="l", lwd=2, xlim=c(0,1.1),
     main="", col="blue",
     axes = FALSE, xlab = "", ylab = "")
axis(side = 4, at = c(0,350,700))      # Add second axis
mtext("Posterior density (blue line)", side = 4, line = 3)
```

1.5 Implementation of the Integrated Population Model with Immigration

We specify the IPM including observation and state-transition matrices for the CJS survival submodel provided within the manuscript. These matrices are identical for IPMs with and without immigration.

1.5.1 Formulation of the IPM with Immigration

This IPM specifies immigration as a rate (Abadi et al. 2010). We estimate this immigration rate (ω) for non-breeders and breeders separately. The number of immigrant breeders (N^{IB}_{t+1}) was specified as

$$N^{IB}_{t+1} \sim \text{Poisson}(N^{B}_{t}\omega^{B}_{t})$$

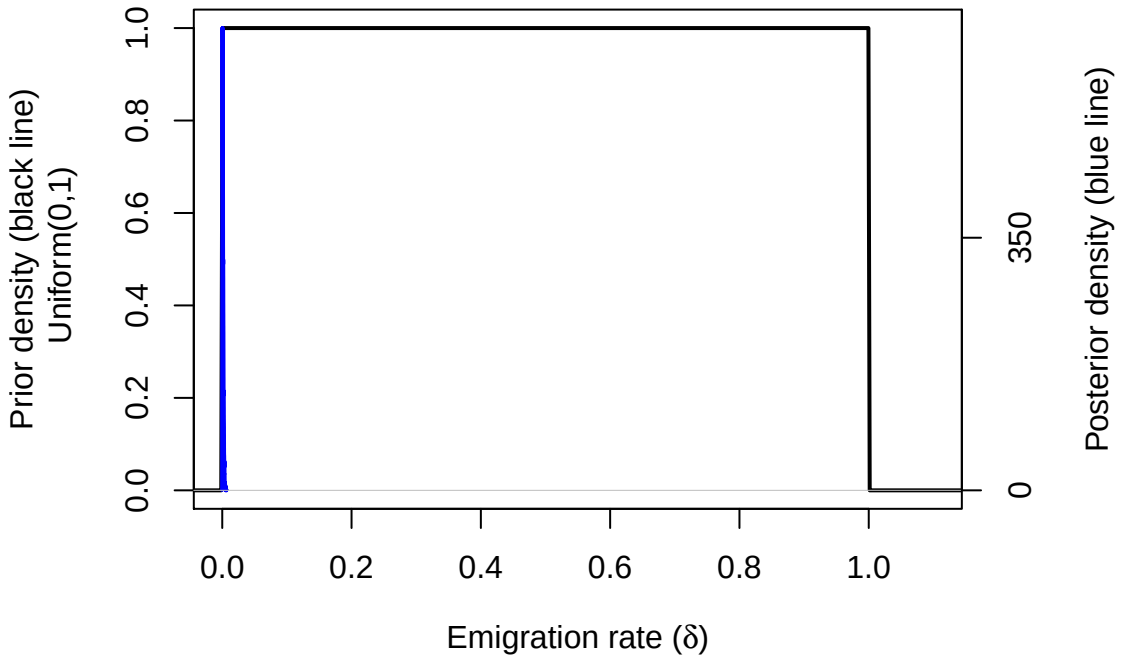


Figure 2: Fig. S3. Prior and posterior densities of emigration.

We assigned the following prior to immigration rates:

$$\omega_t^B = \mu^{IB} + \epsilon_t$$

$$\mu^{IB} \sim \text{halfnormal}(mean = 0, sd = 1)$$

$$\epsilon_t \sim normal(0, \sigma^{IB})$$

A similar parameterization was used for non-breeder immigration. We did not include immigration of first-year birds because they must be recruited as breeders or non-breeders during the following post-breeding survey.

1.5.2 JAGS and R Code for the IPM with Immigration

```
################################
# The model
################################
library (jagsUI)
load("/scratch/brolek/aplo_ipm/data/final-data.Rdata")
datl$pp <- c(0, 0, tapply(datl$prod, datl$year.p, sum, na.rm=T), 42)
datl$pp[c(15,17)] <- NA
datl$countOM <- datl$countJM-datl$aug

m<- c("ipm-ie")
modfl <- paste(".\\", m, ".txt", sep="")
sink(modfl)
cat("
    model{
```

```
#####################################################
#####################################################
# Mark-resight-recovery data
#   Observations (po) = y
#     1 seen first year (age 0)
#     2 seen nonbreeder
#     3 seen breeder
#     4 seen dead
#     5 not seen
#   States (ps)
#     1 alive first year
#     2 alive nonbreeder
#     3 alive breeder
#     4 Recovered recently dead
#     5 Dead not recovered/long dead
#     6 Emigrated and alive
#     7 Emigrated and dead
#   Groups
#     1 wild born
#     2 hacked
#   Sex
#     1 female
#     2 male
#   Effort
#     1 low
#     2 high
#####################################################
# PARAMETERS
#   sO: survival probability first year
#       (this is the letter 'O' rather than zero so jags can parse)
#   sA: survival probability nonbreeders
#   sB: survival probability breeders
#   psiOB: recruitment probability from first-year to breeder
#   psiAB: recruitment probability from nonbreeders to breeder
#   psiBA: recruitment probability from breeder to nonbreeders
#   pA: resight probability nonbreeders
#   pB: resight probability breeder
#   F: fecundity
#   omega: immigration
#####################################################
# Priors and constraints
#####################################################
# Counts for first year are known with certainty
N[1,1] <- 0 # Wild born first-year
N[2,1] <- 0 # first-year hacked to nonbreeder adults
N[3,1] <- 0 # First-year wild to nonbreeder
N[4,1] <- 0  # First-year hacked to breeders
N[5,1] <- 0  # First-year wild to breeders
N[6,1] <- 0  # Nonbreeder wild to nonbreeder
N[7,1] <- 0  # Nonbreeder wild to breeders
N[8,1] <- 0  # Breeders to nonbreeder
N[9,1] <- 0  # Breeders to breeders
N[10,1] <- 0  # Nonbreeder hacked to breeder
```

```
N[11,1] <- 0  # Nonbreeder hacked to nonbreeder
N[12,1] <- 0  # Immigrant to breeder
N[13,1] <- 0  # Immigrant to nonbreeder
N[14,1] <- 0 # breeder to emigrant
N[15,1] <- 0 # nonbreeder to emigrant
N[16,1] <- 0 # first-year wild to emigrant
N[17,1] <- 0 # first-year hacked to emigrant
F[1] <- 0 # No breeding during the first year, all young hacked birds

rNB ~ dunif(0,50)
omegaB1 ~ dnorm(0, 1/(1*1) )T(0,) # upper limit near 5 imm per breeder to help run model
omegaA1 ~ dnorm(0, 1/(1*1) )T(0,)
sigma.omegaB ~ dnorm(0, 1/(2*2) )T(0,)
sigma.omegaA ~ dnorm(0, 1/(2*2) )T(0,)
for (m in 1:2){ mu.F[m] ~ dunif(0,5) } # m # limits to help run model
sigma.F ~ dnorm(0, 1/(2*2) )T(0,)
mu.em <- logit(mu.em1)
mu.em1 ~ dunif(0,1)
sigma.em ~ dnorm(0, 1/(2*2) )T(0,)
r ~ dunif(0,1)

# Survival loops for demographic categories by sex, hacked, effort

sigma.AS.s ~ dnorm(0, 1/(2*2) )T(0,)
ASalpha <- logit(ASalpha1)
ASalpha1 ~ dunif(0, 1)

sigma.BS.s ~ dnorm(0, 1/(2*2) )T(0,)
BSalpha <- logit(BSalpha1)
BSalpha1 ~ dunif(0, 1)

sigma.BAR.psi ~ dnorm(0, 1/(2*2) )T(0,)
BARalpha <- logit(BARalpha1)
BARalpha1 ~ dunif(0, 1)

sigma.pB ~ dnorm(0, 1/(2*2) )T(0,)

for (k in 1:2){
mu.pB[k]<- logit(mu.pB1[k])
mu.pB1[k] ~ dunif(0, 1)
} # k

for (t in 1:(n.yr-1)){
logit(eta.ASalpha[t]) <- ASalpha + eps.AS.phi[t]
eps.AS.phi[t] ~ dnorm(0, 1/(sigma.AS.s*sigma.AS.s) )

logit(eta.BSalpha[t]) <- BSalpha + eps.BS.phi[t]
eps.BS.phi[t] ~ dnorm(0, 1/(sigma.BS.s*sigma.BS.s) )

logit(eta.BARalpha[t]) <- BARalpha + eps.BAR.psi[t]
eps.BAR.psi[t] ~ dnorm(0, 1/(sigma.BAR.psi*sigma.BAR.psi))

logit(eta.pB[t]) <- mu.pB[effort[t]] + eps.pB[t]
```

```
eps.pB[t] ~ dnorm(0, 1/(sigma.pB*sigma.pB) )
} #t

for(s in 1:2){

sigma.OBR.psi[s] ~ dnorm(0, 1/(2*2) )T(0,)
OBRalpha[s] <- logit(OBRalpha1[s])
OBRalpha1[s] ~ dunif(0, 1)

for (t in 1:(n.yr-1)){
logit(eta.OBRalpha[s,t]) <- OBRalpha[s] + eps.OBR.psi[s,t]
eps.OBR.psi[s,t] ~ dnorm(0, 1/(sigma.OBR.psi[s]*sigma.OBR.psi[s]) )
} #t

for (h in 1:2){
sigma.ABR.psi[s,h] ~ dnorm(0, 1/(2*2) )T(0,)
ABRalpha[s,h] <- logit(ABRalpha1[s,h])
ABRalpha1[s,h] ~ dunif(0, 1)

sigma.OS.s[s,h] ~ dnorm(0, 1/(2*2) )T(0,)
OSalpha[s,h] <- logit(OSalpha1[s,h])
OSalpha1[s,h] ~ dunif(0, 1)

for (t in 1:(n.yr-1)){
logit(eta.ABRalpha[s,h,t]) <- ABRalpha[s,h] + eps.ABR.psi[s,h,t]
eps.ABR.psi[s,h,t] ~ dnorm(0, 1/(sigma.ABR.psi[s,h]*sigma.ABR.psi[s,h]) )

logit(eta.OSalpha[s,h,t]) <- OSalpha[s,h] + eps.OS.s[s,h,t]
eps.OS.s[s,h,t] ~ dnorm(0, 1/(sigma.OS.s[s,h]*sigma.OS.s[s,h]) )
} #t
} } #h #s

for (h in 1:2){
sigma.pA[h] ~ dnorm(0, 1/(2*2) )T(0,)
for (k in 1:2){
mu.pA[k,h] <- logit(mu.pA1[k,h])
mu.pA1[k,h] ~ dunif(0,1)
} # k

for (t in 1:(n.yr-1)){
logit(eta.pA[h,t]) <- mu.pA[effort[t], h] + eps.pA[h,t]
eps.pA[h,t] ~ dnorm(0, 1/(sigma.pA[h]*sigma.pA[h]) )
}} #t #h

########################
# Derived params
########################
for (t in 3:(n.yr-1)){
lambda[t] <-  Ntot[t+1]/(Ntot[t])
loglambda[t] <- log(lambda[t])
} #t

################################
```

```
# Likelihood for productivity
################################
for (t in 2:n.yr){
pp[t] ~ dpois( F[t]*NB[t] )
log(F[t]) <- log(mu.F[manage[t]]) + eps.F[t]
eps.F[t] ~ dnorm (0, 1/(sigma.F*sigma.F) )
}
# GOF fecundity
for (t in 1:n.yr){
J.rep[t] ~ dpois( F[t]*NB[t] )
J.exp[t] <- F[t]*NB[t]
d.obs[t] <- pp[t]* log((pp[t]+0.001)/(J.exp[t]+0.001)) - (pp[t]-J.exp[t])
d.rep[t] <- J.rep[t]*log((J.rep[t]+0.001)/(J.exp[t]+0.001)) - (J.rep[t]-J.exp[t])
} # t
dd.obs <- sum(d.obs)
tvm.obs <- sd(pp)^2/mean(pp)
dd.rep <- sum(d.rep)
tvm.rep <- sd(J.rep)^2/mean(J.rep)

################################
# Likelihood for immigration
################################
for (t in 1:(n.yr-1)){
# omegaB[t] <- 0 # set to zero, no breeder immigration
# omegaA[t] <- 0 # set to zero, no nonbreeder immigration
log(omegaB[t]) <-  log(omegaB1) + eps.omegaB[t] # breeder
log(omegaA[t]) <- log(omegaA1) + eps.omegaA[t] # nonbreeder
eps.omegaB[t] ~ dnorm(0, 1/(sigma.omegaB*sigma.omegaB))
eps.omegaA[t] ~ dnorm(0, 1/(sigma.omegaA*sigma.omegaA))
logit(em[t]) <- mu.em + eps.em[t]
eps.em[t] ~ dnorm(0, 1/(sigma.em*sigma.em))
} #t
################################
# Likelihood for counts
################################
for (t in 1:(n.yr-1)){
# Number of wild born juvs
N[1,t+1] ~ dpois(
(NH[t]*eta.OSalpha[2,2,t]*eta.OBRalpha[2,t] +
NW[t]*eta.OSalpha[2,1,t]*eta.OBRalpha[2,t] + # first year males to breeders
NF[t]*eta.ASalpha[t]*eta.ABRalpha[2,2,t] + # nonbreeder male hacked adults to breeder
NF[t]*eta.ASalpha[t]*eta.ABRalpha[2,1,t] + # nonbreeder male wild adults to breeder adults
NB[t]*eta.BSalpha[t]*(1-eta.BARalpha[t])) * # breeders to breeders
F[t+1]/2)
# first year hacked to nonbreeder adults
N[2,t+1] ~ dpois(eta.OSalpha[2,2,t]*(1-eta.OBRalpha[2,t]) * NH[t] )
# first year wild to nonbreeder adults
N[3,t+1] ~ dpois(eta.OSalpha[2,1,t]*(1-eta.OBRalpha[2,t]) * NW[t] )
# first year hacked to breeders
N[4,t+1] ~ dpois(eta.OSalpha[2,2,t]*eta.OBRalpha[2,t] * NH[t] )
# first year wild to breeders
N[5,t+1] ~ dpois(eta.OSalpha[2,1,t]*eta.OBRalpha[2,t] * NW[t] )
# Nonbreeder male wild adults to nonbreeder adults
```

```
N[6,t+1] ~ dbin(eta.ASalpha[t]*(1-eta.ABRalpha[2,1,t]), NF[t] )
# Nonbreeder male wild adults to breeders
N[7,t+1] ~ dbin(eta.ASalpha[t]*eta.ABRalpha[2,1,t], NF[t] )
# Breeders to nonbreeder adults
N[8,t+1] ~ dbin(eta.BSalpha[t]*(eta.BARalpha[t]), NB[t] )
# Breeders to breeders
N[9,t+1] ~ dbin(eta.BSalpha[t]*(1-eta.BARalpha[t]), NB[t] )
# Nonbreeder hacked adults to nonbreeder adults
N[10,t+1] ~ dbin(eta.ASalpha[t]*(1-eta.ABRalpha[2,2,t]), NF[t] )
# Nonbreeder hacked adults to breeders
N[11,t+1] ~ dbin(eta.ASalpha[t]*eta.ABRalpha[2,2,t], NF[t] )
# Immigrants to breeders
N[12,t+1] ~ dpois(NB[t]*omegaB[t])
# Immigrants to nonbreeders
N[13,t+1] ~ dpois(NF[t]*omegaA[t])
# Breeders to emigrants
N[14,t+1] ~ dpois(em[t] * NB[t] )
# NonBreeders to emigrants
N[15,t+1] ~ dpois(em[t] * NF[t] )
# First-year wild to emigrants
N[16,t+1] ~ dpois(em[t] * N[1,t] )
# First-year hacked to emigrants
N[17,t+1] ~ dpois(em[t] * aug[t] )
} # t

for (t in 1:n.yr){
Ntot[t] <- sum(N[c(1,2,3,4,5,6,7,8,9,10,11,12,13),t]) + aug[t] - sum(N[c(14,15,16,17),t]) # total nu
NB[t] <-  sum(N[c(4,5,7,9,11,12),t]) - N[14,t] # number of breeders
NF[t] <-  sum(N[c(2,3,6,8,10,13),t]) - N[15,t] # number of nonbreeders
NO[t] <-  N[1,t] + aug[t] - N[16,t] - N[17,t]# number of total first years. Includes translocated fi
NW[t] <-  N[1,t] - N[16,t] # Number of wild born
NH[t] <- aug[t] - N[17,t] # Number of hacked
NI[t] <- sum(N[c(12,13),t]) # number of immigrants
NE[t] <- sum(N[c(14,15,16,17),t])
} # t

# Observation process
for (t in 2:n.yr){
countBM[t] ~ dnegbin(pNB[t],rNB) # breeding males negative binomial
pNB[t] <- rNB/(rNB+NB[t])
countFM[t] ~ dpois(NF[t]) # nonbreeding adult males
countOM[t] ~ dpois(N[1,t]) # first year wild males
} # t

####################
# Assess GOF of the state-space models for counts
# Step 1: Compute statistic for observed data
# Step 2: Use discrepancy measure: mean absolute error
# Step 3: Use test statistic: number of turns
####################
for (t in 2:n.yr){
c.expB[t] <- NB[t] + 0.001 # expected counts adult breeder
c.expA[t] <- NF[t] + 0.001 # nonbreeder
```

```
c.expO[t] <- N[1,t] + 0.001 # first year
c.obsB[t] <- countBM[t] + 0.001
c.obsA[t] <- countFM[t] + 0.001
c.obsO[t] <- countOM[t] + 0.001
dssm.obsB[t] <- abs( ( (c.obsB[t]) - (c.expB[t]) ) / (c.obsB[t]+0.001)  )
dssm.obsA[t] <- abs( ( (c.obsA[t]) - (c.expA[t]) ) / (c.obsA[t]+0.001)  )
dssm.obsO[t] <- abs( ( (c.obsO[t]) - (c.expO[t]) ) / (c.obsO[t]+0.001)  )
} # t
dmape.obs[1] <- sum(dssm.obsB[2:n.yr])
dmape.obs[2] <- sum(dssm.obsA[2:n.yr])
dmape.obs[3] <- sum(dssm.obsO[2:n.yr])
# Compute fit statistic for replicate data
# Mean absolute error
for (t in 2:n.yr){
c.repB[t] ~ dnegbin(pNB[t],rNB) # expected counts
c.repA[t] ~ dpois(NF[t] )
c.repO[t] ~ dpois(N[1,t] )
} # t
for (t in 2:n.yr){
dssm.repB[t] <- abs( ( (c.repB[t]) - (c.expB[t]) ) / (c.repB[t]+0.001) )
dssm.repA[t] <- abs( ( (c.repA[t]) - (c.expA[t]) ) / (c.repA[t]+0.001) )
dssm.repO[t] <- abs( ( (c.repO[t]) - (c.expO[t]) ) / (c.repO[t]+0.001) )
} # t
dmape.rep[1] <- sum(dssm.repB[2:n.yr])
dmape.rep[2] <- sum(dssm.repA[2:n.yr])
dmape.rep[3] <- sum(dssm.repO[2:n.yr])

# Test statistic for number of turns
for (t in 2:(n.yr-2)){
tt1.obsB[t] <- step(countBM[t+2] - countBM[t+1])
tt2.obsB[t] <- step(countBM[t+1] - countBM[t])
tt3.obsB[t] <- equals(tt1.obsB[t] + tt2.obsB[t], 1)
tt1.obsA[t] <- step(countFM[t+2] - countFM[t+1])
tt2.obsA[t] <- step(countFM[t+1] - countFM[t])
tt3.obsA[t] <- equals(tt1.obsA[t] + tt2.obsA[t], 1)
tt1.obsO[t] <- step(countOM[t+2] - countOM[t+1])
tt2.obsO[t] <- step(countOM[t+1] - countOM[t])
tt3.obsO[t] <- equals(tt1.obsO[t] + tt2.obsO[t], 1)
} # t
tturn.obs[1] <- sum(tt3.obsB[2:(n.yr-2)])
tturn.obs[2] <- sum(tt3.obsA[2:(n.yr-2)])
tturn.obs[3] <- sum(tt3.obsO[2:(n.yr-2)])

for (t in 2:(n.yr-2)){
tt1.repB[t] <- step(c.repB[t+2] - c.repB[t+1])
tt2.repB[t] <- step(c.repB[t+1] - c.repB[t])
tt3.repB[t] <- equals(tt1.repB[t] + tt2.repB[t], 1)
tt1.repA[t] <- step(c.repA[t+2] - c.repA[t+1])
tt2.repA[t] <- step(c.repA[t+1] - c.repA[t])
tt3.repA[t] <- equals(tt1.repA[t] + tt2.repA[t], 1)
tt1.repO[t] <- step(c.repO[t+2] - c.repO[t+1])
tt2.repO[t] <- step(c.repO[t+1] - c.repO[t])
tt3.repO[t] <- equals(tt1.repO[t] + tt2.repO[t], 1)
```

```
} # t
tturn.rep[1] <- sum(tt3.repB[2:(n.yr-2)])
tturn.rep[2] <- sum(tt3.repA[2:(n.yr-2)])
tturn.rep[3] <- sum(tt3.repO[2:(n.yr-2)])


################################
# Likelihood for survival
################################
for (i in 1:nind){
for (t in 1:(n.yr-1)){
#Survival
sO[i,t] <- eta.OSalpha[sex[i],hacked[i],t] # first year
sA[i,t] <- eta.ASalpha[t] # nonbreeder
sB[i,t] <- eta.BSalpha[t] # breeder
#Recruitment
psiOB[i,t] <- eta.OBRalpha[sex[i],t] # first year to breeder
psiAB[i,t] <- eta.ABRalpha[sex[i], hacked[i],t] # nonbreederto breeder
psiBA[i,t] <- eta.BARalpha[t] # breeder to nonbreeder
#Re-encounter
pA[i,t] <- eta.pA[hacked[i],t] # resight of nonbreeders
pB[i,t] <- eta.pB[t]  # resight of breeders
}#t
}#i

# Define state-transition and observation matrices
for (i in 1:nind){
# Define probabilities of state S(t+1) given S(t)
for (t in first[i]:(n.yr-1)){
ps[1,i,t,1] <- 0
ps[1,i,t,2] <- sO[i,t] * (1-psiOB[i,t]) * (1-em[t])
ps[1,i,t,3] <- sO[i,t] * psiOB[i,t] * (1-em[t])
ps[1,i,t,4] <- (1-sO[i,t]) * r * (1-em[t])
ps[1,i,t,5] <- (1-sO[i,t]) * (1-r) * (1-em[t])
ps[1,i,t,6] <- sO[i,t] * em[t]
ps[1,i,t,7] <- (1-sO[i,t]) * (1-r) * em[t]

ps[2,i,t,1] <- 0
ps[2,i,t,2] <- sA[i,t] * (1-psiAB[i,t]) * (1-em[t])
ps[2,i,t,3] <- sA[i,t] * psiAB[i,t] * (1-em[t])
ps[2,i,t,4] <- (1-sA[i,t]) * r * (1-em[t])
ps[2,i,t,5] <- (1-sA[i,t]) * (1-r) * (1-em[t])
ps[2,i,t,6] <- sA[i,t] * em[t]
ps[2,i,t,7] <- (1-sA[i,t]) * (1-r) * em[t]

ps[3,i,t,1] <- 0
ps[3,i,t,2] <- sB[i,t] * psiBA[i,t] * (1-em[t])
ps[3,i,t,3] <- sB[i,t] * (1-psiBA[i,t]) * (1-em[t])
ps[3,i,t,4] <- (1-sB[i,t]) * r * (1-em[t])
ps[3,i,t,5] <- (1-sB[i,t]) * (1-r) * (1-em[t])
ps[3,i,t,6] <- sB[i,t] * em[t]
ps[3,i,t,7] <- (1-sB[i,t]) * (1-r) * em[t]
```

```
    ps[4,i,t,1] <- 0
    ps[4,i,t,2] <- 0
    ps[4,i,t,3] <- 0
    ps[4,i,t,4] <- 0
    ps[4,i,t,5] <- 1
    ps[4,i,t,6] <- 0
    ps[4,i,t,7] <- 0

    ps[5,i,t,1] <- 0
    ps[5,i,t,2] <- 0
    ps[5,i,t,3] <- 0
    ps[5,i,t,4] <- 0
    ps[5,i,t,5] <- 1
    ps[5,i,t,6] <- 0
    ps[5,i,t,7] <- 0

    ps[6,i,t,1] <- 0
    ps[6,i,t,2] <- 0
    ps[6,i,t,3] <- 0
    ps[6,i,t,4] <- 0
    ps[6,i,t,5] <- 0
    ps[6,i,t,6] <- 1
    ps[6,i,t,7] <- 0

    ps[7,i,t,1] <- 0
    ps[7,i,t,2] <- 0
    ps[7,i,t,3] <- 0
    ps[7,i,t,4] <- 0
    ps[7,i,t,5] <- 0
    ps[7,i,t,6] <- 0
    ps[7,i,t,7] <- 1

    # Define probabilities of O(t) given S(t)
    po[1,i,t,1] <- 1
    po[1,i,t,2] <- 0
    po[1,i,t,3] <- 0
    po[1,i,t,4] <- 0
    po[1,i,t,5] <- 0

    po[2,i,t,1] <- 0
    po[2,i,t,2] <- pA[i,t]
    po[2,i,t,3] <- 0
    po[2,i,t,4] <- 0
    po[2,i,t,5] <- 1-pA[i,t]

    po[3,i,t,1] <- 0
    po[3,i,t,2] <- 0
    po[3,i,t,3] <- pB[i,t]
    po[3,i,t,4] <- 0
    po[3,i,t,5] <- 1-pB[i,t]

    po[4,i,t,1] <- 0
    po[4,i,t,2] <- 0
```

```
    po[4,i,t,3] <- 0
    po[4,i,t,4] <- 1
    po[4,i,t,5] <- 0

    po[5,i,t,1] <- 0
    po[5,i,t,2] <- 0
    po[5,i,t,3] <- 0
    po[5,i,t,4] <- 0
    po[5,i,t,5] <- 1

    po[6,i,t,1] <- 0
    po[6,i,t,2] <- 0
    po[6,i,t,3] <- 0
    po[6,i,t,4] <- 0
    po[6,i,t,5] <- 1

    po[7,i,t,1] <- 0
    po[7,i,t,2] <- 0
    po[7,i,t,3] <- 0
    po[7,i,t,4] <- 0
    po[7,i,t,5] <- 1
    } #t
    } #i

    # Likelihood
    for (i in 1:nind){
    # Define latent state at first capture
    z[i,first[i]] <- y[i,first[i]]
    for (t in (first[i]+1):n.yr){
    # State process: draw S(t) given S(t-1)
    z[i,t] ~ dcat(ps[z[i,t-1], i, t-1, 1:7])
    # Observation process: draw O(t) given S(t)
    y[i,t] ~ dcat(po[z[i,t], i, t-1, 1:5])
    } #t
    } #i
    } #model
    ",fill = TRUE)
sink()

get.first <- function(x) min(which(x!=5))
f <- apply(datl$y, 1, get.first)
get.last <- function(x) max(which(x!=5))
l <- apply(datl$y, 1, get.last)
TFmat <- is.na(z.inits) & is.na(datl$z)
for (i in 1:dim(TFmat)[1]){  TFmat[i,1:f[i]] <- FALSE }
z.inits[TFmat] <- sample(size=445, c(2,3), replace=T, prob=c(0.5, 0.5) )

# Set initial values of N
# to small numbers for immigrant and emigrants
# because preliminary analyses suggested low rates
# This is necessary to run model and avoid initial values error.
Ni <- array(NA, dim=c(17,26))
Ni[12:17, 2:26] <- 0
```

```
inits <- function(){list(z = z.inits, N=Ni)}

params <- c( "rNB",
             "sigma.F", "eps.F", "mu.F",
             "r", "l.mu.em", "mu.em", "sigma.em",
             "omegaA1",  "omegaB1", "sigma.omegaA", "sigma.omegaB",
             "OSalpha", "ASalpha", "BSalpha", "OBRalpha", "ABRalpha", "BARalpha",  "mu.pA", "mu.pB",
             "OSalpha1", "ASalpha1", "BSalpha1","OBRalpha1", "ABRalpha1", "BARalpha1","mu.pA1", "mu.pB1
             "sigma.OS.s", "sigma.AS.s", "sigma.BS.s", "sigma.OBR.psi", "sigma.ABR.psi", "sigma.BAR.psi
             "N", "NB", "NF", "NO", "NW", "NH", "NI",  "NE", "Ntot",
             "dmape.obs", "dmape.rep", "tvm.obs", "tvm.rep", "dd.obs", "dd.rep",
             "tturn.obs", "tturn.rep",
             "eps.OS.s", "eps.AS.s", "eps.BS.s", "eps.OBR.psi", "eps.ABR.psi", "eps.BAR.psi", "eps.pA",
             "eps.omegaA", "eps.omegaB", "eps.em",
             "eta.OSalpha", "eta.ASalpha", "eta.BSalpha", "eta.OBRalpha", "eta.ABRalpha", "eta.BARalpha
             "em", "omegaA", "omegaB", "F"
)

# MCMC settings
ni <- 200000; nt <- 50; nb <- 150000; nc <- 3; na <- 1000
out <- jags(datl, inits, params, modfl,
            n.chains = nc, n.thin = nt, n.burnin = nb, n.adapt=na, n.iter=ni,
            parallel=T, module=c("glm", "bugs"))
#save(file=paste("./", m, ".Rdata", sep=""), list="out")
```

1.5.3 The Importance of Immigration

```
## ---- immigration ---------
# plot priors and posteriors of immigration
load("outputs\\ipm-ie.Rdata")
source("R/HDIofMCMC.R")
options(scipen = 100)
x <- seq(from=0, to=4, by=0.01)
df <- data.frame(x=x, prior=dnorm(x,0,1))
df2 <- data.frame(omegaA=out$sims.list$omegaA1,
                  omegaB=out$sims.list$omegaB1)
```

```
par(mar=c(5,5,2,5))
plot(df$x, df$prior, type="l", lwd=2, xlim=c(0,4),
     ylab="Prior density (black line)",
     xlab=expression(paste("Immigration rate (",omega,")", sep="")),
     main="Nonbreeders")
par(new=TRUE)
plot(density(df2$omegaA), type="l", lwd=2, xlim=c(0,4),
     main="", col="blue",
     axes = FALSE, xlab = "", ylab = "")
axis(side = 4, at = c(0,20,40))      # Add second axis
mtext("Posterior density (blue line)", side = 4, line = 3)
```

```
# Plot immigration of breeders
par(mar=c(5,5,2,5))
plot(df$x, df$prior, type="l", lwd=2, xlim=c(0,4),
     ylab="Prior density (black line)",
```

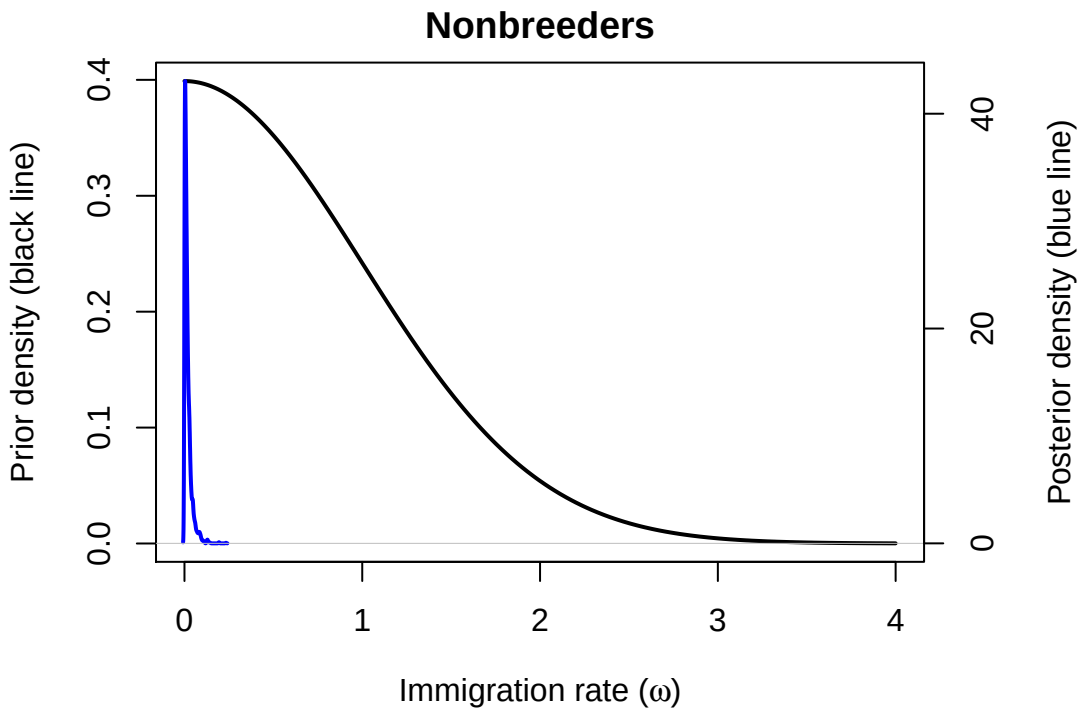


Figure 3: Fig. S4. Immigration priors and posteriors of nonbreeders.

```
     xlab=expression(paste("Immigration rate (",omega,")", sep="")),
     main="Breeders")
par(new=TRUE)
plot(density(df2$omegaB), type="l", lwd=2, xlim=c(0,4),
     main="", col="blue",
     axes = FALSE, xlab = "", ylab = "")
axis(side = 4, at = c(0,4,8))       # Add second axis
mtext("Posterior density (blue line)", side = 4, line = 3)
```

```
hist(out$sims.list$omegaA1, main="Histogram of Immigration\nNonbreeders", xlab="Posterior draws of immi
abline(v=median(out$sims.list$omegaA1), lwd=2)
abline(v=HDIofMCMC(out$sims.list$omegaA1, cred=0.95), lty=2, lwd=2)
median(out$sims.list$omegaA1)
```

```
## [1] 0.01151637
```

```
HDIofMCMC(out$sims.list$omegaA1, cred=0.95)
```

```
## [1] 0.0000007452894 0.0589206724015
```

```
hist(out$sims.list$omegaB1, main="Histogram of Immigration\nBreeders", xlab="Posterior draws of immigra
abline(v=median(out$sims.list$omegaB1), lwd=2)
abline(v=HDIofMCMC(out$sims.list$omegaB1, cred=0.95), lty=2, lwd=2)
median(out$sims.list$omegaB1)
```

```
## [1] 0.1089806
```

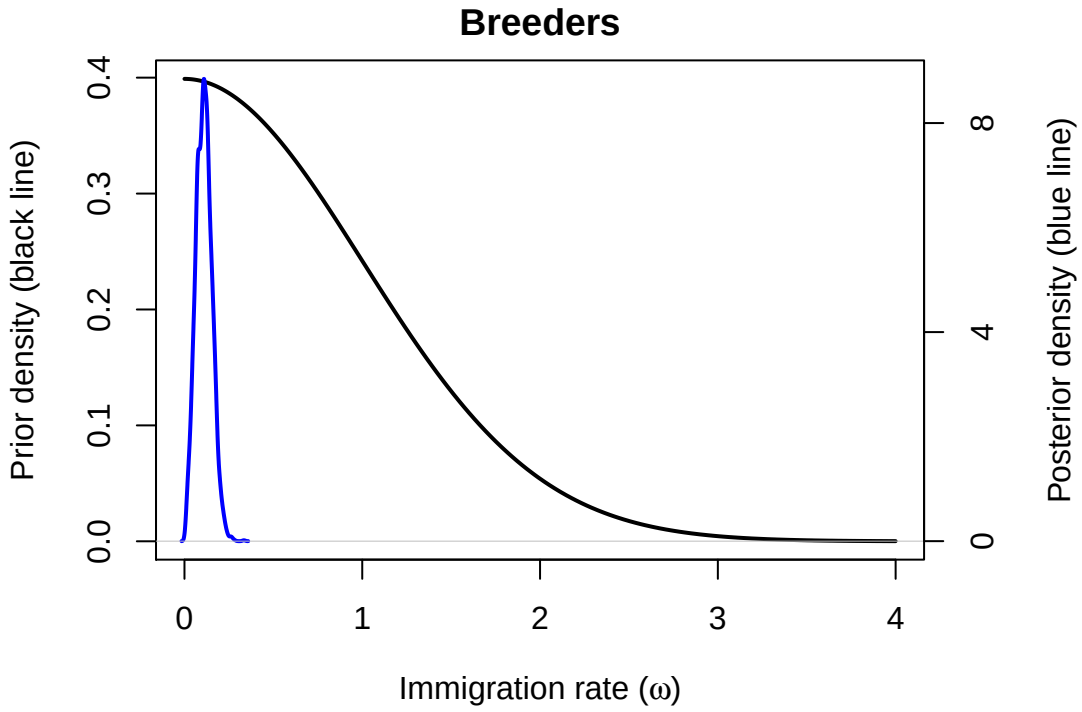


Figure 4: Fig. S5. Immigration priors and posteriors of breeders.

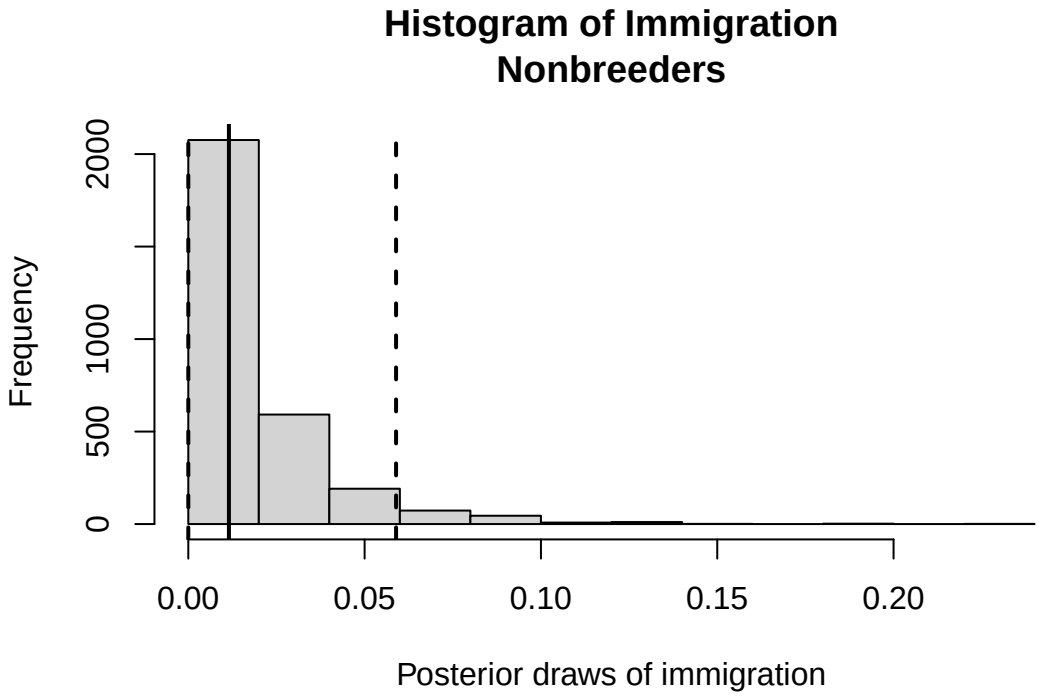


Figure 5: Fig. S6. Histogram and point estimates of Immigration of nonbreeders

```
HDIofMCMC(out$sims.list$omegaB1, cred=0.95)
```

```
## [1] 0.01816769 0.18865711
```

```
db <- density(out$sims.list$omegaB1)
db$x[which(db$y==max(db$y))]
```

```
## [1] 0.1096646
```

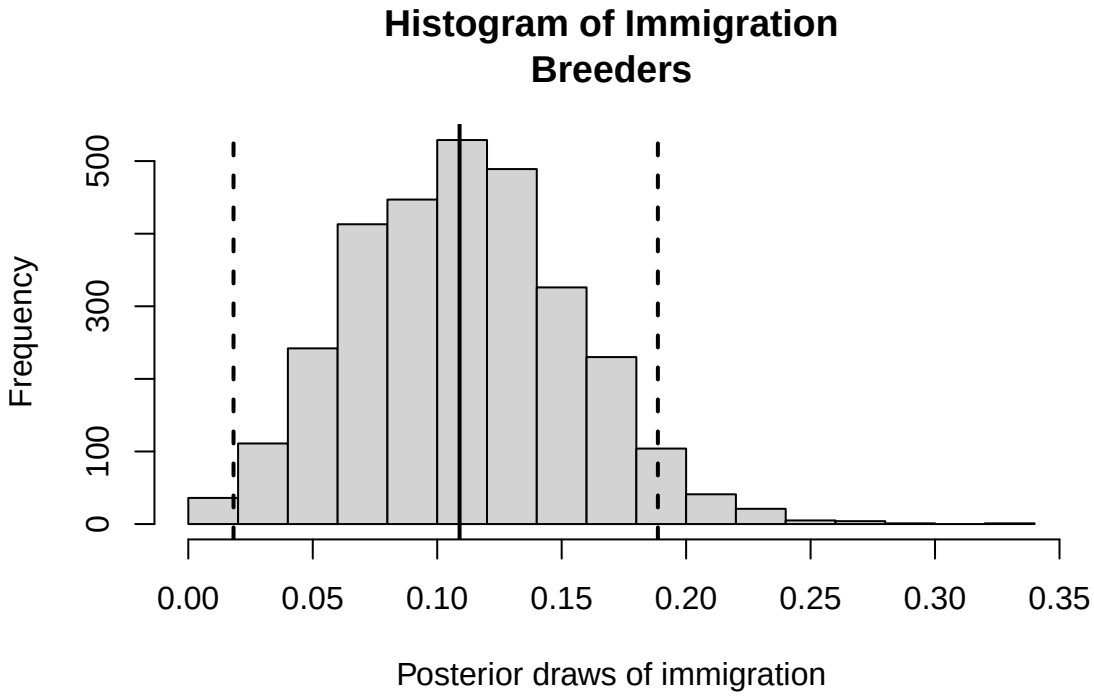


Figure 6: Fig. S7. Histogram and point estimates of Immigration of breeders

2. Implementation of the Integrated Population Model without Immigration for Evaluation of Vital Rates

2.1 JAGS and R Code for the IPM without Immigration

```
#################################
# The model
#################################
library (jagsUI)
load(".\\data\\data-7states.Rdata")
datl$pp <- c(0, 0, tapply(datl$prod, datl$year.p, sum, na.rm=T), 42)
datl$pp[c(15,17)] <- NA
datl$countOM <- datl$countJM-datl$aug

m<- c("ipm-e")
modfl <- paste(".\\", m, ".txt", sep="")
sink(modfl)
```

```
cat("
    model{
    ######################################################
    ######################################################
    # Mark-resight-recovery data
    #   Observations (po) = y
    #     1 seen first year (age 0)
    #     2 seen nonbreeder
    #     3 seen breeder
    #     4 seen dead
    #     5 not seen
    #   States (ps)
    #     1 alive first year
    #     2 alive nonbreeder
    #     3 alive breeder
    #     4 Recovered recently dead
    #     5 Dead not recovered/long dead
    #     6 Emigrated and alive
    #     7 Emigrated and dead
    #   Groups
    #     1 wild hatched
    #     2 hacked
    #   Sex
    #     1 female
    #     2 male
    #   Effort
    #     1 low
    #     2 high
    ######################################################
    # PARAMETERS
    #   sO: survival probability first year
    #       (this is the letter 'O' rather than zero so jags can parse)
    #   sA: survival probability nonbreeders
    #   sB: survival probability breeders
    #   psiOB: recruitment probability from first-year to breeder
    #   psiAB: recruitment probability from nonbreeders to breeder
    #   psiBA: recruitment probability from breeder to nonbreeders
    #   pA: resight probability nonbreeders
    #   pB: resight probability breeder
    #   F: fecundity
    #   omega: immigration
    ######################################################
    # Priors and constraints
    ######################################################
    # Counts for first year are known with certainty
    N[1,1] <- 0 # Wild born first-year
    N[2,1] <- 0 # first-year hacked to nonbreeder adults
    N[3,1] <- 0 # First-year wild to nonbreeder
    N[4,1] <- 0  # First-year hacked to breeders
    N[5,1] <- 0  # First-year wild to breeders
    N[6,1] <- 0  # Nonbreeder wild to nonbreeder
    N[7,1] <- 0  # Nonbreeder wild to breeders
    N[8,1] <- 0  # Breeders to nonbreeder
```

```
N[9,1] <- 0  # Breeders to breeders
N[10,1] <- 0  # Nonbreeder hacked to nonbreeder
N[11,1] <- 0  # Nonbreeder hacked to breeder
N[12,1] <- 0  # Immigrant to breeder
N[13,1] <- 0  # Immigrant to nonbreeder
N[14,1] <- 0 # breeder to emigrant
N[15,1] <- 0 # nonbreeder to emigrant
N[16,1] <- 0 # first-year to emigrant
N[17,1] <- 0 # first-year hacked to emigrant
F[1] <- 0 # No breeding during the first year, all hacked birds

rNB ~ dunif(0,50)
# omegaB1 ~ dnorm(0, 1/(1*1) )T(0,) # upper limit near 5 imm per breeder to help run model
# omegaA1 ~ dnorm(0, 1/(1*1) )T(0,)
# sigma.omegaB ~ dnorm(0, 1/(2*2) )T(0,)
# sigma.omegaA ~ dnorm(0, 1/(2*2) )T(0,)
for (m in 1:2){ mu.F[m] ~ dunif(0,5) } # m # limits to help run model
sigma.F ~ dnorm(0, 1/(2*2) )T(0,)
mu.em <- logit(mu.em1)
mu.em1 ~ dunif(0,1)
sigma.em ~ dnorm(0, 1/(2*2) )T(0,)
r ~ dunif(0,1)

# Survival loops for demographic categories by sex, hacked, effort

sigma.AS.s ~ dnorm(0, 1/(2*2) )T(0,)
ASalpha <- logit(ASalpha1)
ASalpha1 ~ dunif(0, 1)

sigma.BS.s ~ dnorm(0, 1/(2*2) )T(0,)
BSalpha <- logit(BSalpha1)
BSalpha1 ~ dunif(0, 1)

sigma.BAR.psi ~ dnorm(0, 1/(2*2) )T(0,)
BARalpha <- logit(BARalpha1)
BARalpha1 ~ dunif(0, 1)

sigma.pB ~ dnorm(0, 1/(2*2) )T(0,)

for (k in 1:2){
mu.pB[k]<- logit(mu.pB1[k])
mu.pB1[k] ~ dunif(0, 1)
} # k

for (t in 1:(n.yr-1)){
logit(eta.ASalpha[t]) <- ASalpha + eps.AS.phi[t]
eps.AS.phi[t] ~ dnorm(0, 1/(sigma.AS.s*sigma.AS.s) )

logit(eta.BSalpha[t]) <- BSalpha + eps.BS.phi[t]
eps.BS.phi[t] ~ dnorm(0, 1/(sigma.BS.s*sigma.BS.s) )

logit(eta.BARalpha[t]) <- BARalpha + eps.BAR.psi[t]
eps.BAR.psi[t] ~ dnorm(0, 1/(sigma.BAR.psi*sigma.BAR.psi))
```

```
logit(eta.pB[t]) <- mu.pB[effort[t]] + eps.pB[t]
eps.pB[t] ~ dnorm(0, 1/(sigma.pB*sigma.pB) )
} #t

for(s in 1:2){

sigma.OBR.psi[s] ~ dnorm(0, 1/(2*2) )T(0,)
OBRalpha[s] <- logit(OBRalpha1[s])
OBRalpha1[s] ~ dunif(0, 1)

for (t in 1:(n.yr-1)){
logit(eta.OBRalpha[s,t]) <- OBRalpha[s] + eps.OBR.psi[s,t]
eps.OBR.psi[s,t] ~ dnorm(0, 1/(sigma.OBR.psi[s]*sigma.OBR.psi[s]) )
} #t

for (h in 1:2){
sigma.ABR.psi[s,h] ~ dnorm(0, 1/(2*2) )T(0,)
ABRalpha[s,h] <- logit(ABRalpha1[s,h])
ABRalpha1[s,h] ~ dunif(0, 1)

sigma.OS.s[s,h] ~ dnorm(0, 1/(2*2) )T(0,)
OSalpha[s,h] <- logit(OSalpha1[s,h])
OSalpha1[s,h] ~ dunif(0, 1)

for (t in 1:(n.yr-1)){
logit(eta.ABRalpha[s,h,t]) <- ABRalpha[s,h] + eps.ABR.psi[s,h,t]
eps.ABR.psi[s,h,t] ~ dnorm(0, 1/(sigma.ABR.psi[s,h]*sigma.ABR.psi[s,h]) )

logit(eta.OSalpha[s,h,t]) <- OSalpha[s,h] + eps.OS.s[s,h,t]
eps.OS.s[s,h,t] ~ dnorm(0, 1/(sigma.OS.s[s,h]*sigma.OS.s[s,h]) )
} #t
} } #h #s

for (h in 1:2){
sigma.pA[h] ~ dnorm(0, 1/(2*2) )T(0,)
for (k in 1:2){
mu.pA[k,h] <- logit(mu.pA1[k,h])
mu.pA1[k,h] ~ dunif(0,1)
} # k

for (t in 1:(n.yr-1)){
logit(eta.pA[h,t]) <- mu.pA[effort[t], h] + eps.pA[h,t]
eps.pA[h,t] ~ dnorm(0, 1/(sigma.pA[h]*sigma.pA[h]) )
}} #t #h

########################
# Derived params
########################
for (t in 3:(n.yr-1)){
lambda[t] <-  Ntot[t+1]/(Ntot[t])
loglambda[t] <- log(lambda[t])
} #t
```

```
################################
# Likelihood for productivity
################################
for (t in 2:n.yr){
pp[t] ~ dpois( F[t]*NB[t] )
log(F[t]) <- log(mu.F[manage[t]]) + eps.F[t]
eps.F[t] ~ dnorm (0, 1/(sigma.F*sigma.F) )
}
# GOF fecundity
for (t in 1:n.yr){
J.rep[t] ~ dpois( F[t]*NB[t] )
J.exp[t] <- F[t]*NB[t]
d.obs[t] <- pp[t]* log((pp[t]+0.001)/(J.exp[t]+0.001)) - (pp[t]-J.exp[t])
d.rep[t] <- J.rep[t]*log((J.rep[t]+0.001)/(J.exp[t]+0.001)) - (J.rep[t]-J.exp[t])
} # t
dd.obs <- sum(d.obs)
tvm.obs <- sd(pp)^2/mean(pp)
dd.rep <- sum(d.rep)
tvm.rep <- sd(J.rep)^2/mean(J.rep)

################################
# Likelihood for immigration
################################
for (t in 1:(n.yr-1)){
omegaB[t] <- 0 # set to zero, no immigration
omegaA[t] <- 0 # set to zero, no immigration
# log(omegaB[t]) <-  log(omegaB1) + eps.omegaB[t] # breeder
# log(omegaA[t]) <- log(omegaA1) + eps.omegaA[t] # nonbreeder
# eps.omegaB[t] ~ dnorm(0, 1/(sigma.omegaB*sigma.omegaB))
# eps.omegaA[t] ~ dnorm(0, 1/(sigma.omegaA*sigma.omegaA))
logit(em[t]) <- mu.em + eps.em[t]
eps.em[t] ~ dnorm(0, 1/(sigma.em*sigma.em))
} #t
################################
# Likelihood for counts
################################
for (t in 1:(n.yr-1)){
# Number of wild born juvs
N[1,t+1] ~ dpois(
(NH[t]*eta.OSalpha[2,2,t]*eta.OBRalpha[2,t] +
NW[t]*eta.OSalpha[2,1,t]*eta.OBRalpha[2,t] + # first year males to breeders
NF[t]*eta.ASalpha[t]*eta.ABRalpha[2,2,t] + # nonbreeder male hacked adults to breeder
NF[t]*eta.ASalpha[t]*eta.ABRalpha[2,1,t] + # nonbreeder male wild adults to breeder adults
NB[t]*eta.BSalpha[t]*(1-eta.BARalpha[t])) * # breeders to breeders
F[t+1]/2)
# first year hacked to nonbreeder adults
N[2,t+1] ~ dpois(eta.OSalpha[2,2,t]*(1-eta.OBRalpha[2,t]) * NH[t] )
# first year wild to nonbreeder adults
N[3,t+1] ~ dpois(eta.OSalpha[2,1,t]*(1-eta.OBRalpha[2,t]) * NW[t] )
# first year hacked to breeders
N[4,t+1] ~ dpois(eta.OSalpha[2,2,t]*eta.OBRalpha[2,t] * NH[t] )
# first year wild to breeders
N[5,t+1] ~ dpois(eta.OSalpha[2,1,t]*eta.OBRalpha[2,t] * NW[t] )
```

```
# Nonbreeder male wild adults to nonbreeder adults
N[6,t+1] ~ dbin(eta.ASalpha[t]*(1-eta.ABRalpha[2,1,t]), NF[t] )
# Nonbreeder male wild adults to breeders
N[7,t+1] ~ dbin(eta.ASalpha[t]*eta.ABRalpha[2,1,t], NF[t] )
# Breeders to nonbreeder adults
N[8,t+1] ~ dbin(eta.BSalpha[t]*(eta.BARalpha[t]), NB[t] )
# Breeders to breeders
N[9,t+1] ~ dbin(eta.BSalpha[t]*(1-eta.BARalpha[t]), NB[t] )
# Nonbreeder hacked adults to nonbreeder adults
N[10,t+1] ~ dbin(eta.ASalpha[t]*(1-eta.ABRalpha[2,2,t]), NF[t] )
# Nonbreeder hacked adults to breeders
N[11,t+1] ~ dbin(eta.ASalpha[t]*eta.ABRalpha[2,2,t], NF[t] )
# Immigrants to breeders
N[12,t+1] ~ dpois(NB[t]*omegaB[t])
# Immigrants to nonbreeders
N[13,t+1] ~ dpois(NF[t]*omegaA[t])
# Breeders to emigrants
N[14,t+1] ~ dpois(em[t] * NB[t] )
# NonBreeders to emigrants
N[15,t+1] ~ dpois(em[t] * NF[t] )
# First-year wild to emigrants
N[16,t+1] ~ dpois(em[t] * N[1,t] )
# First-year hacked to emigrants
N[17,t+1] ~ dpois(em[t] * aug[t] )
} # t

for (t in 1:n.yr){
Ntot[t] <- sum(N[c(1,2,3,4,5,6,7,8,9,10,11,12,13),t]) + aug[t] - sum(N[c(14,15,16,17),t]) # total nu
NB[t] <-  sum(N[c(4,5,7,9,11,12),t]) - N[14,t] # number of breeders
NF[t] <-  sum(N[c(2,3,6,8,10,13),t]) - N[15,t] # number of nonbreeders
NO[t] <-  N[1,t] + aug[t] - N[16,t] - N[17,t]# number of total first years and translocated first ye
NW[t] <-  N[1,t] - N[16,t] # Number of wild born
NH[t] <- aug[t] - N[17,t] # Number of hacked
NI[t] <- sum(N[c(12,13),t]) # number of immigrants
NE[t] <- sum(N[c(14,15,16,17),t])
} # t

# Observation process
for (t in 2:n.yr){
countBM[t] ~ dnegbin(pNB[t],rNB) # breeding males negative binomial
pNB[t] <- rNB/(rNB+NB[t])
countFM[t] ~ dpois(NF[t]) # nonbreeding adult males
countOM[t] ~ dpois(N[1,t]) # first year males
} # t

####################
# Assess GOF of the state-space models for counts
# Step 1: Compute statistic for observed data
# Step 2: Use discrepancy measure: mean absolute error
# Step 3: Use test statistic: number of turns
####################
for (t in 2:n.yr){
c.expB[t] <- NB[t] + 0.001 # expected counts adult breeder
```

```
c.expA[t] <- NF[t] + 0.001 # nonbreeder
c.expO[t] <- N[1,t] + 0.001 # first year
c.obsB[t] <- countBM[t] + 0.001
c.obsA[t] <- countFM[t] + 0.001
c.obsO[t] <- countOM[t] + 0.001
dssm.obsB[t] <- abs( ( (c.obsB[t]) - (c.expB[t]) ) / (c.obsB[t]+0.001)  )
dssm.obsA[t] <- abs( ( (c.obsA[t]) - (c.expA[t]) ) / (c.obsA[t]+0.001)  )
dssm.obsO[t] <- abs( ( (c.obsO[t]) - (c.expO[t]) ) / (c.obsO[t]+0.001)  )
} # t
dmape.obs[1] <- sum(dssm.obsB[2:n.yr])
dmape.obs[2] <- sum(dssm.obsA[2:n.yr])
dmape.obs[3] <- sum(dssm.obsO[2:n.yr])
# Compute fit statistic for replicate data
# Mean absolute error
for (t in 2:n.yr){
c.repB[t] ~ dnegbin(pNB[t],rNB) # expected counts
c.repA[t] ~ dpois(NF[t] )
c.repO[t] ~ dpois(N[1,t] )
} # t
for (t in 2:n.yr){
dssm.repB[t] <- abs( ( (c.repB[t]) - (c.expB[t]) ) / (c.repB[t]+0.001) )
dssm.repA[t] <- abs( ( (c.repA[t]) - (c.expA[t]) ) / (c.repA[t]+0.001) )
dssm.repO[t] <- abs( ( (c.repO[t]) - (c.expO[t]) ) / (c.repO[t]+0.001) )
} # t
dmape.rep[1] <- sum(dssm.repB[2:n.yr])
dmape.rep[2] <- sum(dssm.repA[2:n.yr])
dmape.rep[3] <- sum(dssm.repO[2:n.yr])

# Test statistic for number of turns
for (t in 2:(n.yr-2)){
tt1.obsB[t] <- step(countBM[t+2] - countBM[t+1])
tt2.obsB[t] <- step(countBM[t+1] - countBM[t])
tt3.obsB[t] <- equals(tt1.obsB[t] + tt2.obsB[t], 1)
tt1.obsA[t] <- step(countFM[t+2] - countFM[t+1])
tt2.obsA[t] <- step(countFM[t+1] - countFM[t])
tt3.obsA[t] <- equals(tt1.obsA[t] + tt2.obsA[t], 1)
tt1.obsO[t] <- step(countOM[t+2] - countOM[t+1])
tt2.obsO[t] <- step(countOM[t+1] - countOM[t])
tt3.obsO[t] <- equals(tt1.obsO[t] + tt2.obsO[t], 1)
} # t
tturn.obs[1] <- sum(tt3.obsB[2:(n.yr-2)])
tturn.obs[2] <- sum(tt3.obsA[2:(n.yr-2)])
tturn.obs[3] <- sum(tt3.obsO[2:(n.yr-2)])

for (t in 2:(n.yr-2)){
tt1.repB[t] <- step(c.repB[t+2] - c.repB[t+1])
tt2.repB[t] <- step(c.repB[t+1] - c.repB[t])
tt3.repB[t] <- equals(tt1.repB[t] + tt2.repB[t], 1)
tt1.repA[t] <- step(c.repA[t+2] - c.repA[t+1])
tt2.repA[t] <- step(c.repA[t+1] - c.repA[t])
tt3.repA[t] <- equals(tt1.repA[t] + tt2.repA[t], 1)
tt1.repO[t] <- step(c.repO[t+2] - c.repO[t+1])
tt2.repO[t] <- step(c.repO[t+1] - c.repO[t])
```

```
tt3.repO[t] <- equals(tt1.repO[t] + tt2.repO[t], 1)
} # t
tturn.rep[1] <- sum(tt3.repB[2:(n.yr-2)])
tturn.rep[2] <- sum(tt3.repA[2:(n.yr-2)])
tturn.rep[3] <- sum(tt3.repO[2:(n.yr-2)])


################################
# Likelihood for survival
################################
for (i in 1:nind){
for (t in 1:(n.yr-1)){
#Survival
sO[i,t] <- eta.OSalpha[sex[i],hacked[i],t] # first year
sA[i,t] <- eta.ASalpha[t] # nonbreeder
sB[i,t] <- eta.BSalpha[t] # breeder
#Recruitment
psiOB[i,t] <- eta.OBRalpha[sex[i],t] # first year to breeder
psiAB[i,t] <- eta.ABRalpha[sex[i], hacked[i],t] # nonbreederto breeder
psiBA[i,t] <- eta.BARalpha[t] # breeder to nonbreeder
#Re-encounter
pA[i,t] <- eta.pA[hacked[i],t] # resight of nonbreeders
pB[i,t] <- eta.pB[t]  # resight of breeders
}#t
}#i

# Define state-transition and observation matrices
for (i in 1:nind){
# Define probabilities of state S(t+1) given S(t)
for (t in first[i]:(n.yr-1)){
ps[1,i,t,1] <- 0
ps[1,i,t,2] <- sO[i,t] * (1-psiOB[i,t]) * (1-em[t])
ps[1,i,t,3] <- sO[i,t] * psiOB[i,t] * (1-em[t])
ps[1,i,t,4] <- (1-sO[i,t]) * r * (1-em[t])
ps[1,i,t,5] <- (1-sO[i,t]) * (1-r) * (1-em[t])
ps[1,i,t,6] <- sO[i,t] * em[t]
ps[1,i,t,7] <- (1-sO[i,t]) * (1-r) * em[t]

ps[2,i,t,1] <- 0
ps[2,i,t,2] <- sA[i,t] * (1-psiAB[i,t]) * (1-em[t])
ps[2,i,t,3] <- sA[i,t] * psiAB[i,t] * (1-em[t])
ps[2,i,t,4] <- (1-sA[i,t]) * r * (1-em[t])
ps[2,i,t,5] <- (1-sA[i,t]) * (1-r) * (1-em[t])
ps[2,i,t,6] <- sA[i,t] * em[t]
ps[2,i,t,7] <- (1-sA[i,t]) * (1-r) * em[t]

ps[3,i,t,1] <- 0
ps[3,i,t,2] <- sB[i,t] * psiBA[i,t] * (1-em[t])
ps[3,i,t,3] <- sB[i,t] * (1-psiBA[i,t]) * (1-em[t])
ps[3,i,t,4] <- (1-sB[i,t]) * r * (1-em[t])
ps[3,i,t,5] <- (1-sB[i,t]) * (1-r) * (1-em[t])
ps[3,i,t,6] <- sB[i,t] * em[t]
ps[3,i,t,7] <- (1-sB[i,t]) * (1-r) * em[t]
```

```
    ps[4,i,t,1] <- 0
    ps[4,i,t,2] <- 0
    ps[4,i,t,3] <- 0
    ps[4,i,t,4] <- 0
    ps[4,i,t,5] <- 1
    ps[4,i,t,6] <- 0
    ps[4,i,t,7] <- 0

    ps[5,i,t,1] <- 0
    ps[5,i,t,2] <- 0
    ps[5,i,t,3] <- 0
    ps[5,i,t,4] <- 0
    ps[5,i,t,5] <- 1
    ps[5,i,t,6] <- 0
    ps[5,i,t,7] <- 0

    ps[6,i,t,1] <- 0
    ps[6,i,t,2] <- 0
    ps[6,i,t,3] <- 0
    ps[6,i,t,4] <- 0
    ps[6,i,t,5] <- 0
    ps[6,i,t,6] <- 1
    ps[6,i,t,7] <- 0

    ps[7,i,t,1] <- 0
    ps[7,i,t,2] <- 0
    ps[7,i,t,3] <- 0
    ps[7,i,t,4] <- 0
    ps[7,i,t,5] <- 0
    ps[7,i,t,6] <- 0
    ps[7,i,t,7] <- 1

    # Define probabilities of O(t) given S(t)
    po[1,i,t,1] <- 1
    po[1,i,t,2] <- 0
    po[1,i,t,3] <- 0
    po[1,i,t,4] <- 0
    po[1,i,t,5] <- 0

    po[2,i,t,1] <- 0
    po[2,i,t,2] <- pA[i,t]
    po[2,i,t,3] <- 0
    po[2,i,t,4] <- 0
    po[2,i,t,5] <- 1-pA[i,t]

    po[3,i,t,1] <- 0
    po[3,i,t,2] <- 0
    po[3,i,t,3] <- pB[i,t]
    po[3,i,t,4] <- 0
    po[3,i,t,5] <- 1-pB[i,t]

    po[4,i,t,1] <- 0
    po[4,i,t,2] <- 0
```

```
    po[4,i,t,3] <- 0
    po[4,i,t,4] <- 1
    po[4,i,t,5] <- 0

    po[5,i,t,1] <- 0
    po[5,i,t,2] <- 0
    po[5,i,t,3] <- 0
    po[5,i,t,4] <- 0
    po[5,i,t,5] <- 1

    po[6,i,t,1] <- 0
    po[6,i,t,2] <- 0
    po[6,i,t,3] <- 0
    po[6,i,t,4] <- 0
    po[6,i,t,5] <- 1

    po[7,i,t,1] <- 0
    po[7,i,t,2] <- 0
    po[7,i,t,3] <- 0
    po[7,i,t,4] <- 0
    po[7,i,t,5] <- 1
    } #t
    } #i

    # Likelihood
    for (i in 1:nind){
    # Define latent state at first capture
    z[i,first[i]] <- y[i,first[i]]
    for (t in (first[i]+1):n.yr){
    # State process: draw S(t) given S(t-1)
    z[i,t] ~ dcat(ps[z[i,t-1], i, t-1, 1:7])
    # Observation process: draw O(t) given S(t)
    y[i,t] ~ dcat(po[z[i,t], i, t-1, 1:5])
    } #t
    } #i
    } #model
    ",fill = TRUE)
sink()

get.first <- function(x) min(which(x!=5))
f <- apply(datl$y, 1, get.first)
get.last <- function(x) max(which(x!=5))
l <- apply(datl$y, 1, get.last)
TFmat <- is.na(z.inits) & is.na(datl$z)
for (i in 1:dim(TFmat)[1]){  TFmat[i,1:f[i]] <- FALSE }
z.inits[TFmat] <- sample(size=445, c(2,3), replace=T, prob=c(0.5, 0.5) )

inits <- function(){list(z = z.inits)}

params <- c( "rNB",
             "sigma.F", "eps.F", "mu.F",
             "r", "l.mu.em", "mu.em", "sigma.em",
             "omegaA1",  "omegaB1", "sigma.omegaA", "sigma.omegaB",
```

```
              "OSalpha", "ASalpha", "BSalpha", "OBRalpha", "ABRalpha", "BARalpha",  "mu.pA", "mu.pB",
              "OSalpha1", "ASalpha1", "BSalpha1","OBRalpha1", "ABRalpha1", "BARalpha1","mu.pA1", "mu.pB1
              "sigma.OS.s", "sigma.AS.s", "sigma.BS.s", "sigma.OBR.psi", "sigma.ABR.psi", "sigma.BAR.psi
              "N", "NB", "NF", "NO", "NW", "NH", "NI",  "NE", "Ntot",
              "dmape.obs", "dmape.rep", "tvm.obs", "tvm.rep", "dd.obs", "dd.rep",
              "tturn.obs", "tturn.rep",
              "eps.OS.s", "eps.AS.s", "eps.BS.s", "eps.OBR.psi", "eps.ABR.psi", "eps.BAR.psi", "eps.pA",
              "eps.omegaA", "eps.omegaB", "eps.em",
              "eta.OSalpha", "eta.ASalpha", "eta.BSalpha", "eta.OBRalpha", "eta.ABRalpha", "eta.BARalpha
              "em", "omegaA", "omegaB", "F"
)

# Set initial values of N
# to small numbers for immigrant and emigrants
# because preliminary analyses suggested low rates
# This is necessary to run model and avoid initial values error.
Ni <- array(NA, dim=c(17,26))
Ni[12:17, 2:26] <- 0

inits <- function(){list(z = z.inits, N=Ni)}

# MCMC settings
ni <- 200000; nt <- 50; nb <- 150000; nc <- 3; na <- 1000
out <- jags(datl, inits, params, modfl,
            n.chains = nc, n.thin = nt, n.burnin = nb, n.adapt=na, n.iter=ni,
            parallel=T, module=c("glm", "bugs"))
#save(file=paste("./", m, ".Rdata", sep=""), list="out")
```

2.2 Postprocess Model Output

```
load(".\\outputs\\ipm-e.Rdata") # load model output
library (jagsUI)
library (ggplot2)
library (gridExtra)
source("R\\HDIofMCMC.R") # Function to compute highest density interval. From Kruschke 2011.
data_summary85 <- function(x) {
  m <- median(x)
  ymin <- HDIofMCMC(x, credMass=0.85)[[1]]
  ymax <- HDIofMCMC(x, credMass=0.85)[[2]]
  return(c(y=m,ymin=ymin,ymax=ymax))
}

data_summary95 <- function(x) {
  m <- median(x)
  ymin <- HDIofMCMC(x, credMass=0.95)[[1]]
  ymax <- HDIofMCMC(x, credMass=0.95)[[2]]
  return(c(y=m,ymin=ymin,ymax=ymax))
}

O.surv <- O.trans <- A.surv <- A.trans <- B.surv <- B.trans <-
  array(NA, dim=c(dim(out$sims.list$OSalpha1)[1],3))
B.recap <-array(NA, dim=c(dim(out$sims.list$mu.pB1)[1],2))
A.recap <-array(NA, dim=c(dim(out$sims.list$mu.pA1)[1],2))
```

```r
# Survival
# wild - hacked
O.surv[,1]<- (out$sims.list$OSalpha1[,1,1] +         out$sims.list$OSalpha1[,2,1])/2 -
  (out$sims.list$OSalpha1[,1,2] + out$sims.list$OSalpha1[,2,2])/2
# female - male
O.surv[,2]<- (out$sims.list$OSalpha1[,1,1] + out$sims.list$OSalpha1[,1,2])/2 -
  (out$sims.list$OSalpha1[,2,1] + out$sims.list$OSalpha1[,2,2])/2
# Recruitment
O.trans[,2]<- out$sims.list$OBRalpha1[,1] - out$sims.list$OBRalpha1[,2]# Recruitment female - male
# Nonbreeder (A) Recruitment
A.trans[,1]<- (out$sims.list$ABRalpha1[,1,1] + out$sims.list$ABRalpha1[,2,1])/2 -
  (out$sims.list$ABRalpha1[,1,2] + out$sims.list$ABRalpha1[,2,2])/2 # wild - hacked
A.trans[,2]<- (out$sims.list$ABRalpha1[,1,1] + out$sims.list$ABRalpha1[,1,2])/2 -
  (out$sims.list$ABRalpha1[,2,1] + out$sims.list$ABRalpha1[,2,2])/2 # female - male
# Resight probabilities of nonbreeders indexed as mu.pA1 [effort, h]
A.recap[,1] <- (out$sims.list$mu.pA1[,1,1] + out$sims.list$mu.pA1[,2,1])/2 -
               (out$sims.list$mu.pA1[,1,2] + out$sims.list$mu.pA1[,2,2])/2 # wild - hacked
A.recap[,2] <- (out$sims.list$mu.pA1[,2,1] + out$sims.list$mu.pA1[,2,2])/2 -
              (out$sims.list$mu.pA1[,1,1] + out$sims.list$mu.pA1[,1,2])/2 # high effort - low # Resight p
B.recap[,1] <- out$sims.list$mu.pB1[,2] - out$sims.list$mu.pB1[,1] # high effort - low effort

tab <- data.frame(stage= c("First-year", "First-year", "First-year", "Nonbreeder", "Nonbreeder", "Nonbr
                  param= c("Survival", "Survival","Recruitment", "Recruitment", "Recruitment", "Resight
                  comp= c("Sex", "Hacked", "Sex", "Hacked", "Sex", "Hacked", "Effort", "Effort" ),
                  median=NA, LHDI95=NA, UHDI95=NA, P=NA, important=NA)
tab[1, 4:6]<- round(data_summary95(O.surv[,2]),3)
tab[2, 4:6]<- round(data_summary95(O.surv[,1]),3)
tab[3, 4:6]<- round(data_summary95(O.trans[,2]),3)
tab[4, 4:6]<- round(data_summary95(A.trans[,1]),3)
tab[5, 4:6]<- round(data_summary95(A.trans[,2]),3)
tab[6, 4:6]<- round(data_summary95(A.recap[,1]),3)
tab[7, 4:6]<- round(data_summary95(A.recap[,2]),3)
tab[8, 4:6]<- round(data_summary95(B.recap[,1]),3)
tab$important <- ifelse( ((tab$LHDI95 >= 0 & tab$UHDI95>=0) |
                            (tab$LHDI95 <= 0 & tab$UHDI95<=0)) &
                           (tab$LHDI95 != 0 & tab$UHDI95!=0),
                         "yes", "no")
tab$P[1] <- round(sum(O.surv[,2]>0)/length(O.surv[,2]),2)
tab$P[2] <- round(sum(O.surv[,2]>0)/length(O.surv[,1]),2)
tab$P[3] <- round(sum(O.trans[,2]<0)/length(O.trans[,2]),2)
tab$P[4] <- round(sum(A.trans[,1]>0)/length(A.trans[,1]),2)
tab$P[5] <- round(sum(A.trans[,2]<0)/length(A.trans[,2]),2)
tab$P[6] <- round(sum(A.recap[,1]>0)/length(A.recap[,1]),2)
tab$P[7] <- round(sum(A.recap[,2]>0)/length(A.recap[,2]),2)
tab$P[8] <- round(sum(B.recap[,1]>0)/length(B.recap[,1]),2)
print(tab)
```

```
##        stage       param   comp median LHDI95 UHDI95    P important
## 1 First-year    Survival    Sex  0.209  0.102  0.309 1.00       yes
## 2 First-year    Survival Hacked  0.113  0.005  0.214 1.00       yes
## 3 First-year Recruitment    Sex -0.294 -0.610 -0.014 0.99       yes
## 4 Nonbreeder Recruitment Hacked  0.183 -0.014  0.373 0.97        no
## 5 Nonbreeder Recruitment    Sex -0.493 -0.677 -0.310 1.00       yes
```

```
## 6 Nonbreeder     Resight Hacked  0.050 -0.128  0.229 0.71        no
## 7 Nonbreeder     Resight Effort  0.185  0.009  0.361 0.98       yes
## 8    Breeder     Resight Effort  0.840  0.666  0.968 1.00       yes
```

```
## ---- ipm postprocess 2 --------
# combine survival data
temp.df<- data.frame(Draws=O.surv[,1], Cat="Hacked", State="First-year")
temp.df1<- data.frame(Draws=O.surv[,2], Cat="Sex", State="First-year")
temp.df2<- data.frame(Draws=0, Cat="Hacked", State="Breeder")
temp.df3<- data.frame(Draws=0, Cat="Sex", State="Breeder")
temp.df4<- data.frame(Draws=0, Cat="Hacked", State="Non-breeder")
temp.df5<- data.frame(Draws=0, Cat="Sex", State="Non-breeder")
df.surv2<- rbind(temp.df, temp.df1, temp.df2, temp.df3, temp.df4, temp.df5)
df.surv2$combined<- factor(paste(df.surv2$State, df.surv2$Cat))
df.surv2$combined <- factor(df.surv2$combined, levels=levels(df.surv2$combined)[c(3,4,5,6,1,2)])

# combine recruitment data
temp.df<- data.frame(Draws=0, Cat="Hacked", State="First-year")
temp.df1<- data.frame(Draws=0, Cat="Sex", State="First-year")
temp.df2<- data.frame(Draws=0, Cat="Hacked", State="Breeder")
temp.df3<- data.frame(Draws=0, Cat="Sex", State="Breeder")
temp.df4<- data.frame(Draws=A.trans[,1], Cat="Hacked", State="Non-breeder")
temp.df5<- data.frame(Draws=A.trans[,2], Cat="Sex", State="Non-breeder")
df.rec2<- rbind(temp.df, temp.df1, temp.df2, temp.df3, temp.df4, temp.df5)
df.rec2$combined<- factor(paste(df.rec2$State, df.rec2$Cat))
df.rec2$combined <- factor(df.rec2$combined, levels=levels(df.rec2$combined)[c(3,4,5,6,1,2)])

# combine resight data
temp.df2<- data.frame(Draws=0, Cat="Hacked", State="Breeder")
temp.df3<- data.frame(Draws=0, Cat="Sex", State="Breeder")
temp.df4<- data.frame(Draws=B.recap[,1], Cat="Effort", State="Breeder")
temp.df5<- data.frame(Draws=0, Cat="Hacked", State="Non-breeder")
temp.df6<- data.frame(Draws=0, Cat="Sex", State="Non-breeder")
temp.df7<- data.frame(Draws=A.recap[,2], Cat="Effort", State="Non-breeder")
df.recap2<- rbind(temp.df2, temp.df3, temp.df4, temp.df5, temp.df6, temp.df7)
df.recap2$combined<- factor(paste(df.recap2$State, df.recap2$Cat))
df.recap2$combined <- factor(df.recap2$combined, levels=c(levels(df.recap2$combined)[c(4,5,6,1,2,3)]))
# fix level order for plots
ord <- c("First-year","Non-breeder","Breeder")
df.surv2$State <-  factor(df.surv2$State, levels=ord, labels=ord)
df.rec2$State <-  factor(df.rec2$State, levels=ord, labels=ord)
df.recap2$State <-  factor(df.recap2$State, levels=ord[-1], labels=ord[-1])
```

```
data_summary <- function(x,...) {
  m <- median(x)
  ymin <- HDIofMCMC(x, credMass=0.95)[[1]]
  ymax <- HDIofMCMC(x, credMass=0.95)[[2]]
  return(c(y=m,ymin=ymin,ymax=ymax))
}
txt <- 30
lwd <- 1.5

ps <- ggplot(df.surv2, aes(x = combined, y = Draws, group=combined )) +
  scale_y_continuous(breaks=c(-0.5, 0, 0.5),  labels=c(-0.5, 0, 0.5)) +
  geom_hline(yintercept=0, linetype="solid", size=lwd) +
```

```
  geom_violin(aes(fill=State)) + scale_fill_manual(values=c("First-year"="#f7f7f7", "Non-breeder"="#ccc
  stat_summary(fun.data=data_summary,  geom="pointrange", size=lwd) +
  theme_classic() + theme (text = element_text(size=txt)) +
  ylab("Survival\ndifference") + xlab ("") +
  scale_x_discrete( labels=c("Hacked", "Sex", "Hacked", "Sex","Hacked", "Sex" ))

pt <- ggplot(df.rec2, aes(x = combined, y = Draws, group=combined )) +
  scale_y_continuous(breaks=c(-0.5, 0, 0.5),  labels=c(-0.5, 0, 0.5)) +
  geom_hline(yintercept=0, linetype="solid", size=lwd) +
  geom_violin(aes(fill=State)) +  scale_fill_manual(values=c("First-year"="#f7f7f7", "Non-breeder"="#cc
  stat_summary(fun.data=data_summary,  geom="pointrange", size=lwd) +
  theme_classic() + theme (text = element_text(size=txt)) +
  ylab("Recruitment\ndifference") + xlab ("") +
  scale_x_discrete( labels=c("Hacked", "Sex", "Hacked", "Sex","Hacked", "Sex" ))

pr <- ggplot(df.recap2, aes(x = combined, y = Draws, group=combined)) +
  scale_y_continuous(breaks=c(-0.5, 0, 0.5),  labels=c(-0.5, 0, 0.5)) +
  geom_hline(yintercept=0, linetype="solid", size=lwd) +
  geom_violin(aes(fill=State)) +  scale_fill_manual(values=c("First-year"="#f7f7f7",
    "Non-breeder"="#cccccc", "Breeder"="#969696")) +
  stat_summary(fun.data=data_summary,  geom="pointrange", size=lwd) +
  theme_classic() + theme (text = element_text(size=txt)) +
  ylab("Resight\ndifference") + xlab ("") +
  scale_x_discrete( labels=c("Effort", "Hacked", "Sex", "Effort", "Hacked", "Sex" ))

grid.arrange(ps + theme(legend.position=c(0.5,0.95), legend.direction="horizontal",legend.background = 
             pt + theme(legend.position="none"),
             pr + theme(legend.position="none"),
             nrow=3,
             layout_matrix = rbind(c(1, 1, 1, 1, 1, 1),
                                   c(2, 2, 2, 2, 2, 2),
                                   c(3, 3, 3, 3, 3, 3)))
```

```
## Warning: Groups with fewer than two data points have been dropped.
## Groups with fewer than two data points have been dropped.
## Groups with fewer than two data points have been dropped.
## Groups with fewer than two data points have been dropped.
## Groups with fewer than two data points have been dropped.
## Groups with fewer than two data points have been dropped.
## Groups with fewer than two data points have been dropped.
## Groups with fewer than two data points have been dropped.
## Groups with fewer than two data points have been dropped.
## Groups with fewer than two data points have been dropped.
## Groups with fewer than two data points have been dropped.
```

```
########################
# Plot mean estimates with 95% HDIs
########################
source("R\\HDIofMCMC.R")
df4<- df3<- df2 <- df1 <- data.frame(param=NA , md=NA, lhdi85=NA, uhdi85=NA, lhdi95=NA, uhdi95=NA)
# params with 1 estimate
for (i in 1:3){
ind <- c(17,18,21)[i]
```

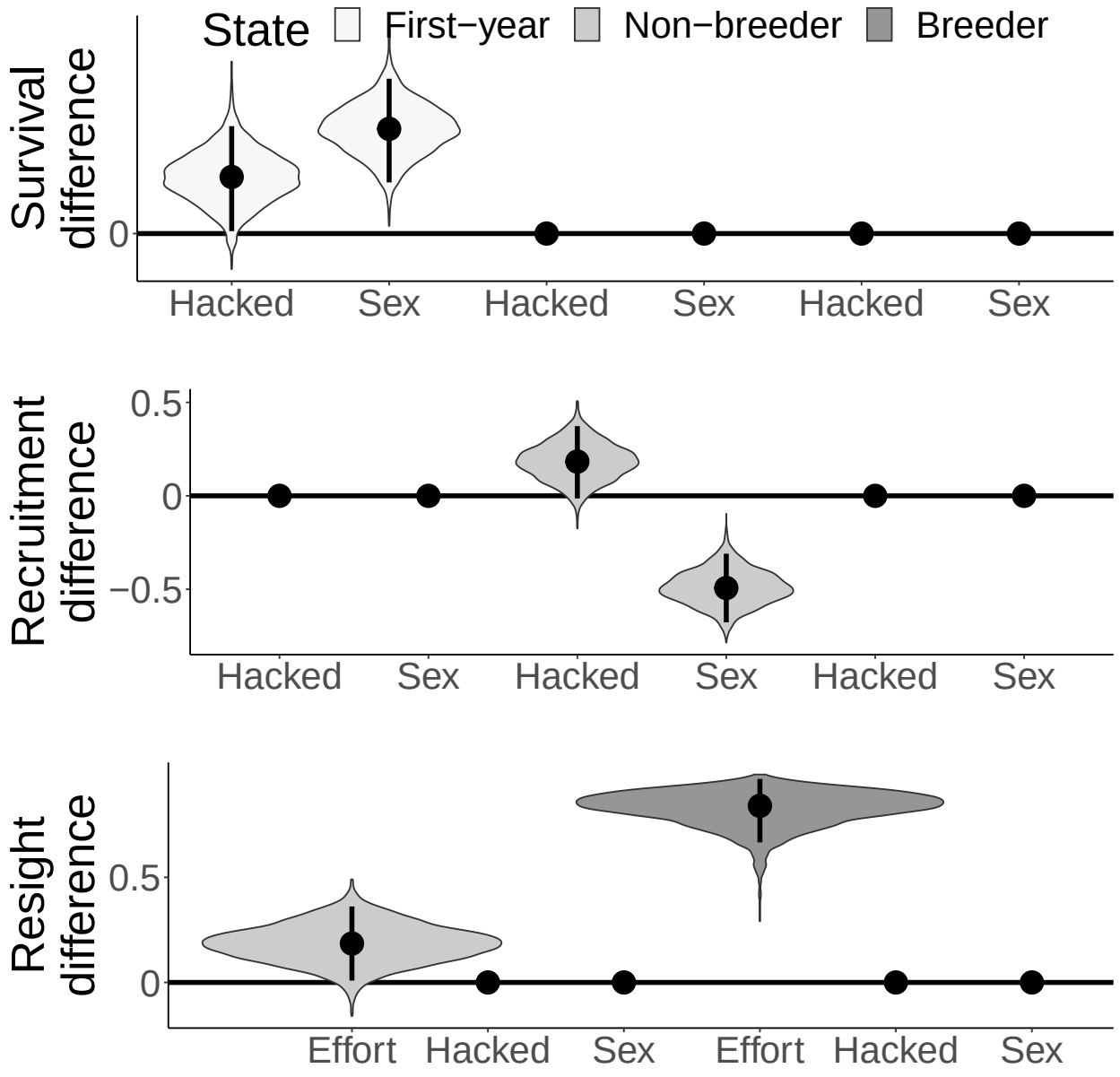


Figure 7: Fig. S8. Violin plots of differences in survival, recruitment, and recapture estimates from IPM without immigration.

```
df1[i,1] <- names(out$sims.list)[ind]
df1[i,2] <- median(out$sims.list[[ind]] )
df1[i,3:4] <- HDIofMCMC(out$sims.list[[ind]], credMass=0.85)
df1[i,5:6] <- HDIofMCMC(out$sims.list[[ind]])
}

# params with 2 estimates
ind <- 19
df2[1,1] <- paste(names(out$sims.list)[ind], "_1", sep="")
df2[1,2] <- median(out$sims.list[[ind]][,1] )
df2[1,3:4] <- HDIofMCMC(out$sims.list[[ind]][,1], credMass=0.85)
df2[1,5:6] <- HDIofMCMC(out$sims.list[[ind]][,1], credMass=0.95)
df2[2,1] <- paste(names(out$sims.list)[ind], "_2", sep="")
df2[2,2] <- median(out$sims.list[[ind]][,2] )
df2[2,3:4] <- HDIofMCMC(out$sims.list[[ind]][,2], credMass=0.85)
df2[2,5:6] <- HDIofMCMC(out$sims.list[[ind]][,2], credMass=0.95)
ind <- 23
df2[3,1] <- paste(names(out$sims.list)[ind], "_1", sep="")
df2[3,2] <- median(out$sims.list[[ind]][,1] )
df2[3,3:4] <- HDIofMCMC(out$sims.list[[ind]][,1], credMass=0.85)
df2[3,5:6] <- HDIofMCMC(out$sims.list[[ind]][,1], credMass=0.95)
df2[4,1] <- paste(names(out$sims.list)[ind], "_2", sep="")
df2[4,2] <- median(out$sims.list[[ind]][,2] )
df2[4,3:4] <- HDIofMCMC(out$sims.list[[ind]][,2], credMass=0.85)
df2[4,5:6] <- HDIofMCMC(out$sims.list[[ind]][,2], credMass=0.95)

ind <- c(16, 20, 22)[1]
df4[1,1] <- paste(names(out$sims.list)[ind], "_1_1", sep="")
df4[1,2] <- median(out$sims.list[[ind]][,1,1] )
df4[1,3:4] <- HDIofMCMC(out$sims.list[[ind]][,1,1], credMass=0.85)
df4[1,5:6] <- HDIofMCMC(out$sims.list[[ind]][,1,1], credMass=0.95)
df4[2,1] <- paste(names(out$sims.list)[ind], "_2_1", sep="")
df4[2,2] <- median(out$sims.list[[ind]][,2,1] )
df4[2,3:4] <- HDIofMCMC(out$sims.list[[ind]][,2,1], credMass=0.85)
df4[2,5:6] <- HDIofMCMC(out$sims.list[[ind]][,2,1], credMass=0.95)
df4[3,1] <- paste(names(out$sims.list)[ind], "_1_2", sep="")
df4[3,2] <- median(out$sims.list[[ind]][,1,2] )
df4[3,3:4] <- HDIofMCMC(out$sims.list[[ind]][,1,2], credMass=0.85)
df4[3,5:6] <- HDIofMCMC(out$sims.list[[ind]][,1,2], credMass=0.95)
df4[4,1] <- paste(names(out$sims.list)[ind], "_2_2", sep="")
df4[4,2] <- median(out$sims.list[[ind]][,2,2] )
df4[4,3:4] <- HDIofMCMC(out$sims.list[[ind]][,2,2], credMass=0.85)
df4[4,5:6] <- HDIofMCMC(out$sims.list[[ind]][,2,2], credMass=0.95)
ind <- c(16, 20, 22)[2]
df4[5,1] <- paste(names(out$sims.list)[ind], "_1_1", sep="")
df4[5,2] <- median(out$sims.list[[ind]][,1,1] )
df4[5,3:4] <- HDIofMCMC(out$sims.list[[ind]][,1,1], credMass=0.85)
df4[5,5:6] <- HDIofMCMC(out$sims.list[[ind]][,1,1], credMass=0.95)
df4[6,1] <- paste(names(out$sims.list)[ind], "_2_1", sep="")
df4[6,2] <- median(out$sims.list[[ind]][,2,1] )
df4[6,3:4] <- HDIofMCMC(out$sims.list[[ind]][,2,1], credMass=0.85)
df4[6,5:6] <- HDIofMCMC(out$sims.list[[ind]][,2,1], credMass=0.95)
df4[7,1] <- paste(names(out$sims.list)[ind], "_1_2", sep="")
```

```
df4[7,2] <- median(out$sims.list[[ind]][,1,2] )
df4[7,3:4] <- HDIofMCMC(out$sims.list[[ind]][,1,2], credMass=0.85)
df4[7,5:6] <- HDIofMCMC(out$sims.list[[ind]][,1,2], credMass=0.95)
df4[8,1] <- paste(names(out$sims.list)[ind], "_2_2", sep="")
df4[8,2] <- median(out$sims.list[[ind]][,2,2] )
df4[8,3:4] <- HDIofMCMC(out$sims.list[[ind]][,2,2], credMass=0.85)
df4[8,5:6] <- HDIofMCMC(out$sims.list[[ind]][,2,2], credMass=0.95)
ind <- c(16, 20, 22)[3]
df4[9,1] <- paste(names(out$sims.list)[ind], "_1_1", sep="")
df4[9,2] <- median(out$sims.list[[ind]][,1,1] )
df4[9,3:4] <- HDIofMCMC(out$sims.list[[ind]][,1,1], credMass=0.85)
df4[9,5:6] <- HDIofMCMC(out$sims.list[[ind]][,1,1], credMass=0.95)
df4[10,1] <- paste(names(out$sims.list)[ind], "_2_1", sep="")
df4[10,2] <- median(out$sims.list[[ind]][,2,1] )
df4[10,3:4] <- HDIofMCMC(out$sims.list[[ind]][,2,1], credMass=0.85)
df4[10,5:6] <- HDIofMCMC(out$sims.list[[ind]][,2,1], credMass=0.95)
df4[11,1] <- paste(names(out$sims.list)[ind], "_1_2", sep="")
df4[11,2] <- median(out$sims.list[[ind]][,1,2] )
df4[11,3:4] <- HDIofMCMC(out$sims.list[[ind]][,1,2], credMass=0.85)
df4[11,5:6] <- HDIofMCMC(out$sims.list[[ind]][,1,2], credMass=0.95)
df4[12,1] <- paste(names(out$sims.list)[ind], "_2_2", sep="")
df4[12,2] <- median(out$sims.list[[ind]][,2,2] )
df4[12,3:4] <- HDIofMCMC(out$sims.list[[ind]][,2,2], credMass=0.85)
df4[12,5:6] <- HDIofMCMC(out$sims.list[[ind]][,2,2], credMass=0.95)

df<- rbind(df1,df2,df4)
df<- df[c(8,9,10,11,1,2,4,5,12,13,14,15,3,16,17,18,19,6,7),]

df$parameter <- c(rep("Survival",6), rep("Recruitment",7), rep("Resight",6))

df$lifestage <- c("First-year", "First-year","First-year",
"First-year", "Nonbreeder", "Breeder",
  "First-year to breeder", "First-year to breeder", "Nonbreeder to breeder" , "Nonbreeder to breeder" ,
  "Nonbreeder", "Nonbreeder", "Nonbreeder", "Nonbreeder","Breeder", "Breeder")

df$she <- c("female, wild","male, wild",
            "female, hacked","male, hacked",
            "","",
            "female", "male",
            "female, wild","male, wild",
            "female, hacked","male, hacked",
            "",
            "female, wild","male, wild",
            "female, hacked","male, hacked",
            "female", "male")
df <- df[,c(7,8,9,2,5,6,3,4)]

df[,c(4:8)] <- format(round(df[,c(4:8)], digits=2), nsmall=2)
# print estimates for survival, recruitment, and resight probabilities
df
```

3. Goodness-of-fit tests for the IPM

```
load(".\\outputs\\ipm-e.Rdata")
par(mfrow=c(2,3), oma=c(0,0,5,0))
plot(out$sims.list$dd.obs, out$sims.list$dd.rep,
     main="Poisson regression model\nfor Productivity",
     ylab="Discrepancy replicate values",
     xlab="Discrepancy observed values",
     xlim=c(0,40), ylim=c(0,40),
     pch=16, cex=0.1, col="gray10")
curve(1*x, from=0, to=40, add=T, lty=2, lwd=2, col="blue")
bp <- round(mean(out$sims.list$dd.rep > out$sims.list$dd.obs),2)
loc <- ifelse(bp < 0.5, "topleft", "bottomright")
legend(loc, legend=bquote(p[B]==.(bp)), bty="n")

hist(out$sims.list$tvm.rep, nclass=50,
     xlab="variance/mean ", main=NA, axes=FALSE)
abline(v=out$mean$tvm.obs, col="red")
axis(1); axis(2)
```

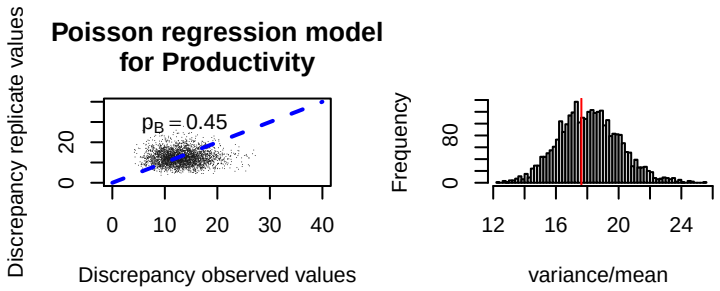


Figure 8: Fig. S9. Goodness-of-fit tests for IPM.

```
plot(jitter(out$sims.list$dmape.obs[,1], amount=300),
     jitter(out$sims.list$dmape.rep[,1], amount=300),
     main="State-space model\n for Breeder counts",
     ylab="Discrepancy replicate values",
     xlab="Discrepancy observed values",
     xlim=c(0,8000), ylim=c(0,8000),
```

```
     pch=16, cex=0.1, col="gray10")
curve(1*x, from=0, to=8000, add=T, lty=2, lwd=2, col="blue")
bp <- round(mean(out$sims.list$dmape.rep[,1] > out$sims.list$dmape.obs[,1]),2)
loc <- ifelse(bp < 0.5, "topleft", "bottomright")
legend(loc, legend=bquote(p[B]==.(bp)), bty="n")
```

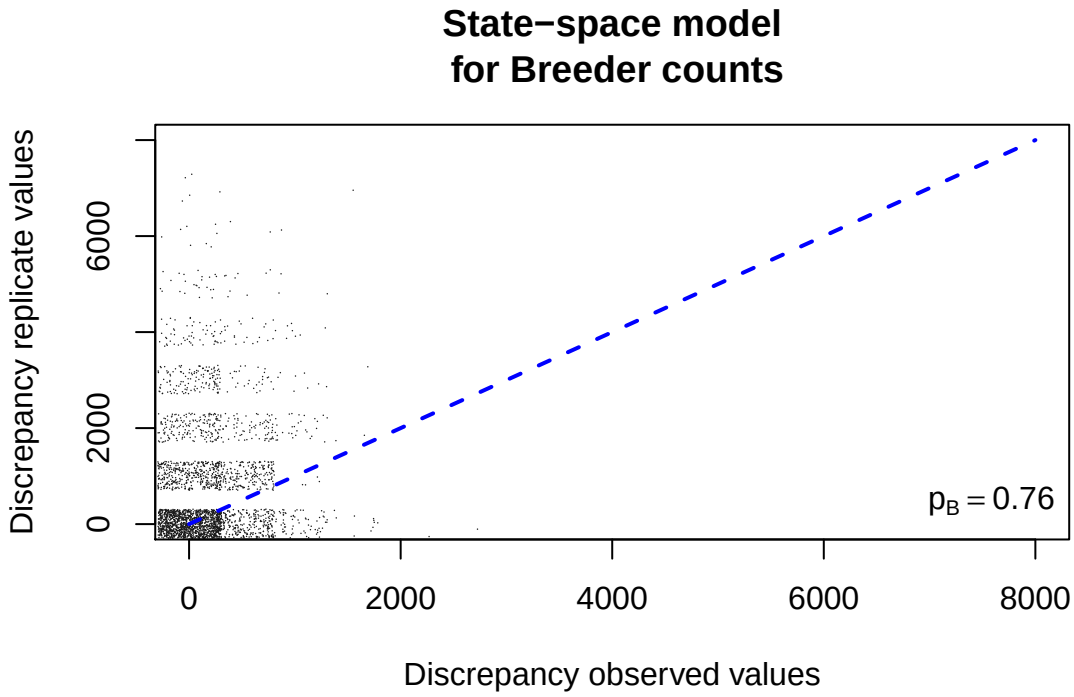


Figure 9: Fig. S10. Goodness-of-fit tests for breeder counts in the IPM.

```
plot(jitter(out$sims.list$dmape.obs[,2], amount=300),
     jitter(out$sims.list$dmape.rep[,2], amount=300),
     main="State-space model\n for non-breeder counts",
     ylab="Discrepancy replicate values",
     xlab="Discrepancy observed values",
     xlim=c(0,15000), ylim=c(0,15000),
     pch=16, cex=0.1, col="gray10")
curve(1*x, from=0, to=30000, add=T, lty=2, lwd=2, col="blue")
bp <- round(mean(out$sims.list$dmape.rep[,2] > out$sims.list$dmape.obs[,2]),2)
loc <- ifelse(bp < 0.5, "topleft", "bottomright")
legend(loc, legend=bquote(p[B]==.(bp)), bty="n")
```

```
plot(jitter(out$sims.list$dmape.obs[,3], amount=200),
     jitter(out$sims.list$dmape.rep[,3], amount=200),
     main="State-space model\n for first-year counts",
     ylab="Discrepancy replicate values",
     xlab="Discrepancy observed values",
     xlim=c(0,12000), ylim=c(0,12000),
     pch=16, cex=0.1, col="gray10")
curve(1*x, from=0, to=15000, add=T, lty=2, lwd=2, col="blue")
```

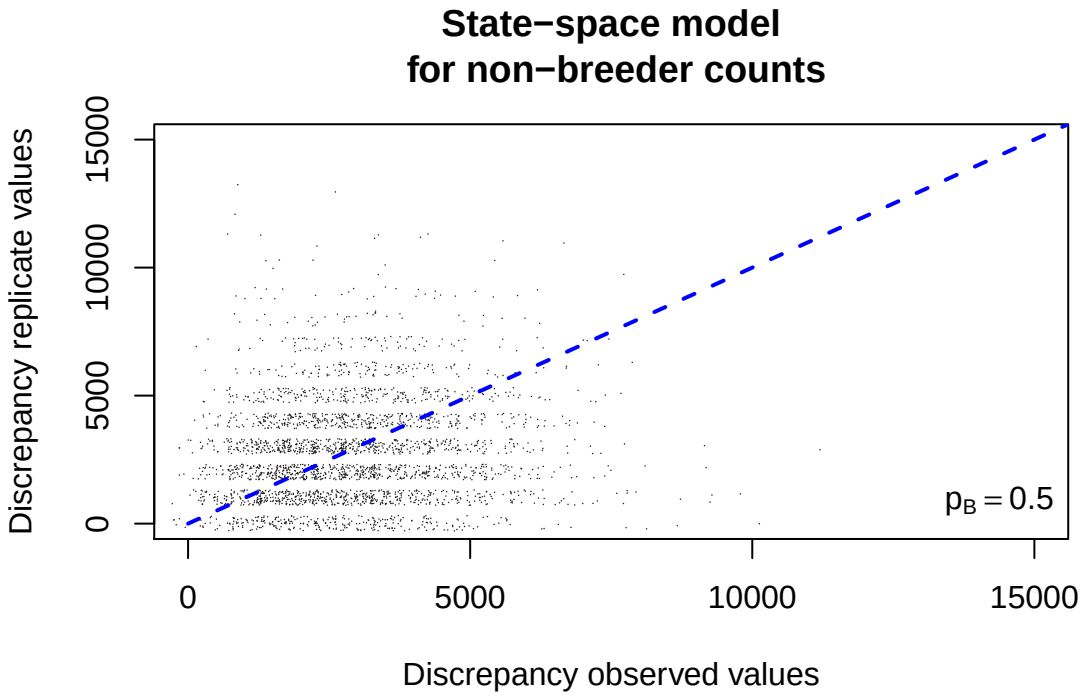


Figure 10: Fig. S11. Goodness-of-fit tests for non-breeder counts in the IPM.

```
bp <- round(mean(out$sims.list$dmape.rep[,3] > out$sims.list$dmape.obs[,3]),2)
loc <- ifelse(bp < 0.5, "topleft", "bottomright")
legend(loc, legend=bquote(p[B]==.(bp)), bty="n")
```

4. Literature cited

Abadi, F., O. Gimenez, B. Ullrich, R. Arlettaz, and M. Schaub. 2010. Estimation of immigration rate using integrated population models. Journal of Applied Ecology 47:393–400.

Gimenez, O., V. Rossi, R. Choquet, C. Dehais, B. Doris, H. Varella, J.-P. Vila, and R. Pradel. 2007. State-space modelling of data on marked individuals. Ecological Modelling 206:431–438.

Kéry, M., and M. Schaub. 2012. Bayesian Population Analysis Using WinBUGS: A hierarchical perspective. First Edit. Elselvier Inc., Oxford, UK.

Krushke, J. K. 2011. Doing Bayesian Data Analysis: A Tutorial with R and BUGS. Elsevier Inc., Burlington, Massachusetts, USA.

Paquet, M., J. Knape, D. Arlt, P. Forslund, T. Pärt, Ø. Flagstad, C. G. Jones, M. A. C. Nicoll, K. Norris, J. M. Pemberton, H. Sand, L. Svensson, V. Tatayah, P. Wabakken, C. Wikenros, M. Åkesson, and M. Low. 2021. Integrated population models poorly estimate the demographic contribution of immigration. Methods in Ecology and Evolution 12:1899–1910.

Royle, J. A. 2008. Modeling Individual Effects in the Cormack–Jolly–Seber Model: A State–Space Formulation. Biometrics 64:364–370.

Schaub, M., and D. Fletcher. 2015. Estimating immigration using a Bayesian integrated population model: choice of parameterization and priors. Environmental and Ecological Statistics 22:535–549.

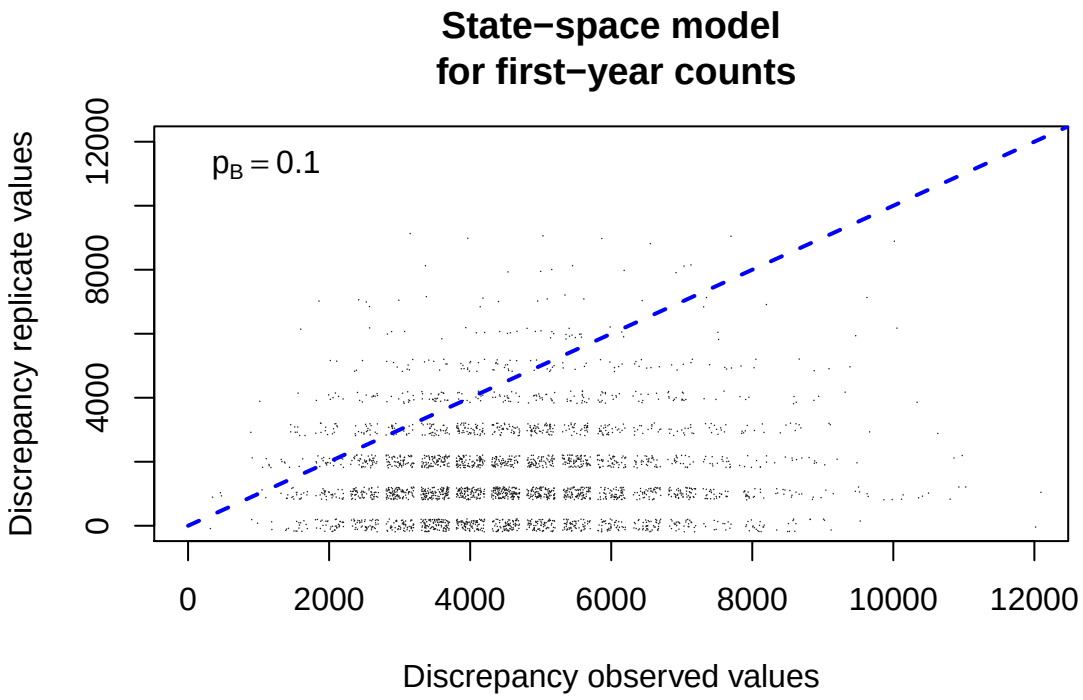


Figure 11: Fig. S12. Goodness-of-fit tests for first-year counts in the IPM.